Logic and Games on Automatic Structures

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften
der RWTH Aachen University zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Dipl.-Math., Dipl.-Inform.
Łukasz Kaiser
aus Wrocław, Polen

Berichter: Prof. Dr. Erich Grädel
Prof. Dr. Joost-Pieter Katoen
Prof. Dr. Damian Niwiński

Tag der mündlichen Prüfung: 26. Juni 2008

Preface

An important connection between logic and games is based on the correspondence between the evaluation of a logical formula and a game played by two opponents, one trying to show that the formula is true and the other trying to prove it false. This relationship has been implicitly known for a long time, even before mathematical logic and game theory were formalized. It was formally established in the 1950s by Paul Lorenzen [52, 53, 54] in the form of dialogue games and later developed in another form by Jaakko Hintikka [38]. Since then, it has inspired numerous research directions, leading both to new logics and interesting insights about the classical ones.

In computer science, there are two main approaches to algorithmically exploiting the correspondence between logic and games. On the one hand, games played on syntactic objects such as formulas, programs or language expressions were studied. Such games, derived from the dialogue games of Lorenzen and their extensions, were used to build theorem provers for classical and intuitionistic first-order logic [27], to give semantics to programming languages and to verify programs [1], and in various other contexts in linguistics and artificial intelligence (see [56] for an overview). On the other hand, games can be played in a more semantic setting, where players choose elements of a mathematical structure. In this way, following the ideas of Hintikka, games are used to evaluate formulas of both first-order and second-order logic on finite structures and to verify temporal properties on Kripke structures.

The algorithmic utility of such semantic games is apparent in the verification of μ-calculus formulas on finite structures. While there is no known polynomial-time algorithm for this problem, parity and mean-payoff games were used to narrow its complexity class [42], to obtain algorithms that are among the most efficient ones in practice [43, 76], and

recently to find the first sub-exponential algorithm for the verification of μ-calculus [44].

Extending the game-based algorithmic approach to first-order logic on *infinite* structures that arise in computer science is the main motivation for this thesis. In structures that are to be stored and manipulated by a computer, even infinite ones, elements and relations must be represented in a finite way. For example, elements can be defined in an inductive way using algebraic datatypes, and relations can be given by programs that compute them. To avoid the problems inherent in theorem-proving with mathematical induction [16], we focus on the semantic setting where games are played using representations of elements of the structure. Since we are interested in algorithmic results, we additionally restrict our consideration to one prominent class of finitely presentable structures that has a decidable first-order theory, namely to *automatic structures*.

Automatic structures, introduced first in [40] and later in [46] and [10], contain elements represented by words over a finite alphabet. Relations in these structures are represented by synchronous automata that perform step-by-step transitions on tuples of symbols from the alphabet. A prominent example of an automatic structure is Presburger arithmetic $(\mathbb{N}, +)$, for which the natural way of writing numbers as sequences of digits and the standard column addition method constitute an automatic presentation. In this thesis we allow words that represent elements of an automatic structure to be infinite; such structures are sometimes called ω-automatic.

The basic fact that first-order logic is decidable on automatic structures follows from the closure properties of automata, both the ones working on finite and those on infinite words [13]. To develop a correspondence between games and logic on automatic structures, we first look for suitable extensions of first-order logic that remain decidable on this class of structures. We study the notion of game quantification and extend the open and closed game quantifiers, known in model theory of infinitary logics, to a regular game quantifier defined on automatic presentations.

This quantifier corresponds to a construction of the words representing

elements of a structure by means of a game played step-by-step with the letters from the alphabet. In this way the game quantifier intuitively captures games played by two players during the construction of elements of an automatic structure. We show that this quantifier effectively preserves regularity, which follows from the possibility to determinize alternating automata.

We study the expressive power of the regular game quantifier. We identify classes of structures on which logic extended with this quantifier collapses to pure first-order logic and distinguish these from those on which it has larger expressive power. We prove that already quite basic structures, for example the binary tree, are complete for first-order logic extended with the game quantifier. To get a better understanding of the expressive power of this extended logic on weaker structures, we identify a class of inductive automorphisms and show that these preserve relations defined using the game quantifier.

Model-checking games for the extension of first-order logic with the game quantifier on automatic presentations can be defined in a more natural way than for pure first-order logic. To introduce them, we first recall the classical two-player parity games, which arise as the model-checking games for modal μ-calculus. We extend parity games to the multiplayer setting where two coalitions play against each other with a special kind of *hierarchical* imperfect information about actions of the players. This extension allows us to define the appropriate model-checking games for the first-order logic with the regular game quantifier on automatic presentations.

We look closely at the definition of hierarchical games to identify the influence of various factors on algorithmic properties of these games. On the one hand, it is essential to assume that the information is hidden in a hierarchical way and that players take moves in a prescribed alternating order. We show that allowing non-alternating moves of players makes the problem of determining the winners of these games undecidable. On the other hand, hierarchical games are robust under manipulations of the kind of the winning condition in the game. The winning condition is

often represented by listing which sets of positions must appear infinitely often for one coalition to win the play (the Muller winning condition), by assigning priorities to positions and requiring that the minimal priority that appears infinitely often is even (the parity condition), or in other forms (e.g. the Streett and Rabin conditions). The complexity of establishing the winner in hierarchical games is not significantly affected by the kind of winning condition, as far as it is an ω-regular set of paths through the game graph.

One reason for the robustness of these games under changes of the winning condition is that adding a finite memory to strategies suffices to reduce games with complex winning conditions to games with easier ones. For example, it is a well-known result that games with Muller winning condition can be reduced to games with parity winning condition using finite memory. Gurevich and Harrington proved this in [35] using a special kind of memory structure called the latest appearance record, which follows the idea of order vectors introduced by McNaughton. Later, Zielonka introduced split trees [78], which form another memory structure that allows to reduce Muller games to parity games and gives more insight into the amount of memory that is needed for the reduction when the Muller condition is fixed.

While these results are well-known for games over finitely many priorities, it has not been known how to extend these memory structures to games on infinite arenas with infinitely many priorities. We generalize the latest appearance record to a memory structure that can store a finite number of priorities that appeared in the play. Memory structures of this kind are sufficient for winning Muller games with a finite or co-finite number of sets in the Muller condition, and additionally for a few other classes of games with infinitely many priorities. Zielonka trees can be extended to certain classes of games with infinitely many priorities as well. We investigate these memory structures and show that the reductions known for the case of finitely many priorities can be generalized to games with infinitely many priorities, assuming certain constraints on the structure of the Zielonka tree for the winning condition.

Another common direction in which first-order logic can be extended is by adding generalized Lindström quantifiers [51]. This extension has been widely studied both in logic and in descriptive complexity theory, where it is used to describe complexity classes for machines with oracles [30]. We address the following question: which generalized unary quantifiers can be added to first-order logic without introducing non-regular relations on automatic structures. We answer this question with a complete characterization of such quantifiers. These are the cardinality and modulo counting quantifiers, i.e. "there exist infinitely many", "there exist uncountably many" and "there exist k mod m many". We show that these quantifiers indeed preserve regularity on all automatic structures, including the non-injective ω-automatic ones. As a corollary we answer a question of Blumensath [8] and prove that all countable ω-automatic structures are in fact finite-word automatic.

A natural and often considered question is which other classes of structures still have decidable first-order logic with extensions. To investigate it, we study a large class of structures introduced by Colcombet and Löding [18], called generalized-automatic structures. These structures are given by interpretations transforming their first-order theory to monadic second-order theory of a tree or of a linear order with additional labels. Because of these arbitrary labellings that are included, the methods from the previous chapters can not be directly generalized. Instead, we use the composition method for monadic second-order theory over linear orders and trees.

Using the composition method we show that the second-order cardinality quantifiers, both the infinity and the uncountability quantifier on the number of sets X satisfying a formula $\varphi(X)$, can be effectively eliminated from monadic second-order logic, both over trees and over countable linear orders. This elimination procedure can be transferred to first-order logic only on injectively presented generalized-automatic structures and is thus a partial generalization of the previous results. We discuss the outlook on further extensions and applications of our work in the final chapter.

Acknowledgment. I wish to express my deep gratitude to Erich Grädel, not only for giving me the opportunity to meet inspiring people and to develop my interests, but also for his support and focus on the quality of scientific work. I am also grateful to Wolfgang Thomas for his constant encouragement and help.

I would like to thank Alexander Rabinovich and Sasha Rubin, I have gained much from their fascinating ways of thinking. I also thank Diana Fischer, Sophie Pinchinat, Dietmar Berwanger, Michael Ummels and Tobias Ganzow for their illuminating comments and remarks which have contributed to this thesis, and for carefully correcting this text. I am especially grateful to my brother-in-arms Vince Bárány for his insightful thoughts, thorough discussions and for his uncommon companionship during the years in Aachen.

For making my time in Aachen so enjoyable, I thank my friends Alex, Antje, Michaela, Christof, Jan, Frank, Kari, Michael, Nico, Philipp, Stefan, and also Bernd, Roman and Wenyun. My deepest gratitude goes to those at home, especially to my parents Andrzej and Teresa, for their unending love, trust and encouragement.

Contents

List of Figures

1 Logics, Structures and Presentations

In this chapter we review the standard notions of first-order and monadic second-order logic. We introduce linear orders and trees and state a few elementary properties of these structures. We recall the correspondence between monadic second-order logic and automata on infinite words together with basic facts from automata theory . We introduce automatic structures using presentations by automata and characterize them both by first-order and by monadic second-order to first-order interpretations. Finally, we discuss the composition method for monadic second-order logic over linear orders and trees.

As most of these notions are standard, we assume that the reader is already familiar with them and thus we do not discuss them in detail, but concentrate instead on fixing the notation. More thorough introductions to logic and automata can be found in many textbooks and surveys, e.g. in [24, 23, 72, 74].

1.1 First-order and monadic second-order logic

A structure $\mathfrak{A} = (A, R_1, \ldots, R_n, f_1, \ldots, f_m)$ is given by a set A, called the domain of $\mathfrak{A}$, a number of relations R_i and a number of functions f_j. If we denote the arity of R_i by r_i and the arity of f_j by s_j then $R_i \subseteq A^{r_i}$ and $f_j : A^{s_j} \to A$. We say that $\mathfrak{A}$ is a *relational structure* if it contains no functions, only relations. Every structure can be coded as a relational structure by replacing each function f_j by its graph G_{f_j}, which is a relation of arity $s_j + 1$ such that $G_{f_j}(\overline{x}, y) \iff f_j(\overline{x}) = y$. The signature of the structure $\mathfrak{A}$ is denoted by $\sigma(\mathfrak{A}) = \{R_i^{(r_i)}\} \cup \{f_j^{(s_j)}\}$, where R_i and f_j are now just symbols with appropriate arities. Note that we only consider structures with finite signatures in this thesis, even though some of the

results do not rely on this assumption.

Both first-order and monadic second-order logic formulas over a signature τ are built from atomic formulas using boolean connectives and quantifiers. Atomic formulas in first-order logic (FO) have either the form $t_1 = t_2$ or $R_i(t_1, \ldots, t_{r_i})$, where t_k are terms, i.e. either first-order variables, denoted $x, y, x_1, y_1, \ldots$, or expressions of the form $f_j(t_1, \ldots, t_{s_j})$. In monadic second-order logic (MSO) there are additional atomic formulas of the form $t \in X$, where t is again a term and X is a second-order variable. We use the standard boolean connectives $\wedge, \vee, \neg$ to denote conjunction, disjunction and negation and we write $\varphi \to \psi$ for $(\neg\varphi) \vee \psi$ and $\varphi \leftrightarrow \psi$ for $(\varphi \to \psi) \wedge (\psi \to \varphi)$. In first-order logic it is possible to quantify over first-order variables using either existential or universal quantifiers, i.e. to write $\exists x\varphi$ or $\forall x\varphi$. In second-order logic there is the additional possibility to quantify over second-order variables, i.e. to write $\exists X\varphi$ or $\forall X\varphi$. When writing formulas we use the convention that negation and quantifiers bind stronger than $\wedge$ and $\vee$, which in turn bind stronger than $\to$ and $\leftrightarrow$, so that $\forall xRx \wedge \neg Ry \to Rz$ is to be understood as $((\forall xRx) \wedge (\neg Ry)) \to Rz$. We say that a variable in a formula φ that appears in scope of a quantifier is *bound* and in the other case it is *free*. When writing $\varphi(x_1, \ldots, x_n)$ (or sometimes shorter $\varphi(\overline{x})$) we mean that all free variables of φ are contained in $\{x_1, \ldots, x_n\}$.

Whether a formula $\varphi(\overline{x}, \overline{X})$ holds in a structure $\mathfrak{A}$, given an assignment $\theta : \overline{x} \to A$ for first-order variables and an assignment $\Theta : \overline{X} \to \mathcal{P}(A)$ for second-order ones, denoted $\mathfrak{A}, \theta, \Theta \vDash \varphi$, is defined inductively. First, we extend the assignment θ to all terms by putting $\theta(f_j(t_1, \ldots, t_{s_j})) = f_j(\theta(t_1), \ldots, \theta(t_{s_j}))$. Note that f_j on the left side of the equation is only a symbol, while on the right side it is a function of $\mathfrak{A}$. We will often abuse the notation in this way and we sometimes write $f_i^{\mathfrak{A}}$ to point out that we have the function (or relation) in the structure in mind, rather than the symbol. Moreover, we extend this notation to formulas, so given a formula $\varphi(\overline{x})$ we write $\varphi^{\mathfrak{A}}$ for the relation defined by $\varphi^{\mathfrak{A}}(\overline{a}) \iff \mathfrak{A}, \overline{x} \leftarrow \overline{a} \vDash \varphi$. In the inductive definition of the semantics below we write $\theta[x \leftarrow a]$ (or analogous for Θ) for the assignment $\theta' : \overline{x} \cup \{x\} \to A$ that maps x to a

and every other variable x' to $\theta(x')$.

- $\mathfrak{A}, \theta, \Theta \models t_1 = t_2$ whenever $\theta(t_1) = \theta(t_2)$,
- $\mathfrak{A}, \theta, \Theta \models R_i(t_1, \ldots, t_{r_i})$ whenever $R_i^{\mathfrak{A}}(\theta(t_1), \ldots, \theta(t_{r_i}))$ holds,
- $\mathfrak{A}, \theta, \Theta \models t \in X$ whenever $\theta(t) \in \Theta(X)$,
- $\mathfrak{A}, \theta, \Theta \models \neg\varphi$ whenever it is not the case that $\mathfrak{A}, \theta, \Theta \models \varphi$,
- $\mathfrak{A}, \theta, \Theta \models \varphi \wedge \psi$ whenever $\mathfrak{A}, \theta, \Theta \models \varphi$ and $\mathfrak{A}, \theta, \Theta \models \psi$,
- $\mathfrak{A}, \theta, \Theta \models \varphi \vee \psi$ whenever $\mathfrak{A}, \theta, \Theta \models \varphi$ or $\mathfrak{A}, \theta, \Theta \models \psi$,
- $\mathfrak{A}, \theta, \Theta \models \exists x\varphi$ whenever $\mathfrak{A}, \theta[x \leftarrow a], \Theta \models \varphi$ for some $a \in A$,
- $\mathfrak{A}, \theta, \Theta \models \forall x\varphi$ whenever $\mathfrak{A}, \theta[x \leftarrow a], \Theta \models \varphi$ for all $a \in A$,
- $\mathfrak{A}, \theta, \Theta \models \exists X\varphi$ whenever $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for some $B \subseteq A$,
- $\mathfrak{A}, \theta, \Theta \models \forall X\varphi$ whenever $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for all $B \subseteq A$.

Sometimes we evaluate MSO formulas in the *weak semantics* and in such case only finite sets are substituted for second-order variables. In this setting the last two items above must be replaced by the following:

- $\mathfrak{A}, \theta, \Theta \models \exists X\varphi$ in the weak semantics if $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for some *finite* $B \subseteq A$,
- $\mathfrak{A}, \theta, \Theta \models \forall X\varphi$ in the weak semantics if $\mathfrak{A}, \theta, \Theta[X \leftarrow B] \models \varphi$ for all *finite* $B \subseteq A$.

We often call formulas that we evaluate using the weak semantics *weak monadic second-order logic* (WMSO) formulas.

1.2 Linear orders and trees

Linear orders are a prominent example of structures that appear throughout this thesis. The standard ordering of natural numbers is denoted $(\omega, <)$ or $(\mathbb{N}, <)$, the orderings of integers and rational numbers are denoted $(\mathbb{Z}, <)$ and $(\mathbb{Q}, <)$, respectively. Recall that $(\mathbb{Q}, <)$ is dense, meaning that between any two elements $x < y$ there is another element z such that $x < z < y$. On the other hand, we say that a linear order $(L, <)$ is *scattered* if it does not embed any dense order, or equivalently if it does not embed $(\mathbb{Q}, <)$. Given a linear order $(I, <)$ and orders $(L_i, <_i)$ for every $i \in I$, the sum $\sum_I L_i$ is defined as the linear ordering of $\bigcup_{i \in I} L_i \times \{i\}$ such that

$$(l, i) < (l', i') \iff i < i' \text{ or } i = i' \text{ and } l <_i l'.$$

We write $(L_0, <_0) + (L_1, <_1)$ for the sum over $(\{0, 1\}, <)$. For example $(\mathbb{Z}, <)$ is isomorphic to $\omega^* + \omega$, where ω^* is the standard ordering of negative integers, $(\{-1, -2, -3, \ldots\}, <)$. Note that sometimes we use the same name for the structure and its universe when the meaning is clear from the context, as in the case of ω above. For a linear order L and $a, b \in L$ we use the standard notation for intervals, so for example $[a, b)$ is a left-closed and right-open interval. Moreover, we write $L|_{[a,b)}$ for the order $L \cap [a, b)$, and for $X \subseteq L$ we use analogous notation, i.e. $X|_{[a,b]}$ for $X \cap [a, b]$.

One can classify countable linear orders using the sum operation defined above in the following way. Every countable linear order can be written as a dense sum of scattered linear orders, i.e. as $\sum_{\mathbb{Q}} L_i$ where each $(L_i, <_i)$ is a scattered linear order. Moreover, Hausdorff classified countable scattered linear orders in classes VD_α defined inductively as follows. $\mathsf{VD}_0 = \{\mathbf{1}\}$ consists of the linear order having one element (we leave out the empty linear order). For each ordinal $\alpha > 0$, VD_α consists of those linear orders that can be written as a sum $\sum_{\mathbb{Z}} L_i$ with each $L_i \in \mathsf{VD}_\beta$ for some $\beta < \alpha$. Let VD be the union of all the VD_α. Hausdorff has shown that VD contains every countable scattered linear order, and the *Hausdorff-rank* of a linear order $L \in \mathsf{VD}$ is defined as the smallest α such that $L \in \mathsf{VD}_\alpha$.

A linear order is *complete* if every one of its subsets has a least upper bound. Given a linearly ordered set $(L, <)$ its *Dedekind cuts* are subsets $C \subseteq L$ that are downward closed. The *completion* of $(L, <)$ is the set $DC(L)$ of Dedekind cuts of L ordered by inclusion, containing $\varnothing$ if there no least element in L and excluding $\varnothing$ in the other case. Note that the completion of L, which we denote by $\overline{L}$ and consider only up to isomorphism, is a complete linear order with both endpoints, i.e. a least and a greatest element.

Every linear order is naturally equipped with the *order topology* generated by open intervals. This allows us to speak of neighborhoods, open sets, limit (alternatively condensation or accumulation) points, and all other topological notions on every linear order.

For a given set A we denote by A^* the set of all finite sequences of elements of A, by A^ω the set of all infinite sequences of elements of A (i.e. functions $\omega \to A$), and $A^{\leq\omega} = A^* \cup A^\omega$. For any sequence $s = s_0 s_1 s_2 \ldots \in A^{\leq\omega}$ we denote by $|s|$ the length of s (either a natural number or ω) and by $s|_n = s_0 \ldots s_{n-1}$ the finite sequence composed of the first n elements of s, with $s|_0 = \varepsilon$, the empty sequence. We write $s[n]$ for the $(n+1)$st element of s (as we start counting from 0), so $s[n] = s_n$ for $n \in \mathbb{N}$. Similarly, $s[n, m]$ is the factor $s[n]s[n+1]\cdots s[m]$ and $s[n, m)$ is defined as $s[n, m-1]$, therefore in our notation $s|_n = s[0, n)$. Given a finite sequence s and a sequence $r \in A^{\leq\omega}$ we denote by $s \cdot r$ (or just sr) the concatenation of s and r. For the n-times concatenation $s\cdots s$ we use the symbol s^n. Moreover, we write $s \sqsubseteq t$ if s is a prefix of t, i.e. if there exists a sequence r such that $t = sr$, and in such case we denote the difference by $t - s = r$. A subset B of $A^{\leq\omega}$ is said to be prefix-closed if for every $t \in B$ and $s \sqsubseteq t$ it holds that $s \in B$. For an infinite sequence $s \in A^\omega$ the set of elements that appear infinitely often in this sequence is denoted by $\inf(s)$.

We sometimes extend all notations introduced above to vectors of sequences, so for example if $\bar{s}$ is a vector then $\overline{s[n]}$ or equivalently $\bar{s}[n]$ is the vector consisting of the $(n+1)$st element of each sequence in $\bar{s}$. Moreover, given a function $f : A \to B$ and $u \in A^{\leq\omega}$ we denote by $f(u)$ the sequence $f(u[0])\ f(u[1])\ f(u[2])\ \ldots \in B^{\leq\omega}$.

A tree $\mathfrak{T} = (T, \sqsubseteq, P_1, \ldots, P_n)$ over a set A is a structure where the universe T is a prefix-closed subset of A^*. The relation $\sqsubseteq$ is interpreted as the standard prefix ordering and P_i are arbitrary unary predicates, sometimes called labels. We say that a tree $\mathfrak{T}$ includes successor labels if for each $a \in A$ there is a predicate P_a in the structure such that $P_a(u)$ holds exactly when $u = va$, which allows to distinguish all immediate successors of any node $v \in T$. Given a finite set A we denote by $\mathfrak{T}(A)$ the complete tree over A, which is a structure with the universe A^*, with successor labels and without any other predicates. An important example of such a tree is the complete binary tree, $\mathfrak{T}(\{0,1\}) = (\{0,1\}^*, \sqsubseteq, S_0, S_1)$ where $S_0(u)$ holds if $u = v0$ and $S_1(u)$ if $u = v1$. In the case of complete k-ary trees we may write $\mathfrak{T}(k)$ for $\mathfrak{T}(\{0, \ldots, k-1\})$, so the complete binary tree is sometimes denoted $\mathfrak{T}(2)$.

For a node $v \in \mathfrak{T}$ the restriction $\mathfrak{T}|_v$ is the substructure of $\mathfrak{T}$ with universe $\{w \mid v \sqsubseteq w\}$. The universe of this substructure is not prefix-closed for $v \neq \varepsilon$ (and so formally it is not a tree), but it is isomorphic to a tree $\mathfrak{T}_v$ with universe $\{w \mid vw \in \mathfrak{T}\}$ and predicates derived from $\mathfrak{T}$ by putting $P_i^{\mathfrak{T}_v}(w) \iff P_i^{\mathfrak{T}}(vw)$. We say that $\mathfrak{T}_v$ (and sometimes $\mathfrak{T}|_v$ as well) is the subtree of $\mathfrak{T}$ rooted at v. Similarly, we use this notation for every set $X \subseteq T$, i.e. $X_v = \{u \mid vu \in X\}$ and $X|_v = X \cap T_v = vX_v$.

A *path* of $\mathfrak{T}$ is a prefix-closed subset P of T that is linearly ordered by $\sqsubseteq$, thus either finite or of order-type ω. We identify a path P with the sequence $\pi = n_0 n_1 n_2 \ldots \in A^{\leq\omega}$ such that $P = \{n_0 \ldots n_s \mid s \leq |\pi|\}$, and given a sequence $\pi \in A^{\leq\omega}$ we write $\mathrm{Pref}(\pi)$ for the corresponding path P. Let π be a sequence corresponding to a path P through the complete binary tree $\mathfrak{T}(2)$. Given a subset Q of the path P and a sequence X_i of subsets of $\mathfrak{T}(2)$, we say that X is a *sum of X_i along π, Q*,

$$X = \sum_{\pi,Q} X_i,$$

if $X \cap P = Q$ and, for every i, the labels aside from the path P on step i are exactly X_i, i.e. if $\pi[i] = 0$ then $X_{\pi|_i 1} = X_i$ and if $\pi[i] = 1$ then $X_{\pi|_i 0} = X_i$, as illustrated for a path $\pi = 010\ldots$ and $Q = \mathrm{Pref}(\pi)$ in Figure 1.1.

In addition to the trees defined above, we study trees over infinite

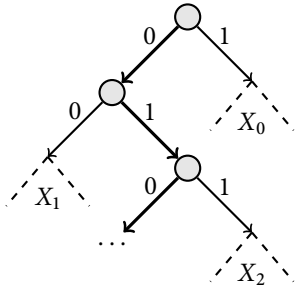


Figure 1.1: Sum of sets along a path in a tree.

words, called ω-trees. A complete ω-tree over a finite set A is defined in an analogous way to the standard tree as $\mathfrak{T}^\omega(A) = (A^{\leq\omega}, \sqsubseteq, \{S_a\}_{a\in A})$, where S_a are again successor labels, i.e. $S_a(u)$ holds only for finite words $u = va$. Moreover, we sometimes extend the trees with the equal-length relation el, defined by $\mathsf{el}(u,w) \iff |u| = |w|$. We denote a tree $\mathfrak{T}$ extended with this binary relation $\mathfrak{T}_{\mathsf{el}}$, so for example $\mathfrak{T}^\omega_{\mathsf{el}}(\{0,1\})$ is the complete binary ω-tree extended with the equal-length relation, i.e. $(\{0,1\}^{\leq\omega}, \sqsubseteq, S_0, S_1, \mathsf{el})$.

1.3 Automata on ω-words

The order $(\omega, <)$ and the binary tree $\mathfrak{T}(2)$ play an important role in computer science and logic because the monadic second-order theory of both of these structures is decidable, as proved by Büchi [13] and Rabin [65] respectively. These proofs use the notion of an automaton, either a word automaton for $(\omega, <)$ or a tree automaton for $\mathfrak{T}(2)$, and establish a one-to-one correspondence between relations recognized by automata and the ones definable in monadic second-order logic.

An ω-word automaton $\mathcal{A}$ over a finite alphabet Σ is a tuple $(Q, \Delta, q_0, \mathcal{F})$ where Q is a finite set of states, Δ is a transition relation $\Delta \subseteq Q \times \Sigma \times Q$, $q_0 \in Q$ is an initial state and $\mathcal{F}$ is an acceptance condition. An automaton is *deterministic* if Δ is a function $Q \times \Sigma \to Q$. In the case of the standard

finite-word automata, the acceptance condition consists only of a set of final states and a word is accepted if some run ends in a final state. For ω-word automata the acceptance condition $\mathcal{F}$ is a set of runs, i.e. infinite sequences of states, which are considered accepting for the automaton, so $\mathcal{F} \subseteq Q^\omega$. In practice $\mathcal{F}$ is described in a finite way and there are a few representations that are often used for this purpose.

- The *Büchi condition* is represented by a set $F \subseteq Q$ and $\mathcal{F} = \{s \in Q^\omega \mid \inf(s) \cap F \neq \varnothing\}$.

- The *parity condition* is defined using a mapping $\Omega : Q \to \{0, \ldots, d\}$ and $\mathcal{F} = \{s \in Q^\omega \mid \min(\inf(\Omega(s))) \text{ is even}\}$.

- The *Rabin condition* is given by a set of pairs $\{(E_1, F_1), \ldots, (E_k, F_k)\}$ and $\mathcal{F} = \{s \in Q^\omega \mid \inf(s) \cap F_i \neq \varnothing \text{ and } \inf(s) \cap E_i = \varnothing \text{ for some } i\}$.

- The *Streett condition*, dual to the Rabin condition, is again represented by a set of pairs $\{(E_1, F_1), \ldots, (E_k, F_k)\}$, but in this case $\mathcal{F} = \{s \in Q^\omega \mid \inf(s) \cap F_i \neq \varnothing \text{ or } \inf(s) \cap E_i = \varnothing \text{ for every } i\}$.

- The *Muller condition* is defined by listing $F \subseteq \mathcal{P}(Q)$ and $\mathcal{F} = \{s \in Q^\omega \mid \inf(s) \in F\}$.

A *run* of an automaton $\mathcal{A}$ on a word $w \in \Sigma^\omega$ is defined as any sequence of states $q_0 q_1 \ldots \in Q^\omega$ in which $\Delta(q_i, w[i], q_{i+1})$ holds for all i. The word w is *accepted* by $\mathcal{A}$ if there is a run ρ of $\mathcal{A}$ on w that is accepting, i.e. $\rho \in \mathcal{F}$, and we denote by $L(\mathcal{A})$ the set of all words accepted by an automaton $\mathcal{A}$.

It is well-known that non-deterministic Büchi, parity, Rabin, Streett and Muller automata all recognize the same class of languages, the ω-regular languages. The deterministic variants have the same expressive power for all the representations of acceptance conditions introduced above except for the case of Büchi condition, as deterministic Büchi automata are strictly weaker than non-deterministic ones. Moreover, the class of ω-regular languages is closed under union, intersection and complementation.

1.3.1 Alternating automata

In addition to the standard notion of automata, we use *alternating* automata as a tool in our proofs. The intuition behind alternating automata is that, unlike in the deterministic case where only one run on a given word is possible, there are more possibilities of transitions from each state for a given letter. But unlike non-deterministic automata, an alternating automaton does not only accept a word if there *exists* an accepting run among all possible ones, or if *all* possible runs are accepting (as in universal automata), but it allows to alternate such conditions with respect to states of the automaton and has both existential and universal branching choices.

To define alternating automata we have to consider, for a given set of states Q, the set $\mathcal{B}^+(Q)$ of all *positive boolean formulas* over Q. By definition $\mathcal{B}^+(Q)$ is the set of all boolean formulas built using elements of Q, the boolean connectives $\wedge$ and $\vee$ and the constants $\top$ (true) and $\bot$ (false). Note that negation is not allowed. We say that a subset $X \subseteq Q$ *satisfies* a formula $\varphi \in \mathcal{B}^+(Q)$ if φ is satisfied by the assignment that assigns true to all elements of X and false to $Q \setminus X$.

An *alternating automaton* $\mathcal{A}$ over an alphabet Σ is a tuple $(Q, \delta, q_0, \mathcal{F})$, where Q is the set of states, q_0 is the initial state, $\mathcal{F}$ is the acceptance condition, but this time δ does not point to a single next state but specifies a positive boolean formula as transition condition, $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$. Intuitively, a *correct run* of $\mathcal{A}$ on a word w is a tree labeled with Q where the successors of each node form a satisfying set for the boolean condition related to the state in this node and to the corresponding letter in w.

To capture this intuition formally and simplify notation, we define runs as sets of infinite sequences of states, so a run ρ is a subset of Q^ω. When one thinks of runs as trees, our definition corresponds to defining runs directly as the set of branches of the run-tree. For a run ρ represented in this way we write $s_\rho(u)$ for the set of all states appearing in ρ that prolong $u \in Q^*$, i.e. the successors of u when thinking of a run as a tree,

$$s_\rho(u) = \{q \in Q : \exists v\, u \cdot q \cdot v \in \rho\}.$$

We define that ρ is a correct run of $\mathcal{A}$ on the word w if for each infinite branch $u \in \rho$ and each prefix $u|_i$ the successors after that prefix satisfy the corresponding boolean constraint, i.e. if $s_\rho(u|_i)$ satisfies $\delta(u[i], w[i])$ for all i and $u \in \rho$. We say that $\mathcal{A}$ accepts a word w if there is a correct, non-empty run ρ on w starting from q_0 such that each branch $u \in \rho$ is accepted, i.e. $u \in \mathcal{F}$, and again we write $L(\mathcal{A})$ for the language recognized by $\mathcal{A}$.

Alternating automata may seem more powerful than deterministic ones and it is often much easier to express problems in terms of alternating automata than in terms of deterministic ones, but they are in fact equal in expressiveness to the standard automata.

Theorem 1.1. *Every language recognized by an alternating Büchi, parity, Rabin, Streett or Muller automaton is ω-regular.*

The above theorem can be proved by expressing acceptance of alternating automata in monadic second-order logic on infinite words and then going back from the logic to automata [13]. Alternatively, one can give an explicit construction which shows that (for all acceptance conditions except the Büchi condition) the size of the deterministic automaton constructed for a language recognized by an alternating one is at most doubly exponential in the size of the original automaton, as first shown in [57].

1.3.2 Omega-semigroups

There is a fundamental correspondence between recognizability of sets by finite-word automata and by finite semigroups. It has been extended to recognizability of ω-regular sets, first using Wilke algebras [77] and later based on the notion of *ω-semigroups*. The theory of ω-semigroups was first well presented in [62] and is thoroughly discussed in [63], we only mention what is most necessary.

An ω-semigroup $S = (S_f, S_\omega, \cdot, *, \pi)$ is a two-sorted algebra, where $(S_f, \cdot)$ is a semigroup, $* : S_f \times S_\omega \mapsto S_\omega$ is the *mixed product* satisfying for every $s, t \in S_f$ and every $\alpha \in S_\omega$ the equality

$$s * (t * \alpha) = (s \cdot t) * \alpha$$

and where $\pi : S_f^\omega \mapsto S_\omega$ is the *infinite product* satisfying

$$s_0 * \pi(s_1, s_2, \ldots) = \pi(s_0, s_1, s_2, \ldots)$$

as well as the associativity rule

$$\pi(s_0, s_1, s_2, \ldots) = \pi(s_0 s_1 \cdots s_{k_1}, s_{k_1+1} s_{k_1+2} \cdots s_{k_2}, \ldots)$$

for every sequence $(s_i)_{i\geq 0}$ of elements of S_f and every strictly increasing sequence $(k_i)_{i\geq 0}$ of indices. For $s \in S_f$ we denote $s^\omega = \pi(s, s, \ldots)$.

Morphisms of ω-semigroups are defined to preserve all three products as expected. There is a natural way to extend finite semigroups and their morphisms to ω-semigroups. As in semigroup theory, idempotents play a central role in this extension. An *idempotent* is a semigroup element $e \in S$ satisfying $ee = e$. For every element s in a finite semigroup the sub-semigroup generated by s contains a unique idempotent s^k. The least $k > 0$ such that s^k is idempotent for every $s \in S_f$ is called the *exponent* of the semigroup S_f. Another useful notion is absorption of semigroup elements: we say that *s absorbs t (on the right)* if $st = s$.

There is a natural extension of the free semigroup Σ^+ to the free ω-semigroup $(\Sigma^+, \Sigma^\omega)$ with $*$ and π determined by concatenation. An ω-semigroup $S = (S_f, S_\omega)$ *recognizes* a language $L \subseteq \Sigma^\omega$ via a morphism $\phi : (\Sigma^+, \Sigma^\omega) \to (S_f, S_\omega)$ if $\phi^{-1}(\phi(L)) = L$. This notion of recognizability coincides, as for finite words, with recognizability by non-deterministic Büchi automata and translations from Büchi automata to ω-semigroups and back can be done effectively.

Theorem 1.2 ([62]). *A language $L \subseteq \Sigma^\omega$ is ω-regular if and only if it is recognized by a finite ω-semigroup.*

This correspondence allows one to engage in an algebraic study of varieties of ω-regular languages, and also has the advantage of hiding complications of cutting apart and stitching together runs of Büchi automata. This is precisely the reason for which we use this algebraic framework. Most remarkably, one does not need to understand the exact relationship between automata and ω-semigroups and the technical details of

the constructions behind Theorem 1.2 to use ω-semigroups to simplify calculations on ω-regular sets.

1.4 Automatic structures

Before we define automatic presentations and automatic structures, let us introduce relations on finite and ω-words recognized by ω-word automata operating in a synchronized letter-to-letter fashion. Formally, R is an *ω-regular relation* of arity r over the domain Σ^ω if there exists an ω-automaton $\mathcal{A}$ over the alphabet Σ^r accepting the *convolution* $\otimes\overline{w}$ of ω-words $w_1, \ldots, w_r$ exactly when $R(w_1, \ldots, w_r)$ holds. The convolution is defined as

$$\otimes\overline{w}[i] = (w_1[i], \ldots, w_r[i]) \text{ for all } i.$$

For pairs of words w, w' we sometimes write $w \otimes w'$ or $\binom{w}{w'}$ for $\otimes(w, w')$.

Example 1.3. Words $u, v \in \Sigma^\omega$ have *equal ends*, written $u \sim_{\mathrm{e}} v$, if $u[n] = v[n]$ for all but a finitely many natural numbers n. This is an important example of an ω-regular equivalence relation. For $S, T \subseteq \mathbb{N}$ we extend the notation and write $S \sim_{\mathrm{e}} T$ if for all but finitely many $n \in \mathbb{N}$, $n \in S \iff n \in T$. The non-deterministic Büchi automaton depicted in Figure 1.2 accepts the equal-ends relation over $\{0, 1\}$.

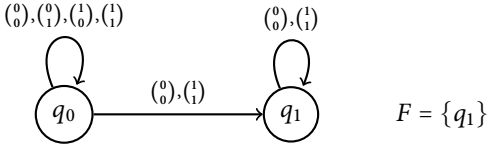


Figure 1.2: An automaton for the equal ends relation.

To define ω-regular relations over finite words one needs to add a padding end-of-word symbol $\square \notin \Sigma$ to formally define the convolution of words of different length. For simplicity, we will sometimes identify

a finite word $w \in \Sigma^*$ with its infinite padding $w^\square = w\square^\omega \in \Sigma_\square^\omega$ where $\Sigma_\square = \Sigma \cup \{\square\}$. A language $L \subseteq \Sigma^*$ is regular, i.e. recognized by a standard finite-word automaton, exactly if the language $L^\square = \{w^\square \mid w \in L\}$ is ω-regular over $\Sigma_\square$. Thus, we say that an r-ary relation $R \subseteq (\Sigma^*)^r$ is *regular* whenever its extension with padding is ω-regular over $\Sigma_\square$. This is equivalent to defining a convolution for finite words by padding each word with $\square$ to be as long as the longest one.

Definition 1.4 (Automatic Presentation)**.**
For any relational structure $\mathfrak{A} = (A, R_1, \ldots, R_k)$, a tuple of ω-automata $\mathfrak{d} = (\mathcal{A}, \mathcal{A}_\approx, \mathcal{A}_1, \ldots, \mathcal{A}_k)$ together with a surjective naming function $f : L(\mathcal{A}) \to A$ constitutes an *(ω-)automatic presentation* of $\mathfrak{A}$ over Σ if the following criteria are met:

(i) the equivalence relation denoted $\approx$ and defined as

$$\{(u, w) \in L(\mathcal{A})^2 \mid f(u) = f(w)\}$$

is recognized by $\mathcal{A}_\approx$,

(ii) every automaton $\mathcal{A}_i$ recognizes a relation $\mathcal{R}_i \subseteq (\Sigma^\omega)^{r_i}$ with the same arity r_i as the relation R_i,

(iii) f is an isomorphism between $\mathfrak{A}_\mathfrak{d} = (L(\mathcal{A}), \mathcal{R}_1, \ldots, \mathcal{R}_k)/_\approx$ and $\mathfrak{A}$.

The presentation is said to be *injective* whenever f is, in which case $\mathcal{A}_\approx$ can be omitted. Observe that the relation $\approx$ needs to be a congruence of the structure $(L(\mathcal{A}), \mathcal{R}_1, \ldots, \mathcal{R}_k)$ for item (iii) to hold.

In case $L(\mathcal{A})$ only consists of words of the form $w^\square$ where $w \in \Sigma^*$, we say that the presentation is (finite-word) automatic. We call a structure *(ω-)automatic* if it has an (ω-)automatic presentation.

There may exist different automatic presentations of a single structure, and different relations might be regular in each of these presentations. For example, for every number $p > 1$ there is a presentation of Presburger arithmetic $(\mathbb{N}, +)$ where numbers are coded in base p. The relation $|_2$ defined as $x|_2 y \iff x|y$ and $x = 2^n$ is a regular relation when numbers

are coded in binary, but it is not regular in the presentation that uses ternary coding. Relations that are regular in each automatic presentation of a structure are called *intrinsically regular* and were first studied in [47, 48]. Every FO-definable relation is intrinsically regular, but on some structures there are intrinsically regular relations that are not definable in FO. Remarkably, this is not the case for Presburger arithmetic, where all relations that are intrinsically regular are definable in FO. An accessible survey of results on presentations of Presburger arithmetic is given in [12], and the general problem of different automatic presentations of a structure is studied in [5, 6].

The basic advantage of having an (ω-)automatic presentation of a structure lies in the fact that first-order formulas can be effectively evaluated using classical automata constructions. This is expressed by the following fundamental theorem.

Theorem 1.5 (Cf. [40, 46, 11]). *There is an effective procedure that given an (ω-)automatic presentation $\mathfrak{d}$, f of a structure $\mathfrak{A}$, and given a* FO*-formula $\varphi(\overline{x})$ constructs an (ω-)automaton recognizing $f^{-1}(\varphi^{\mathfrak{A}})$. The* FO*-theory of every (ω-)automatic structure is decidable.*

1.5 Interpretations and complete structures

In this section, instead of explicitly representing a structure by a finite object, as done above using automata, we consider operations for transforming structures. More precisely, we fix an underlying family of structures and a class of operations that transform structures, and investigate the class of structures obtained by applying the transformations to structures in the underlying family. When the operations preserve decidability of first-order or monadic second-order logic, and structures in the underlying family have decidable first-order or monadic second-order theory, then we obtain a class of structures on which the corresponding logic is again decidable.

An important and well studied way of transforming structures is the model-theoretic interpretation, where one structure is interpreted in

another one using formulas of either first-order or monadic second-order logic. Interpretations preserve decidability of the corresponding logics, or transform structures with decidable monadic second-order theory to structures with decidable first-order theory. We will show a few ways in which automatic structures can be characterized and extended by means of interpretations in trees and linear orders.

Definition 1.6. Let $\mathfrak{A} = (A, R_1, \ldots, R_k)$ and $\mathfrak{B}$ be relational structures and let r_i be the arity of R_i. A tuple of first-order formulas over $\sigma(\mathfrak{B})$,

$$\mathcal{I} = (\delta(\overline{x}), \varepsilon(\overline{x_1}, \overline{x_2}), \varphi_1(\overline{x}_1, \ldots, \overline{x}_{r_1}), \ldots, \varphi_k(\overline{x}_1, \ldots, \overline{x}_{r_k})),$$

where each vector $\overline{x}, \overline{x_i}$ is of the same length n is an *n-dimensional* FO *interpretation of* $\mathfrak{A}$ in $\mathfrak{B}$ if $\mathcal{I}(\mathfrak{B}) = (\delta^{\mathfrak{B}}, \varphi_1^{\mathfrak{B}}, \ldots, \varphi_k^{\mathfrak{B}})/\varepsilon^{\mathfrak{B}}$ is isomorphic to $\mathfrak{A}$.

In an analogous way, a tuple of MSO or WMSO formulas over $\sigma(\mathfrak{B})$,

$$\mathcal{J} = (\delta(X), \varepsilon(X_1, X_2), \varphi_1(X_1, \ldots, X_{r_1}), \ldots, \varphi_k(X_1, \ldots, X_{r_k})),$$

where X and X_i are single second-order variables, is an MSO-to-FO or a WMSO-to-FO interpretation if $\mathcal{J}(\mathfrak{B}) = (\delta^{\mathfrak{B}}, \varphi_1^{\mathfrak{B}}, \ldots, \varphi_k^{\mathfrak{B}})/\varepsilon^{\mathfrak{B}}$ is isomorphic to $\mathfrak{A}$, with the formulas evaluated using the standard or the weak semantics respectively. If there exists an FO, MSO-to-FO or WMSO-to-FO interpretation of $\mathfrak{A}$ in $\mathfrak{B}$ we denote this by $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{B}$, $\mathfrak{A} \leq_{\mathsf{MSO}\to\mathsf{FO}} \mathfrak{B}$ or $\mathfrak{A} \leq_{\mathsf{WMSO}\to\mathsf{FO}} \mathfrak{B}$, respectively.

Let ψ be a first-order formula over $\sigma(\mathfrak{A})$ and $\mathcal{I}$ an interpretation of $\mathfrak{A}$ in $\mathfrak{B}$. We construct the formula $\psi^{\mathcal{I}}$ by replacing every relation symbol R_i in ψ by the corresponding formula φ_i of $\mathcal{I}$, replacing every equality $t_1 = t_2$ in ψ by $\varepsilon(t_1, t_2)$ and relativizing the quantifiers in the following way. In the case of FO interpretations we replace $\exists x \varphi$ by $\exists \overline{x}(\delta(\overline{x}) \wedge \varphi)$ and $\forall x \varphi$ by $\forall \overline{x}(\delta(\overline{x}) \to \varphi)$, and in the second-order case we use second-order variables and thus $\exists X(\delta(X) \wedge \varphi)$ and $\forall X(\delta(X) \to \varphi)$, respectively. The standard interpretation lemma states that $\mathfrak{A} \vDash \psi$ exactly if $\mathfrak{B} \vDash \psi^{\mathcal{I}}$. This allows us to deduce decidability of the FO-theory of $\mathfrak{A}$ from decidability

of the FO, MSO or WMSO theory of $\mathfrak{B}$, given an FO, MSO-to-FO or WMSO-to-FO interpretation of $\mathfrak{A}$ in $\mathfrak{B}$, respectively.

A class $\mathcal{K}$ of structures is *closed under* FO*-interpretations* if for all $\mathfrak{B} \in \mathcal{K}$ whenever $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{B}$ then $\mathfrak{A} \in \mathcal{K}$ as well. For such a class $\mathcal{K}$ we say that a structure $\mathfrak{B}$ is *complete for* $\mathcal{K}$ if all $\mathfrak{A} \in \mathcal{K}$ are FO-interpretable in $\mathfrak{B}$. It follows from Theorem 1.5 that the class of (ω-)automatic structures is closed under FO-interpretations, because every relation defined in FO using only (ω-)regular relations is again (ω-)regular.

One way to characterize automatic structures by means of interpretations is to find complete automatic structures, and such structures were presented in [8, 11]. It was shown there that for any finite alphabet Σ with at least two letters, the complete tree over Σ extended with the equal-length relation, $\mathfrak{T}_{\mathsf{el}}(\Sigma)$, is complete for the class of automatic structures. Analogously, the complete ω-tree with the equal-length relation, $\mathfrak{T}^{\omega}_{\mathsf{el}}(\Sigma)$, is complete for the class of ω-automatic structures.

It is natural to ask whether Presburger arithmetic is a complete automatic structure. The answer is negative, but there are extensions of Presburger arithmetic that are complete. Let us define the following structures: $\mathfrak{N}_p = (\mathbb{N}, +, |_p)$ where $x|_p y \iff x|y$ and $x = p^n$ for some $n \in \mathbb{N}$, and $\mathfrak{R}_p = (\mathbb{R}, +, \leq, |_p, 1)$ where $x|_p y \iff y = kx$ and $x = p^l$ for some $k, l \in \mathbb{Z}$. It was shown in [8, 11] that for all integers $p \geq 2$ the extensions $\mathfrak{N}_p$ and $\mathfrak{R}_p$ of Presburger arithmetic and the real arithmetic are indeed complete, for the class of automatic and ω-automatic structures respectively.

Another way to characterize automatic structures, where WMSO-to-FO and MSO-to-FO interpretations are used, was first mentioned in [69] and more systematically introduced in [18]. This characterization extends the intimate connection between ω-automata over words and MSO over $(\omega, <)$, as well as between finite-word automata and WMSO over $(\omega, <)$, to (ω-)automatic structures. A structure $\mathfrak{A}$ is finite-word automatic if there is a WMSO-to-FO interpretation of $\mathfrak{A}$ in $(\omega, <)$, and a structure $\mathfrak{B}$ is ω-automatic if there is an MSO-to-FO interpretation of $\mathfrak{B}$ in $(\omega, <)$. Let us summarize the characterizations of automatic structures by means

of interpretations in the following theorems.

Theorem 1.7 (Cf. [8, 11, 69, 18]). *For any relational structure* $\mathfrak{A}$ *the following statements are equivalent:*

- $\mathfrak{A}$ *is finite-word automatic,*
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{T}_{\mathsf{el}}(2)$,
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{N}_2$,
- $\mathfrak{A} \leq_{\mathsf{WMSO}\to\mathsf{FO}} (\omega, <)$.

Theorem 1.8 (Cf. [8, 11, 69, 18]). *For any relational structure* $\mathfrak{A}$ *the following statements are equivalent:*

- $\mathfrak{A}$ *is* ω*-automatic,*
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{T}^{\omega}_{\mathsf{el}}(2)$,
- $\mathfrak{A} \leq_{\mathsf{FO}} \mathfrak{R}_2$,
- $\mathfrak{A} \leq_{\mathsf{MSO}\to\mathsf{FO}} (\omega, <)$.

The characterization of automatic structures by MSO-to-FO interpretations was used in [18] to define *generalized-automatic structures*. We say that $\mathfrak{A}$ is an (ω-)generalized-automatic structure if there is a WMSO-to-FO (or MSO-to-FO) interpretation of $\mathfrak{A}$ in some tree $\mathfrak{T}$. In particular, we say that the structure is (ω-)tree-automatic if this is the case for $\mathfrak{T}(2)$, the complete binary tree. By the result of Rabin and the interpretation lemma, (ω-)tree-automatic structures have a decidable first-order theory. Moreover, in chapter 6 we show that certain extensions of first-order logic collapse to FO on all ω-generalized-automatic structures.

1.6 Composition in monadic second-order logic

To study logic on linear orders and trees with arbitrary additional predicates it is convenient to depart from automata and use related methods

from mathematical logic, especially the composition method. The history of the composition method starts with the introduction of Ehrenfeucht games [26], which are an intuitive formulation of Fraïssé's characterization of elementary equivalence, i.e. indistinguishability of relational structures by first-order formulas. These games were first defined for first-order logic and extended to weak monadic second-order logic [26]. Later, other logical systems were covered, such as modal, temporal and infinitary logics that we discuss in chapter 2. Here we focus on the extension of this method, now usually called the Ehrenfeucht-Fraïssé method, to full monadic second-order logic over linear orders and trees.

While Ehrenfeucht proved decidability of the first-order and the weak monadic second-order theory of countable ordinals using logical methods [25, 26], decidability of the full monadic second-order theory of these orderings was first shown by Büchi using automata [13, 14, 15]. Only later Shelah gave, in his celebrated and difficult paper [71], alternative proofs of Büchi's results (and many more) using an extension of the Ehrenfeucht-Fraïssé method to full monadic second-order logic, which he called the composition of monadic theories. This method was subsequently used by Gurevich and Shelah to obtain even more results, for example in [33, 37] and with Magidor in [36]. Theoretical computer scientists long preferred the automata theoretic approach, even after the composition method was well presented in Gurevich's survey [34]. It was only after the more accessible survey by Wolfgang Thomas [73] that the merits of the composition method started to be appreciated in theoretical computer science, which resulted in numerous papers. One example is the characterization of all extensions of $(\omega, <)$ by unary predicates that have a decidable monadic second-order theory [66].

For a structure $\mathfrak{A}$ and a tuple $\overline{U}$ of m subsets of $\mathfrak{A}$ the *monadic n-theory of* $\overline{U}$, $\mathrm{Th}^n(\mathfrak{A}, \overline{U})$, is the set of all MSO formulas $\varphi(\overline{X})$ having no more than n nested quantifiers in any subformula and no free variables other than $X_1, \ldots, X_m$ for which $\mathfrak{A} \vDash \varphi(\overline{U})$.

Given a finite relational signature there are only finitely many n-theories in m variables. Moreover, every n-theory $\mathrm{Th}^n(\mathfrak{A}, \overline{U})$ is definable by a

single MSO formula $\tau(\overline{X})$ having m free variables and quantifier depth at most n. Hintikka formulas are canonical formulas defining n-theories. We denote the finite set of all Hintikka formulas for n-theories in m variables by $\mathrm{Tp}(n,m)$ and say that $\overline{U}$ has *type* $\tau \in \mathrm{Tp}(n,m)$, denoted $\tau = \mathrm{Th}^n(\mathfrak{A},\overline{U})$, if $\mathfrak{A} \models \tau(\overline{U})$, i.e. if τ and $\mathrm{Th}^n(\mathfrak{A},\overline{U})$ are equivalent.

The essence of the composition method is that certain operations on structures, such as disjoint union and ordered sums of linear orders, can be projected to n-theories, i.e. there are corresponding operations mapping n-theories of constituent structures to the n-theory of the resulting structure. In other words, n-theories can be composed.

We mostly use a very simple form of the composition method on linear orders and on the complete binary tree, as stated below, which can be proved directly using Ehrenfeucht-Fraïssé games. As mentioned before, there are more powerful theorems also known as the composition method, e.g. the effective ones presented later and other in [71, 33, 37, 36].

Theorem 1.9 (Composition on linear orders)**.** *Let L be a countable linear order and let $x_i \in \overline{L}$ be a sequence of elements in the completion of L such that $L = \bigcup_{i\in\omega}[x_i, x_{i+1})$. Then for every n and $\overline{X}$ the theory $\mathrm{Th}^n(L,\overline{X})$ is uniquely determined by the theories $\mathrm{Th}^n(L|_{[x_i,x_{i+1})}, \overline{X}|_{[x_i,x_{i+1})})$.*

Theorem 1.10 (Composition on the binary tree)**.** *Let π be an infinite path through $\mathfrak{T}(2)$, $\overline{P}$ a tuple of subsets of π, $\overline{X}_i$ a sequence of tuples of subsets of $\mathfrak{T}(2)$ and let $\overline{X} = \sum_{\pi,\overline{P}} \overline{X}_i$. Then $\mathrm{Th}^n(\mathfrak{T}(2),\overline{X})$ is uniquely determined by π, $\overline{P}$ and the theories $\mathrm{Th}^n(\mathfrak{T}_{\pi[i]},\overline{X}_i)$.*

Please note that when considering substructures, e.g. $\mathfrak{A} \subseteq \mathfrak{B}$, and given sets $\overline{X} \subseteq \mathfrak{B}$, we write $\mathrm{Th}^n(\mathfrak{A},\overline{X})$ to denote $\mathrm{Th}^n(\mathfrak{A},\overline{X}\cap\mathfrak{A})$. For instance for $X \subseteq \mathfrak{T}$ we have $\mathrm{Th}^n(\mathfrak{T}_v, X) = \mathrm{Th}^n(\mathfrak{T}_v, X|_v)$.

As the number of n-types in m variables is finite, it is possible to operate on types as on any finite object. The more constructive version of the composition theorem allows to decompose an MSO formula over a sum of structures into a formula on the index structure labelled by types.

Theorem 1.11. *For every* MSO *formula $\varphi(\overline{X})$ of quantifier rank n and enumeration $\tau_1,\ldots,\tau_k$ of $\mathrm{Tp}(|\overline{X}|,n)$ there is an* MSO *formula $\theta(T_1,\ldots,T_k)$*

such that for every linear order $\mathfrak{I}$, family $\{(L_i, <_i) : i \in \mathfrak{I}\}$ of linear orders, and subsets $\overline{A}$ of $\sum_{i\in\mathfrak{I}} L_i$, it holds that

$$\sum_{i\in\mathfrak{I}} L_i \models \varphi(\overline{A}) \quad \Longleftrightarrow \quad \mathfrak{I} \models \theta(T_1, \ldots, T_k),$$

where for each $m \in \{1, \ldots, k\}$

$$T_m = \{i \in \mathfrak{I} : (L_i, <_i) \models \tau_m(\overline{A}|_{L_i})\}.$$

Moreover, the formula θ is computable from φ.

On the tree, it is possible to encode types of subtrees in a similar way. Given a tuple of m variables $\overline{X}$ on $\mathfrak{T}$ we will denote by $T^n(\overline{X})$ the tuple of $|\mathrm{Tp}(|\overline{X}|, n)|$ sets encoding, for each node v, the n-type of $\overline{X}$ on $\mathfrak{T}_v$, i.e.

$$T^n(\overline{X})[m] = \{v \in \mathfrak{T} : \mathfrak{T}_v \models \tau_m(X|_v)\}.$$

As types are formulas, for any type τ one can construct a formula $\tau(\overline{X}, v)$ checking that $\overline{X}$ has type τ on $\mathfrak{T}_v$ by guarding all first-order quantifiers in τ to talk about suffixes of v. Thus, for all $n \in \mathbb{N}$, there exists an MSO formula $\mathrm{TP}_n(\overline{X}, T^n(\overline{X}))$ that checks whether $T^n(\overline{X})$ indeed codes the n-types of $\overline{X}$. Since there is a unique $T^n(\overline{X})$ for which $\mathrm{TP}_n(\overline{X}, T^n(\overline{X}))$ holds, we use the notation $T^n(X)$ in a functional way in MSO, i.e. instead of writing $\exists Z\ (\mathrm{TP}_n(X, Z) \wedge \varphi(X, Y, Z))$ we write $\varphi(X, Y, T^n(X))$.

Again, using the above coding and the fact that MSO can simulate finite automata on the binary tree, we can extend Theorem 1.10 to the following effective statement that is useful when we wish to express statements about types in MSO (see Figure 1.1 for a picture of X with respect to X_i).

Theorem 1.12. *Let π be an infinite path through $\mathfrak{T}(2)$, $\overline{P}$ a tuple of subsets of π, $\overline{X}_i$ a sequence of tuples of subsets of $\mathfrak{T}(2)$ and let $\overline{X} = \sum_{\pi,\overline{P}} \overline{X}_i$. Moreover, let $\overline{Z}$ be a tuple of subsets of π coding in each node $\pi|_i$ the n-types of $\overline{X}_i$ below $\pi|_i$,*

$$\overline{Z}(\pi|_i) = T^n(\overline{X})(\pi|_i(1 - \pi[i])).$$

Then, for any node $v \in \pi$, the theory $\mathrm{Th}^n(\mathfrak{T}(2)_v, \overline{X})$ is uniquely determined by $\pi, v, \overline{P}$ and $\overline{Z}$, and for each type τ there exists an MSO formula $\psi_\tau(\pi, v, \overline{P}, \overline{Z})$ that holds on $(\omega, <)$ if and only if $\mathrm{Th}^n(\mathfrak{T}(2)_v, \overline{X}) = \tau$.

2 Game Quantifiers on Automatic Presentations

This chapter is devoted to the study of game quantification on automatic presentations. We start with a historic survey on game quantification in the context of infinitary logics, based on [49]. Then, we define a new game quantifier on automatic presentations, the regular game quantifier. We show that the basic advantage of first-order logic on automatic structures, namely decidability and regularity of definable relations, still holds for the logic extended with the game quantifier.

Further, we investigate the expressive power of game quantification on automatic presentations. We show that the prefix and equal-length relations on a presentation are definable using only equality and the regular game quantifier. It follows that much simpler structures are complete for the extended first-order logic than for the standard one. In contrast to the binary ω-tree with equal-length $\mathfrak{T}^{\omega}_{\mathsf{el}}(2)$ needed for pure first-order logic, already the binary ω-tree $\mathfrak{T}^{\omega}(2)$, even without the prefix relation, is complete for first-order logic extended with the game quantifier.

To understand which relations are not definable using the game quantifier, we study the automorphisms of structures that preserve formulas of the extended logic. For this reason we introduce the notion of inductive automorphisms and show that all relations definable in first-order logic extended with the regular game quantifier are preserved under such automorphisms. In addition to the explicit definition, we characterize inductive automorphisms as exactly those automorphisms that preserve the prefix relation on an automatic presentation.

2.1 Open and closed game quantifier

It is natural to ask how first-order logic can be extended without allowing second-order quantification. Two possible extensions are well studied: one where new unary quantifiers are allowed in addition to $\exists$ and $\forall$, which is discussed in more detail in section 5.1, and another where infinitely long formulas can be written, either by allowing infinite conjunctions and disjunctions or by allowing infinite strings of quantifiers.

The extension of first-order logic where conjunctions and disjunctions of size less than κ and *homogeneous* strings of quantifiers of length less than λ are allowed is denoted $\mathsf{L}_{\kappa\lambda}$ for any cardinals κ and λ. For example $\mathsf{FO} = \mathsf{L}_{\omega\omega}$, and the extension where only countable conjunctions and disjunctions are allowed is $\mathsf{L}_{\omega_1\omega}$. Allowing countable boolean operations means that the syntax of $\mathsf{L}_{\omega_1\omega}$ allows to build formulas of the form $\bigwedge_{i\in\mathbb{N}} \varphi_i$ and $\bigvee_{i\in\mathbb{N}} \varphi_i$ in addition to what is allowed by the standard FO syntax. The definition of semantics for $\mathsf{L}_{\omega_1\omega}$ contains two new rules in addition to the ones presented in section 1.1 for FO, namely

- $\mathfrak{A}, \theta \vDash \bigwedge_{i\in\mathbb{N}} \varphi_i$ whenever $\mathfrak{A}, \theta \vDash \varphi_i$ for all $i \in \mathbb{N}$,
- $\mathfrak{A}, \theta \vDash \bigvee_{i\in\mathbb{N}} \varphi_i$ whenever $\mathfrak{A}, \theta \vDash \varphi_i$ for some $i \in \mathbb{N}$.

The definition of semantics for conjunctions and disjunctions of greater length and homogeneous strings of quantifiers is similar. In addition to allowing infinite first-order formulas one can also allow second-order quantification and mix it with infinitary formulas. One interesting possibility is to allow one existential second-order quantifier over a countable set of second-order relations followed by an infinite conjunction of FO formulas. Such formulas, $\exists\{R_i\}_{i\in\mathbb{N}} \bigwedge_{j\in\mathbb{N}} \varphi_j$, where φ_j are FO formulas over the extended alphabet $\tau \cup \{R_i\}_{i\in\mathbb{N}}$, are called PC_Δ formulas and are intimately related to game quantification, as will be explained later.

One interesting extension of FO that is not directly included in $\mathsf{L}_{\kappa\lambda}$ is the case of an infinite string of alternating quantifiers, i.e. formulas of the form $\exists x_0 \forall x_1 \exists x_2 \forall x_3 \ldots R(x_0, x_1, \ldots)$, where $R \subseteq A^\omega$ is a relation on sequences of elements of A. The semantics of such formulas is given using

two-person games known as Gale-Stewart games. The intuition behind the game is that the first player, sometimes called the Verifier, starts by choosing $x_0 \in A$. Then the second player, called the Falsifier, answers with $x_1 \in A$. After this the Verifier chooses x_2, the Falsifier answers and so on. Finally, the winner is determined depending on whether the infinite sequence that was chosen belongs to R or not. Formally, given a strategy for one player, defined as a function $f : \bigcup_{n\in\mathbb{N}} A^{2n} \to A$, and one for the other player defined as $g : \bigcup_{n\in\mathbb{N}} A^{2n+1} \to A$, we construct the sequence $\pi = \pi(f, g) = x_0x_1x_2\ldots$ given by $x_{2n} = f(\pi|_{2n})$ and $x_{2n+1} = g(\pi|_{2n+1})$ for all $n \in \mathbb{N}$. This allows us to define the semantics to a formula $\varphi = \exists x_0 \forall x_1 \ldots R(\overline{x})$ by saying that φ holds in $\mathfrak{A}$ whenever there exists a strategy f for the first player such that for all counter-strategies g of the second player the play $\pi(f, g) \in R$.

In the paragraph above we did not specify the relation R, so we did not give a precise extension of FO. For this purpose, we are going to use infinitary conjunctions and disjunctions, as presented before in the context of $\mathsf{L}_{\omega_1\omega}$. Therefore we define that $\varphi(\overline{z})$ is an *open game formula* if

$$\varphi(\overline{z}) = \exists x_0 \forall y_0 \exists x_1 \forall y_1 \ldots \bigvee_{i\in\mathbb{N}} \varphi_i(\overline{z}, x_0, y_0, \ldots, x_{i-1}, y_{i-1}),$$

and $\psi(\overline{z})$ is a *closed game formula* if

$$\psi(\overline{z}) = \forall x_0 \exists y_0 \forall x_1 \exists y_1 \ldots \bigwedge_{i\in\mathbb{N}} \psi_i(\overline{z}, x_0, y_0, \ldots, x_{i-1}, y_{i-1}),$$

where φ_i, ψ_i are standard FO formulas. A structure $\mathfrak{A}$ with a mapping $\theta : \overline{z} \to \overline{a}$ is a model of $\varphi(\overline{z})$ if there exists a strategy $f : \bigcup_{n\in\mathbb{N}} A^{2n} \to A$ such that for all strategies $g : \bigcup_{n\in\mathbb{N}} A^{2n+1} \to A$ there exists $i \in \mathbb{N}$ for which

$$\varphi_i^{\mathfrak{A}}(\overline{a}, \pi(f, g)[0], \pi(f, g)[1], \ldots, \pi(f, g)[2i - 1])$$

holds. Similarly, $\psi(\overline{z})$ holds if there exists a strategy $g : \bigcup_{n\in\mathbb{N}} A^{2n+1} \to A$ such that for all strategies $f : \bigcup_{n\in\mathbb{N}} A^{2n} \to A$ and for all $i \in \mathbb{N}$ the relation

$$\psi_i^{\mathfrak{A}}(\overline{a}, \pi(f, g)[0], \pi(f, g)[1], \ldots, \pi(f, g)[2i - 1])$$

holds in $\mathfrak{A}$.

Observe that the definition above does not imply that a negation of an open game formula is a closed game formula, as for each of them to be true there must exist a strategy for one player that is winning against all counter-strategies. For the negation of such a formula to be true it suffices that a player can counter all strategies of the opponent with a winning strategy. The property of a two-player game that either one or the other player has a winning strategy is called *determinacy*.

It is an important result of Gale and Stewart [29] that all Gale-Stewart games with open and closed winning conditions, i.e. for relations R denoting open or closed sets in the product topology on A^ω, are determined. Since relations defined by infinite disjunctions of FO formulas are indeed open in the product topology and the ones defined by infinite conjunctions are closed, this result implies that the negation of a closed game formula is indeed an open game formula, and vice versa. This determinacy theorem, later extended by Martin [55] to all Borel winning conditions, gives a foundation for the study of game quantification.

As mentioned before, there is an intimate relationship between closed game formulas and PC_Δ formulas, which was first shown by Svenonius. For any formula $\varphi(\overline{z}) \in \mathsf{PC}_\Delta$ there exists a closed game formula $\psi(\overline{z})$ such that for all structures $\mathfrak{A}$ it holds that $\mathfrak{A} \vDash \varphi(\overline{z}) \to \psi(\overline{z})$ and for all *countable* structures $\mathfrak{A}$ it holds that $\mathfrak{A} \vDash \varphi(\overline{z}) \leftrightarrow \psi(\overline{z})$.

The above result was extended by Vaught to an analogous relationship between formulas of the form $\exists\{R_i\}_{i\in\mathbb{N}}\varphi$ where $\varphi \in \mathsf{L}_{\omega_1\omega}$ is written using a signature extended with $\{R_i\}_{i\in\mathbb{N}}$, and *closed Vaught formulas*. These formulas are extensions of closed game formulas where countable conjunctions are allowed after each $\forall$ quantifier and countable disjunction after each $\exists$ quantifier. Their semantics is again given using a Gale-Stewart game, where the strategies additionally pick a branch of the infinite conjunction or disjunction in each step.

For any closed Vaught formula φ one can consider a finite part of the infinite sequence of alternating quantifiers in the prefix, which is an $\mathsf{L}_{\omega_1\omega}$ formula. We say that these formulas *approximate* φ and Vaught proved that on countable structures the conjunction of all these approximating

formulas is equivalent to φ. This is a strong result and combined with the Svenonius-Vaught theorem mentioned before it allows to approximate existential second-order quantification added to $\mathsf{L}_{\omega_1\omega}$ using only pure $\mathsf{L}_{\omega_1\omega}$ formulas. This has interesting applications to the model theory of $\mathsf{L}_{\omega_1\omega}$, for example allowing to prove compactness and interpolation theorems for this logic. A more thorough introduction to game quantification and its applications in logic is given in [49].

2.2 Game quantification over infinite words

In the closed or open game formulas $\exists x_0 \forall x_1 \ldots R(\overline{x})$ the relation R is either an open or a closed set over A^ω because it is expressed by an infinite conjunction or disjunction of FO formulas. As we are interested in automatic presentations, i.e. structures over $\Sigma^{\leq\omega}$ where all relations are ω-regular, it is natural to consider ω-regular relations R instead of open or closed ones.

To extend the notion of game quantification to an automatic presentation $\mathfrak{A} = (\Sigma^{\leq\omega}, R_1, \ldots, R_k)$, we make explicit use of the fact that elements of the universe are words and so already have an inductive structure to play the game on, and we introduce $\mathsf{FO}[\partial]$, first-order logic extended with the regular game quantifier ∂. We define the meaning of the formula $\partial xy\ \varphi(x,y)$ by saying that $\partial xy\ \varphi(x,y)$ holds if φ can be satisfied by two arguments x and y which are words constructed stepwise by two opposing players. The first letter of x is given by the first player, then the first letter of y is given by the second player, then another letter of x by the first player, and so on. Formally, to capture both finite and infinite words over Σ we again define $\Sigma_\square = \Sigma \cup \{\square\}$ and set

$$\begin{aligned}\partial xy\ \varphi(x,y) \iff (&\exists \text{ well-formed } f : \Sigma_\square^* \times \Sigma_\square^* \to \Sigma_\square) \\ &(\forall \text{ well-formed } g : \Sigma_\square^* \times \Sigma_\square^* \to \Sigma_\square) \\ &\varphi(x_{fg}, y_{fg}),\end{aligned}$$

where x_{fg} and y_{fg} are the Σ-words constructed inductively using f and

g up to the first appearance of $\Box$,

$$x_{fg}[n] = f(x_{fg}|_n, y_{fg}|_n),$$

$$y_{fg}[n] = g(x_{fg}|_{n+1}, y_{fg}|_n),$$

and well-formedness means that if any of the functions f or g outputs $\Box$ then the word x_{fg} resp. y_{fg} is considered to be finite and the function must then continue to output $\Box$ infinitely. Formally, we say that h is well-formed when

$$h(w,u) = \Box \implies (\forall w' \sqsupseteq w)\ (\forall u' \sqsupseteq u)\ h(w',u') = \Box.$$

This definition coincides with the traditional one for an infinite string of alternating quantifiers over letters and a regular relation in scope of all the quantifiers,

$$\eth x y\ \varphi(x,y) \iff (\exists a_0 \forall b_0 \exists a_1 \forall b_1 \ldots)\ \varphi(a_0 a_1 \ldots, b_0 b_1 \ldots).$$

Moreover, using our notation $\eth x y\ \varphi(x)$ is equivalent to $\exists x\ \varphi(x)$ as we can always forget opponent moves and play letters from x or conversely use any g to obtain the witness x. Similarly, $\eth x y\ \varphi(y)$ is equivalent to $\forall y\ \varphi(y)$. Thus, we do not need to consider the standard quantifiers when the regular game quantifier is present.

On some structures it is possible to encode a pair of words into a single one, but that is not always the case. Therefore we might sometimes need to use the game quantifier with more variables:

$$\begin{aligned}\eth x_1 \ldots x_k y_1 \ldots y_m\ \varphi(\overline{x}, \overline{y}) \iff\ & \\ & (\exists f : (\Sigma_\Box^*)^k \times (\Sigma_\Box^*)^m \to \Sigma_\Box^k) \\ & \quad (\forall g : (\Sigma_\Box^*)^k \times (\Sigma_\Box^*)^m \to \Sigma_\Box^m) \\ & \quad\quad \varphi(\overline{x_{fg}}, \overline{y_{fg}}),\end{aligned}$$

where again the functions must be well–formed in each column and

$$\overline{x_{fg}}[n] = f(\overline{x_{fg}}|_n, \overline{y_{fg}}|_n), \qquad \overline{y_{fg}}[n] = g(\overline{x_{fg}}|_{n+1}, \overline{y_{fg}}|_n).$$

Example 2.1. To illustrate the use of game quantifier let us consider the following relation

$$R(u, w, s, t) \text{ defined by } \Game x y\ (y = u \to x = s) \wedge (y = w \to x = t).$$

We claim that R means that the common prefix of s and t is longer than the common prefix of u and w. Denoting by $v \sqcap r$ the common prefix of v and r and by $|v|$ the length of v we can say that

$$R(u, w, s, t) \equiv |u \sqcap w| < |s \sqcap t|,$$

with the additional necessary assumption that $u \neq w$ and $s \neq t$.

The intuitive way to evaluate such a formula is by means of a game played by two players – the Verifier choosing letters of x and the Falsifier choosing letters of y. To see the above equivalence, let us assume that indeed the common prefix of s and t is longer than the common prefix of u and w. In this case, the Falsifier will have to choose whether $y = u$ or $y = w$ before the Verifier chooses if $x = s$ or if $x = t$, and therefore the Verifier is going to win. In the other case, the Falsifier can win and prove the formula false as he knows if the prefix of x can be prolonged to s or to t before choosing whether $y = u$ or $y = w$.

2.3 Decidability and determinacy for $\mathsf{FO}[\Game]$

The two basic properties of $\mathsf{FO}[\Game]$ that interest us are decidability of the model-checking problem for this logic on ω-automatic presentations and the existence of a negation normal form, which semantically corresponds to the determinacy of the underlying games.

To be able to state the existence of a negation normal form, let us introduce another variation of the regular game quantifier, namely one where it is the Falsifier who makes the moves first. Formally, let

$$\begin{aligned}\Game^{\forall} x y\ \varphi(x, y) \iff &(\exists f : \Sigma_{\square}^* \times \Sigma_{\square}^* \to \Sigma_{\square}) \\ &(\forall g : \Sigma_{\square}^* \times \Sigma_{\square}^* \to \Sigma_{\square})\ \ \varphi(x_{fg}^{\forall}, y_{fg}^{\forall}),\end{aligned}$$

where again the functions must be well-formed, but this time the words are constructed in reverse order,

$$y^{\forall}_{fg}[n] = g(x^{\forall}_{fg}|_n, y^{\forall}_{fg}|_n), \qquad x^{\forall}_{fg}[n] = f(x^{\forall}_{fg}|_n, y^{\forall}_{fg}|_{n+1}).$$

If we denote the game quantifier introduced before by $\partial^{\exists}$ then the intended relation that leads to negation normal form can be stated in the following way (note that the variables are reversed after $\partial^{\forall}$ below):

$$\partial^{\exists} xy\ \varphi(x,y) \equiv \neg\partial^{\forall} yx\ \neg\varphi(x,y).$$

When the relation of prefixing a word with a letter, written $y = ax$ for a letter $a \in \Sigma$, is present, the quantifier $\partial^{\forall}$ is superfluous and can be eliminated by adding one letter,

$$\partial^{\forall} xy\ \varphi(x,y) \iff \partial^{\exists} zy\ \exists x\ z = ax \wedge \varphi(x,y).$$

To verify this equivalence, observe that on the right side the Verifier must start with an a and later play a strategy that ensures that φ is satisfied, so the same strategy without the first a can be used on the left side. Conversely, if Verifier's strategy on the left side is given then playing an a and later the same strategy is winning for the right side.

The observation that we use to prove both decidability and the existence of negation normal is that if one starts with ω-regular relations then anything defined in the FO[∂] logic remains ω-regular. The proof relies on the fact that, when applied to an automaton, the game quantifier indeed constructs a game and changes the automaton to an alternating one.

Lemma 2.2. *If the relation $R(\overline{x}, \overline{y}, \overline{z})$ is ω-regular over $\overline{x} \otimes \overline{y} \otimes \overline{z}$ then the relation $S(\overline{z}) \iff \partial\overline{xy}\ R(\overline{x}, \overline{y}, \overline{z})$ is ω-regular over $\otimes\overline{z}$.*

Proof. Let us take the deterministic automaton $\mathcal{A}_R$ for R over $\overline{x} \otimes \overline{y} \otimes \overline{z}$ and construct an alternating automaton $\mathcal{A}_S$ for S over $\otimes\overline{z}$ in the following way. The set of states, acceptance condition and initial state remain the same and the new transition relation is defined by

$$\delta_S(q, \overline{c}) = \bigvee_{\overline{a} \in \Sigma^k_{\square}} \bigwedge_{\overline{b} \in \Sigma^l_{\square}} \delta_R(q, \overline{a} \otimes \overline{b} \otimes \overline{c}),$$

where k is the length of $\overline{x}$ and l is the length of $\overline{y}$.

By definition, the semantics of the relation S is

$$S(\overline{z}) \iff (\exists f : (\Sigma_\square^*)^k \times (\Sigma_\square^*)^l \to \Sigma_\square^k) \\ (\forall g : (\Sigma_\square^*)^k \times (\Sigma_\square^*)^l \to \Sigma_\square^l)\ \varphi(\overline{x_{fg}}, \overline{y_{fg}}, \overline{z}).$$

Assuming for a tuple of words $\overline{w}$ that $S(\overline{w})$ holds, we construct an accepting run ρ of the automaton $\mathcal{A}_S$ on $\overline{w}$. The run ρ is constructed inductively starting from q_0, and in parallel we construct the tuples of words $\overline{x}$ and $\overline{y}$, starting from empty words. Assuming that we are on level n in the run-tree ρ on the branch $q_0 \dots q_{n-1}$ and that the prefixes $\overline{x}|_n$ and $\overline{y}|_n$ were constructed, we let $\overline{x}[n] = f(\overline{x}|_n, \overline{y}|_n)$ and for each $\overline{a} \in \Sigma_\square^l$ we add a branch in ρ from q_{n-1} to $q_n = \delta_R(q_{n-1}, \overline{x}[n] \otimes \overline{a} \otimes \overline{w}[n])$. Finally, we progress to one of these q_n and store $\overline{y}[n] = \overline{a}$. You can see that the function f in this definition of $S(\overline{z})$ corresponds to the choice of a branch to satisfy in the disjunction over the letters for $\overline{x}$ in the boolean formula when selecting the run of the alternating automaton, and that the function g corresponds to the choice of the branch of the run, as all branches must be accepted. The converse direction, constructing a function f from the run ρ is analogous, the run-tree is in fact a representation of such a function. □

From this lemma, together with Theorem 1.1, the decidability of $\mathsf{FO}[\game]$ on automatic presentations follows. The doubly exponential bound on the size of the deterministic automaton constructed from an alternating one gives a bound on the complexity of model-checking $\mathsf{FO}[\game]$ when the quantifier depth of a formula is fixed. It is necessary to bound the quantifier depth to get elementary complexity, as the model-checking problem on automatic structures in general is non-elementary even for FO. As regular relations on ω-words are Borel, we can derive from the previous lemma and the result of Martin [55] that the games corresponding to regular game quantifier are determined, which proves game quantifier inversion as stated below.

Corollary 2.3. FO[∋] *is decidable on ω-automatic presentations, all relations definable in it are ω-automatic and model-checking formulas with a fixed quantifier depth k is in* 2kEXPTIME.

Corollary 2.4. *For each* FO[∋] *formula φ, each automatic presentation* $\mathfrak{A}$ *and each valuation θ*

$$\mathfrak{A}, \theta \vDash \ni^{\exists} \overline{xy}\ \varphi(\overline{x}, \overline{y}, \overline{z}) \iff \mathfrak{A}, \theta \vDash \neg \ni^{\forall} \overline{yx}\ \neg\varphi(\overline{x}, \overline{y}, \overline{z}).$$

2.4 Expressive power of FO[∋]

As mentioned in the introduction, the binary ω-tree with the equal-length relation $\mathfrak{T}^{\omega}_{\mathrm{el}}(2)$ is a complete ω-automatic structure. In particular, every ω-automatic relation over $\{0,1\}^{\omega}$ can be defined by an FO formula over this structure. Since FO[∋] preserves regularity by Lemma 2.2, it follows that FO[∋] is equally expressive as just FO on $\mathfrak{T}^{\omega}_{\mathrm{el}}(2)$.

The situation changes when the equal-length relation is not included, as already $\mathfrak{T}^{\omega}(2)$ is not a complete automatic structure for FO. It turns out that we can even leave out the prefix relation and still define all regular relations in FO[∋] using only the successor predicates.

Theorem 2.5. *On the structure* $(\{0,1\}^{\leq\omega}, S_0, S_1)$ *where* $S_i(v)$ *holds exactly when* $v = ui$*, all regular relations can be defined in* FO[∋].

Proof. First let us recall a few basic formulas that we are going to use. As we have already shown in Example 2.1, we can use the game quantifier to talk about the length of common prefix of words, i.e. for $u \neq w, s \neq t$ we can say $|s \sqcap t| < |u \sqcap w|$ and the other variants with $\leq, =, \geq$ and $>$ can be expressed using boolean combinations and argument permutations of the above.

To say that x is a prefix of y we are going to say that no word $z \neq x$ has a longer common prefix with x than y,

$$x \sqsubseteq y \ \equiv\ (x = y) \ \vee\ \forall z \neq x\ |x \sqcap z| \leq |x \sqcap y|.$$

To define equal length we again use the $|s \sqcap t| < |u \sqcap w|$ relation to define that $|x| \leq |y|$. Note that so far we expressed $|s \sqcap t| < |u \sqcap w|$ only for

$s \neq t$ and $u \neq w$, so we can not just write $|x \sqcap x| \leq |y \sqcap y|$. Instead, we say that for any $x' \neq x$ there is an $y' \neq y$ that has common prefix with y not shorter that the common prefix of x' and x:

$$|x| \leq |y| \;\equiv\; \forall x' \neq x \; \exists y' \neq y \; |x \sqcap x'| \leq |y \sqcap y'|.$$

Now we can use a boolean combination to define $|x| = |y|$, and in this way we obtain both $\sqsubseteq$ and el. As all ω-regular relations over $\mathfrak{T}^{\omega}_{\mathsf{el}}(2)$ can be defined in FO, this completes the proof. □

Note that the above formulas for $\sqsubseteq$ and el did not involve the successor predicates. Definability of these two relations on any automatic presentation in $\mathsf{FO}[\exists^{\infty}]$ just with equality is an important property which will be used subsequently.

One interesting example of an automatic presentation is the binary coding of natural numbers where the least significant digit comes first. We look at the expressive power of $\mathsf{FO}[\exists^{\infty}]$ on such presentations over $\{0,1\}^{\leq\omega}$. To speak meaningfully about numbers, as opposed to words representing them, there must be a relation eq in such presentation that defines the equality between numbers as opposed to equality over words with redundant zeros, $\mathsf{eq}(x,y) \equiv (x = n0^k \text{ and } y = n0^l)$ for some $k, l \in \mathbb{N}$. Using eq and $\sqsubseteq$ it is possible to define S_0 and S_1 over $\{0,1\}^{\leq\omega}$ as words ending with one are exactly those without redundant zeros. Thus in any such presentation with eq it is again possible to define all regular relations in $\mathsf{FO}[\exists^{\infty}]$. This can as well be used to define $+$ and thus adding other strong non-regular relations to the structure, for example multiplication, makes model-checking undecidable.

Corollary 2.6. *On the binary (lower-endian) presentation of* $(\mathbb{N}, =)$ *the relations* $+$ *and* $|_2$ *(and all relations regular in this presentation) are definable in* $\mathsf{FO}[\exists^{\infty}]$. *On the binary presentation of Skolem arithmetic* $(\mathbb{N}, =, \cdot)$ *the logic* $\mathsf{FO}[\exists^{\infty}]$ *is undecidable.*

2.5 Inductive automorphisms

After analyzing what can be expressed in $\mathsf{FO}[\eth]$, we want to look for methods of establishing which relations can *not* be expressed in this logic. For example, one could ask whether a^ω can be expressed in $\mathsf{FO}[\eth]$ without any relations other than equality of words on $\{a, b\}^{\leq\omega}$. We are going to develop a general method to answer such questions by showing that there is a class of automorphisms of a structure that extend to all relations definable in $\mathsf{FO}[\eth]$.

First of all, observe that not all automorphisms of an automatic presentation, when considered just as a first-order structure $(\Sigma^{\leq\omega}, R_1, \ldots, R_k)$, extend to relations definable in $\mathsf{FO}[\eth]$. For example, on a presentation with no relations, the bijection of $\Sigma^{\leq\omega}$ that swaps a^ω with b^ω and leaves other elements untouched is an automorphism. On the other hand, the relation $|s \sqcap t| < |u \sqcap w|$ is definable in $\mathsf{FO}[\eth]$ just with equality, but the bijection described above is not an automorphism of the structure extended with this relation, since

$$|b^\omega \sqcap ab^\omega| < |a^\omega \sqcap ab^\omega| \text{ but } |a^\omega \sqcap ab^\omega| > |b^\omega \sqcap ab^\omega|.$$

The example above is not surprising since we proved that the prefix relation is definable in $\mathsf{FO}[\eth]$ on any presentation, so any automorphism that preserves $\mathsf{FO}[\eth]$-definable relations on $(\Sigma^{\leq\omega}, R_1, \ldots, R_k)$ must be an automorphism of $(\Sigma^{\leq\omega}, R_1, \ldots, R_k, \sqsubseteq)$. We are going to explicitly define the class of *inductive automorphisms* that do extend to relations definable in $\mathsf{FO}[\eth]$ by restricting the bijections of $\Sigma^{\leq\omega}$ to a special form. It turns out that this class is precisely the class of all automorphisms of $(\Sigma^{\leq\omega}, \sqsubseteq)$, so extending the presentation with the prefix relation assures that all $\mathsf{FO}[\eth]$-definable relations are preserved under automorphism.

Definition 2.7. The bijection $\phi : \Sigma^{\leq\omega} \to \Sigma^{\leq\omega}$ is *inductive* whenever it does not change the length of the words, i.e. $|\phi(u)| = |u|$ for every word u, and additionally there exists a family of permutations

$$\{\pi_w\}_{w\in\Sigma^*} \ \pi_w : \Sigma \to \Sigma,$$

such that for each word u with at least n letters the nth letter of $\pi(u)$ is given by the appropriate permutation,

$$\phi(u)[n] = \pi_{u|_{n-1}}(u[n]).$$

Observe that the inverse bijection ϕ^{-1} of any inductive bijection ϕ is again inductive as inverse permutations $\{\pi_w^{-1}\}$ can be used.

If we restrict our attention to an automorphism ϕ that is an inductive bijection then the structure can be extended with any $\mathsf{FO}[\exists]$ definable relation and ϕ will still be an automorphism of the extended structure, as formulated below.

Theorem 2.8. *Let ϕ be an inductive automorphism of a structure $\mathfrak{A} = (\Sigma^{\leq\omega}, R_1, \ldots, R_k)$ and $R(\overline{x})$ a relation defined by an $\mathsf{FO}[\exists]$ formula $\varphi(\overline{x})$. Then ϕ is an automorphism of the extended structure $(\Sigma^{\leq\omega}, R_1, \ldots, R_k, R)$.*

Proof. We proceed by induction on the structure of formulas and it is enough to consider the inductive step for the game quantifier. Let $\varphi(\overline{x}, \overline{y}, \overline{z})$ be a formula such that

$$\varphi^{\mathfrak{A}}(\overline{a}, \overline{b}, \overline{c}) \iff \varphi^{\mathfrak{A}}(\phi(\overline{a}), \phi(\overline{b}), \phi(\overline{c})).$$

We show that for $\psi(\overline{z}) = \exists\overline{x}\,\forall\overline{y}\,\varphi(\overline{x}, \overline{y}, \overline{z})$ it holds $\psi^{\mathfrak{A}}(\overline{c}) \iff \psi^{\mathfrak{A}}(\phi(\overline{c}))$.

To prove it let us define for any strategies f of the Verifier and g of the Falsifier used in $\exists\overline{x}\,\forall\overline{y}\,\varphi(\overline{x}, \overline{y}, \overline{z})$ the transposed strategies f_ϕ, g_ϕ in the following way:

$$f_\phi(\overline{u}, \overline{w}) = \pi_{\phi^{-1}(\overline{u})} f(\phi^{-1}(\overline{u}), \phi^{-1}(\overline{w})),$$

$$g_\phi(\overline{u}, \overline{w}) = \pi_{\phi^{-1}(\overline{w})} g(\phi^{-1}(\overline{u}), \phi^{-1}(\overline{w})),$$

where π_w is the permutation for word w associated with ϕ and $\pi_{\overline{w}}$ applied to a tuple $\overline{v}$ of the same length means applying π_{w_i} to each element v_i. You should observe that when the players play with strategies f_ϕ, g_ϕ then the resulting words are exactly images of the words that result from using f and g under ϕ,

$$\overline{x_{f_\phi g_\phi}} = \phi(\overline{x_{fg}}), \qquad \overline{y_{f_\phi g_\phi}} = \phi(\overline{y_{fg}}).$$

In this way we can use the winning strategy f for the first player in $\psi(\overline{z})$ and play with f_ϕ in $\psi(\phi(\overline{z}))$. If the opponent chooses to play g then in the end the formula $\varphi(\overline{x_{f_\phi g}}, \overline{y_{f_\phi g}}, \phi(\overline{z}))$ will be evaluated, but

$$\begin{aligned}\varphi(\overline{x_{f_\phi g}}, \overline{y_{f_\phi g}}, \phi(\overline{z})) &\equiv \varphi(\phi(\overline{x_{f g_{\phi^{-1}}}}), \phi(\overline{y_{f g_{\phi^{-1}}}}), \phi(\overline{z})) \\ &\equiv \varphi(\overline{x_{f g_{\phi^{-1}}}}, \overline{y_{f g_{\phi^{-1}}}}, \overline{z}),\end{aligned}$$

which holds as f is winning against any strategy, in particular against $g_{\phi^{-1}}$. □

While the explicit definition of inductive automorphisms given above was useful for the proof, we can characterize these automorphisms in another way, namely as automorphisms of $(\Sigma^{\leq\omega}, \sqsubseteq)$. On the one hand, any inductive automorphism preserves $\sqsubseteq$ because this is an $\mathsf{FO}[\partial]$-definable relation. On the other hand, it can be shown by induction on the prefix order that any automorphism of $(\Sigma^{\leq\omega}, \sqsubseteq)$ is inductive. First, the empty word is preserved by any automorphism that preserves the prefix relation as it is the minimal element of $\sqsubseteq$. Secondly, if a word w is mapped to w' by an automorphism preserving $\sqsubseteq$, then all $\sqsubseteq$-successors of w must be mapped to $\sqsubseteq$-successors of w', which defines the permutation π_w. Thus, all automorphisms of an automatic presentation $(\Sigma^{\leq\omega}, \sqsubseteq, R_1, \ldots, R_k)$ preserve $\mathsf{FO}[\partial]$-definable relations.

The standard way to show that a relation is not definable in a logic using automorphisms is to find an automorphism the relation is not invariant under. Theorem 2.8 makes it possible to use this standard method for $\mathsf{FO}[\partial]$ on automatic presentations, as shown in the following example, where we answer the question asked at the beginning of this section.

Example 2.9. Let us consider the automorphism ϕ of $\{a, b\}^{\leq\omega}$ that just swaps the first letter of all words, i.e. $\phi(au) = bu$, $\phi(bv) = av$, $\phi(\varepsilon) = \varepsilon$. The mapping ϕ is an inductive bijection; the appropriate permutations π_w are identities for all $w \neq \varepsilon$, and π_ε is given by $\pi_\varepsilon(a) = b$ and $\pi_\varepsilon(b) = a$. This automorphism maps a^ω to ba^ω and thus the set $\{a^\omega\}$ is not preserved under ϕ. By Theorem 2.8 we conclude that a^ω is not definable in $\mathsf{FO}[\partial]$ just with equality.

3 Games for Model Checking on Automatic Structures

In the previous chapter we used games as a tool to define the semantics of game quantification and to investigate questions in logic. In this chapter we focus on games in their own right.

We start by defining games played on graphs by two players with perfect information. The connection between such games and logic is illustrated on two well-known examples: the game-theoretical semantics of first-order logic where games of finite duration are used, and model-checking of modal μ-calculus where parity games are appropriate. These two examples show that studying the relation to games can both lead to better insight into the expressive power of a logic and also have an algorithmic utility for model checking. This motivates us to look for games for model-checking on automatic structures.

To find an appropriate game model for first-order logic on an automatic structure, we fix a presentation of the structure and investigate the extended logic $\mathsf{FO}[\exists]$. For this setting, we introduce multiplayer games played by two coalitions with opposing objectives and with imperfect information exchanged according to a hierarchical constraint [45]. On the one hand, this constraint is suitable for defining model-checking games for the extended first-order logic, and it is necessary for the problem of establishing the winning coalition to be decidable. On the other hand, this constraint alone is not sufficient for establishing the winners to be decidable.

To identify the properties needed to make hierarchical games decidable, we study a restricted version of these games where players are forced to alternate. We show that this constraint is required both for determinacy of hierarchical games and for decidability of the problem of establishing the

winning coalition. Finally, we prove that hierarchical games where players alternate are indeed model-checking games for $\mathsf{FO}[\circlearrowright]$ on automatic presentations.

3.1 Games on graphs and logic

In the previous chapter we discussed game quantification and used Gale-Stewart games to provide semantics for game formulas. Since we were only interested in the existence of winning strategies, we did not give a formal definition of what a game is in that context. In this section we want to take a step back and define games, more precisely games played on graphs. We also give an overview of the well-known connection between two-player zero-sum games with complete information and fixed-point logics.

The intuition behind a two-player zero-sum turn-based game played on a graph is very natural. Two players, let us call them Player 0 and Player 1, play by moving a token around a graph of positions. There is a position singled out in which the game starts and every position is assigned to one of the players. When the token is in a position that belongs to one of the players, this player is required to move by choosing an edge going out from this position. If there are no outgoing edges, the player who can not move loses. If the players manage to keep playing infinitely long, then the winner is decided based on a winning condition that specifies which infinite plays are winning for Player 0 and which for Player 1.

Definition 3.1. A Büchi, parity, Streett, Rabin or Muller game is given by a tuple $\mathcal{G} = (V_0, V_1, E, \mathcal{F})$ where V_0 is the set of positions of Player 0 and V_1, disjoint from V_0, contains the positions of Player 1. $E \subseteq V \times V$ is the edge relation denoting possible moves between positions $V = V_0 \cup V_1$, and $\mathcal{F} \subseteq V^\omega$ is a winning condition, represented in the same way as Büchi, parity, Streett, Rabin and Muller acceptance conditions for automata described in section 1.3.

To avoid tedious case distinctions, we often assume that all plays are infinite, i.e. that $vE \neq \varnothing$ for all $v \in V$.

You can see that the Gale-Stewart game for a structure $\mathfrak{A}$ can be viewed as a graph game, either as a game on the tree $\mathfrak{T}(A)$ with players alternating their moves or as a game on the complete bipartite graph $A \times A$ with one side belonging to Player 0 and the other to Player 1.

A strategy for player $i \in \{0,1\}$ in the game $\mathcal{G}$ is a function $\sigma : V^* V_i \to V$ with $(v, \sigma(hv)) \in E$ for all $h \in V^*$ and $v \in V_i$. A play $\pi = v_0 v_1 \ldots$ is *consistent with a strategy* σ for player i if $v_{n+1} = \sigma(v_0 \ldots v_n)$ for every n such that $v_n \in V_i$. Given strategies σ, ρ for Player 0 and Player 1, respectively, we denote by $\pi_{\sigma,\rho}(v_0)$ the unique play starting in position v_0 which is consistent with both σ and ρ.

We say that a strategy σ is winning for Player 0 from v_0 if for all strategies ρ of the opponent $\pi_{\sigma,\rho}(v_0) \in \mathcal{F}$. Analogously, a strategy ρ is winning for Player 1 from v_0 if for all strategies σ of the opponent $\pi_{\sigma,\rho}(v_0) \notin \mathcal{F}$. The set of all positions from which player i has a winning strategy is called the winning region of player i. A game $\mathcal{G}$ is determined if from every position either Player 0 or Player 1 has a winning strategy. Thus, in a determined game, the game graph can be partitioned into winning regions of Player 0 and Player 1.

In many cases one is interested not only in arbitrary winning strategies, but in strategies of a special kind. One prominent example are *positional strategies*, where the strategy depends only on the current position and not on the previous positions of the play, i.e. $\sigma(hv) = \sigma(v)$ for any history h. In a stronger version of determinacy one requires the winning strategies to belong to a certain class. For example, games with parity winning conditions are determined in positional strategies [28, 59], i.e. from every position either Player 0 or Player 1 has a positional strategy that is winning against all strategies of the opponent. For games with Muller winning conditions on finitely many priorities a larger class of strategies is needed, namely such where a finite number of memory states is allowed. We investigate various kinds of determinacy and memory for strategies in chapter 4.

There is an intimate connection between zero-sum games and logic. The idea to give semantics to logics using games was mentioned already

in the last decade of the 19th century by C.S. Pierce, and about sixty years later Paul Lorenzen gave a game-theoretical semantics for first-order logic. Giving a game-theoretical semantics to a logic means that for the evaluation of a formula φ on a structure $\mathfrak{A}$ one constructs a model-checking game $\mathsf{MC}(\mathfrak{A}, \varphi)$ such that Player 0 has a winning strategy in $\mathsf{MC}(\mathfrak{A}, \varphi)$ from an initial position exactly if $\mathfrak{A} \models \varphi$.

The model-checking game for an FO formula φ on $\mathfrak{A}$ is constructed in a very intuitive way. The positions of the game consist of subformulas of φ together with a valuation of all free variables in the subformula. If the position is of the form $(\varphi_1 \vee \varphi_2, \theta)$ then Player 0 moves either to (φ_1, θ) or to (φ_2, θ). Analogously, from $(\varphi_1 \wedge \varphi_2, \theta)$ Player 1 moves to one of the subformulas. In a position of the form $(\exists x \varphi, \theta)$, Player 0 moves by choosing an element $a \in \mathfrak{A}$. The next position is then $(\varphi, \theta[x \leftarrow a])$. For $(\forall x \varphi, \theta)$, the other player can make analogous moves. When the game reaches a position (φ, θ) for an atomic formula φ, the winner is determined depending on whether or not $\mathfrak{A}, \theta \models \varphi$.

On finite structures first-order logic is often too weak to express properties of interest. Before we proceed to show model-checking games for first-order logic on infinite structures, let us recall how a more expressive logic, the modal fixed-point logic, can be model-checked on finite structures using parity games.

In computer science, real-world systems are often modeled using finite Kripke structures, which are directed graphs labeled by a set of predicates. Formally, a Kripke structure is a tuple $\mathcal{K} = (V, E, P_1, \ldots, P_k)$ with $E \subseteq V \times V$ and $P_i \subseteq V$. Important properties that often need to be checked on such systems include *reachability*, i.e. the question whether a node where a predicate P_i holds can be reached from an initial node, and *safety*, i.e. the question whether nodes where a predicate P_j holds can be avoided on all possible paths from an initial node. These properties are not definable in FO, but there are well-known temporal logics, like the linear time logic LTL and the branching-time logic CTL, which can express these properties. There is an elegant modal logic that subsumes all these temporal logics and can express many interesting properties,

the modal μ-calculus L_μ. Formulas φ of L_μ are formed according to the following syntax,

$$\varphi \;=\; P_i \mid \neg P_i \mid X \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\, \varphi \mid \Diamond\, \varphi \mid \mu X \varphi \mid \nu X \varphi,$$

and evaluated on a Kripke structure $\mathcal{K}$ using the following semantics.

- $\mathcal{K}, v \vDash P_i$ whenever $P_i(v)$ holds and $\mathcal{K}, v \vDash \neg P_i$ in the other case,
- $\mathcal{K}, v \vDash \varphi \wedge \psi$ ($\varphi \vee \psi$) whenever $\mathcal{K}, v \vDash \varphi$ and (or) $\mathcal{K}, v \vDash \psi$,
- $\mathcal{K}, v \vDash \Diamond\varphi$ whenever there is a $w \in vE$ for which $\mathcal{K}, w \vDash \varphi$ holds,
- $\mathcal{K}, v \vDash \Box\varphi$ whenever $\mathcal{K}, w \vDash \varphi$ for all $w \in vE$,
- $\mathcal{K}, v \vDash \mu X \varphi$ whenever $(\mathcal{K}, X), v \vDash \varphi$, where X is the *smallest* subset of V for which the equation $X = \{w \,:\, (\mathcal{K}, X), w \vDash \varphi\}$ holds,
- $\mathcal{K}, v \vDash \nu X \varphi$ whenever $(\mathcal{K}, X), v \vDash \varphi$, where X is the *biggest* subset of V for which the equation $X = \{w \,:\, (\mathcal{K}, X), w \vDash \varphi\}$ holds.

Note that in the syntax we use X to denote a set variable, while in the definition of semantics we write $(\mathcal{K}, X)$ for the Kripke structure $\mathcal{K}$ extended with the predicate X. The semantics above is well defined only if the smallest and biggest solutions to the fixed-point equation exist, but this is indeed the case due to the monotonicity of all the operators of L_μ.

The modal μ-calculus is a very expressive logic, in fact it can express all MSO-definable properties that are invariant under bisimulation [41], and most properties of practical interest belong to this class. To define a model-checking game $\mathsf{MC}(\mathcal{K}, \varphi)$ for an L_μ formula φ on a Kripke structure $\mathcal{K}$ one proceeds in an analogous way to first-order logic. Player 0 chooses a successor for $\Diamond$ and $\vee$, while Player 1 moves for $\Box$ and $\wedge$. Additionally, to handle fixed-point operators, from any set variable X a new edge is added back to the formula $\mu X \varphi$ or $\nu X \varphi$ where the variable X was introduced. These back-edges make infinite plays possible and it turns out that the winner of such an infinite play is decided depending on whether the outermost fixed-point variable occurring infinitely often in the play is

introduced in a μ or in a ν formula. This corresponds exactly to the parity condition and indeed, not only are parity games powerful enough for model-checking L_μ, the converse holds as well, i.e. winning in any parity game (with a fixed number of priorities) can be expressed in the μ-calculus.

The correspondence between L_μ and parity games is not only an interesting extension of the analogous relation between first-order logic and games of finite duration, it also has interesting algorithmic consequences. While it is still open whether there exists a polynomial-time algorithm for model-checking L_μ, all of the most efficient algorithms known so far [42, 43, 76, 44] rely on the representation of the problem as a game. In particular, one very efficient algorithm [76] heavily exploits the structure of the game. This algorithm does not compute the fixed-points in an iterative way, as suggested by the structure of the L_μ formula. Instead, it starts by guessing a positional strategy in the parity game and then it improves this strategy, which often takes fewer steps than the iterative fixed-point evaluation. The fact that the structure of the game can be of algorithmic use is an additional motivation to look for model-checking games for $\mathsf{FO}[\exists]$ on automatic structures.

3.2 Games with hierarchical imperfect information

Our goal in this section is to describe a class of games that will later be used for model-checking first-order logic with the game quantifier on presentations of automatic structures. To define such games we go beyond two-player perfect information games and use multiplayer games with imperfect information. Even though there are multiple players, in the games we define they form two coalitions with strictly opposing objectives. For this reason one could use a different metaphor with just two players for the same class of games. We use the multiplayer setting in this chapter and discuss the other possibilities in the final chapter.

While imperfect information is a standard element of classical game theory, especially for games in extensive form, in computer science games

with imperfect information played on graphs have first been studied in the context of alternating Turing machines with private states [67, 68]. At that time only the reachability condition was considered. Algorithmic solutions for imperfect information games with ω-regular winning conditions were presented only recently [17], however only for the case of observable winning conditions.

The standard way to represent imperfect information in games is by means of *information sets*, equivalence relations describing which states can not be distinguished by a given player. We find it more convenient to use a different representation, in which players see some of the actions of their opponents and other actions are hidden. It is possible to transform between these two representations, but the transformation may increase the size of the game.

Definition 3.2. A hierarchical Büchi, parity, Rabin, Streett or Muller game with actions in a finite set Σ is given by a tuple

$$(V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}}, \mu, \mathcal{F}).$$

The game is played by two coalitions, I and II, each consisting of N players, with the set of players denoted

$$\Pi = (1,\mathrm{I}), (2,\mathrm{I}), \ldots, (N,\mathrm{I}), (1,\mathrm{II}), (2,\mathrm{II}), \ldots, (N,\mathrm{II})$$

and the arena of the game given by the pairwise disjoint sets of positions of each player, $V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}}$. Positions of coalition I are denoted by $V_{\mathrm{I}} = V_{1,\mathrm{I}} \cup \ldots \cup V_{N,\mathrm{I}}$ and the ones of coalition II by $V_{\mathrm{II}} = V_{1,\mathrm{II}} \cup \ldots \cup V_{N,\mathrm{II}}$, with all positions denoted $V = V_{\mathrm{I}} \cup V_{\mathrm{II}}$. The function $\mu : V \times \Sigma \to V$ defines the possible moves, so that when a player chooses an action $a \in \Sigma$ in his position v then the token is moved and the play proceeds to position $\mu(v, a)$. The objective of coalition I is given by the winning condition $\mathcal{F} \subseteq V^\omega$, represented in a finite way as a parity, Streett, Rabin or Muller condition, depending on the type of the game.

When a hierarchical game is played infinitely long, an infinite sequence of actions is taken by the players during the play, which we call the *play*

actions sequence and denote by $\alpha \in \Sigma^\omega$. Conversely, with every play actions sequence α and a starting position v_0, we associate the unique play $\pi_\alpha(v_0)$. It is the infinite sequence of positions that results from making the moves according to α,

$$\pi_\alpha(v_0) = v_0 v_1 \ldots \iff v_{i+1} = \mu(v_i, \alpha[i]) \text{ for all } i \in \mathbb{N}.$$

During the play $\pi_\alpha(v_0)$ we encounter a sequence of players that take the moves in each step, defined by $\Pi_\alpha(v_0)[i] = p \Leftrightarrow \pi_\alpha(v_0)[i] \in V_p$.

In a hierarchical game each player p has to decide on a strategy $\sigma_p : \Sigma^* \to \Sigma$. In a game with perfect information one says that play actions α are consistent with a strategy σ_p in a play starting in v_0 if for each move i taken by player p the action taken is given by the strategy acting on the history of actions, $\alpha[i] = \sigma_p(\alpha|_i)$.

Since the players do not have perfect information, we additionally assume that for each player p there is a *view function* v_p that extracts the information visible for this player from the history of play actions. More precisely, let $v_p : (\Sigma \times \Pi)^* \to \Sigma^*$ be the function that extracts the information visible to player p from the history of play actions labeled by players who took these actions. We say that a sequence of play actions α is consistent with a strategy σ_p of player p in a play starting in v_0 if, for each i for which $\pi_\alpha[i] \in V_p$, it holds that

$$\alpha[i+1] = \sigma_p(v_p((\alpha[0], \Pi_\alpha[0]) \ldots (\alpha[i], \Pi_\alpha[i]))).$$

The above definition of views of play history is very general, but we will only use a concrete special case of *hierarchical* view functions. These hierarchical views allow player k in each coalition to see the moves of players $1, \ldots, k$ in both coalitions, but do not allow him to see the moves of players with numbers $j > k$. Formally, for a player $p = (k, c)$, i.e. player number k in coalition c,

$$v_p((a_0, p_0)(a_1, p_1) \ldots (a_n, p_n)) = a_{i_1} a_{i_2} \ldots a_{i_l}$$

if for all $i \in \{i_1, \ldots, i_l\}$ the player $p_i = (l, d)$ has number $l \leq k$, and for all other $j \notin \{i_1, \ldots, i_l\}$ the player $p_j = (m, e)$ has number $m > k$.

There is a good reason to use hierarchical view functions, namely that for most other kinds of information flow, determining the winner, even in a reachability game with three players, is undecidable [4, 2].

To define when coalition I wins a hierarchical game we can not require from all players in this coalition to put forth their winning strategies before players in coalition II do, as it is often done in games with perfect information. Intuitively, in that case players with higher numbers would lose their advantage of information as their strategies would be disclosed too early. Therefore, we use the following definition that requires that strategies are given stepwise, level by level in the information hierarchy.

Definition 3.3. Coalition I wins the hierarchical game

$$(V_{1,\mathrm{I}}, \ldots, V_{N,\mathrm{I}}, V_{1,\mathrm{II}}, \ldots, V_{N,\mathrm{II}}, \mu, \mathcal{F})$$

starting from position v_0 if the following condition holds. There exists a strategy $\sigma_{1,\mathrm{I}}$ for player 1, I, such that for each strategy $\sigma_{1,\mathrm{II}}$ of player 1, II, there exists a strategy $\sigma_{2,\mathrm{I}}$, such that for each strategy $\sigma_{2,\mathrm{II}}$, ..., there exists a strategy $\sigma_{N,\mathrm{I}}$, such that for each strategy $\sigma_{N,\mathrm{II}}$, the play actions sequence α that starts from v_0 and is consistent with all strategies $\sigma_{1,\mathrm{I}}, \sigma_{1,\mathrm{II}}, \ldots, \sigma_{N,\mathrm{I}}, \sigma_{N,\mathrm{II}}$ results in a play winning for I, i.e. $\pi_\alpha(v_0) \in \mathcal{F}$.

The definition for coalition II is analogous, i.e. there exists a $\sigma_{1,\mathrm{II}}$, such that for all $\sigma_{1,\mathrm{I}}$, ..., the play is winning for II, i.e. $\pi_\alpha(v_0) \notin \mathcal{F}$.

Example 3.4. To get an intuition about the kind of interactions that appear in hierarchical games, let us consider the simple game depicted in Figure 3.1 in two variants. The positions of coalition I are round, the positions of coalition II are square, there are two levels of information, and the positions on the upper level are dotted.

You can think of this game as played using a coin with two sides, A and B. Each of the players can choose to either flip the coin (F) or leave it as it is (L). Formally, there are four players in this game, two in each coalition. The top position belongs to 2, II and the two bottom positions belong to 1, II. The game proceeds as follows: first the second player of coalition II chooses either to flip the coin or to leave it intact. Afterward,

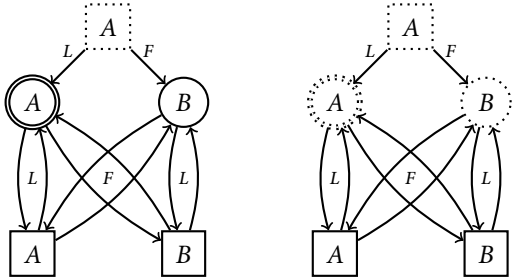


Figure 3.1: Example of a hierarchical game in two variants.

only the other two players play by either flipping the coin or leaving it as it is. Coalition I wins if the A side of the coin is seen infinitely often in positions where players in coalition I move, as marked in Figure 3.1.

To illustrate the importance of hierarchical information levels we consider two variants of this game. In the first one (left), the bottom strongly connected component belongs to players on the same information level, i.e. to 1, II and 1, I. In this scenario, coalition II can win, because first player 2, II can flip the coin to B and later player 1, II can always repeat the last move of player 1, I.

In the other variant (right), the player in coalition I has more information, i.e. the bottom strongly connected component belongs to 1, II and 2, I, with $V_{1,\mathrm{I}} = \varnothing$. In this case coalition I can win, because the strategy of player 2, I is given after the strategy of 1, II is set. Therefore, player 2, I can assure that the coin will be flipped after each two moves, which guarantees that I holds the coin on the A side infinitely often, independent of the first move of 2, II.

3.3 Alternation of moves in hierarchical games

In games with perfect information it is not necessary to assume that the players move in any fixed order. Moreover, the assumption that players

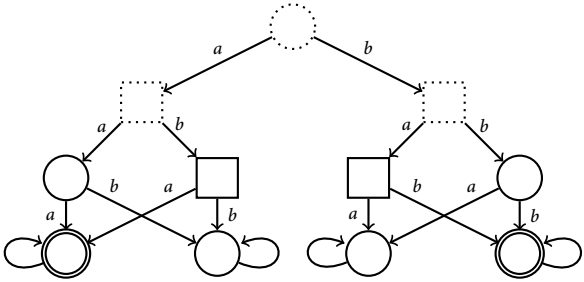


Figure 3.2: Non-determined hierarchical game.

move in an alternating way can be made without loss of generality. We show that this is not the case for hierarchical games. Thus, we define an *alternating hierarchical game* as a hierarchical game, where for each letter $a \in \Sigma$ and each level $i = 1, \ldots, N$ the following alternation conditions hold:

$$v_i \in V_{i,\mathrm{I}} \implies \mu(v_i, a) \in V_{i,\mathrm{II}},$$
$$v_i \in V_{i,\mathrm{II}} \implies \mu(v_i, a) \in V_{(i \bmod N)+1,\mathrm{I}}.$$

To see that non-alternating hierarchical games can not be reduced to alternating ones, let us consider the game depicted in Figure 3.2. The leftmost and the rightmost bottom position is winning for coalition I, while in the other two bottom positions coalition I loses. This simple hierarchical game is not alternating and we show that it is not determined. To win this game, the player on the lower level of information, i.e. 1, I or 1, II, has to predict the move of the opponent, i.e. 1, II or 1, I. In particular, his strategy has to start with an a exactly if the opponent starts with an a. As this holds for players in both coalitions, it leads to a non-determined game as each player can counter the strategy of the opponent, once it is known.

Introducing alternation of moves, even in the simplest possible way, changes this situation. The game depicted in Figure 3.3 is identical to the

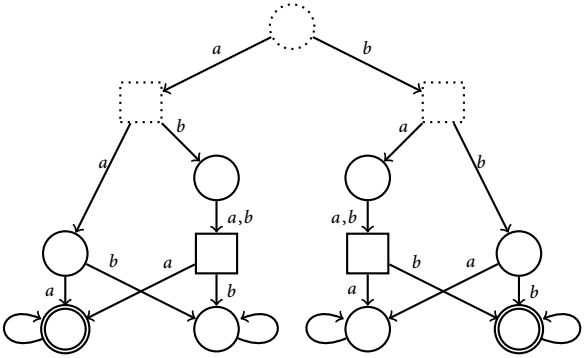


Figure 3.3: Alternation makes hierarchical games determined.

one in Figure 3.2 except for two additional positions of player 1, I. These positions may seem useless as there is no choice to be made there, but the new game is determined. To convince yourself that, in the extended game, coalition II can indeed win, take the following strategy of player 1, II: let him always play the opposite move to the one that was taken before by player 1, I. For player 2, II take the following strategy: if player 1, I declared that he will play a first, then play b, and else play a first. You can check that these strategies are indeed winning for coalition II, but this is possible only because when constructing the strategy for 1, II the first letter played by 1, I was already known.

Another important difference between alternating and non-alternating hierarchical games is decidability of the problem of establishing whether coalition I wins the game. We show in the next section that this problem is decidable for alternating hierarchical games, and here we prove that in the general non-alternating case it is undecidable. The differences between alternating and non-alternating hierarchical games can be explained on the level of logic and model-checking, as alternating hierarchical games correspond to model-checking on automatic presentations, while non-alternating games correspond to model-checking on presentations that

use asynchronous automata, known as rational structures, which have undecidable first-order theory. It is also interesting to observe that the proof of undecidability uses the fact that all players in hierarchical games as we defined them choose actions from the same alphabet Σ. If we assume that in a hierarchical game every player chooses actions from his own alphabet, which does not overlap with the alphabet of any other player, then establishing which coalition wins is decidable even for non-alternating games, cf. [64].

Theorem 3.5. *The question whether coalition* I *wins in a hierarchical Büchi game is undecidable.*

Proof. We reduce the Post correspondence problem for $\overline{u} = u_1, \ldots, u_K$ and $\overline{v} = v_1, \ldots, v_K$, where $u_i, v_i \in \{a, b\}^*$, to the problem whether coalition I wins in the hierarchical game $\mathcal{G}_{\overline{u},\overline{v}}$. The possible actions in $\mathcal{G}_{\overline{u},\overline{v}}$ are $\Sigma = \{a, b, \square, 1, 2, \ldots, K\}$ and they intuitively correspond to the players choosing letters of the words u_i, v_i, a special delimiter $\square$, and choosing which word to play next.

In constructing $\mathcal{G}_{\overline{u},\overline{v}}$, we are going to use subgames such that, for a given word u, the subgame enforces that u is played, or else the player that moves loses. Such a subgame has one more position than the length of u, and if the wrong letter is chosen then the move leads to a position where the player loses. There is only one outgoing edge in such a subgame, the one taken when the last letter of u is played. In Figure 3.4 we depicted an example subgame for $u = aba$ and player 1, I, who loses in the rightmost position.

We start the construction of the game $\mathcal{G}_{\overline{u},\overline{v}}$ with a position belonging to player 3, II with two possible (non-losing) moves. In this position, coalition II can decide if the test will be done for the words $\overline{u}$ or for the words $\overline{v}$. All other positions will be on lower levels of information and we construct them in such a way that coalition I will never be able to deduce in which component the play is taking place.

Each of the two components, for $\overline{u}$ and for $\overline{v}$, starts with a position of player 2, I where this player chooses if he wants to play a word with index

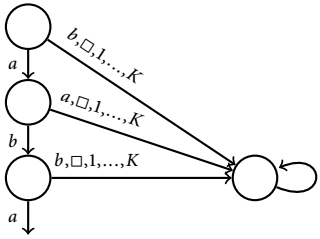


Figure 3.4: Example subgame for $u = aba$.

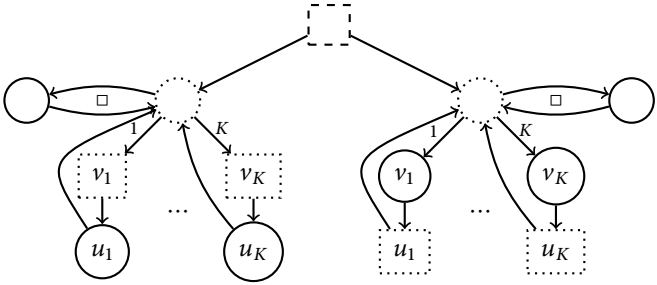


Figure 3.5: Complete game $\mathcal{G}_{\overline{u},\overline{v}}$.

$1, \ldots, K$ or the special symbol □. If the special symbol is chosen, player 1, I must play the same symbol □ and the play returns back to the position, where 2, I chooses a word. When an index L is chosen, then in each of the components first the word v_L and then the word u_L is played. The difference is that, in the first component (for $\overline{u}$), it is player 2, II who must play v_L and player 1, I must play u_L, while in the other component (for $\overline{v}$), it is player 1, I who must play v_L and player 2, II who must play u_L. After the two words were played, the play returns to the position where 2, I chooses the index of a word to be played. The complete game is depicted in Figure 3.5, using subgames for u_i and v_i.

The winning condition is defined as follows: the special symbol □ must

be chosen by 2, I infinitely often and additionally there must be another action L, different from $\Box$, that is played infinitely often. While this is not directly a Büchi condition, the game can be transformed into a game with Büchi winning condition. In the modified game, one more position for player 2, I is added in each component, with the same moves as in the original one except for the possibility of choosing $\Box$. In the transformed game, when 1, I chooses $\Box$ in the only position where he is allowed to do so, the play proceeds from the new position of 2, I where $\Box$ is not allowed, thus ensuring that a non-$\Box$ action is taken.

Let us first show that if there is a solution for the Post correspondence problem for $\overline{u}$ and $\overline{v}$ then coalition I has winning strategies for $\mathcal{G}_{\overline{u},\overline{v}}$. Indeed, let $i_1, i_2, \ldots, i_M$ be the indices for the solution of the correspondence problem, so that $u_{i_1}u_{i_2}\ldots u_{i_M} = v_{i_1}v_{i_2}\ldots v_{i_M}$. Let player 2, I choose i_1 in his first move, then i_2, i_3, and so on up to i_M, then the special symbol $\Box$, and then again i_1, i_2, and so on. Player 1, I is going to play the letters from the word $u_{i_1}u_{i_2}\ldots u_{i_M}$ in turn, and then $\Box$, and then again the letters $u_{i_1}u_{i_2}\ldots u_{i_M}$, and $\Box$, and so on. Clearly, player 2, I chooses $\Box$ and non-$\Box$ infinitely often, so to show that coalition I wins we only need to prove that player 1, I will never play the wrong letter in a subgame for some word w. If the play is taking place in the $\overline{u}$ component this is clear from the definition of the strategies given above, as player 1, I plays exactly the words indices of which player 2, I chooses. When the play takes place in the $\overline{v}$ component, the indices chosen by player 2, I force player 1, I to play the words $v_{i_1}, v_{i_2}, \ldots, v_{i_M}$. But since $u_{i_1}u_{i_2}\ldots u_{i_M} = v_{i_1}v_{i_2}\ldots v_{i_M}$, this is equivalent to playing the u_i words with the same indices, which is exactly the strategy that player 1, I uses.

To prove the converse, namely that if there is a winning strategy for coalition I then the correspondence problem has a solution, observe two intuitive facts. First, 2, I can never deduce in which component the play is taking place, because what he can see after each of his moves is the same in both components. Secondly, $\Box$ can be played by 2, I only if the words played up to that point have the same length in both components. Otherwise, coalition I would lose as $\Box$ can not be played in a subgame for

any word.

Formally, let us first fix the only rational strategy for 2, II, namely that if a number L was the most recent action in the play, then 2, II plays v_L, and if there were other actions from $\{a, b\}^*$ taken after the last time a number L was played, then he plays u_L. Note that the above construction implies that player 2, II knows in which component the play takes place, even if the move of 3, II is not visible for him. With this strategy fixed, the condition that coalition I has a winning strategy for $\mathcal{G}_{\overline{u},\overline{v}}$ means that there exists a strategy σ_1 for player 1, I and a strategy σ_2 for player 2, I such that the play corresponding to these two strategies and the one fixed for 2, II is winning for coalition I, independent of the component chosen by 3, II.

Let us first concentrate on the strategy σ_2. Since, according to the winning condition, $\Box$ can not be the only action played infinitely often, and in each component the only possible answer to $\Box$ is again $\Box$, let us assume without loss of generality that the first move taken by σ_2 is not $\Box$ and let it be L_1. After choosing L_1 the play goes through v_{L_1} and u_{L_1} and does not stop, since player 2, II uses a fixed strategy that prevents him from losing in a subgame and player 1, I plays a winning strategy. Let us denote by L_2 the next move of 2, I, i.e. $L_1 = \sigma_2(\varepsilon)$, $L_2 = \sigma_2(L_1 v_{L_1} u_{L_1})$, and continue the play denoting the subsequent moves of 2, I by $L_2, \ldots, L_M$, up to the point where he plays $\Box$. Formally,

$$L_1 = \sigma_2(\varepsilon), \qquad L_{i+1} = \sigma_2(L_1 v_{L_1} u_{L_1} \ldots L_i v_{L_i} u_{L_i}), \qquad L_{M+1} = \Box.$$

After extracting the sequence $L_1, \ldots, L_M$ of moves of 2, I from his winning strategy σ_2, let us look at player 1, I. This is the only player on information level 1 so he only sees his own previous moves. In this case, the strategy σ_1 is in fact completely described by the word $t \in \{a, b, \Box\}^\omega$ such that

$$t[i] = \sigma_1(t|_i) \text{ for all } i \in \mathbb{N}.$$

Due to the structure of the game, no $\Box$ can be played by 1, I before 2, I decides to play $\Box$, and then $\Box$ must be played. Therefore, if w is the prefix of t up to the first occurence of $\Box$, then w is exactly the word played by

1, I while 2, I played the moves $L_1, \ldots, L_M$. But due to the structure of the game $\mathcal{G}_{\overline{u},\overline{v}}$, coalition II can decide if $w = u_{L_1} \ldots u_{L_M}$ or if $w = v_{L_1} \ldots v_{L_M}$. Since we extracted both $L_1, \ldots, L_M$ and w independent of this choice, w has to be good for both cases. Therefore it is the solution for the Post correspondence problem as requested. □

3.4 Model checking with hierarchical games

We observed that non-alternating hierarchical games are neither determined nor decidable, so we concentrate on the alternating version. Indeed, we prove that alternating hierarchical games are exactly the games needed for model-checking $\mathsf{FO}[\circlearrowright]$ on presentations of automatic structures.

To start with, observe that in an alternating game every infinite sequence of play actions can be divided into blocks of $2N$ actions, each taken by a different player,

$$\alpha = a_0^{1,\mathrm{I}} a_0^{1,\mathrm{II}} a_0^{2,\mathrm{I}} a_0^{2,\mathrm{II}} \ldots a_0^{N,\mathrm{I}} a_0^{N,\mathrm{II}} a_1^{1,\mathrm{I}} \ldots a_1^{N,\mathrm{II}} a_2^{1,\mathrm{I}} \ldots .$$

Let the $2N$-*split* of these play actions be the tuple of $2N$ words of actions played by each of the players,

$$\mathrm{split}_{2N}(\alpha) = (a_0^{1,\mathrm{I}} a_1^{1,\mathrm{I}} \ldots, \{a_i^{1,\mathrm{II}}\}_{i\in\mathbb{N}}, \ldots, \{a_i^{N,\mathrm{I}}\}_{i\in\mathbb{N}}, \{a_i^{N,\mathrm{II}}\}_{i\in\mathbb{N}}).$$

Observe that since the set of plays winning for coalition I and starting from a fixed v_0 is ω-regular, also the set of corresponding $2N$-splits of play actions is ω-regular. This is a known property of ω-regular languages, and it can be proved by taking each $2N$th state of the automaton recognizing the plays and making a product with Σ^{2N} to store the states that were omitted from the original automaton. For an alternating hierarchical game $\mathcal{G}$ with winning condition $\mathcal{F}$ let us denote the $2N$ary relation recognizing the $2N$-split of plays winning for coalition I by $W_{\mathrm{I}}^{\mathcal{G},v_0}(\beta_1, \ldots, \beta_{2N})$, formally defined by

$$W_{\mathrm{I}}^{\mathcal{G},v_0}(\overline{\beta}) \iff \forall \alpha \, (\, \mathrm{split}_{2N}(\alpha) = \overline{\beta} \Rightarrow \pi_\alpha(v_0) \in \mathcal{F} \,).$$

The definition for coalition II is analogous with $\pi_\alpha(v_0)) \notin \mathcal{F}$.

Using the relation W_{I}^{G,v_0} we can express in $\mathsf{FO}[\partial]$ that coalition I wins in the alternating hierarchical game $\mathcal{G}$, which results in the following theorem.

Theorem 3.6. *For any alternating hierarchical game $\mathcal{G}$ and the relation $W_{\mathrm{I}}^{\mathcal{G},v_0}$ defined as above, coalition* I *wins the game $\mathcal{G}$ starting from v_0 if and only if the following formula φ_{I} holds in $(\Sigma^\omega, W_{\mathrm{I}}^{\mathcal{G},v_0})$:*

$$\varphi_{\mathrm{I}} = \partial x_1 y_1 \ldots \partial x_N y_N \; W_{\mathrm{I}}^{\mathcal{G},v_0}(x_1, y_1, \ldots, x_N, y_N).$$

Proof. Let us recapitulate the definition of coalition I winning a hierarchical game and the semantics of the formula φ_{I}. Coalition I wins $\mathcal{G}$ if there is a strategy σ_1 for player on level 1 of coalition I, so that for any counter-strategy ρ_1, there exists a strategy σ_2, and so on up to σ_N, such that for all ρ_N the resulting play must be won by coalition I. On the other hand, the formula φ_{I}, according to the definition of ∂, says that there is a function f_1, so that for all functions g_1, there is a function f_2, and so on up to a function f_N, such that for all g_N, if we construct the words according to $\overline{f}$ and $\overline{g}$ then they form a $2N$-split of a play that is won by coalition I.

As the structure and the final condition in both definitions are equivalent, due to the definition of $W_{\mathrm{I}}^{\mathcal{G},v_0}$, the only remaining task is to show how the functions f_i, g_i and the strategies σ_i, ρ_i are related. It is intuitively clear that the functions and the strategies are closely related, the only difference is that the functions f_i, g_i operate on prefixes of x_i, y_i while the strategies σ_i, ρ_i take all actions of all players $j \leq i$ as arguments, which corresponds to prefixes of all words x_j, y_j with $j \leq i$. Intuitively, this makes no difference since the words x_j, y_j are completely fixed before the function f_i is constructed, and we are going to prove it formally.

Let us construct, given the function f_i, a strategy $\sigma_i^{f_i}$. The strategy $\sigma_i^{f_i}$ applied to a view h of the history of play actions extracts from h the sequences h_{I}^i and h_{II}^i of actions of players i, I and i, II, respectively, and chooses $f_i(h_{\mathrm{I}}^i, h_{\mathrm{II}}^i)$ as the next action. It is possible to extract h_{I}^i and h_{II}^i from h due to the alternation condition, because we know that h is of the form $a_0^{1,\mathrm{I}} a_0^{1,\mathrm{II}} a_0^{2,\mathrm{I}} a_0^{2,\mathrm{II}} \ldots a_0^{i,\mathrm{I}} a_0^{i,\mathrm{II}} a_1^{1,\mathrm{I}} \ldots$ and the sequences $h_{\mathrm{I}}^i = a_0^{i,\mathrm{I}} a_1^{i,\mathrm{I}} \ldots$ and h_{II}^i can be computed by taking every $2i$th position in h starting from

$2i-1$ and $2i$, respectively. Note that extracting these sequences would not be possible if it was not clear which player made which move, which we used in the previous proof of undecidability.

Let us now do the converse and construct, given the strategy σ_i, the function $f_i^{\sigma_i}$. For this construction we need to have all the f_j, g_j with $j < i$ already constructed, thus we write $f_i^{\{\sigma_j,\rho_j\}_{j\leq i}}$. Using the constructed functions f_j, g_j, we can assume that the words x_j, y_j are already fixed. The result of

$$f_i^{\{\sigma_j,\rho_j\}_{j\leq i}}(x_i[0]\dots x_i[n], y_i[0],\dots y_i[n])$$

is given by

$$\sigma_i(x_1[0]y_1[0]\dots x_i[0]y_i[0]x_1[1]y_1[1]\dots x_i[1]y_i[1]\dots x_i[n]y_i[n]).$$

The constructions relating g_i and ρ_i are analogous. Observe that if

$$W_{\mathrm{I}}^{\mathcal{G},v_0}(x_{f_1g_1}y_{f_1g_1},\dots,x_{f_Ng_N}y_{f_Ng_N})$$

holds for some functions $\overline{f},\overline{g}$ then, by the above definition, the play $\pi(v_0,\sigma_1^{f_1},\rho_1^{g_1},\dots,\sigma_N^{f_N},\rho_N^{g_N})$ is in $\mathcal{F}$. Moreover, the converse holds as well, i.e. if for some strategies $\overline{\sigma},\overline{\rho}$ we have

$$\pi(v_0,\sigma_1,\rho_1,\dots,\sigma_N,\rho_N)\in\mathcal{F},$$

then $W_{\mathrm{I}}^{\mathcal{G},v_0}(x_{f_1g_1}y_{f_1g_1},\dots,x_{f_Ng_N}y_{f_Ng_N})$ holds, where $f_i = f_i^{\{\sigma_j,\rho_j\}_{j\leq i}}$ and $g_i = g_i^{\{\sigma_j,\rho_j\}_{j\leq i}}$ are the functions constructed above.

This correspondence allows to exploit the similarity of the structure of the definition of the $\mathsf{FO}[\mathrm{Ↄ}]$ formula φ_{I} and the definition of coalition I winning in $\mathcal{G}$. Intuitively, it is enough to insert the transformed functions and strategies into the definition to arrive at a contradiction and finish this proof. To avoid cluttered notation, we formally present only one direction in the case of two levels, the other direction and the proof for more levels is analogous.

Let us assume that φ_{I} holds and coalition I does not win $\mathcal{G}$, formally

(1) $\exists f_1\,\forall g_1\exists f_2\,\forall g_2\; W_{\mathrm{I}}^{\mathcal{G},v_0}(x_{f_1g_1},y_{f_1g_1},x_{f_2g_2},y_{f_2g_2})$,

(2) $\forall\sigma_1\exists\rho_1\forall\sigma_2\exists\rho_2\;\pi(\sigma_1,\rho_1,\sigma_2,\rho_2)\notin\mathcal{F}$.

Let us fix f_1 that exists by our first assumption, set $\sigma_1 = \sigma_1^{f_1}$ and fix ρ_1 that exists by the second assumption for this σ_1. Let us now set $g_1 = g_1^{\rho_1}$ and fix f_2 that exists by the first assumption. Finally, let us set $\sigma_2 = \sigma_2^{f_2}$ and fix ρ_2 that exists by the second assumption. By the previous observation

$$W_{\mathrm{I}}^{\mathcal{G},v_0}(x_{f_1g_1}, y_{f_1g_1}, x_{f_2g_2}, y_{f_2g_2}) \iff \pi(\sigma_1, \rho_1, \sigma_2, \rho_2) \in \mathcal{F},$$

but this contradicts the two assumptions above. □

Observe that the same proof works for the other coalition and an analogous relation $W_{\mathrm{II}}^{\mathcal{G},v_0}$. Thus, the negation normal form of $\mathsf{FO}[\partial]$ corresponds to determinacy of alternating hierarchical games.

Corollary 3.7. *Alternating hierarchical games are determined.*

After we captured winning in alternating games in $\mathsf{FO}[\partial]$ let us do the converse and construct the model-checking game for a given $\mathsf{FO}[\partial]$ formula on an automatic presentation $\mathfrak{A}$. At first, we restrict ourselves to formulas of the form

$$\varphi = \partial x_1 y_1 \partial x_2 y_2 \dots \partial x_N y_N \; R(x_1, y_1, \dots, x_N, y_N)$$

and construct a game so that the *split* of the winning plays will allow us to use the previous theorem.

Intuitively, the construction can be understood as prefixing each variable with all possible letters in the order of information hierarchy and making a step of the automaton when all the variables are prefixed. To define these games precisely, let us take the deterministic automaton for R, denoted $\mathcal{A}_R = (Q, q_0, \delta, \mathcal{F}_R)$, and construct the model-checking game $\mathcal{G}_\varphi$ for φ in the following way.

For each tuple of letters $c_1, d_1, c_2, d_2, \dots, c_M, d_M$ of even length, with $0 \le M < N$, and for every state $q \in Q$, we have in $\mathcal{G}_\varphi$ the position

$$R^q(c_1x_1, d_1y_1, \dots, c_Mx_M, d_My_M, x_{M+1}, \dots, y_N). \tag{3.1}$$

Moreover, for each tuple $c_1, d_1, c_2, d_2, \dots, c_M, d_M, c_{M+1}$ of odd length, we have the position

$$R^q(c_1x_1, \dots, d_My_M, c_{M+1}x_{M+1}, y_{M+1}, \dots, y_N). \tag{3.2}$$

In each of these positions, the next move is made by the player corresponding to the next variable that is not yet prefixed by a letter, e.g. in position 3.1 it is the player $M+1$ of coalition I who makes the move for x_{M+1} and in position 3.2 it is the player $M+1$ of coalition II. We can formally define the set of positions of players on each level i as $V_{i,\mathrm{I}} = Q \times \Sigma^{2(i-1)}$, $V_{i,\mathrm{II}} = Q \times \Sigma^{2i-1}$.

The moves in $\mathcal{G}_\varphi$ intuitively correspond to the player choosing a letter to prefix his variable with, so for $0 \leq M < N$

$$\mu(R^q(c_1x_1, \ldots, d_My_M, x_{M+1}, \ldots, y_N), c_{M+1}) =$$

$$R^q(c_1x_1, \ldots, d_My_M, c_{M+1}x_{M+1}, y_{M+1}, \ldots, y_N),$$

and for $0 \leq M < N-1$

$$\mu(R^q(c_1x_1, \ldots, c_{M+1}x_{M+1}, y_{M+1}, \ldots, y_N), d_{M+1}) =$$

$$R^q(c_1x_1, \ldots, c_{M+1}x_{M+1}, d_{M+1}y_{M+1}, x_{M+2}, \ldots, y_N).$$

The only special case is the final position $R^q(c_1x_1, d_1y_1, \ldots, c_Nx_N, y_N)$. When player N, II chooses the final letter d_N, it will not be appended, but instead all prefixing letters will be removed and the state of the automaton will be changed as follows, with $\overline{\alpha} = c_1d_1 \ldots c_Nd_N$:

$$\mu(R^q(c_1x_1, d_1y_1, \ldots, c_Nx_N, y_N), d_N) = R^{\delta(q,\overline{\alpha})}(x_1, \ldots, y_N).$$

We derive the winning condition $\mathcal{F}$ of the game $\mathcal{G}_\varphi$ from the acceptance condition $\mathcal{F}_R$ of the automaton for R in the following way. Only the state component of each position in the game is taken into account, i.e. a sequence π of positions of $\mathcal{G}_\varphi$ is in $\mathcal{F}$ if and only if π projected to the state component is in $\mathcal{F}_R$.

To see that the game $\mathcal{G}_\varphi$ is indeed the model-checking game for φ, we use Theorem 3.6 and observe that the $2N$-split of the winning paths in $\mathcal{G}_\varphi$ is exactly the relation R, $W_{\mathrm{I}}^{\mathcal{G}_\varphi, R^{q_0}(x_1, y_1, \ldots, x_N, y_N)} = R$.

In this way, the model-checking game for formulas in the considered form is constructed. As we proved, any formula in $\mathsf{FO}[\exists]$ can be written

in negation normal form and additionally, by renaming variables, it can be transformed into prenex normal form. Let us therefore consider a general formula in the form $\varphi = \eth x_1 y_1 \ldots \eth x_N y_N \; \psi(x_1, y_1, \ldots, x_N, y_N)$, where ψ is in negation normal form and does not contain quantifiers. We construct the game $\mathcal{G}_\varphi$ inductively with respect to ψ.

In the case of $\psi(\overline{x}) = R(\overline{x})$ or $\psi(\overline{x}) = \neg R(\overline{x})$ the solution was already presented, when considering $\neg R$ we just have to complement the acceptance condition of the automaton for R. Let us show how to construct the game for boolean connectives, i.e. for $\psi_1 \wedge \psi_2$ and for $\psi_1 \vee \psi_2$. We want to adhere to the usual convention of model-checking games and to have only one additional position for any boolean connective. The game for $\psi_1 \circ \psi_2$, where $\circ = \wedge, \vee$, is therefore constructed as follows: we take the two games for ψ_1 and ψ_2 and we add one more position on higher level of information that has two possible moves — to the starting position of ψ_1 and to the starting position of ψ_2. The new position belongs to coalition I when $\circ = \vee$ and to coalition II when $\circ = \wedge$ and in both cases the other coalition does not play on that information level. With the construction described above we face a problem, as the game is not strictly alternating any more, but this time it can be made alternating by adding dummy positions, as presented in Example 3.8.

To formally prove that the resulting games are indeed model-checking games for formulas with boolean connectives, we replace the connectives with a new variable and the formula with a relation where only the first letter of the new variable corresponding to the boolean connective is considered. Then the automaton for such a relation corresponds to the defined game and Theorem 3.6 can be used again.

Example 3.8. To illustrate the construction of model-checking games and the method to overcome the technical problem with non-alternating games mentioned above, let us consider the simple formula $\exists x \; (R_1(x) \wedge R_2(x))$ over $\{a, b\}^\omega$ with $R_1 = \{a^\omega\}$ and $R_2 = \{a, b\}^\omega \backslash \{a^\omega\}$. Both the automaton for R_1 and the one for R_2 has two states and the transition functions are identical. On any b the automata go from q_0 to q_1 and stay there forever. Only the Büchi acceptance conditions of these automata

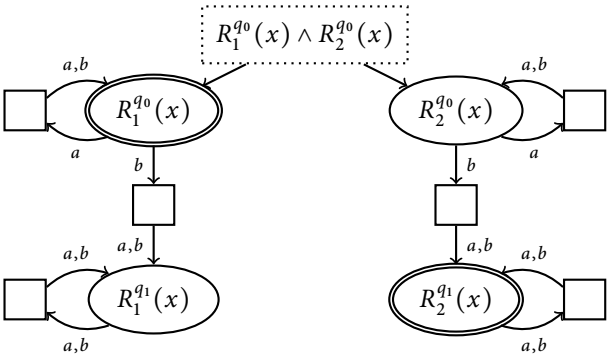


Figure 3.6: Model-checking game for $\exists x(R_1(x) \wedge R_2(x))$.

differ, with $F_1 = \{q_0\}$ and $F_2 = \{q_1\}$.

In Figure 3.6, the game for this formula is depicted. We show dummy moves for the second player, as formally $\exists x\varphi(x) \equiv \Im xy\varphi(x)$. Note that this is actually a four-player game and the top position belongs to player 2, II. Since the formula is false, coalition II wins this game. Indeed, for coalition I to win, player 1, I would have to present a strategy to visit both of the double-circled vertices infinitely often without knowing in which branch he is, and that is impossible.

To fix the problem with alternation, let us add positions where there is no choice for the player. The alternating game for $\psi_1 \circ \psi_2$ is depicted in Figure 3.7. In this game, dummy positions are added there, where it is necessary to make the game alternating. It is clear that winning strategies in these two games can be transferred, as in each move on each level of visibility the players know how many moves on the other levels were made, both in the original game depicted in Figure 3.6 and in the modified one in Figure 3.7.

The tight correspondence between alternating hierarchical games and $\mathsf{FO}[\Im]$ makes it possible to use our knowledge about this logic to reason about the games. In particular, we can transfer the results about complex-

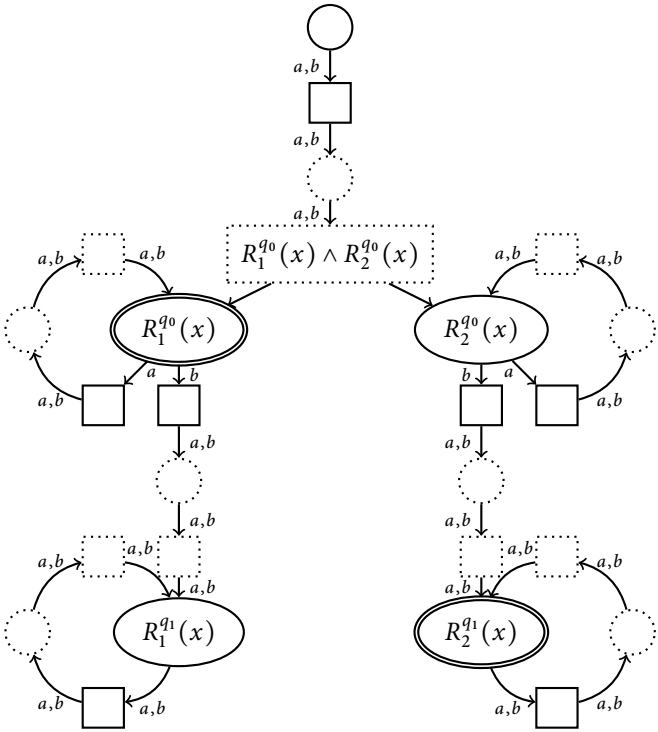


Figure 3.7: Alternating game for $\exists x(R_1(x) \wedge R_2(x))$.

ity, including the non-elementary lower bound on deciding $\mathsf{FO}[\mathfrak{D}]$ on automatic presentations, which allows us to conclude with the following corollary.

Corollary 3.9. *The question whether coalition* I *wins in an alternating hierarchical game on a finite arena is decidable and has non-elementary complexity when the number of levels is not fixed. It can be decided in* $2k\mathsf{EXPTIME}$ *for games with at most k levels.*

4 Memory Structures for Infinitary Games

In the previous chapters, we explored the connections between logic and games in a generic way, without relating to a specific representation of winning conditions in games. In this chapter, we investigate explicitly given winning conditions in terms of the complexity of strategies that are needed to win games with a fixed condition.

This question has been answered to a large extent for winning conditions defined over finite sets of priorities. We look at games with winning conditions defined over infinite sets of priorities and construct memory structures for different types of conditions in this case. Inspired by the notion of latest appearance record [35] used for games with finitely many priorities, we define the finite appearance record [31] and investigate which types of winning conditions are determined with such memory. The class of these conditions includes:

- downward cones,
- singleton conditions,
- finite unions of upwards cones,
- Muller conditions with finitely many winning sets,
- max-parity condition on graphs with bounded moves.

It remains open whether arbitrary max-parity games are determined via finite appearance records and a complete classification of Muller conditions over an infinite set of priorities with this property is not obtained. Still, the reduction for Muller conditions containing finitely many (possibly infinite) sets is a strong generalization of the classical case over finitely many priorities.

In addition to finite appearance records, we investigate winning conditions for which a certain representation, called the Zielonka tree [78], exists. These include all conditions over a finite set of priorities, for which the connection between the Zielonka tree and the memory needed for strategies is well understood [78]. We show that under certain assumptions this classical result can be transferred to infinite number of priorities as well.

4.1 Memory structures and determinacy

The most general representation of ω-regular acceptance conditions (for automata) and winning conditions (for games) that we considered so far was the Muller condition, which we defined as a class of subsets of states of the automaton or positions of the game. In this chapter, we study these conditions more thoroughly, and for this reason we give a slightly more general definition and extend the notation.

To start with, we assume that every game we consider has an arena labeled by priorities from a set C. Formally, a game is now not only the tuple $(V_0, V_1, E, \mathcal{F})$ but it consists of the *game graph* $G = (V, V_0, V_1, E)$ (with $V = V_0 \cup V_1$) which, together with a labeling function $\Omega : V \to C$, forms the *game arena* (G, Ω). A game is defined as a game arena together with a winning condition, $\mathcal{G} = (G, \Omega, \mathcal{F})$, and we focus on a Muller winning condition $\mathcal{F}$ over C defined as follows.

Definition 4.1. A Muller condition over a finite set C of priorities is written in the form $(\mathcal{F}_0, \mathcal{F}_1)$ where $\mathcal{F}_0 \subseteq \mathcal{P}(C)$ and $\mathcal{F}_1 = \mathcal{P}(C) - \mathcal{F}_0$. A play π in a game with Muller winning condition $(\mathcal{F}_0, \mathcal{F}_1)$ is won by Player σ if, and only if, $\inf(\pi)$, the set of priorities occurring infinitely often in π, belongs to $\mathcal{F}_\sigma$. A *Streett-Rabin condition* is now defined as a Muller condition $(\mathcal{F}_0, \mathcal{F}_1)$ such that $\mathcal{F}_0$ is closed under union.

Definition 4.2. A *memory structure* for a game $\mathcal{G}$ with positions in V is a triple $\mathfrak{M} = (M, \text{update}, \text{init})$, where M is a set of *memory states*, $\text{update} : M \times V \to M$ is a *memory update function* and $\text{init} : V \to$

M is a *memory initialization function*. The *size* of the memory is the cardinality of the set M. A *strategy with memory* $\mathfrak{M}$ for Player σ is given by a next-move function $F : V_\sigma \times M \to V$ such that $F(v, m) \in vE$ for all $v \in V_\sigma, m \in M$. If a play, from starting position v_0, has gone through positions $v_0 v_1 \dots v_n$ the memory state is $m(v_0 \dots v_n)$, defined inductively by $m(v_0) = \text{init}(v_0)$, and $m(v_0 \dots v_i v_{i+1}) = \text{update}(m(v_0 \dots v_i), v_{i+1})$. In case $v_n \in V_\sigma$, the next move from $v_1 \dots v_n$, according to the strategy, leads to $F(v_n, m(v_0 \dots, v_n))$. In case $|M| = 1$, the strategy is *positional*; it can be described by a function $F : V_\sigma \to V$.

We will say that a game is determined via memory $\mathfrak{M}$ if it is determined and both players have winning strategies with memory $\mathfrak{M}$ on their winning regions. A game is *positionally determined* if it is determined via positional winning strategies.

Given a game graph $G = (V, V_0, V_1, E)$ and a memory structure $\mathfrak{M} = (M, \text{update}, \text{init})$ we obtain a new game graph $G \times \mathfrak{M} = (V \times M, V_0 \times M, V_1 \times M, E_{\text{update}})$ where

$$E_{\text{update}} = \{(v, m)(v', m') : (v, v') \in E \text{ and } m' = \text{update}(m, v')\}.$$

Obviously, every play $(v_0, m_0)(v_1, m_1) \dots$ in $G \times \mathfrak{M}$ has a unique projection to the play $v_0 v_1 \dots$ in G. Conversely, every play $v_0, v_1, \dots$ in G has a unique extension to a play $(v_0, m_0)(v_1, m_1) \dots$ in $G \times \mathfrak{M}$ with $m_0 = \text{init}(v_0)$ and $m_{i+1} = \text{update}(m_i, v_{i+1})$.

Consider two games $\mathcal{G} = (G, \Omega, W)$ and $\mathcal{G}' = (G', \Omega', W')$. We say that $\mathcal{G}$ *reduces via memory* $\mathfrak{M}$ *to* $\mathcal{G}'$, (in short $\mathcal{G} \leq_{\mathfrak{M}} \mathcal{G}'$) if $G' = G \times \mathfrak{M}$ and every play in $\mathcal{G}'$ is won by the same player as the projected play in $\mathcal{G}$.

Given a memory structure $\mathfrak{M}$ for G and a memory structure $\mathfrak{M}'$ for $G \times \mathfrak{M}$ we obtain a memory structure $\mathfrak{M}^* = \mathfrak{M} \times \mathfrak{M}'$ for G. The set of memory locations is $M \times M'$ and we have memory initialization $\text{init}^*(v) = (\text{init}(v), \text{init}'(v, \text{init}(v))$ and the update function

$$\begin{aligned} &\text{update}^*((m, m'), v) \; = \\ &\qquad = (\text{update}(m, v), \text{update}'(m', (v, \text{update}(m, v)))). \end{aligned}$$

Proposition 4.3. *Suppose that a game $\mathcal{G}$ reduces to $\mathcal{G}'$ via memory $\mathfrak{M}$ and that Player σ has a winning strategy for $\mathcal{G}'$ with memory $\mathfrak{M}'$ from $(v_0, \mathrm{init}(v_0)))$. Then Player σ has a winning strategy for $\mathcal{G}$ with memory $\mathfrak{M} \times \mathfrak{M}'$ from position v_0.*

Proof. Given a strategy $F' : (V_\sigma \times M) \times M' \to (V \times M)$ for Player σ on $\mathcal{G}'$ we have to construct a strategy $F : (V_\sigma \times (M \times M')) \to V \times (M \times M')$.

For a pair $(v, m) \in V_\sigma \times M$, we have that $F'(v, m) = (w, \mathrm{update}(m, w))$ where $w \in vE$. We now put $F(v, mm') = w$. If a play in $\mathcal{G}$ that is consistent with F proceeds from position v, with current memory location (m, m'), to a new position w, then the memory is updated to (n, n') with $n = \mathrm{update}(m, w)$ and $n' = \mathrm{update}'(m', (w, n))$. In the extended play in $\mathcal{G}'$ we have an associated move from position (v, m) to (w, n) with memory update from m' to n'. Thus, every play in $\mathcal{G}$ from initial position v_0 that is consistent with F is the projection of a play in $\mathcal{G}'$ from $(v_0, \mathrm{init}(v_0))$ that is consistent with F'. Therefore, if F' is a winning strategy from $(v_0, \mathrm{init}(v_0))$, then F is a winning strategy from v_0. □

Corollary 4.4. *Every game that reduces via memory $\mathfrak{M}$ to a positionally determined game, is determined via memory $\mathfrak{M}$.*

Obviously, memory reductions between games compose. If $\mathcal{G}$ reduces to $\mathcal{G}'$ with memory $\mathfrak{M}$ and $\mathcal{G}'$ reduces to $\mathcal{G}''$ with memory $\mathfrak{M}'$ then $\mathcal{G}$ reduces to $\mathcal{G}''$ with the memory $\mathfrak{M}^* = \mathfrak{M} \times \mathfrak{M}'$ defined above.

4.2 Latest appearance record for Muller games

One of the reasons for the interest in parity games is the fact that parity games over a finite set of priorities $C = \{0, \ldots, d\}$ are positionally determined [28, 59]. The classical example of a game reduction with finite memory on the other hand is the reduction of Muller games to parity games via latest appearance records [35]. Intuitively, a latest appearance record (LAR) is a list of priorities ordered by their latest occurrence.

More formally, for a finite set C of priorities, $\mathsf{LAR}(C)$ is the set of sequences $c_1 \ldots c_k \natural c_{k+1} \ldots c_\ell$ of elements from $C \cup \{\natural\}$ in which each

priority $c \in C$ occurs at most once, and $\natural$ occurs precisely once. At a position v, the LAR $c_1 \dots c_k \natural c_{k+1} \dots c_\ell$ is updated by moving the priority $\Omega(v)$ to the end, and moving $\natural$ to the previous position of $\Omega(v)$ in the sequence. For instance, at a position with priority c_2, the LAR $c_1c_2c_3 \natural c_4c_5$ is updated to $c_1 \natural c_3c_4c_5c_2$. (If $\Omega(v)$ did not occur in the LAR, we simply append $\Omega(v)$ at the end). Thus, the LAR-memory for an arena with priority labeling $\Omega : V \to C$ is the triple $(LAR(C), \text{init}, \text{update})$ with $\text{init}(v) = \natural\, \Omega(v)$ and

$$\text{update}(c_1 \dots c_k \natural c_{k+1} \dots c_\ell, v) = c_1 \dots c_k \natural c_{k+1} \dots c_\ell \Omega(v)$$

in case $\Omega(v) \notin \{c_1 \dots c_\ell\}$, and

$$\text{update}(c_1 \dots c_k \natural c_{k+1} \dots c_\ell, v) = c_1 \dots c_{m-1} \natural c_{m+1} \dots c_\ell c_m$$

if $\Omega(v) = c_m$.

The *hit-set* of a LAR $c_1 \dots c_k \natural c_{k+1} \dots c_\ell$ is the set $\{c_{k+1} \dots c_\ell\}$ of priorities occurring after the symbol $\natural$. Observe that if in a play $\pi = v_0v_1 \dots$, the LAR at position v_n is $c_1 \dots c_k \natural c_{k+1} \dots c_\ell$ then $\Omega(v_n) = c_\ell$ and the hit-set $\{c_{k+1} \dots c_\ell\}$ is the set of priorities that have been seen since the latest previous occurrence of c_ℓ in the play.

Lemma 4.5. *Let π be a play of a Muller game $\mathcal{G}$, and let* $\inf(\pi)$ *be the set of priorities occurring infinitely often in π. On π the hit-set of the latest appearance record is, from some point onwards, always a subset of* $\inf(\pi)$ *and infinitely often coincides with* $\inf(\pi)$.

Proof. For each play $\pi = v_0v_1v_2 \dots$ there is a position v_m such that $\Omega(v_n) \in \inf(\pi)$ for all $n \geq m$. Since no priority outside $\inf(\pi)$ is seen anymore after position v_m, the hit-set will from that point onwards always be contained in $\inf(\pi)$, and the LAR will always have the form $c_1 \dots c_{j-1}c_j \dots c_k \natural c_{k+1} \dots c_\ell$ where $c_1, \dots c_{j-1}$ remain fixed and the set $\{c_j, \dots, c_k, c_{k+1}, \dots c_\ell\} = \inf(\pi)$. Since all priorities in $\inf(\pi)$ are seen again and again, it happens infinitely often that, among these, the one occurring leftmost in the LAR is hit. At such positions, the LAR is updated to $c_1, \dots, c_{j-1} \natural c_{j+1} \dots c_\ell c_j$ and the hit-set then coincides with $\inf(\pi)$. □

Theorem 4.6. *Every Muller game with finitely many priorities reduces via LAR memory to a parity game.*

Proof. Let $\mathcal{G}$ be a Muller game with game graph G, priority labeling $\Omega : V \to C$ and winning condition $(\mathcal{F}_0, \mathcal{F}_1)$. We have to prove that $\mathcal{G} \leq_{\mathsf{LAR}} \mathcal{G}'$ for a parity game $\mathcal{G}'$ with game graph $G \times \mathsf{LAR}(C)$ and an appropriate priority labeling Ω' on $V \times \mathsf{LAR}(C)$ which is defined as follows.

$$\Omega'(v, c_1 c_2 \dots c_k \natural c_{k+1} \dots c_\ell) = \begin{cases} 2k & \text{if } \{c_{k+1}, \dots, c_\ell\} \in \mathcal{F}_0, \\ 2k+1 & \text{if } \{c_{k+1}, \dots, c_\ell\} \in \mathcal{F}_1. \end{cases}$$

Let $\pi = v_0 v_1 v_2 \dots$ be a play on $\mathcal{G}$ and fix a number m such that, for all numbers $n \geq m$ and $\Omega(v_n) \in \inf(\pi)$, the LAR at position v_n has the form $c_1 \dots c_j c_{j+1} \dots c_k \natural c_{k+1} \dots c_\ell$ where $\inf(\pi) = \{c_{j+1}, \dots c_\ell\}$ and the prefix $c_1 \dots c_j$ remains fixed. In the extended play $\pi' = (v_0 r_0)(v_1, r_1) \dots$ all nodes (v_n, r_n) for $n \geq$ will therefore have a priority $2k + \rho$ with $k \geq j$ and $\rho \in \{0, 1\}$. Assume that the play π is won by Player σ, i.e. $\inf(\pi) \in \mathcal{F}_\sigma$. Since infinitely often the hit-set of the LAR coincides with $\inf(\pi)$, the minimal priority seen infinitely often on the extended play is $2j + \sigma$. Thus the extended play in the parity game $\mathcal{G}'$ is won by the same player as the original play in the Muller game $\mathcal{G}$. □

Observe that for a Muller game on n priorities, an LAR-memory has $n!$ memory states. Dziembowski, Jurdziński, and Walukiewicz [22] have shown that with this respect LAR-strategies are essentially optimal for Muller games.

Theorem 4.7. *There exists a sequence $(\mathcal{G}_n)_{n \in \omega}$ of Muller games such that the game graph of $\mathcal{G}_n$ is of size $O(n)$ and every winning strategy for $\mathcal{G}_n$ requires a memory of size at least $n!$*

4.3 Games with infinitely many priorities

The definition of the Muller condition (Definition 4.1) directly generalizes to countable sets C of priorities. Note that with minor modifications it

can also be generalized to uncountable sets C, see [32] for a discussion of this. But some properties that hold for a finite set of priorities C do not generalize even to countable sets. One of them is the possibility to represent any Muller condition by a Zielonka tree, which we discuss in section 4.6.

For finitely many priorities, the condition that $\mathcal{F}_0$ and $\mathcal{F}_1$ are both closed under finite unions is sufficient for positional determinacy of any game with this Muller condition. To see that this is not the case for infinite sets C, let us discuss the possible generalizations of parity games to the case of priority assignments $\Omega : V \to \omega$. For parity games with finitely many priorities it is of course purely a matter of taste whether we let the winner be determined by the least priority seen infinitely often or by the greatest one. Here this is no longer the case. Based on priority assignments $\Omega : V \to \omega$, we consider the following classes of games.

Infinity games are games where Player 0 wins those infinite plays in which no priority at all appears infinitely often, i.e.

$$\mathcal{F}_0 = \{\varnothing\},$$
$$\mathcal{F}_1 = \mathcal{P}(\omega) \setminus \{\varnothing\}.$$

Parity games are games where Player 0 wins the plays in which the least priority seen infinitely often is even, or where no priority appears infinitely often. Thus,

$$\mathcal{F}_0 = \{X \subseteq \omega : \min(X) \text{ is even}\} \cup \{\varnothing\},$$
$$\mathcal{F}_1 = \{X \subseteq \omega : \min(X) \text{ is odd}\}.$$

Max-parity games are games where Player 0 wins if the maximal priority occurring infinitely often is even, or does not exist, i.e.

$$\mathcal{F}_0 = \{X \subseteq \omega : \text{ if } X \neq \varnothing \text{ and } X \text{ is finite then } \max(X) \text{ is even}\},$$
$$\mathcal{F}_1 = \{X \subseteq \omega : X \text{ is finite, non-empty, and } \max(X) \text{ is odd}\}.$$

It is easy to see that infinity games are a special case of parity games, via a simple reassignment of priorities. Further, we note that for both parity games and max-parity games, $\mathcal{F}_0$ and $\mathcal{F}_1$ are closed under finite unions. Nevertheless the conditions behave quite differently.

Proposition 4.8. *Max-parity games with infinitely many priorities in general do not admit finite memory winning strategies.*

Proof. Consider the max-parity game with positions $V_0 = \{0\}$ and $V_1 = \{2n+1 : n \in \mathbb{N}\}$ (where the name of a position is also its priority), such that Player 0 can move from 0 to any position $2n+1$ and Player 1 can move back from $2n+1$ to 0. Clearly Player 0 has a winning strategy from each position but no winning strategy with finite memory. □

On the other hand it has been shown in [32] that infinity games and parity games with priorities in ω do admit positional winning strategies for both players on all game graphs. In fact, parity games over ω turn out to be the only Muller games with this property.

Theorem 4.9. *[32] Let $(\mathcal{F}_0, \mathcal{F}_1)$ be a Muller winning condition over a countable set C of priorities. Then the following are equivalent.*

- *Every game with winning condition $(\mathcal{F}_0, \mathcal{F}_1)$ is positionally determined.*
- *Both $\mathcal{F}_0$ and $\mathcal{F}_1$ are closed under finite unions, unions of chains, and non-empty intersections of chains.*
- *The Zielonka tree of $(\mathcal{F}_0, \mathcal{F}_1)$ exists, and is a path of co-finite sets (and possibly the empty set at the end).*
- *$(\mathcal{F}_0, \mathcal{F}_1)$ reduces to a parity condition over $n \leq \omega$ priorities.*

4.4 Finite appearance records

Although over an infinite set of priorities one can easily define Muller games that do not admit finite memory strategies, these games are often

solvable by strategies with very simple infinite memory structures. For instance, for the max-parity game described in the proof of Proposition 4.8, it suffices for Player 0 to store the maximal priority seen so far, in order to determine the next move in her winning strategy. One can readily come up with other games where the memory required by a winning strategies is essentially a finite collection of previously seen priorities.

This motivates the definition of an infinite memory structure that we call *finite appearance records* (FAR) which generalizes the LAR-memory for games with finitely many priorities. In a FAR we store tuples of previously encountered priorities or some other symbols from a finite set. Additionally the update function in the appearance record is restricted, so that new values of the memory can be equal only to the values stored before or to the currently seen priority.

Definition 4.10. A d-dimensional FAR-memory for a game $\mathcal{G}$ with priorities in C is a memory structure $(M, \text{update}, \text{init})$ for $\mathcal{G}$ with $M = (C \cup N)^d$ for some finite set N such that whenever

$$\text{update}(m_1, \ldots, m_d, v) = (m'_1, \ldots, m'_d)$$

then $m'_i \in \{m_1, \ldots, m_d\} \cup N \cup \{\Omega(v)\}$.

Observe that an LAR-memory over a finite set C is a special case of an FAR-memory, with $d = |C| + 1$ and $N = \{\natural, B\}$, where B is a blank symbol used to pad latest appearance records in which some priorities are missing. Here the dimension of the FAR depends on the size of C. Hence, the question arises whether there is a fixed dimension d and a fixed additional set N such that every Muller game over finitely many priorities reduces to a parity game via d-dimensional FAR-memory. From Theorem 4.7 it follows that his is not the case. Indeed, since $n!$ grows faster than n^d for any constant d, we infer that for any dimension d there is a Muller game $\mathcal{G}_d$ that can not be reduced to a parity game via d-dimensional FAR-memory. From this we obtain the following conclusion.

Proposition 4.11. *There exists a Muller game $\mathcal{G}$ that does not reduce to a parity game with any* FAR*-memory.*

Proof. Take $\mathcal{G}$ to be the disjoint sum of the games $\mathcal{G}_d$, assuming that all these games have disjoint sets of priorities. Suppose that $\mathcal{G}$ reduces to a parity game via some FAR-memory of dimension d. Since game extensions preserve connectivity it follows that the extension of the connected component $\mathcal{G}_d$ of $\mathcal{G}$ will also be a parity game. But this contradicts the fact that $\mathcal{G}_d$ does not reduce to a parity game via d-dimensional FAR-memory. □

4.5 FAR reductions for infinitary Muller games

In this section we consider some cases of Muller games with priorities in ω that admit FAR-reductions to positionally determined games.

To illustrate the idea consider any *downwards cone* $\mathcal{F}_0 = \{X : X \subseteq A\}$ for a fixed set $A \subseteq \omega$. Again it is easy to see that such games may require infinite-memory strategies. To reduce such a game to a parity game $\mathcal{G}'$ it suffices to store the maximal priority m seen so far, and to define priorities in $\mathcal{G}'$ by

$$\Omega'(v, m) = \begin{cases} 2m + 2 & \text{if } \Omega(v) \in A, \\ 2\Omega(v) + 1 & \text{otherwise.} \end{cases}$$

If $\inf(\pi) \subseteq A$ then Player 0 wins π' since no odd priority is seen infinitely often in π'. If there is some $a \in \inf(\pi) \setminus A$, then $2a + 1$ occurs infinitely often in π', and since $a \leq m$ from some point onwards, no smaller even priority can have this property, so Player 1 wins π'.

Hence any Muller game such that $\mathcal{F}_0$ (or $\mathcal{F}_1$) is a downwards cone is determined via one-dimensional FAR-memory.

4.5.1 Visiting sequences and singleton Muller conditions

Our next example for winning conditions that are amenable for an approach via FAR-reductions are Muller games where the winning condition of Player 0 is a singleton, i.e. $\mathcal{F}_0 = \{A\}$, $\mathcal{F}_1 = \mathcal{P}(\omega) \setminus \{A\}$.

We first observe that such games may require infinite memory.

Theorem 4.12. *For any $A \neq \emptyset$, there exists a (solitaire) Muller game with $\mathcal{F}_0 = \{A\}$ whose winning strategies all require infinite memory.*

Proof. If $A = \{a_1, a_2, \dots\}$ is infinite, take the game with set of positions $V = V_0 = A$ (where the name of a position indicates also its priority), and moves (a_1, a_n) and (a_n, a_1) for all $n \geq 2$. If $A = \{a_1, \dots, a_n\}$ is finite, let $\omega \setminus A = \{b_1, b_2, \dots\}$. We consider instead the game with $V = V_0 = A \cup (\omega \setminus A)$, and set of moves

$$\begin{aligned} E = &\{(a_i, a_{i+1}) : 1 \leq i < n\} \cup \\ &\{(a_n, b) : b \in (\omega \setminus A)\} \cup \{(b, a_1) : b \in (\omega \setminus A)\}. \end{aligned}$$

In both cases, Player 0 wins, but requires infinite memory to do so. □

We will prove that singleton Muller games can be reduced via FAR-memory to parity games with priorities in ω which, as shown in [32], are positionally determined. The FAR-memory that we use for this reduction is based on a particular order in which the elements of the winning sets have to be seen infinitely often, which is specified by a visiting sequence.

Definition 4.13. Let $A = \{a_1 < a_2 < \dots\}$ be an infinite subset of ω. For each $n \in \omega$, let $p(a_n) := a_1 a_2 \dots a_n$ be the *prefix* of a_n. The *visiting sequence* of A is the concatenation of the prefixes of all elements of A

$$\mathrm{visit}(A) = p(a_1)p(a_2)p(a_3)\dots$$

For a finite set $\{a_1 < a_2 < \dots < a_n\} \subseteq \omega$ we define $\mathrm{visit}(A) = p(a_n)^\omega$.

Let $\mathcal{G}$ be a Muller game over ω.

Lemma 4.14. *For any play $\pi = v_1 v_2 \dots$ of $\mathcal{G}$ the set $\mathrm{Inf}(\pi)$ is the unique set A with the following two properties:*

(1) There exists a sequence of indices $i_1 < i_2 < \dots$ such that the priorities $\Omega(v_{i_1})\Omega(v_{i_2})\dots$ form the visiting sequence of A.

(2) If $\Omega(v_k) \in \omega \setminus A$ then there is only a finite number of indices $i > k$ such that $\Omega(v_i) \in \{0, \dots, \Omega(v_k)\} \cap \omega \setminus A$.

Proof. First we notice that $A = \mathrm{Inf}(\pi)$ indeed fulfills these two properties. The visiting sequence can be chosen from the play as all elements of $\inf(\pi)$ appear infinitely often. Since all elements of $\omega \setminus \inf(\pi)$ occur only finitely often in the play, the second property must also hold.

Conversely, if a set A satisfies property (1), then all elements of A appear infinitely often in π, so $A \subseteq \inf(\pi)$. If there were an element $a \in \inf(\pi) \setminus A$, then for any k with $\Omega(v_k) = a$, there were infinitely many indices $i > k$, with $\Omega(v_i) = a$ which contradicts property (2). Thus if A satisfies properties (1) and (2), then $A = \inf(\pi)$. □

Let $A \subseteq \omega$ be infinite. Any initial segment of the visiting sequence of A can be written in the form $p(a_1)p(a_2)\dots p(a_i)a_1a_2\dots a_j$ where $1 \leq j \leq i+1$. It can be represented by a pair (p, c) where $c = a_j$ indicates the position of the last letter in the current prefix $p(a_{i+1})$, and $p = a_i$ indicates the last previously completed prefix (or ε if we are at the first element). For instance, the initial segment $a_1\, a_1a_2\, a_1a_2a_3\, a_1a_2a_3$ of the visiting sequence of A is encoded by (a_3, a_3), the initial segment a_1 is encoded by (ε, a_1), and the empty initial segment by $(\varepsilon, \varepsilon)$. We write $\mathrm{visit}_n(A)$ for the initial segment of length A of $\mathrm{visit}(A)$.

Given a (finite or infinite) winning set A, we want to use a three-dimensional FAR-memory to check whether $\inf(\pi) = A$. For infinite A, the memory state after an initial segment of a play is a triple (p, c, q) where (p, c) encode the initial segment of the visiting sequence of A that has been seen so far, and q is the maximal priority that has occurred.

Definition 4.15. For any infinite set $A \subseteq \omega$, we define a three-dimensional FAR-memory $\mathsf{FAR}(A) = (M, \mathrm{init}, \mathrm{update})$ with $M = \{(p, c, q) : p, c \in \omega \cup \{\varepsilon\}, q \in \omega\}$. The initialization function is defined by

$$\mathrm{init}(v) = \begin{cases} (\varepsilon, \Omega(v), \Omega(v)) & \text{if } \Omega(v) = a_1 \\ (\varepsilon, \varepsilon, \Omega(v)) & \text{if } \Omega(v) \neq a_1 \end{cases}$$

The update function is defined by

$$\mathrm{update}(p, c, q, v) = (p', c', q'),$$

where $q' = \max(q, \Omega(v))$, and where (p, c) and (p', c') encode, for some n, the initial segments $\text{visit}_n(A)$ and $\text{visit}_{n+1}(A)$, respectively, of the visiting sequence of A such that $\text{visit}_{n+1}(A) = \text{visit}_n(A)\Omega(v)$, or otherwise, $(p', c') = (p, c)$.

For a more formal description, let

$$\text{up}(p, c, v) = \begin{cases} 2 & \text{if, for some } i,\ p = a_i, c = a_{i+1}, \Omega(v) = a_1 \\ 1 & \text{if, for some } j \leq i,\ p = a_i, c = a_j, \Omega(v) = a_{j+1} \\ 0 & \text{otherwise} \end{cases}$$

(where, to simplify notation, we identify ε with a_0). Note that $\text{up}(p, c, v) = 2$ if, at node v, the visiting sequence is updated with an a_1 (i.e. a prefix $p(a_i)$ has been completed and a new one is started), that $\text{up}(p, c, v) = 1$ if the visiting sequence is updated by another value, and that $\text{up}(p, c, v) = 0$ if no update of the visiting sequence happens at v. Then we can define $\text{update}(p, c, q, v) := (p', c', q')$ by

$$\begin{aligned} (p', c') &= \begin{cases} (c, \Omega(v)) & \text{if up}(p, c, v) = 2 \\ (p, \Omega(v)) & \text{if up}(p, c, v) = 1 \\ (p, c) & \text{if up}(p, c, v) = 0 \end{cases} \\ q' &= \max(q, \Omega(v)) \end{aligned}$$

For finite $A = \{a_1 < a_2 < \cdots < a_n\}$ this has to be modified since once cannot really encode the part of the visiting sequence that one has seen with priorities in A. In this case the value (p, c, q) is so that c is the last element of the visiting sequence, q is the maximal priority that has occurred so far, and p is the maximal priority that had occurred up to the last time when, in the visiting sequence of A, a prefix $p(a_n)$ had been completed and c had been updated from a_n to a_1. Thus we set

$$\text{up}(p, c, v) = \begin{cases} 2 & \text{if } c = a_n, \Omega(v) = a_1 \\ 1 & \text{if, for some } i < n,\ c = a_i, \Omega(v) = a_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

and $\mathrm{update}(p, c, q, v) := (p', c', q')$ with

$$\begin{aligned}(p', c') &= \begin{cases}(q, \Omega(v)) & \text{if } \mathrm{up}(p, c, v) = 2\\ (p, \Omega(v)) & \text{if } \mathrm{up}(p, c, v) = 1\\ (p, c) & \text{if } \mathrm{up}(p, c, v) = 0\end{cases}\\ q' &= \max(q, \Omega(v)).\end{aligned}$$

Theorem 4.16. *Any singleton Muller game with $\mathcal{F}_0 = \{A\}$ can be reduced, via memory $\mathsf{FAR}(A)$, to a parity game.*

Proof. The given Muller game $\mathcal{G}$ with arena (G, Ω) and Muller condition such that $\mathcal{F}_0 = \{A\}$ is reduced via memory $\mathsf{FAR}(A)$ to a parity game $\mathcal{G}'$ with priority function $\Omega' : V \times \mathsf{FAR}(A) \to \omega$ defined as follows.

$$\Omega'(v, p, c, q) = \begin{cases}2p + 2 & \text{if } \Omega(v) \in A, \mathrm{up}(p, c, v) \in \{1, 2\}\\ 2p + 3 & \text{if } \Omega(v) \in A, \mathrm{up}(p, c, v) = 0\\ \min(2p + 3, 2\Omega(v) + 1) & \text{if } \Omega(v) \notin A\end{cases}$$

We have to prove that any play $\pi = v_0 v_1 v_2 \dots$ of $\mathcal{G}$ is won by the same player as the extended play $\pi' = (v_0, p_0, c_0, q_0)(v_1, p_1, c_1, q_1) \dots$ of $\mathcal{G}'$.

We first assume that $\inf(\pi) = A$ and prove that either no priority at all occurs infinitely often in π' or the minimal such is even. If A is infinite, then the sequence of the values p_n diverges and therefore no priority will be seen infinitely often in π'. If A is finite then it may be the case that the sequence $(p_n)_{n\in\omega}$ converges, i.e. $p_n = p$ from some point onwards. But since the visiting sequence will be updated again and again this means that infinitely often the priority $2p + 2$ occurs in π', and the only other priority that may occur infinitely often is $2p + 3$. Hence Player 0 wins π'.

For the converse, we assume that Player 1 wins π. We distinguish several cases. If there exist some $a \in A \setminus \inf(\pi)$ then from some point onwards, the visiting sequence cannot be updated anymore, so the sequence $(p_n)_{n\in\omega}$ stabilizes at some value p. Then the minimal priority seen infinitely often is either $2p + 3$, or $2\Omega(v) + 1$ for some $\Omega(v) \in \omega \setminus A$ and Player 1 also wins π'. If no such element a exists, then $A \subsetneq \inf(\pi)$ and there is a

minimal element $b \in \inf(\pi) \setminus A$. If the sequence $(p_n)_{n\in\omega}$ diverges (which is always the case for infinite winning sets A) then the minimal priority seen infinitely often in π' is $2b+1$. If A is finite then the sequence p_n may stabilize at some value p which coincides with the largest priority ever occurring in π. Hence $b \leq p$ and therefore $2b+1 < 2p+2$, so the minimal priority seen infinitely often in π' is $2b+1$. Again Player 1 wins the associated play in the parity game. □

Corollary 4.17. *Singleton Muller games are determined with* FAR *memory.*

4.5.2 Finite unions of upwards cones

Visiting sequences can also be used for the case where $\mathcal{F}_0$ is a finite union of upwards cones, i.e.

$$\mathcal{F}_0 = \bigcup_{i=1}^{k} \{X : A_i \subseteq X \subseteq \omega\}$$

for some finite collection of sets $A_1, \ldots, A_k$.

The FAR-memory stores the pairs (p_i, c_i) encoding the visiting sequences of $A_1, \ldots, A_k$. All that has to checked is whether $A_i \subseteq \inf(\pi)$ for some i, which is the case if, and only if, one of the visiting sequences is updated infinitely often. Thus we can define priorities by

$$\Omega'(v, p_1, c_1, \ldots, p_k, c_k) = \begin{cases} 0 & \text{if } \mathrm{up}(p_i, c_i, v) = 2 \text{ for some } i \\ 1 & \text{otherwise.} \end{cases}$$

Theorem 4.18. *Any Muller game such that $\mathcal{F}_\sigma$ is a finite union of upwards cones is determined via* FAR*-memory.*

4.5.3 Muller conditions with finitely many winning sets

We now consider the case of Muller games whose winning conditions are defined by a finite collection of (possibly infinite) sets, $\mathcal{F}_0 = \{A_1, \ldots, A_k\}$. To extend the idea presented above to this case we are going to use the memory $\mathsf{FAR}(A_i)$ for each set A_i and additionally we have to remember

when the set A_i is *active*, as is described below. The property of being active is stored in a value $a_i \in \{0,1,2\}$.

Definition 4.19. For a finite collection $\{A_1,\dots,A_k\}$ of sets $A_i \subseteq \omega$, we define a $4k$-dimensional FAR-memory $\mathsf{FAR}(A_1,\dots,A_k) = (M, \mathrm{init}, \mathrm{update})$. We denote the FAR-memory of A_i by $\mathsf{FAR}(A_i) = (M_i, \mathrm{init}_i, \mathrm{update}_i)$. Then $M = M_1 \times M_2 \times \ldots \times M_k \times \{0,1,2\}^k$. The initialization function is defined by

$$\mathrm{init}(v) = (\mathrm{init}_1(v),\dots,\mathrm{init}_k(v),\bar{0}).$$

The update function is defined by

$$\mathrm{update}(m_1,\dots,m_k,a_1,\dots,a_k,v) = \\ (\mathrm{update}_1(m_1,v),\dots,\mathrm{update}_k(m_k,v),a_1',\dots,a_k'),$$

where a_i' is the new activation value for sequence i defined by

$$a_i' = \begin{cases} 0 & \text{if } v \notin A_i \text{ and for some } j \le k\ \mathrm{up}_j(m_j,v) > 0 \\ \min(2, a_i+1) & \text{if } \mathrm{up}_i(m_i,v) = 2 \\ a_i & \text{otherwise.} \end{cases}$$

Theorem 4.20. *Any Muller game with $\mathcal{F}_0 = \{A_1,\dots,A_k\}$ can be reduced, via memory $\mathsf{FAR}(A_1,\dots,A_k)$, to a parity game.*

Proof. The given Muller game $\mathcal{G}$ with arena (G,Ω) and Muller condition such that $\mathcal{F}_0 = \{A_1,\dots,A_k\}$ is reduced to a parity game $\mathcal{G}'$ with priority function Ω' defined by

$$\Omega'(v,\overline{m},\overline{a}) = \begin{cases} \max_{\mathrm{Act}(v,\overline{a})}(2kp_i + 2r_i + 2) & \text{if exists } j \text{ such that} \\ & \Omega(v) \in A_j, a_j = 2, \\ & \mathrm{up}_j(m_j,v) \in \{1,2\} \\ \max_{\mathrm{Act}(v,\overline{a})}(2kp_i + 2r_i + 3) & \text{if exists } j \text{ such that} \\ & \Omega(v) \in A_j, a_j = 2, \\ & \text{and } \mathrm{up}_j(m_j,v) = 0 \\ & \text{for all such } j \\ \min(2kp_{\max} + 3, 2\Omega(v) + 1) & \text{otherwise} \end{cases}$$

where $\mathrm{Act}(v, \overline{a}) = \{i : \Omega(v) \in A_i \wedge a_i = 2\}$ are the indices of active sets to which v belongs, p_i is the first component of the i-th memory $m_i = (p_i, c_i, q_i)$, $p_{\max} = \max\{p_1 \dots p_k\}$ and for each $A_i \in \mathcal{F}_0$ we have $r_i = |\{A_j \in \mathcal{F}_0 : A_i \subseteq A_j\}|$.

We have to prove that any play $\pi = v_0 v_1 v_2 \dots$ of $\mathcal{G}$ is won by the same player as the extended play

$$\pi' = (v_0, m_{10}, \dots, m_{k0}, a_{10}, \dots a_{k0})(v_1, m_{11}, \dots, m_{k1}, a_{11}, \dots a_{k1}) \dots .$$

For a given play π of $\mathcal{G}$, we divide the sets $A_1, \dots, A_k \in \mathcal{F}_0$ into three classes.

The good: A_i is a good set if A_i is active ($a_i = 2$) only finitely often in π.

The bad: A_i is a bad set, if $A_i \subseteq \inf(\pi)$ and A_i is not a good set.

The ugly: A_i is an ugly set if there is a priority $c \in A_i \setminus \inf(\pi)$ and A_i is not a good set.

Lemma 4.21. *If A_i is bad and A_j is ugly, then $A_i \subseteq A_j$.*

Proof. Assume that there is a $b \in A_i \setminus A_j$. Since $A_i \subseteq \inf(\pi)$ the visiting sequence for A_i is updated infinitely often, hence infinitely often with b, and whenever this happens then a_j is reset to 0. By definition there is a $c \in A_j$ that is seen only finitely many times in π. Therefore $a_j = 0$ from some point onwards. But this contradicts the assumption that A_j is not good. □

We first assume that $\inf(\pi) = A_i$ and prove that either no priority at all occurs infinitely often in π' or the minimal such priority is even.

Since from some point on there is no priority $d \notin A_i$ that occurs infinitely often, then for all sets A_j that are not subsets of A_i the visiting sequence will not be updated any more, and so the sequence $(p_{jn})_{n\in\omega}$ stabilizes at some value p_j. Since the visiting sequence of A_i is updated infinitely often, we get that from some point on $a_i = 2$. Hence A_i is a bad set. We can now argue as in the proof of Theorem 4.16: if infinitely many priorities appear in π, then the sequence $(p_{in})_{n\in\omega}$ diverges and no priority

at all will be seen infinitely often in π'. It remains to consider the case where only finitely many priorities occur in π. Then the sequence $(p_{in})_{n\in\omega}$ stabilizes at some value p, which is the maximal priority appearing in π. For any $A_j \subsetneq A_i$, the sequence $(p_{jn})_{n\in\omega}$ will then also stabilize at the same value p, and $r_j > r_i$. It follows that some priority of form $2kp + 2r_\ell + 2$ occurs infinitely often in π', where $r_\ell \geq r_i$.

Suppose now that some smaller odd priority occurs infinitely often in π'. Then it would have to be of the form $2kp + 2r_j + 3$ with $r_j < r_\ell$ such that $a_j = 2$ infinitely often. However, only finitely many priorities appear in π. Hence if there are infinitely many positions v such that $\Omega(v) \in A_j$ and $a_j = 2$, then from some point onwards all these positions v satisfy that $\Omega(v) \in A_j \cap A_i$ and $a_i = 2$. On infinitely many such positions an update happens, and therefore, also the priority $2kp + 2r_j + 2$ appears infinitely often. Hence Player 0 wins π'.

For the converse, we now assume that Player 1 wins π.

Lemma 4.22. *Suppose that some even priority $2kq + 2r + 2$ is seen infinitely often in π'. Then q is the maximal priority that occurs in π and $r = r_\ell$ for some bad set A_ℓ.*

Proof. If there are infinitely many occurrences of $2kq + 2r + 2$ in π', then q is the maximal priority that occurs in π and some A_i is updated infinitely often (i.e. $A_i \subseteq \inf(\pi)$) and active infinitely often. Obviously A_i is bad and $r \geq r_i$. If $r \neq r_\ell$ for all bad set A_ℓ, then $r = r_j$ for some other A_j that is active infinitely often. Thus A_j has to be ugly. But then by Lemma 4.21 $A_i \subseteq A_j$ and thus $r_i > r_j = r$. But $r \geq r_i$. □

Let $r = \min\{r_\ell : A_\ell \text{ is bad}\}$. To show that Player 1 wins π' it suffices to prove that there is an odd priority occurring infinitely often in π' which, in case there exists a bound q on all priorities appearing in π, is smaller than $2kq + 2r + 2$.

Notice that for any ugly set A_i, the sequence $(p_{in})_{n\in\omega}$ stabilizes at some value p_i. Let $p = \max\{p_i : A_i \text{ is ugly}\}$.

We distinguish two cases. First we assume that there exists some priority

$$b \in \inf(\pi) \setminus \bigcup\{A_i : A_i \text{ is bad}\}.$$

Fix n_0 such that, for all $n > n_0$, $p_{in} = p_i$ for all ugly sets A_i and $a_{in} \neq 2$ for all good sets A_i. Since $b \in \inf(\pi)$ there exist infinitely many v_n with $n > n_0$ and $\Omega(v_n) = b$. For such v_n we have $\Omega'(v_n, \bar{m}_n, \bar{a}_n) = 2kp+2r_i+3$ if there is a set A_i (which has to be ugly) such that $b \in A_i$ and $a_i = 2$.

Otherwise $\Omega'(v_n, \bar{m}_n, \bar{a}_n)$ is odd and $\leq 2b+1$. Since A_i is ugly and A_ℓ is bad it follows that $A_\ell \subseteq A_i$. Thus, $r_i < r$. Further $p \leq q$. It follows that there exists some odd priority $s \leq \max(2kp + 2r_i + 3, 2b+1) < 2kq + r + 2$ that appears in π' infinitely often.

Now we consider the other case: every $b \in \inf(\pi)$ is contained in some bad set $A_{i(b)}$. Let $A_1, \ldots, A_\ell$ be the bad sets. Without loss of generality, we assume that A_1 is a maximal bad set, i.e. $A_1 \not\subseteq A_i$ for $i = 2, \ldots, \ell$. Since A_1 is a strict subset of $\inf(\pi)$, we can fix a priority $d \in \inf(\pi) \setminus A_1$. Since any priority $d \in \inf(\pi)$ is contained in some bad set, we can assume that $d \in A_2$. Further, by the maximality of A_1, we can fix priorities $e_2, \ldots, e_\ell$ where $e_i \in A_1 \setminus A_i$.

We consider a suffix of π that starts at a position where

- all sequences $(p_{in})_{n\in\omega}$ that stabilize at some value p_i have already reached that value,
- all good sets A_i have become inactive for good (i.e. $a_i \neq 2$),
- in the visiting sequence for A_1 the prefixes $p(e_1), \ldots p(e_\ell)$ have already been completed.

Note that A_1 is updated infinitely often, and between any two consecutive points in this suffix at which $\mathrm{up}_1 = 2$ all priorities $e_2, \ldots, e_\ell$ are seen at least once. Since the priority d appears infinitely often in π and A_2 is updated infinitely often, we are going to see infinitely many points v_{n_0} in the considered suffix of π for which $\Omega(v_{n_0}) = d$ and $a_1 = 0$ (since a_1 is reset with an update of A_2). Since a_1 increases to 2 infinitely often, there

are infinitely many tuples $n_0 < n_1 < n_2$ such that $a_1 = i$ at all positions v_n with $n_i \leq n < n_{i+1}$ and $a_1 = 2$ at v_{n_2}.

By definition $\mathrm{up}_1 = 2$ at v_{n_1} and v_{n_2} and there cannot be any updates on priority d between v_{n_1} and v_{n_2}, as then a_1 would be reset to 0. By our choice of the considered suffix of π, there are updates on all $e_2, \ldots, e_\ell$ between v_{n_1} and v_{n_2}. Therefore, for any bad set A_j that contains d, we have that $a_j < 2$ between position v_{n_2} and the first position v_n with $\Omega(v_n) = d$ that comes after v_{n_2}. This is the case because between v_{n_1} and v_{n_2} the value a_j was reset to 0 by the update of the visiting sequence for A_1 by priority e_j, and since then it has not increased by more than 1 since there was no update on priority d.

Let us now consider the new priority at v_n. Since all bad sets A_j containing d are inactive, we have the same situation as in the first case: $\Omega'(v_n, \bar{m}_n, \bar{a}_n) = 2kp + 2r_i + 3$ if there is a set A_i (which has to be ugly) such that $d \in A_i$ and $a_i = 2$. Otherwise $\Omega'(v_n, \bar{m}_n, \bar{a}_n)$ is odd and $\leq 2d+1$. Since A_i is ugly and A_ℓ is bad it follows that $A_\ell \subseteq A_i$ and thus $r_i < r_\ell = r$. Further $p \leq q$.

There are infinitely many such positions v_n. Thus there must exist some odd priority $s \leq \max(2kp + 2r_i + 3, 2d + 1) < 2kq + r + 2$ that appears in π' infinitely often. □

Of course, the same arguments apply to the case where $\mathcal{F}_1$ is finite.

Corollary 4.23. *Let $(\mathcal{F}_0, \mathcal{F}_1)$ be a Muller winning condition such that either $\mathcal{F}_0$ or $\mathcal{F}_1$ is finite. Then every Muller game with this winning condition is determined via* FAR *memory.*

4.5.4 Max-parity games with bounded moves

We say that a an arena (G, Ω) has *bounded moves* if there is a natural number d such that $|\Omega(v) - \Omega(w)| \leq d$ for all moves (v, w) of G.

We have shown in Proposition 4.8 that, in general, winning strategies for max-parity games require infinite memory, but we do not know whether max-parity games are determined via FAR-memory.

For max-parity games with bounded moves, it is still the case that winning strategies may require infinite memory, but now we can prove determinacy via FAR-memory.

Proposition 4.24. *There exist max-parity games with bounded moves whose winning strategies require infinite memory.*

Proof. Consider a (solitaire) max-parity game with a single node v_0 of priority 0 from which Player 0 has, for every odd number $2n+1$, the option to go through a cycle C_n consisting of nodes with priorities $2, 4, \ldots, 2n, 2n+1, 2n, 2n-2, \ldots, 4, 2$ and back to the node v_0. All these cycles intersect only at v_0. Clearly Player 0 has a winning strategy, namely to go successively through cycles $C_1, C_2, \ldots$ with the result that there is no maximal priority occurring infinitely often. However, if Player 0 moves according to a finite-memory strategy then only finitely many cycles will be visited and there is a maximal n such that the cycle C_n will be visited infinitely often. Thus the maximal priority seen infinitely often will be $2n+1$ and Player 0 loses. □

Lemma 4.25. *Let π be a play of a max-parity game $\mathcal{G}$ with bounded moves such that infinitely many different priorities occur in π. Then* $\max(\inf(\pi))$ *does not exist, so π is won by Player 0.*

Proof. Assume that moves of $\mathcal{G}$ are bounded by d and $\inf(\pi) \neq \varnothing$ and let q be any priority occurring infinitely often on π. Since infinitely many different priorities occur on π it must happen infinitely often that from a position with priority q the play eventually reaches a priority larger than $q+d$. Since moves are bounded by d, this means that on the way the play has to go through at least one of the priorities $q+1, \ldots, q+d$. Hence at least one of these priorities also occurs infinitely often, so q cannot be maximal in $\inf(\pi)$. □

Theorem 4.26. *Every max-parity game with bounded moves can be reduced via a one-dimensional* FAR*-memory to a parity game. Hence max-parity games are determined via strategies with one-dimensional* FAR*-memory.*

Proof. The FAR-memory simply stores the maximal priority m that has been seen so far. To reduce a max-parity game $\mathcal{G}$ with bounded moves, via this memory, to a parity game $\mathcal{G}'$ we define the priorities of $\mathcal{G}'$ by

$$\Omega'(v, m) = 2m - \Omega(v).$$

Let π be a play of $\mathcal{G}$ and let π' be the extended play in $\mathcal{G}'$. We distinguish two cases. First, we assume that on π the sequence of values for m is unbounded. This means that infinitely many different priorities occur on π, so by Lemma 4.25, Player 0 wins π. But since $m \leq \Omega'(v, m)$ and m never stabilizes there is no priority that occurs infinitely often on π, so π' is also won by Player 0.

In the second case there exists a suffix of π on which m remains fixed on the maximal priority of π. In that case $\inf(\pi)$ is a non-empty subset of $\{0, \ldots, m\}$ and $\inf(\pi')$ is a non-empty subset of $\{m, \ldots, 2m\}$. Further, $\Omega'(v, m)$ is even if, and only if $\Omega(v)$ is even, and $\Omega'(v_1, m) < \Omega'(v_2, m)$ if, and only if, $\Omega(v_1) > \Omega(v_2)$. Thus, $\min(\inf(\pi'))$ is even if, and only if, $\max(\inf(\pi))$ is even. Hence π is won by the same Player as π'. □

4.6 Infinitary Zielonka-tree memory

In this section we stop investigating finite appearance records and study Muller conditions represented by Zielonka trees. These trees, with nodes labeled by sets of priorities, were introduced by Zielonka in [78] under the name of split trees to establish how much memory is needed for strategies in games with a fixed Muller condition.

Definition 4.27 (Cf. [78])**.** The *Zielonka tree* for a Muller condition $(\mathcal{F}_0, \mathcal{F}_1)$ over a set C of priorities is a tree $Z(\mathcal{F}_0, \mathcal{F}_1)$ whose nodes are labeled with pairs (X, σ) such that $X \in \mathcal{F}_\sigma$. We define $Z(\mathcal{F}_0, \mathcal{F}_1)$ inductively as follows. Let $C \in \mathcal{F}_\sigma$ and $C_0, \ldots, C_k$ be the maximal sets in $\{X \subseteq C : X \in \mathcal{F}_{1-\sigma}\}$. Then $Z(\mathcal{F}_0, \mathcal{F}_1)$ consists of a root, labeled by (C, σ), to which we attach as subtrees the Zielonka trees $Z(\mathcal{F}_0 \cap \mathcal{P}(C_i), \mathcal{F}_1 \cap \mathcal{P}(C_i))$ for $i = 0, \ldots, k$. Moreover, if the intersection of all sets on an infinite

branch of a Zielonka tree is not empty, then the intersection is added as the final point on the branch (so a Zielonka tree may be an ω-tree).

Every Muller condition over a finite set of priorities can be represented using a Zielonka tree, but this is not always possible for infinite sets C. This happens because there may be sets $D \in \mathcal{F}_\sigma$ that have subsets in $\mathcal{F}_{1-\sigma}$ but no maximal ones.

We will analyze Muller conditions $(\mathcal{F}_0, \mathcal{F}_1)$ over ω for which the tree $Z(\mathcal{F}_0, \mathcal{F}_1)$ exists, is finitely branching, and all internal nodes of the tree are labeled with co-finite sets. Slightly abusing notation, we will identify a node (X, σ) in the Zielonka tree with X, if the Muller condition is clear in the context. For such a node X, we denote its successors by $X^0, X^1, \ldots, X^k$ and, as all internal sets are co-finite, we know that $X \setminus X^i$ is finite for all i. Please note that with this notation we have implicitly fixed an ordering of the successors of each vertex.

We define the *height* of a node X in a Zielonka tree, denoted $h(X)$, as its distance to the root, starting with 0 if $\omega \in \mathcal{F}_0$ or with 1 if $\omega \in \mathcal{F}_1$. In this way the height of a node (X, σ) is even for $\sigma = 0$ and odd for $\sigma = 1$.

Example 4.28. The parity condition has a very simple Zielonka tree, namely just a Zielonka path

$$\omega \to \omega \setminus \{0\} \to \omega \setminus \{0,1\} \to \omega \setminus \{0,1,2\} \to \cdots$$

and $h(X) = \min(X)$. There is no Zielonka tree for the max-parity condition since $\omega \in \mathcal{F}_0$ has no maximal subset in $\mathcal{F}_1$ as $\mathcal{F}_1$ is not closed under unions of chains.

To define a memory structure for Muller conditions that have Zielonka trees with the properties described above we slightly deviate from the previous definition of a memory structure. In the rest of this chapter we use memory structures with *move update functions*. We define this memory in the same way as in Definition 4.2, with the only exception that this time the memory update function, update : $M \times V \times V \to M$, depends on both the start and on the end position of a move, not only on the final position as in the previous definition. The inductive definition

of the memory state is thus changed to $m(v_0) = \text{init}(v_0)$ and

$$m(v_0 \dots v_i v_{i+1}) = \text{update}(m(v_0 \dots v_i), v_i, v_{i+1}).$$

The notion of a reduction presented before generalizes to memory structures with move update functions in the following way. For a game graph $G = (V, V_0, V_1, E)$ and a memory structure $\mathfrak{M} = (M, \text{update}, \text{init})$, we again define $G \times \mathfrak{M} = (V \times M, V_0 \times M, V_1 \times M, E_{\text{update}})$, but

$$E_{\text{update}} = \{(v, m)(v', m') : (v, v') \in E \text{ and } m' = \text{update}(m, v, v')\}.$$

While this is a natural generalization and it preserves all the properties of memory reductions described previously, it does not allow to fully operate on moves because the priority function Ω is still defined on positions. For this reason, we define *games with move labeling* as games where the labeling function $\Omega : E \to C$ assigns priorities to moves instead of positions.

The notion of memory reduction applies to games with move labellings in the same way as to games with labellings of positions. Moreover, parity games with move labellings are again positionally determined. Remarkably, in the setting where multiple moves with different priorities are allowed between any two positions, even a stronger relationship between parity winning conditions and positional determinacy can be established. Not only are parity winning conditions in this setting the only positionally determined Muller winning conditions, in the same sense as in Theorem 4.9, but this statement holds for all prefix-independent winning conditions, even if these are not Muller conditions and over arbitrary sets of priorities [19]. This motivates us to define a Zielonka tree memory with move update function.

Definition 4.29. For a Muller condition $(\mathcal{F}_0, \mathcal{F}_1)$ over ω with finitely branching Zielonka tree $\mathcal{Z} = Z(\mathcal{F}_0, \mathcal{F}_1)$ we define the infinitary Zielonka-tree memory as any memory structure

$$\mathsf{ZTM}(Z(\mathcal{F}_0, \mathcal{F}_1)) = (Z, \text{update}, \text{init})$$

where Z are the nodes of $\mathcal{Z}$, $\text{init}(v) = \omega$ and the move update function satisfies the following constraint. If $\text{update}(X, v, w) = X'$ then the following conditions hold:

(1) $\Omega(w) \in X'$ and X' is a minimal node with this property, i.e. there is no successor Y of X' in $\mathcal{Z}$ for which $\Omega(w) \in Y$,

(2) if $\Omega(w) \in X$ then X' lies in the subtree $\mathcal{Z}|_X$,

(3) if $\Omega(w) \notin X$ then let Y be the minimal predecessor of X for which $\Omega(w) \in Y$ and let Y^i be the successor of Y for which $X \in \mathcal{Z}|_{Y_i}$; in this case if Y has k successors then

$$X' \in \mathcal{Z}|_{Y^{i+1 \bmod k}}.$$

Intuitively, a Zielonka tree memory assigns to every play a corresponding walk on the Zielonka tree. At each step of this walk (except for the first one) if the play is in a position v then the walk is in a node $X \ni \Omega(v)$ that has no successors containing $\Omega(v)$, as guaranteed by Condition (1) above. If the play moves from v to w and the priority $\Omega(w)$ is already contained in the current position X then the walk moves down in $\mathcal{Z}|_X$ to any minimal position containing $\Omega(w)$, as guaranteed by Condition (2). If $\Omega(w)$ is not contained in the current position X then the walk first goes up to the minimal predecessor Y containing $\Omega(w)$ and then chooses the next successor of Y and moves down to any minimal position containing $\Omega(w)$ in the subtree corresponding to that next successor. This structure of the walk can be exploited to reduce games with winning conditions that have Zielonka trees with certain properties to parity games with move labeling as follows.

Theorem 4.30. *Every game $\mathcal{G}$ with a Muller winning condition $(\mathcal{F}_0, \mathcal{F}_1)$ over ω such that $Z(\mathcal{F}_0, \mathcal{F}_1)$ is finitely branching, all its internal nodes are co-finite, and the intersection of sets on every infinite branch belongs to $\mathcal{F}_0$, reduces via* ZTM *memory to a parity game with move labeling.*

Proof. The given Muller game $\mathcal{G}$ is reduced to a parity game $\mathcal{G}'$ with the move labeling function defined by

$$\Omega'((v, X), (w, X')) = h(Y),$$

where Y is again the minimal common ancestor of X and X', the same as in the definition above.

We prove that any play $\pi = v_0 v_1 \dots$ of $\mathcal{G}$ is won by the same player as the extended play $\pi' = (v_0, X_0)(v_1, X_1) \dots$ in $\mathcal{G}'$. Let us denote by Y_i the lowest common ancestor of X_i and X_{i+1}. This notation allows us to imagine the ZTM memory corresponding to π as a walk through the Zielonka tree,

$$X_0 \to Y_0 \to X_1 \to Y_1 \to X_2 \to \dots.$$

We consider two cases.

Case 1: $\inf(\Omega'(\pi')) = \varnothing$. In this case the walk through the Zielonka tree progresses downwards and there exists a unique infinite branch of $Z(\mathcal{F}_0, \mathcal{F}_1)$ to which infinitely many Y_i belong. The set of priorities seen infinitely often in π is then the intersection of the sets on this branch. Thus, by assumption, Player 0 wins π, and by definition the same player wins π'.

Case 2: there exists a minimal priority $m = \min \inf \Omega'(\pi')$.
First, we claim that there is exactly one node Y of the Zielonka tree with $h(Y) = m$ that appears infinitely often in the sequence of nodes $Y_0 Y_1 \dots$. Indeed, let $Y_i Y_{i+1} \dots$ be the suffix of this sequence such that in π' there are no positions with priority smaller than m after step i. Let us assume that Y_k and Y_{k+l} are two different nodes with priority m that appear consecutively in $Y_i Y_{i+1} \dots$. Since

$$Y_k \to X_{k+1} \to \dots \to X_{k+l} \to Y_{k+l}$$

is a walk in the Zielonka tree connecting two different nodes with equal heights, there must be a node Y_m in this walk that is both an ancestor of Y_k and Y_{k+l}, but then $h(Y_m) < h(Y_k) = m$, which contradicts the way the suffix $Y_i Y_{i+1} \dots$ was chosen.

Let us now look at the suffix $Y_i Y_{i+1} \dots$ where $Y_i = Y$ and the only node with priority m in this suffix is Y. By definition of the update function this means that all priorities in $\Omega(v_i v_{i+1} \dots)$ are contained in Y and therefore $\inf(\pi) \subseteq Y$.

Moreover, again by Condition (3) of Definition 4.29, when visiting Y the update function always chooses the next successor and then the first successor again. Since the tree $Z(\mathcal{F}_0, \mathcal{F}_1)$ is finitely branching, each successor Y^i is chosen infinitely often, and the subtree rooted at Y^i is left infinitely many times during the walk through the Zielonka tree that corresponds to π. By definition of the update function, this means that for each Y^i we infinitely often encounter priorities in $Y \setminus Y^i$ during the play π. As $Y \setminus Y^i$ is finite, this means that there is a priority $c \in Y \setminus Y^i$ encountered infinitely often and thus $\inf(\pi) \notin Y^i$. Therefore the play π is won by the player σ for which $Y \in \mathcal{F}_\sigma$, which, by the definition of height for the Zielonka tree, is the same player who wins π' with priority m. □

Note that in the theorem above we reduce a Muller game with labels on positions to a parity games with labels on moves. Since parity games with labels on moves are positionally determined, one can use the positional strategy from the parity game to obtain a strategy σ for the original Muller game that is a function depending only on the current position in the game v and on the node of the Zielonka tree X.

Since in a ZTM memory it holds that X is the minimal node with $\Omega(v) \in X$ (except for the first step which is irrelevant), we can encode the position X using only v and the current branch of the Zielonka tree. If we denote by $[\mathcal{Z}]$ the set of all branches, i.e. maximal paths through the Zielonka tree $\mathcal{Z}$, then we can modify the strategy σ to get a strategy σ' that depends only on the branches, instead of the positions in the tree. The following consequence follows for Zielonka trees where $[Z(\mathcal{F}_0, \mathcal{F}_1)]$ is finite.

Corollary 4.31. *Every game $\mathcal{G}$ with a Muller winning condition $(\mathcal{F}_0, \mathcal{F}_1)$ over ω such that $Z(\mathcal{F}_0, \mathcal{F}_1)$ has finitely many (possibly infinite) branches, all its internal nodes are co-finite, and the intersection of sets on every*

infinite branch belongs to $\mathcal{F}_0$, is determined with a finite memory of size $|[Z(\mathcal{F}_0, \mathcal{F}_1)]|$.

Please note that this is both a generalization of positional determinacy for parity games with countably many priorities and of the classical result [78] that Muller games over a finite set of priorities are determined with finite memory equal in size to the number of leafs of $Z(\mathcal{F}_0, \mathcal{F}_1)$, which is of course the same as $|[Z(\mathcal{F}_0, \mathcal{F}_1)]|$ for finite trees.

5 Counting Quantifiers on Automatic Structures

In chapter 2 we asked how first-order logic can be extended and analyzed using infinitary logic, which lead us to the regular game quantifier and to clarifying the connection to games.

In this chapter we consider extensions of first-order logic in another direction, by generalized unary quantifiers. As usual, we require these extensions to preserve regularity on automatic structures. It turns out that the only generalized unary quantifiers with this property are the counting quantifiers:

- the *modulo counting quantifiers* "there exist k mod m many",
- the *infinity quantifier* "there exist infinitely many", and
- the *uncountability quantifier* "there exist uncountably many".

While it is known that all counting quantifiers indeed preserve regularity over finite-word automatic structures, and even over injectively presented ω-automatic structures, this was open for general ω-automatic structures. Our proof [7] uses ω-semigroups and leads to an additional corollary that all countable ω-automatic structures have injective presentations. It follows that countable ω-automatic structures have automatic presentations over finite words, which answers a question of Blumensath [8].

5.1 Generalized quantifiers preserving regularity

To extend first-order logic with additional quantifiers it is useful to have an abstract definition of a *generalized quantifier*. We borrow the definition given by Lindström [51].

Definition 5.1. A generalized quantifier Q over a relational signature $\tau = \{R_1, \ldots, R_k\}$ is a class of structures with signature τ that is closed under isomorphism. Let $\mathfrak{A}$ be a structure and $\varphi_1(\overline{x}_1, \overline{z}), \ldots, \varphi_k(\overline{x}_k, \overline{z})$ formulas over the signature $\sigma(\mathfrak{A})$ possibly different from τ, such that $|\overline{x}_i| = r_i$, i.e. the length of the vector $\overline{x}_i$ is the same as the arity of R_i. In first-order logic extended with the quantifier Q we allow to write formulas of the form $Q\overline{x}_1 \ldots \overline{x}_k(\varphi_1, \ldots, \varphi_k)$ and define their semantics in the following way. If θ maps $\overline{z}$ to the tuple $\overline{a}$ of elements of $\mathfrak{A}$ then

$$\mathfrak{A}, \theta \vDash Q\overline{x}_1 \ldots \overline{x}_k(\varphi_1, \ldots, \varphi_k) \Leftrightarrow (A, \varphi_1^{\mathfrak{A}}(-, \overline{a}), \ldots, \varphi_k^{\mathfrak{A}}(-, \overline{a})) \in Q,$$

where by $\varphi_i^{\mathfrak{A}}(-, \overline{a})$ we denote the relation satisfied by exactly those tuples $\overline{b}$ for which $\varphi_i^{\mathfrak{A}}(\overline{b}, \overline{a})$ holds. The arity of the quantifier Q is the maximum of the lengths $|\overline{x}_i|$, so a *unary quantifier* is one where each of the vectors $\overline{x}_i$ is just a single variable.

To illustrate this definition observe that the classical quantifier $\exists$ is given by $\{(A, X) : X \neq \varnothing\}$ and $\forall$ is given by $\{(A, X) : X = A\}$. The quantifiers "there exist infinitely many" or "there exist k mod m many" can be represented in a similar way, but we give the standard definition.

The extension of first-order logic with counting quantifiers, denoted $\mathsf{FO}[\mathsf{C}]$, allows to write all quantifiers of the following form:

- $\exists^{(r \bmod m)} x\ \varphi$ meaning that the number of x satisfying φ is finite and is congruent to r mod m,
- $\exists^{\infty} x\ \varphi$ meaning that there are infinitely many x satisfying φ,
- $\exists^{\leq \aleph_0} x\ \varphi$ and $\exists^{> \aleph_0} x\ \varphi$ meaning that the cardinality of the set of all x satisfying φ is countable, or uncountable, respectively.

The logic $\mathsf{FO}[\mathsf{C}]$ has intimate relation to quantifiers that preserve regularity. To define this relation we first need to say that a generalized quantifier Q preserves (ω-)regularity if for every (ω-)automatic presentation $\mathfrak{d}, f$ of a structure $\mathfrak{A}$ every formula

$$\psi(\overline{z}) = Q\overline{x}_1 \ldots \overline{x}_k\big(\varphi_1(\overline{x}_1, \overline{z}) \ldots \varphi_k(\overline{x}_1, \overline{z})\big)$$

defines a relation $\psi^{\mathfrak{A}}$ that is (ω-)regular in the presentation $\mathfrak{d}, f$, i.e. such that $f^{-1}(\psi^{\mathfrak{A}})$ is (ω-)regular.

Moreover, we say that a quantifier Q over a signature τ is definable in $\mathsf{FO}[\mathsf{C}]$ (or any other extension of FO) if there exists a formula $\varphi_Q \in \mathsf{FO}[\mathsf{C}]$ over the same signature τ such that $Q = \{\mathfrak{A} : \mathfrak{A} \models \varphi\}$. We can now state the result that shows the relationship between $\mathsf{FO}[\mathsf{C}]$ and regularity-preserving quantifiers: every unary quantifier that preserves (ω-)regularity is definable in $\mathsf{FO}[\mathsf{C}]$. This result is proved in [70] for regularity preserving quantifiers and the proof extends to ω-regularity preserving ones.

The remaining question is whether every quantifier in $\mathsf{FO}[\mathsf{C}]$ preserves regularity, and whether it does so in an effective way. For finite-word automatic structures the basic Theorem 1.5 can be extended to $\mathsf{FO}[\mathsf{C}]$ as follows.

Theorem 5.2 (Cf. [40, 46, 11])**.**

- *There is an effective procedure that given an automatic presentation $\mathfrak{d}, f$ of a structure $\mathfrak{A}$, and given an $\mathsf{FO}[\mathsf{C}]$ formula $\varphi(\overline{x})$ defining a k-ary relation R over $\mathfrak{A}$, constructs a k-tape synchronous automaton recognizing $f^{-1}(R)$.*
- *The $\mathsf{FO}[\mathsf{C}]$-theory of every automatic structure is decidable.*
- *The class of automatic structures is closed under $\mathsf{FO}[\mathsf{C}]$-interpretations.*

It has been observed that Theorem 5.2 can be extended to *injective* ω-automatic presentations [48, 50]. Moreover, Kuske and Lohrey show that the cardinality of any set definable in $\mathsf{FO}[\mathsf{C}]$ is either countable or equal to that of the continuum. In the next section we work to extend this result to all, not necessarily injective automatic structures.

5.2 Defining uncountability using equal ends

We characterize when there exist countably many words x satisfying a given formula *with parameters* $\varphi(x, \overline{z})$ in some ω-automatic structure $\mathfrak{A}$.

The characterization is first-order expressible in an ω-automatic extension of $\mathfrak{A}$ by the equal ends relation $\sim_e$ and the quantifier rank of the resulting formula depends on a constant C, which itself depends on φ and on the given presentation of $\mathfrak{A}$.

Let us fix an ω-automatic presentation $\mathfrak{d}$ of a structure $\mathfrak{A}$ with congruence $\approx$, and a first-order formula $\varphi(x, \overline{z})$ in the language of $\mathfrak{A}$ with x and $\overline{z}$ as free variables.

Proposition 5.3. *There is a constant C, computable from the presentation $\mathfrak{d}$, so that for all tuples $\overline{z}$ of infinite words the following are equivalent:*

(i) $\varphi(-, \overline{z})$ is satisfiable and $\approx$ restricted to the domain $\varphi(-, \overline{z})$ has countably many equivalence classes,

(ii) there exist C-many words $x_1, \ldots, x_C$, each satisfying $\varphi(-, \overline{z})$, so that every x satisfying $\varphi(-, \overline{z})$ is $\approx$-equivalent to some $y \sim_e x_i$; formally, the structure $(\mathfrak{A}, \approx, \sim_e)$ models the sentence

$$\forall \overline{z} \Bigg(\exists^{\leq \aleph_0} w \; \varphi(w, \overline{z}) \longleftrightarrow \exists x_1 \ldots x_C \Bigg(\bigwedge_i \varphi(x_i, \overline{z}) \wedge \forall x \left(\varphi(x, \overline{z}) \to \exists y \; (y \approx x \wedge \bigvee_i y \sim_e x_i) \right) \Bigg) \Bigg).$$

Proof. Suppose $\mathfrak{d}$, $\mathfrak{A}$, and φ are given. Define C to be c^2, where c is the size of the largest ω-semigroup corresponding to any of the given automata from the presentation $\mathfrak{d}$ or corresponding to φ. We fix the parameters $\overline{z}$ and let $\approx$ denote the equivalence relation $\approx$ restricted to the domain of $\varphi(-, \overline{z})$.

(ii) $\Rightarrow$ (i): Condition *(ii)* and the fact that every $\sim_e$-class is countable imply that all words satisfying $\varphi(-, \overline{z})$ are contained in a countable number of $\approx$-classes.

(i) $\Rightarrow$ *(ii)*: The negation of Condition *(ii)* says that given $D < C$ many words $x_1, \ldots, x_D$, each satisfying $\varphi(-, \overline{z})$, there exists a word x_{D+1} also satisfying $\varphi(-, \overline{z})$ whose $\approx$-class does not meet any of the $\sim_e$-classes of the x_i for $i \leq D$.

Thus we can inductively define words $x_1, \ldots, x_C$, each satisfying the formula $\varphi(-, \overline{z})$, and such that for $1 \leq i < j \leq C$ the $\approx$-class of x_j does not meet the $\sim_e$-class of x_i. In particular, the x_is are pairwise non-equivalent with respect to $\sim_e$.

The plan is to produce uncountably many pairwise non-$\approx$ words that satisfy $\varphi(-, \overline{z})$. In the first 'Ramsey step', similar to what is done in [50], we find two words from the given C many, say $x_1, x_2 \in \Sigma^*$, and a factorization $H \subset \mathbb{N}$ so that both words behave the same way along the factored subwords with respect to the $\approx$- and φ-semigroups. In the second 'Coarsening step' we identify a technical property of finite semigroups recognizing transitive relations. This allows us to produce an altered factorization G and new, well-behaving words y_1, y_2. In the final step, the new words are 'shuffled along G' to produce continuum many pairwise non-$\approx$ words, each satisfying $\varphi(-, \overline{z})$.

5.2.1 Ramsey step

This step effectively allows us to discard the parameters $\overline{z}$. Before we use Ramsey's theorem, we introduce a convenient notation to talk about factorizations of words.

Definition 5.4. Let $A = a_1 < a_2 < \ldots$ be any infinite subset of $\mathbb{N}$ and $h : \Sigma^* \to S$ be a morphism into a finite semigroup S. For an ω-word $\alpha \in \Sigma^\omega$, and element $e \in S$, say that *A is an h, e-homogeneous factorization of α* if for all $n \in \mathbb{N}^+$, $h\big(\alpha[a_n, a_{n+1})\big) = e$.

Observe the following facts.

- If A is an h, s-homogeneous factorization of α and $k \in \mathbb{N}^+$ then the set $\{a_{k \cdot i}\}_{i \in \mathbb{N}^+}$ is an h, s^k-homogeneous factorization of α.

– If A is an h, e-homogeneous factorization of α and e is idempotent, then every infinite $B \subset A$ is also an h, e-homogeneous factorization of α.

In the following we write w^{φ} and $w^{\approx}$ to denote the image of w under the semigroup morphism into the finite semigroup associated to φ and $\approx$, respectively, as determined by the presentation. Accordingly, we will speak of e.g. φ, s_i-homogeneous factorizations.

Let us now color every $\{n, m\} \in [\mathbb{N}]^2$ with $n < m$ by the tuple of ω-semigroup elements

$$\Big(\big(\otimes (x_i, \overline{z})[n, m)^{\varphi} \big)_{0 \leq i \leq C} , \big(\otimes (x_i, x_j)[n, m)^{\approx} \big)_{0 \leq i \leq j \leq C} \Big).$$

By Ramsey's theorem there exists an infinite $H \subset \mathbb{N}$ and a tuple of ω-semigroup elements

$$\big((s_i)_{1 \leq i \leq C}, (t_{(i,j)})_{1 \leq i \leq j \leq C}\big)$$

so that for all $0 \leq i \leq j \leq C$,

– H is a φ, s_i-homogeneous factorization of the word $\otimes(x_i, \overline{z})$,

– H is a $\approx, t_{(i,j)}$-homogeneous factorization of the word $\otimes(x_i, x_j)$.

Note that by virtue of additivity of our coloring and Ramsey's theorem each of the s_i and $t_{(i,j)}$ above are idempotents. Since there are at most c-many s_is and c-many $t_{(i,i)}$s, there are at most c^2 many pairs $(s_i, t_{(i,i)})$ and so there must be two indices, we may suppose 1 and 2, with $s_1 = s_2$ and $t_{(1,1)} = t_{(2,2)}$.

5.2.2 Coarsening step

For technical reasons we now refine H and alter x_1, x_2 so that the semigroup elements have certain additional properties.

To start with, using the fact that $x_1 \not\sim_e x_2$ and the facts we observed on homogeneous factorizations, we assume without loss of generality that H is coarse enough so that $x_1[h_n, h_{n+1}) \neq x_2[h_n, h_{n+1})$ for all $n \in \mathbb{N}$.

Lemma 5.5. *There exists a subset $G \subset H$, listed as $g_1 < g_2 < \ldots$, and ω-words y_1, y_2 with the following properties:*

(1) The words y_1 and y_2 are neither $\approx$-equivalent nor $\sim_e$-equivalent, and each satisfies $\varphi(-, \overline{z})$.

(2) There exists an idempotent φ-semigroup element s such that G is a φ, s-homogeneous factorization for each of $\otimes(y_1, \overline{z})$ and $\otimes(y_2, \overline{z})$.

(3) There exist idempotent $\approx$-semigroup elements $t, t^{\uparrow}, t^{\downarrow}$ so that for $y_j \in \{y_1, y_2\}$

- *both $t^{\uparrow}$ and $t^{\downarrow}$ absorb t*
- *$\otimes(y_j, y_j)[0, g_1)^{\approx}$ absorbs t*
- *G is an $\approx, t$-homogeneous factorization of $\otimes(y_j, y_j)$*
- *G is an $\approx, t^{\uparrow}$-homogeneous factorization of $\otimes(y_1, y_2)$*
- *G is an $\approx, t^{\downarrow}$-homogeneous factorization of $\otimes(y_2, y_1)$.*

Proof. Define ω-words $y_1 := x_2[0, h_2)x_1[h_2, \infty)$, and y_2 by

$$\begin{aligned} y_2[0, h_2) &:= x_2[0, h_2) \text{ and} \\ y_2[h_{2n}, h_{2n+2}) &:= x_2[h_{2n}, h_{2n+1})x_1[h_{2n+1}, h_{2n+2}) \text{ for } n > 0. \end{aligned}$$

Item 1. Clearly, $y_1 \not\sim_e y_2$ and each $y_j \in \{y_1, y_2\}$ satisfies $\varphi(y_j, \overline{z})$ since by homogeneity and $s_1 = s_2$

$$\begin{aligned} \otimes(y_1, \overline{z})^{\varphi} &= \otimes(x_2, \overline{z})[0, h_2)^{\varphi} s_1^{\omega} \\ &= \otimes(x_2, \overline{z})[0, h_2)^{\varphi} s_2^{\omega} \\ &= \otimes(x_2, \overline{z})^{\varphi}, \end{aligned}$$

and similarly

$$\begin{aligned} \otimes(y_2, \overline{z})^{\varphi} &= \otimes(x_2, \overline{z})[0, h_2)^{\varphi} (s_2 s_1)^{\omega} \\ &= \otimes(x_2, \overline{z})[0, h_2)^{\varphi} s_2^{\omega} \\ &= \otimes(x_2, \overline{z})^{\varphi}. \end{aligned}$$

Next we check that $y_1 \not\approx y_2$.

$$\begin{aligned}
\otimes(y_1,y_2)^{\approx} &= \pi_{\approx}\Big(\otimes(x_2,x_2)[0,h_2)^{\approx}, \\
&\qquad \big(\otimes(x_1,x_2)[h_{2n},h_{2n+1})^{\approx}, \\
&\qquad\quad \otimes(x_1,x_1)[h_{2n+1},h_{2n+2})^{\approx} \big)_{n\in\mathbb{N}^+} \Big) \\
&= \otimes(x_2,x_2)[0,h_1)^{\approx}\, t_{(2,2)}\, (t_{(1,2)}t_{(1,1)})^{\omega} \\
&= \otimes(x_2,x_2)[0,h_1)^{\approx}\, t_{(2,2)}t_{(2,2)}\, (t_{(1,2)}t_{(1,1)})^{\omega} \\
&= \otimes(x_2,x_2)[0,h_1)^{\approx}\, t_{(2,2)}t_{(2,2)}\, (t_{(1,2)}t_{(2,2)})^{\omega} \\
&= \otimes(x_2,x_2)[0,h_1)^{\approx}\, t_{(2,2)}\, (t_{(2,2)}t_{(1,2)})^{\omega} \\
&= \pi_{\approx}\Big(\otimes(x_2,x_2)[0,h_2)^{\approx}, \\
&\qquad \big(\otimes(x_2,x_2)[h_{2n},h_{2n+1})^{\approx}, \\
&\qquad\quad \otimes(x_1,x_2)[h_{2n+1},h_{2n+2})^{\approx} \big)_{n\in\mathbb{N}^+} \Big) \\
&= \otimes(y_2,x_2)^{\approx}
\end{aligned}$$

Thus, if $y_1 \approx y_2$ then also $y_2 \approx x_2$ and so *by transitivity* $y_1 \approx x_2$. But since $y_1 \sim_{\mathrm{e}} x_1$, the $\approx$-class of x_2 meets the $\sim_{\mathrm{e}}$-class of x_1, contradicting the initial choice of the x_is.

Items 2 and 3. Define intermediate semigroup elements $q := s_1$, $r := t_{(1,1)}$, $r^{\uparrow} := t_{(1,2)}t_{(1,1)}$ and $r^{\downarrow} := t_{(2,1)}t_{(1,1)}$. Then

1. both $r^{\uparrow}$ and $r^{\downarrow}$ absorb r, since $t_{(1,1)}$ is idempotent,
2. $\otimes(y_j,y_j)[0,h_2)^{\approx} = \otimes(y_j,y_j)[0,h_1)^{\approx}t_{(2,2)}$ and thus absorbs r (for $y_j \in \{y_1,y_2\}$).

In this notation, for all $i \in \mathbb{N}^+$ and $y_j \in \{y_1,y_2\}$,

- $\otimes(y_j,\overline{z})[h_{2i},h_{2i+2})^{\varphi}$ is $qq = q$,
- $\otimes(y_j,y_j)[h_{2i},h_{2i+2})^{\approx}$ is $rr = r$,
- $\otimes(y_1,y_2)[h_{2i},h_{2i+2})^{\approx}$ is $t_{(1,2)}t_{(1,1)} = r^{\uparrow}$,

– $\otimes(y_2, y_1)[h_{2i}, h_{2i+2})^{\approx}$ is $t_{(2,1)}t_{(1,1)} = r^{\downarrow}$.

Finally, define the set $G := \{h_{2ki}\}_{i>1}$, i.e. $g_i = h_{2k(i+1)}$, and the semigroup elements $t := r^k$, $t^{\uparrow} := (r^{\uparrow})^k$, $t^{\downarrow} := (r^{\downarrow})^k$ and $s := q^k$. The extra multiple of k (defined as the product of the exponents of the semigroups for $\sim_e$ and $\approx$) ensures all these semigroup elements (in particular $t^{\uparrow}$ and $t^{\downarrow}$) are idempotent. We now verify the absorption properties:

$$t^{\uparrow}t = r^{\uparrow k}r^k = r^{\uparrow k} = t^{\uparrow} \qquad \text{because } r^{\uparrow} \text{ absorbs } r.$$

Similarly, $t^{\downarrow}t$ absorbs t. Further, since $g_1 = h_{4k}$, we have

$$\begin{aligned} \otimes(y_j, y_j)[0, g_1)^{\approx} &= \otimes(y_j, y_j)[0, h_2)^{\approx} \otimes (y_j, y_j)[h_2, h_{4k})^{\approx} \\ &= \otimes(y_j, y_j)[0, h_2)^{\approx} r^{4k-2} \\ &= \otimes(y_j, y_j)[0, h_2)^{\approx} r^{3k-2}t \end{aligned}$$

and thus absorbs t.

Finally, we verify the homogeneity properties. Observe that G is an $\approx, t^{\downarrow}$-homogeneous factorization of $\otimes(y_2, y_1)$ since for $i \in \mathbb{N}^+$

$$\begin{aligned} \otimes(y_2, y_1)[g_i, g_{i+1})^{\approx} &= \otimes(y_2, y_1)[h_{2k(i+1)}, h_{2k(i+2)})^{\approx} \\ &= (r^{\downarrow})^k = t^{\downarrow}. \end{aligned}$$

The other cases are similar. □

5.2.3 Shuffling step

We continue the proof of Proposition 5.3 by shuffling the words y_1 and y_2 along G resulting in continuum many pairwise distinct words that are pairwise not $\approx$-equivalent, each satisfying $\varphi(-, \overline{z})$. To this end, we define for $S \subset \mathbb{N}^+$ the characteristic word χ_S by

$$\begin{aligned} \chi_S[0, g_1) &:= y_2[0, g_1)\text{, and} \\ \chi_S[g_n, g_{n+1}) &:= \begin{cases} y_2[g_n, g_{n+1}) & \text{if } n \in S \\ y_1[g_n, g_{n+1}) & \text{otherwise} \end{cases} \end{aligned}$$

First observe that $\mathfrak{A} \vDash \varphi(\chi_S, \overline{z})$. Indeed, by item (2) of Lemma 5.5

$$\begin{aligned} \otimes(\chi_S, \overline{z})^\varphi &= \otimes(y_2, \overline{z})[0, g_1)^\varphi s^\omega \\ &= \otimes(y_2, \overline{z})^\varphi \end{aligned}$$

and $\mathfrak{A} \vDash \varphi(y_2, \overline{z})$ by item (1) of Lemma 5.5. Moreover, for $S \not\sim_e T$ the construction gives that $\chi_S \not\sim_e \chi_T$. This is due to our initial choice of $x_1 \not\sim_e x_2$ and the assumption that the factorization $(h_n)_{n\in\mathbb{N}}$ is coarse enough so that $x_1[h_n, h_{n+1}) \neq x_2[h_n, h_{n+1})$ and thus also $y_1[g_n, g_{n+1}) \neq y_2[g_n, g_{n+1})$ for all n.

The following two lemmas establish that if both $S \setminus T$ and $T \setminus S$ are infinite then $\chi_S \not\approx \chi_T$. We denote by $x_{\circ\bullet}$ the word $\chi_{2\mathbb{N}^+}$ and by $x_{\bullet\circ}$ the word $\chi_{2\mathbb{N}^+-1}$, and we write p for $\otimes(y_2, y_2)[0, g_1)^\approx$.

Lemma 5.6. *For all S, T such that both $S \setminus T$ and $T \setminus S$ are infinite*

$$\otimes(\chi_S, \chi_T)^\approx = \begin{cases} \otimes(x_{\circ\bullet}, x_{\bullet\circ})^\approx & \text{or} \\ \otimes(x_{\bullet\circ}, x_{\circ\bullet})^\approx \end{cases}$$

Proof. Define semigroup-elements p_n for $n \in \mathbb{N}$ by

$$p_n := \begin{cases} t^\downarrow & \text{if } n \in S \setminus T \\ t^\uparrow & \text{if } n \in T \setminus S \\ t & \text{otherwise} \end{cases}$$

Let m be the smallest number in $S \vartriangle T$. Suppose that $m \in S \setminus T$. Because both $t^\uparrow$ and $t^\downarrow$ are idempotent and since t is absorbed by both p, $t^\uparrow$ and $t^\downarrow$, and both $t^\uparrow$ and $t^\downarrow$ appear infinitely often, we have

$$\begin{aligned} \otimes(\chi_S, \chi_T)^\approx &= \pi_\approx(p, (p_n)_{n\in\mathbb{N}}) = p(t^\downarrow t^\uparrow)^\omega \\ &= \otimes(x_{\bullet\circ}, x_{\circ\bullet})^\approx. \end{aligned}$$

The case that $m \in T \setminus S$ similarly results in $\otimes(x_{\circ\bullet}, x_{\bullet\circ})^\approx$. □

Lemma 5.7. $x_{\circ\bullet} \not\approx x_{\bullet\circ}$.

Proof. Define an intermediate word $x_{\circ\bullet\circ\circ} := \chi_{4\mathbb{N}^+-2}$. By computations similar to the above we find that

$$\begin{aligned}\otimes(x_{\bullet\circ}, x_{\circ\bullet\circ\circ})^{\approx} &= p(t^{\downarrow}t^{\uparrow}t^{\downarrow}t)^{\omega} = p(t^{\downarrow}t^{\uparrow}t^{\downarrow})^{\omega} = p(t^{\downarrow}t^{\uparrow})^{\omega} \\ &= \otimes(x_{\bullet\circ}, x_{\circ\bullet})^{\approx}\end{aligned}$$

and

$$\begin{aligned}\otimes(x_{\circ\bullet}, x_{\circ\bullet\circ\circ})^{\approx} &= p(tttt^{\downarrow})^{\omega} = p(t^{\downarrow})^{\omega} \\ &= \otimes(y_2, y_1)^{\approx}.\end{aligned}$$

Therefore, if $x_{\bullet\circ} \approx x_{\circ\bullet}$ then also $x_{\bullet\circ} \approx x_{\circ\bullet\circ\circ}$ and so *by symmetry and by transitivity* $x_{\circ\bullet} \approx x_{\circ\bullet\circ\circ}$. But in this case also $y_2 \approx y_1$, contradicting item (1) of Lemma 5.5. □

We are now able to complete the proof of Proposition 5.3. There are continuum many classes in $\mathcal{P}(\mathbb{N})/\sim_e$, thus there is a continuum of pairwise non-$\sim_e$-equivalent sets S. To construct sets with pairwise infinite differences, we define for a set $S \subseteq \mathbb{N}$ the swap set

$$\widehat{S} = \{2n+1 : n \in S\} \cup \{2n+2 : n \notin S\}.$$

Observe that if $S \not\sim_e T$ then both $\widehat{S} \setminus \widehat{T}$ and $\widehat{T} \setminus \widehat{S}$ are infinite. Therefore taking the words $\chi_{\widehat{S}}$ for the continuum of pairwise non-$\sim_e$-equivalent sets S yields a continuum of non-$\approx$-equivalent words, each satisfying $\varphi(-, \overline{z})$. □

5.3 FO[C] over ω-automatic structures

Using the results about countability of the previous section, we are finally able to extend Theorem 5.2 to ω-automatic structures.

Theorem 5.8. *The statements of Theorem 5.2 hold true for* FO[C] *over all (not necessarily injective) ω-automatic presentations.*

Proof. We prove the first item, i.e. give the procedure for constructing automata for formulas, from which the rest of the theorem follows immediately. We inductively eliminate occurrences of cardinality and modulo quantifiers in the following way.

The countability quantifier $\exists^{\leq \aleph_0}$ and uncountability quantifier $\exists^{>\aleph_0}$ can be eliminated (in an extension of the presentation by $\sim_e$) by the formula given in Proposition 5.3.

For the remaining quantifiers we further expand the presentation with the ω-regular relations

- $\pi(a, b, c)$ saying that $a \sim_e b \sim_e c$ and the last position where a differs from c is no larger than the last position where b differs from c, and
- $\lambda(a, b, c)$ saying that $\pi(a, b, c)$ and $\pi(b, a, c)$ and that the word $a[0, k]$ is lexicographically smaller than the word $b[0, k]$, where k is the common last position where a and b differ from c.

Now $\exists^{<\infty} x\ \varphi(x, \overline{z})$ is equivalent to

$$\exists x_1 \dots x_C\ \Psi(x_1, \dots, x_C, \overline{z})$$

where Ψ expresses that $x_1, \dots x_C$ satisfy $\varphi(-, \overline{z})$ and there exists a position, say $k \in \mathbb{N}$, so that every $\approx$-class contains a word satisfying $\varphi(-, \overline{z})$ that coincides with one of the x_i from position k onwards. This additional condition can be expressed by

$$\exists y_1 \dots y_C \forall x \exists y \left(\varphi(x, \overline{z}) \to x \approx y \wedge \bigvee_i \pi(y, y_i, x_i) \right).$$

Consequently, $\exists^{(r \bmod m)} x\ .\ \varphi(x, \overline{z})$ can be eliminated since we can pick out unique representatives of the $\approx$-classes. We write $i(w)$ for the smallest index i for which $w \sim_e x_i$. The representatives are those x that satisfy the following properties for every $y \neq x$ in the same $\approx$-class as x.

- Either the index $i(x) < i(y)$, or

– the index $i(x) = i(y)$ and $\lambda(x, y, x_{i(x)})$ holds.

Now we can apply the construction of [50] or [48] for elimination of the $\exists^{(r \bmod m)}$ quantifier. □

5.4 Presentations of countable ω-automatic structures

As a corollary of Proposition 5.3 we obtain that for every ω-regular equivalence relation with countably many classes a set of unique representatives is definable.

Corollary 5.9. *Let $\approx$ be an ω-automatic equivalence relation on Σ^ω. There is a constant C, depending on the presentation, so that the following are equivalent:*

(1) $\approx$ has countably many equivalence classes,

(2) there exist C many $\sim_e$-classes so that every $\approx$-class has a non-empty intersection with at least one of these $\sim_e$-classes.

If one of these conditions holds, then there exists an ω-regular set of representatives of $\approx$. Moreover, an automaton for this set can be effectively constructed given an automaton for $\approx$.

Proof. The first two items are simply a specialization of Proposition 5.3. We construct the ω-regular set of representatives as follows.

Write A for the domain of $\approx$ and consider the formula $\psi(x_1, \ldots, x_C)$ with free variables $x_1, \ldots, x_C$:

$$\bigwedge_i x_i \in A \ \wedge \ \forall x \in A \, \exists y \, (y \approx x \wedge \bigvee_i y \sim_e x_i)$$

The relation defined by ψ is ω-regular since it is a first order formula over ω-regular relations. By assumption it is non-empty, and therefore it contains an ultimately periodic word of the form $\otimes(a_1, \ldots, a_C)$. Each of these a_is is thus ultimately periodic, and we write $a_i = v_i(u_i)^\omega$.

By definition of ψ, every word has now an $\approx$-representative in $B = \bigcup_i \Sigma^*(u_i)^\omega$. It remains to prune B to select unique representatives for each $\approx$-class.

It is easy to construct an ω-regular well-founded linear order on B. For every $w \in B$, let $p(w) \in \Sigma^*$ be the length-lexicographically smallest word such that w has period $p(w)$. Also let $t(w) \in \Sigma^*$ be the length-lexicographically smallest word so that $w = t(w) \cdot p(w)^\omega$. Define an order $\prec$ on B by $w \prec w'$ if $p(w)$ is length-lexicographically smaller than $p(w')$, or otherwise if $p(w) = p(w')$ and $t(w)$ is length-lexicographically smaller than $t(w')$. The ordering $\prec$ is ω-regular since it is FO-definable in terms of ω-regular relations. Finally, the required set of representatives may be defined as the set of $\prec$-minimal elements of every $\approx$-class. An automaton for this set can be constructed from an automaton for $\approx$ as all the steps we made used definable relations. □

Corollary 5.9 immediately yields an *injective* ω-automatic presentation from a given ω-automatic presentation. This is especially interesting together with the following proposition by which countable injective ω-automatic presentations can be transformed to automatic ones.

Proposition 5.10. *([8, Theorem 5.32]) Let $\mathfrak{d}$ be an injective ω-automatic presentation of a countable structure $\mathcal{A}$. Then, an (injective) automatic presentation $\mathfrak{d}'$ of $\mathcal{A}$ can be effectively constructed.*

Combining Proposition 5.10 and Corollary 5.9 we are able to answer affirmatively a question of Blumensath [8] and conclude that every countable ω-automatic structure is already automatic.

Corollary 5.11. *A countable structure is ω-automatic if and only if it is automatic. Transforming a presentation of one type into the other can be done effectively.*

Remarkably, the existence of injective presentations can not be extended from countable to arbitrary ω-automatic structures. Consider a disjoint sum of the boolean algebra of sets of natural numbers $\mathfrak{B} = (\mathcal{P}(\mathbb{N}), \cup, \cap, ^C)$ and the uncountable atomless boolean algebra $\mathfrak{B}/\sim_e$,

where sets with finite symmetric difference are identified. Let us define the structure $\mathfrak{A} = (\mathfrak{B} \sqcup (\mathfrak{B}/\sim_e), B_1, B_2, f)$ which extends the disjoint sum $\mathfrak{B} \sqcup (\mathfrak{B}/\sim_e)$ with two predicates denoting the universes of the two components of the sum and a function that takes elements of $\mathfrak{B}$ to their corresponding classes in the other component of the disjoint sum, i.e. $f(B) = [B]_{\sim_e}$ for $B \in \mathfrak{B}$.

Observe that there is an ω-automatic presentation of $\mathfrak{A}$. Elements of $\mathfrak{A}$ are represented as ω-words over $\{0, 1\}$ with the first bit indicating whether the word represents an element of $\mathfrak{B}$ or of $\mathfrak{B}/\sim_e$ and the other bits listing which numbers belong to the represented subset. Equality is defined as equality of words for words staring with 1, i.e. representing elements of $\mathfrak{B}$, and as $\sim_e$ for words starting with 0. Boolean operations can be represented by automata in the standard way and the function f must only check that the $\sim_e$-classes of the components coincide, which can be done by the $\sim_e$ automaton ignoring the first bit.

The fact that there is no injective ω-automatic presentation of the structure $\mathfrak{A}$ was recently shown by Hjörth, Khoussainov, Montalban and Nies [39]. The proof is based on the topological observation that certain morphism between $\mathfrak{B}/\sim_e$ and $\mathfrak{B}$ can not be Borel, which would be contradicted by an injective presentation of $\mathfrak{A}$. It follows that decidability of $\mathsf{FO}[\mathsf{C}]$ on the structure $\mathfrak{A}$, which is a consequence of Theorem 5.8, can not be deduced from the previous Theorem 5.2, and so it shows that Theorem 5.8 is a strong generalization of the previous result.

6 Cardinality Quantifiers in MSO on Trees and Linear Orders

The intimate connection between first-order logic on automatic structures and MSO on $(\omega, <)$ can be used to define generalized-automatic structures, which we introduced in section 1.5 as the ones that are MSO-to-FO interpretable in a tree. It is therefore a natural extension of the previous work to ask whether counting quantifiers preserve regularity on such generalized-automatic structures.

In this chapter we give a partial positive answer to this problem, as we investigate the cardinality quantifiers on injectively presented generalized-automatic structures. The restriction to injective presentations allows us to work directly with MSO and therefore, instead of extending first-order logic, we study the extension of MSO with the following quantifiers:

- the second-order infinity quantifier
  "there exist infinitely many sets X for which $\varphi(X)$ holds",
- and the second-order uncountability quantifier
  "there exist uncountably many sets X for which $\varphi(X)$ holds".

When working directly with MSO we do not use automata or semigroups as previously, but rely on the composition method instead, which allows us to consider trees labeled with arbitrary predicates. First, we prove that on arbitrary countable structures the second-order infinity quantifier can be eliminated from MSO using the predicate that expresses infiniteness of a set, which is definable in MSO on finitely branching trees and linear orders. Further, we show that on arbitrarily labeled trees and on arbitrary countable linear orders the second-order uncountability quantifier can be eliminated as well, i.e. it can be expressed using pure MSO formulas. This result, obtained together with Vince Bárány

and Alexander Rabinovich, is a generalization of a result of Niwiński [60] for unlabeled trees and a first step in studying extensions of FO on generalized-automatic structures.

6.1 Infinity quantifier

We consider the extension of MSO with cardinality quantifiers $\exists^\kappa$, so that the meaning of a formula $\exists^\kappa X\, \varphi$ is that there are *at least* κ many sets X satisfying φ, for each cardinal $\kappa \in \{\aleph_0, \aleph_1, 2^{\aleph_0}\}$. For $\exists^{\aleph_0}$ we sometimes write $\exists^\infty$. It will follow from our proofs that for MSO-definable families of subsets of countable linear orders and trees the continuum hypothesis holds, i.e. $\exists^{\aleph_1}$ and $\exists^{2^{\aleph_0}}$ are equivalent.

To start with, let us show how to eliminate the infinity quantifier $\exists^\infty$ from monadic second-order formulas over any structure where infiniteness of a set is expressible in MSO. This yields a uniform elimination of the infinity quantifier from MSO formulas over the binary tree and countable linear orders, as finiteness of a set is expressible over these structures. Over the binary tree, infiniteness of a set X is, by König's Lemma, equivalent to the existence of an infinite path every element of which is a prefix of some node in X. Over a countable linear order, by Ramsey's theorem, infiniteness of X is equivalent to the existence of a subset $Y \subseteq X$ of type ω or ω^*, i.e. a set with every element having a direct successor (or predecessor) and with at most one limit point.

Proposition 6.1. *Over all structures, the infinity quantifier $\exists^\infty$ can be effectively eliminated from monadic second-order formulas using the "set X is finite" predicate.*

To prove this proposition, we show how occurrences of the infinity quantifier can be eliminated from formulas inductively, according to the following claim.

Claim 6.2. *An* MSO *formula $\varphi(X, \overline{Y})$ is satisfied on a structure $\mathfrak{A}$ for fixed parameters $\overline{Y}$ by finitely many sets X if and only if there is a finite set Z*

such that any two distinct sets both satisfying φ differ on Z, i.e.

$$\exists \textit{ finite } Z\ \forall X_1 X_2 \left((\varphi(X_1, \overline{Y}) \wedge \varphi(X_2, \overline{Y}) \wedge X_1 \neq X_2) \rightarrow \exists z \in Z((z \in X_1 \wedge z \notin X_2) \vee (z \in X_2 \wedge z \notin X_1))\right).$$

Proof.

($\Rightarrow$) Let $X_1, \ldots, X_k$ be the sets that satisfy $\varphi(X_i, \overline{Y})$. For every pair of distinct sets X_i, X_j choose an element $z_{i,j}$ which belongs to X_i but not to X_j. Define Z as the set of all chosen elements.

($\Leftarrow$) We show by induction that if the cardinality of Z is k then there are at most 2^k sets X which satisfy $\varphi(X, \overline{Y})$. If $k = 0$ then no two distinct sets, so at most one set, can satisfy φ. For the induction step, assume that the cardinality of Z is $k+1$ and pick any element $z \in Z$. Observe that each pair of sets satisfying φ and including z has to differ on $Z \setminus \{z\}$, thus by the inductive assumption there are at most 2^k such sets. Analogously, any pair of sets satisfying φ and not including z has to differ on $Z \setminus \{z\}$, so there are at most 2^k such sets. In total there are at most $2 \cdot 2^k = 2^{k+1}$ sets that satisfy φ and differ on Z. □

Observe that the following converse of Proposition 6.1 holds as well. The predicate "the set X is infinite" can be defined using the $\exists^\infty$ quantifier, e.g. by saying that there exist infinitely many singletons in X. Moreover, if we have only the quantifier $\exists^{2^{\aleph_0}}$ we can define "X is infinite" by $\exists^{2^{\aleph_0}} Y\ Y \subseteq X$ and thus we can define $\exists^\infty$ as well.

6.2 Uncountability quantifier on linear orders

In this section, we show how to eliminate the uncountability quantifier from $\exists^{\aleph_1} X\ \varphi(X, \overline{Y})$ over a countable linear order L. We start by associating certain colorings of intervals of L with each X that satisfies φ and defining a condition on these colorings that guarantees the existence of uncountably many X satisfying φ. When this condition is not satisfied, we show how to reduce the problem to establishing whether there exist uncountably many Dedekind cuts of L satisfying certain formula. The

problem for Dedekind cuts is then solved for countable linear orders using a topological lemma on the binary tree, which is proved in the next section.

6.2.1 U-D coloring of intervals

We will consider the following colorings of intervals of L. When a set X and parameters $\overline{Y}$ are fixed, we color an interval I of L as unique, with color U, if X is the only set with type $\mathrm{Th}^N(I, \overline{Y}, X)$ on I. In the other case, there is another set with this type on I and we say that I is a D-interval. To give a formal definition, let us first fix the parameters $\overline{Y}$, N as the quantifier rank of φ, M as the length of $\overline{Y}$, and K as the number of N-types in $M+1$ variables.

Definition 6.3. For fixed parameters $\overline{Y}$ and set X satisfying $\varphi(X, \overline{Y})$, and $a, b \in L$ with $a < b$, we say that $[a, b]$ is a *U-interval* if for every X'

$$\mathrm{Th}^N(L|_{[a,b]}, \overline{Y}, X) = \mathrm{Th}^N(L|_{[a,b]}, \overline{Y}, X') \Rightarrow X \cap L|_{[a,b]} = X' \cap L|_{[a,b]},$$

otherwise we call $[a, b]$ a *D-interval.*

D-intervals ensure that there are at least two different possibilities of instantiating X without changing the theory on the given interval. Thus, if we find an infinite set of disjoint D-intervals, we can, by composition, shuffle these possibilities to obtain uncountably many X satisfying φ.

Condition A (linear orders)

$$\begin{aligned}&\exists X\ \varphi(X, \overline{Y}) \text{ and } \exists A \text{ of type } \omega \text{ or } \omega^* \text{ such that}\\ &\big(A = \{a_0 < a_1 < \ldots\} \text{ and } [a_{2i}, a_{2i+1}] \text{ is a D-interval for all } i, \text{ or}\\ &\ A = \{a_0 > a_1 > \ldots\} \text{ and } [a_{2i+1}, a_{2i}] \text{ is a D-interval for all } i\big)\end{aligned}$$

Observe that this condition is expressible in MSO and it indeed guarantees the existence of uncountably many sets satisfying φ.

Lemma 6.4. *When Condition A is satisfied on a linear order L then there exist uncountably many sets X satisfying $\varphi(X, \overline{Y})$.*

Proof. If Condition A is satisfied over a linear order then, on the disjoint D-intervals $[a_{2i}, a_{2i+1}]$ (or $[a_{2i+1}, a_{2i}]$), one can independently choose either X or the other set with the same type. By composition, the theory on the whole order remains unchanged, i.e. the same as $\mathrm{Th}(L, \overline{Y}, X)$. Thus every one of these $2^{\aleph_0}$ different possible choices yields a set satisfying φ. □

In addition to the above lemma, we are interested in the number of sets X satisfying $\varphi(X, \overline{Y})$ for which certain intervals are all U-intervals. To analyze it more precisely, let us introduce the notion of a finite U-regular cover of L.

Definition 6.5. A finite number of pairwise disjoint intervals $I_1, \ldots, I_n$ (with endpoints not necessarily in L) is a *U-regular cover* of L for a fixed formula φ and sets $X, \overline{Y}$ if it is a cover of L, i.e. $\bigcup_{j=1,\ldots,n} I_j = L$, and for each I_j the following holds. Either $\inf I_j \in L$ and for each $p \in I_j \cap L$ the interval $[\inf I_j, p]$ is a U-interval, or $\sup I_j \in L$ and for each $p \in I_j \cap L$ the interval $[p, \sup I_j]$ is a U-interval.

With fixed φ and $\overline{Y}$, a finite cover of L can be a U-regular cover for various X. But as the intervals are required to be almost U-intervals for such X, there can be only finitely many of them, as expressed by the following lemma.

Lemma 6.6. *Let $I_1, \ldots, I_k$ be a U-regular cover of L for a formula φ, parameters $\overline{Y}$ and each of the distinct sets $X_1, \ldots, X_n$. Then there is a number $N(k, \varphi)$ such that $n < N(k, \varphi)$.*

Proof. Let us recall that K is the number of $\mathrm{qr}(\varphi)$-types in $|\overline{Y}|+1$ variables. Thus if $[a, b]$ is a U-interval for $K+1$ sets, then two of these sets must be equal on $[a, b]$, as at least two have the same type. We extend this remark to the sets I_j as follows.

Denote $\inf I_j$ by a_j and $\sup I_j$ by b_j, and assume that $a_j \in L$ and that there are $K+1$ sets $X_1, \ldots, X_{K+1}$ such that for each $p \in I_j \cap L$ the interval $[a_j, p]$ is a U-interval for each of these sets X_i. If for each pair X_i, X_j there is a point $p_{i,j} \in I_j \cap L$ on which these two sets differ, then on the interval $[a_j, \max\{p_{i,j} \ : \ i, j \leq K+1\}]$ all the $K+1$ sets differ, which

contradicts the fact that it is a U-interval for all of them combined with the previous remark.

By Ramsey's theorem, for each l there exists a number $R(l,m)$ such that every clique of size bigger than $R(l,m)$ with edges colored with l colors contains as subgraph a clique of size at least m colored uniformly. Let us assume that there are more than $R(k, 2K+1)$ distinct sets X_i for which $I_1, \ldots, I_n$ is a U-regular cover. Let us imagine an edge between i and j colored with the first j such that X_i and X_j differ on I_j. By the above, there is a j such that $2K+1$ sets X_i differ on I_j. For each of these sets either $[a_j, p]$ is a U-interval for all $p \in I_j \cap L$, or this holds for $[p, b_j]$. Therefore, for at least $K+1$ sets one of the above conditions holds. But this is a contradiction as proved before, thus the lemma holds with $N(k,\varphi) = R(k, 2K+1)$. □

6.2.2 D-points and countable-to-one property

Definition 6.7. For fixed parameters $\overline{Y}$ and a set X satisfying $\varphi(X, \overline{Y})$, we say that a point $p \in \overline{L}$ in the completion of L is a *U-point* if it is contained in some U-interval $[a, b]$. Otherwise a point $p \in \overline{L}$ is a *D-point*, and then every interval containing it is a D-interval. By $D(X)$ we denote the set of all D-points for a given X.

When interpreted over a linear order L, $D(X)$ is a set of points in the completion $\overline{L}$ and thus a third-order set, but membership in $D(X)$ can still be expressed in MSO in the following sense.

$$D(X) = \{C \subset L \,:\, C \text{ is downward closed and } \sup C \text{ is a D-point for } X\}.$$

Thus, given φ, there is a formula $\delta(C, X, \overline{Y})$ satisfied by those triples $C, X, \overline{Y}$ such that C is a downward closed set of points, i.e. a Dedekind-cut, $\varphi(X, \overline{Y})$ holds, and $\sup C$ is a D-point for X.

Observe that Condition A over linear orders is satisfied when the set of D-points $D(X)$ is infinite for some X. This follows from the fact that every infinite set has a subset of type ω or ω^*. Thus one can find D-points

$p_0 < p_1 < \ldots$ in $\overline{L}$ and also points $a_i, b_i \in L$ such that $a_0 < p_0 < b_0 < a_1 < p_1 < b_1 < \ldots$ and so $[a_i, b_i]$ form the required D-intervals.

If Condition A is not satisfied over a countable linear order L, then there is no X satisfying φ that has an infinite set of pairwise disjoint D-intervals. In particular, the set of D-points for each X satisfying φ is finite. We aim to reduce the problem of finding uncountably many X to the problem of finding uncountably many $D(X)$, which is easier due to its topological structure as shown later in section 6.3.4. For this reduction we need to establish that for each set D there are only countably many sets X for which $D(X) = D$, and in this section our goal is to guarantee this as follows.

Lemma 6.8. *For a countable linear order L, a formula $\varphi(X, \overline{Y})$ and fixed sets $\overline{Y}$, if Condition A is not satisfied, then the mapping $X \mapsto D(X)$ is countable-to-one.*

Proof. Since Condition A is not satisfied, each set $D = D(X)$ is finite. For each X such that $D = D(X)$ and each point $\alpha \in D$ we fix an open interval $R_{\alpha,X} \subseteq L$ containing α, so that all intervals contained in $R_{\alpha,X} \setminus \{\alpha\}$ are U-intervals for X.

Let us first show that it is possible to find such intervals $R_{\alpha,X}$ if Condition A is not satisfied. For this let $a_0 < a_1 < \ldots$ be a sequence of elements of L converging to α from below and $b_0 > b_1 > \ldots$ another sequence from L converging to α from above (the special case when α is the minimal or maximal element of $\overline{L}$ uses only one sequence, but is analogous). Since each interval (a_i, b_i) contains a D-interval, there are either D-intervals $[a_j, a_{j+k}]$ or $[b_j, b_{j+k}]$ for arbitrary large j. But then there is an infinite family of pairwise disjoint D-intervals for X, which satisfy Condition A. Thus, the required intervals $R_{\alpha,X}$ exist, and, as the set D is finite, we can assume without loss of generality that for each X the intervals $R_{\alpha,X}$ are pairwise disjoint and that both $\inf R_{\alpha,X}$ and $\sup R_{\alpha,X}$ are in L (if not equal to α).

For each X and every point $p \in L$ not included in any of the $R_{\alpha,X}$ intervals there is a U-interval $U_{p,X}$ for X containing p, as p is a U-point.

For each X the interiors of $U_{p,X}$ provide an open cover of the closed set obtained by removing from $\overline{L}$ all the $R_{\alpha,X}$ intervals. Recall that the order topology of the completion $\overline{L}$ of a countable linear order L is compact. By compactness, there is a finite set P_X such that the interiors of $U_{p,X}$ with $p \in P_X$ constitute an open cover of the same set as all interiors of $U_{p,X}$. Consequently, the family of intervals

$$C_X = \{U_{p,X} \,:\, p \in P_X\} \cup \{[\inf R_{\alpha,X}, \alpha),\ (\alpha, \sup R_{\alpha,X}]\}$$

is a finite U-regular cover of L (and $\overline{L}$).

Since the endpoints of all the intervals in C_X are from L or D, there are only countably many choices of C_X for each X. According to Lemma 6.6 each finite U-regular cover C_X is shared by at most finitely many X. Thus there are at most countably many X with $D = D(X)$. □

As a direct consequence of Lemma 6.8, we can solve the case when the completion $\overline{L}$ of L is countable, i.e. the case of scattered linear orders. If Condition A does not hold, each set $D(X)$ is a finite set of points in $\overline{L}$, and there cannot exist uncountably many such sets if $\overline{L}$ is countable. Thus, over scattered linear orders, Condition A not only guarantees, but is also necessary for uncountability.

Corollary 6.9 (Cf. [50]). MSO *extended with the quantifier* $\exists^{\aleph_1}$ *effectively reduces to* MSO *over countable scattered linear orders, in particular* $(\omega, <)$.

Let us now extend the corollary above that holds for MSO formulas over ω to a specific case of MSO formulas over trees. If an MSO formula over $\mathfrak{T}$ talks only about subsets of a fixed path π then, by the composition theorem for trees, it behaves exactly like a formula on ω. Please observe that our signature for the tree does not differentiate between the left and the right successor. Predicates for the left and the right successor (and any predicates in general) will be added in the main theorem in the next section, but the corollary that we describe here depends on the fact that the whole labelling is only on subsets of π, which allows to use Theorem 1.12 for the reduction.

Corollary 6.10. *Let $\varphi(\pi, \overline{X}, \overline{Y})$ be an* MSO *formula interpreted over the binary tree such that π is a path and all X_i and Y_j are subsets of π, i.e.*

$$\varphi(\pi, \overline{X}, \overline{Y}) = path(\pi) \wedge \left(\bigwedge_i X_i \subseteq \pi \right) \wedge \left(\bigwedge_j Y_j \subseteq \pi \right) \wedge \psi(\pi, \overline{X}, \overline{Y}).$$

Then there exists an MSO *formula over the tree that expresses*

$$\exists^{\aleph_1} \overline{X}\ \varphi(\pi, \overline{X}, \overline{Y}).$$

6.2.3 Counting Dedekind cuts on linear orders

On countable linear orders we shall formulate a condition for uncountability of the set of cuts satisfying a formula taking advantage of the topological Lemma 6.15 proved later. We transfer this lemma to countable linear orders by first proving it for the rationals using a standard embedding into the binary tree, and later extending it to all countable orders by composition.

Lemma 6.11. *For every* MSO *formula $\varphi(C, \overline{Y})$ and tuple of sets $\overline{Y}$, if there are uncountably many Dedekind cuts C of $\mathbb{Q}$ such that $\mathbb{Q} \models \psi(C, \overline{Y})$ then there is a subset $D \subseteq \mathbb{Q}$ such that $(D, <^{\mathbb{Q}})$ is dense and without endpoints and for every irrational (with supremum not in D) Dedekind cut C^D of $(D, <^{\mathbb{Q}})$ the cut*

$$C = \{q \in \mathbb{Q} \ :\ \exists p \in C^D\ q < p\}$$

satisfies $\mathbb{Q} \models \psi(C, \overline{Y})$.

Proof. Let us sketch the standard embedding of $(\mathbb{Q}, <)$ into the complete binary tree $\mathfrak{T}(2)$. Since $(\mathbb{Q}, <)$ is isomorphic to the natural ordering of $\{\frac{l}{2^n} \ :\ l \in \mathbb{Z}, n \in \mathbb{N}\}$ it is enough to embed the rationals that are finite sequences over $\{0,1\}$ when written in binary coding. These are embedded directly as $(\{0,1\}^*1, <_{\text{lex}})$ based on the lexicographic ordering, definable in MSO on $\mathfrak{T}(2)$. Let $\phi : (\mathbb{Q}, <) \to (\{0,1\}^*1, <_{\text{lex}})$ be such an embedding.

First, observe that the non-empty irrational Dedekind cuts of $\mathbb{Q}$ (i.e. the non-empty downward-closed subsets of $\mathbb{Q}$ without supremum in $\mathbb{Q}$) are

in bijection with branches of $\mathfrak{T}(2)$ containing infinitely many ones, i.e. not ultimately zero. Indeed, every such branch α determines a Dedekind cut $C(\alpha) = \{q : \phi(q) <_{\text{lex}} \alpha\}$ without a maximal element. Conversely, every Dedekind cut C of $\mathbb{Q}$ determines a unique $\alpha \in \{0,1\}^\omega$ such that for every n, $\alpha|_n$ is the lexicographically least word of length n with $C \subseteq C(\alpha|_n 1^\omega)$. Therefore $C = C(\alpha) = \bigcap_n C(\alpha|_n 1^\omega)$, and C has a maximal element if and only if α is ultimately zero. Moreover, $C_\alpha \subsetneq C_\beta$ exactly if $\alpha <_{\text{lex}} \beta$.

Let $\varphi^*(P, \overline{Z})$ be an MSO-formula to be interpreted over $\mathfrak{T}(2)$ expressing that $P = \text{Pref}(\alpha)$ is a branch and that $C(\alpha)$ satisfies $\varphi(C(\alpha), \overline{Y})$ if $\phi(\overline{Y}) = \overline{Z}$. This formula is obtained from $\varphi(C, \overline{X})$ by replacing every occurrence of $<$ with the definition of the lexicographic order.

By Lemma 6.15, if for any parameters $\overline{Y}$ over $\mathbb{Q}$ the set of branches B satisfying $\mathfrak{T}(2) \vDash \varphi^*(B, \phi(\overline{Y}))$ is uncountable, then it contains a perfect subset $\mathcal{P}$ (in the Cantor topology).

Let D' be a complete set of representatives of the equivalence $v \sim w$ on $\{0,1\}^*1$, where $v \sim w$ if they are not separated by an element of $\mathcal{P}$. Since $\mathcal{P}$ is perfect, D' is densely ordered. Removing endpoints, we obtain a densely ordered set D such that

(i) between any two $\alpha <_{\text{lex}} \beta$ from $\mathcal{P}$ there is some $v \in D$, and

(ii) between any two $v <_{\text{lex}} w$ from D there lies some $\alpha \in \mathcal{P}$.

We show that these properties ensure that for every Dedekind cut C^D of $(D, <_{\text{lex}})$ having no maximal element there is some $\alpha \in \mathcal{P}$ such that

$$C = \{w \in \{0,1\}^*1 : \exists v \in C^D \; w <_{\text{lex}} v\}$$

is identical to $C(\alpha)$. Note that with this claim, the lemma follows.

To prove the above claim let α be the supremum of the set

$$\mathcal{C} = \{\beta \in \mathcal{P} : \exists v \in C^D \; \beta <_{\text{lex}} v\}.$$

Since $\mathcal{P}$ is closed, $\alpha \in \mathcal{P}$ and C^D has no maximal element (by (ii)), we conclude that $\alpha \notin \mathcal{C}$. Also, since α is the supremum of a non-empty set of binary ω-sequences, it is not ultimately zero. Thus $C(\alpha)$ has no maximal element either. It remains to prove that indeed $C(\alpha) = C$.

($\supseteq$) By (ii) each $w <_{\text{lex}} v$ with $v \in C^D$ is separated by some $\beta \in \mathcal{C}$, therefore $\beta <_{\text{lex}} \alpha$ and so $w <_{\text{lex}} \alpha$.

($\subseteq$) Since $C(\alpha)$ has no greatest element, for every $w \in C(\alpha)$ there is some $v \in C(\alpha)$ such that $w <_{\text{lex}} v$. By (i) there is some $\beta \in \mathcal{P}$ separating w and v. Then $\beta <_{\text{lex}} \alpha$, i.e. $\beta \in \mathcal{C}$, and so there is some $u \in C^D$ such that $\beta <_{\text{lex}} u$ thus also $w <_{\text{lex}} u$. □

Having established the lemma for $\mathbb{Q}$, we use the composition theorem and the characterization by Hausdorff to generalize it to all countable linear orders.

Lemma 6.12. *For every countable linear order L,* MSO *formula $\varphi(C, \overline{Y})$ and tuple of sets $\overline{Y}$, if there are uncountably many Dedekind cuts C of L such that $L \vDash \psi(C, \overline{Y})$ then there is a subset $D \subseteq L$ such that $(D, <^L)$ is dense and without endpoints and for every irrational Dedekind cut C^D of $(D, <^L)$ the cut*

$$C = \{p \in L \,:\, \exists q \in C^D \; p < q \text{ or } ([q, p], <^L) \text{ is scattered}\}$$

satisfies $L \vDash \psi(C, \overline{Y})$.

Proof. As mentioned in section 1.2, Hausdorff has shown that every countable order L is a sum $L = \sum_{q \in \mathbb{Q}} L_q$, where each L_q is a scattered countable order. Since each L_q is scattered, the completion of L_q is countable and so there are only countably many cuts inside all the L_q components. Thus if there are uncountably many cuts C with $\varphi(C, \overline{Y})$ on L, then there are uncountably many cuts C that, for each q, either contain L_q fully or not at all, i.e. $L_q \subseteq C$ or $L_q \cap C = \varnothing$. For such cuts C we write $C^{\mathbb{Q}} = \{q \in \mathbb{Q} \,:\, L_q \subseteq C\}$ for the set of rationals corresponding to the components contained in C.

We transform the formula $\varphi(C, \overline{Y})$ to a formula $\varphi'(C, \overline{Y})$ that is satisfied on L only by cuts of the form specified above. Note that by the above remark $\exists^{\aleph_1} C \; \varphi(C, \overline{Y})$ is equivalent to $\exists^{\aleph_1} C \; \varphi'(C, \overline{Y})$. To construct φ' we first define a set Q containing exactly one point of each L_q. First, we require that Q is dense and without endpoints. Then, for any two points in Q, there is a dense subset of Q between them, so they cannot belong

to the same scattered L_q. Moreover, we require that for each $p \in L$ there is a $q \in Q$ such that either $[p, q]$ or $[q, p]$ is scattered, meaning that it contains no dense subset. This ensures that for each (non-empty) L_q the set $Q \cap L_q \neq \varnothing$. Observe that expressing "scattered" in the same way, we can construct a formula $\gamma(q, X)$ that ensures for each $q \in Q$ that X contains exactly the component $L_{q'}$ that contains q. We can now construct $\varphi'(C, \overline{Y})$ as a conjunction of $\varphi(C, \overline{Y})$ and a formula that requires for each $q \in Q$ that the set X satisfying $\gamma(q, X)$ is either contained in C or disjoint from it.

We now apply the effective composition theorem (Theorem 1.11) to the formula φ' and the order $L = \sum_{q \in \mathbb{Q}} L_q$ and get the formula $\theta(T_1, \ldots, T_k)$ such that for each $C, \overline{Y}$ if $T_m = \{q : L_q \vDash \tau_m(C|_{L_q}, \overline{Y}|_{L_q})\}$ then

$$L \vDash \varphi'(C, \overline{Y}) \iff \mathbb{Q} \vDash \theta(T_1, \ldots, T_k).$$

Since we want C to be a parameter, we define *two* type labels that depend only on $\overline{Y}$. Let

$$T_m^- = \{q \in \mathbb{Q} : L_q \vDash \tau_m(\varnothing, \overline{Y}|_{L_q})\}, \text{ and}$$

$$T_m^+ = \{q \in \mathbb{Q} : L_q \vDash \tau_m(L_q, \overline{Y}|_{L_q})\}.$$

Since all cuts C satisfying $\varphi'(C, \overline{Y})$ either contain L_q or are disjoint from it (for all q), it is either the case that $T_m = T_m^-$ or that $T_m = T_m^+$, and $C^{\mathbb{Q}}$ is enough to establish which case it is. Thus, replacing T_m by $(T_m^+ \cap C') \cup (T_m^- \setminus C')$ in θ, we get a formula $\theta'(C', T_1^-, \ldots, T_k^-, T_1^+, \ldots, T_k^+)$ such that for each $\overline{Y}$ and $\overline{T^-}, \overline{T^+}$ as defined above

$$L \vDash \varphi'(C, \overline{Y}) \iff \mathbb{Q} \vDash \theta'(C^{\mathbb{Q}}, \overline{T^-}, \overline{T^+})$$

for each Dedekind cut C of L.

Applying Lemma 6.11 to θ' and the labels $\overline{T^-}, \overline{T^+}$ we get a set $D^{\mathbb{Q}}$, dense and without endpoints, such that for every irrational Dedekind cut C' of $D^{\mathbb{Q}}$ if $C^{\mathbb{Q}} = \{q \in \mathbb{Q} : \exists p \in C'\ q < p\}$ then $\varphi(C, \overline{Y})$ holds. Using the previously defined set of representants Q and taking $D = \{q \in Q : q \in L_{q'} \text{ and } q' \in D^{\mathbb{Q}}\}$ we get the desired set. □

As shown in the proof above, given the set D and a cut C^D of D it is possible to define the cut C satisfying the condition of the lemma in MSO. Thus the required properties of D for a formula $\varphi(C, \overline{Y})$ can be defined by an MSO formula $\gamma_\varphi(D, \overline{Y})$ stating that D is dense, without endpoints, and has cuts satisfying the condition of the lemma. Therefore, the existence of an uncountable set of Dedekind cuts C satisfying $\varphi(C, \overline{Y})$ over an arbitrary countable linear order can be expressed by the existence of a suitable dense set. For a formula $\varphi(X, \overline{Y})$ let

$$\varphi'(C, \overline{Y}) = \exists X \, (\varphi(X, \overline{Y}) \ \wedge \ C \in D(X))$$

be the formula defining D-points of all X satisfying φ. We can now formulate the second condition for linear orders simply as follows.

Condition B (linear orders)

$$\exists D \, \gamma_{\varphi'}(D, \overline{Y})$$

The set of Dedekind cuts of a dense set D is always of continuum cardinality, thus Condition B indeed implies the existence of uncountably many points in all $D(X)$ and thus of uncountably many X. The converse direction was proved in the previous lemmas. Thus we have shown that on the class of all countable linear orders the uncountability quantifier can be eliminated from monadic second-order formulas.

Theorem 6.13. *Over the class of countable linear orders, the following equivalence holds.*

$$\exists^{\aleph_1} X \, \varphi(X, \overline{Y}) \ \equiv \ \exists^{2^{\aleph_0}} X \, \varphi(X, \overline{Y}) \ \equiv \ \textit{Condition A or Condition B.}$$

6.3 Uncountability quantifier on the binary tree

In this section, we show how the uncountability quantifier can be eliminated from MSO on the binary tree. We use U-D colorings of the tree similar to the case of linear orders, and reduce the problem to counting

the number of *paths* in the tree satisfying a given formula. The case of paths is then solved using a topological lemma, which we present first together with basic topological definitions.

6.3.1 Descriptive complexity

Let us recall a few basic notions from descriptive set theory and prove a lemma about the topological complexity of definable sets of paths. A thorough introduction to descriptive set theory can be found in [58]. We already mentioned Borel sets and the product topology before, but let us recapitulate it more formally here. Note that we work only on the binary tree here to avoid unnecessary technical notation, even though the results hold for any finitely branching tree.

The Cantor space is the topological space with the product topology on $\{0,1\}^\omega$. The topology is generated by basic neighborhoods $w\{0,1\}^\omega$ with the prefix $w \in \{0,1\}^*$. Alternatively, it can be defined by the metric

$$d(\alpha,\beta) = 2^{-\min\{n \,:\, \alpha[n]\neq\beta[n]\}}.$$

The hierarchy of Borel sets is generated starting from open sets, i.e. unions of basic neighborhoods, denoted $\mathbf{\Sigma}^0_1$, and closed sets, which are complements of open sets and denoted $\mathbf{\Pi}^0_1$. Further, for any countable cardinal α, we say that a set is in $\mathbf{\Sigma}^0_{\alpha+1}$ if it can be expressed as a countable sum of sets from $\mathbf{\Pi}^0_\alpha$, and it is in $\mathbf{\Pi}^0_{\alpha+1}$ if it is a countable intersection of sets from $\mathbf{\Sigma}^0_\alpha$. For limit cardinals, the sum of lower the classes is taken. The Borel hierarchy defined in this way can further be extended to the projective hierarchy, where $\mathbf{\Sigma}^1_1$ is used to denote *analytic* sets, which are projections of Borel sets, and $\mathbf{\Pi}^1_1$ are complements of analytic sets, called *co-analytic* sets. The hierarchy goes on with a set in $\mathbf{\Sigma}^1_{\alpha+1}$ being a projection of a set from $\mathbf{\Pi}^1_\alpha$ and $\mathbf{\Pi}^1_{\alpha+1}$ being complements of sets from $\mathbf{\Sigma}^1_{\alpha+1}$. All these definitions extend from labellings of the line, i.e. words in $\{0,1\}^\omega$, to labellings of the binary tree.

The connection between the topological complexity of MSO-definable tree languages and the complexity of tree-automata recognizing them is

well understood [75, 61]. By Rabin's complementation theorem all MSO-definable tree languages are in $\mathbf{\Sigma}^1_2 \cap \mathbf{\Pi}^1_2$. There are $\mathbf{\Sigma}^1_1$-complete as well as $\mathbf{\Pi}^1_1$-complete regular tree languages, e.g. the $\mathbf{\Pi}^1_1$-complete language of trees over $\{a, b\}$ that, on each path, have only finitely many a's [3, 61]. There exist regular tree languages not contained in $\mathbf{\Sigma}^1_1 \cup \mathbf{\Pi}^1_1$, but deterministic tree automata accept only $\mathbf{\Pi}^1_1$ sets of trees. In contrast, by McNaughton's theorem, ω-regular languages, i.e. MSO-definable sets of ω-words, are boolean combinations of $\mathbf{\Pi}^0_2$ sets [75].

The Cantor-Bendixson Theorem states that closed subsets of the Cantor space have the *perfect set property*: they are either countable or contain a perfect subset and thus have cardinality continuum. Recall that a set P is *perfect* if it is closed and if every point $p \in P$ is a condensation point of P, i.e. if in every neighborhood of p there is another point from P. We shall rely on the following fundamental result.

Theorem 6.14 (Souslin, 1916/17, e.g. in [58])**.** *A subset of $\{0,1\}^\omega$ is Borel if and only if it is both analytic and co-analytic. Moreover, every uncountable analytic set contains a perfect subset.*

Let us remark that whether co-analytic sets, or all sets on higher levels of the projective hierarchy, satisfy the continuum hypothesis is independent of ZFC. A key observation that we need in this chapter is that even though there are non-Borel sets of trees recognizable by tree automata, the sets of recognizable paths are still Borel. Recall that for a sequence $\pi \in \{0,1\}^*$ we denote by $\mathrm{Pref}(\pi)$ the path through $\mathfrak{T}(2)$ that corresponds to this sequence, which formally is the set of prefixes of π.

Lemma 6.15. *Let $U_1, \ldots, U_M$ be subsets of $\mathfrak{T}(2)$ and let $\psi(P, Y_1, \ldots, Y_M)$ be an* MSO *formula over $\mathfrak{T}(2)$. Then the set*

$$\mathcal{X} = \{\pi \in \{0,1\}^\omega \,:\, \mathfrak{T}(2) \vDash \psi(\mathrm{Pref}(\pi), \overline{U})\}$$

of paths though the binary tree satisfying $\psi(-, \overline{U})$ is analytic and therefore has the perfect set property.

Note that negation is allowed in the formula ψ above, so the set $\mathcal{X}$ is co-analytic as well, and thus in fact Borel.

Proof. We will show that $\mathcal{X}$ is a projection of the intersection $\mathcal{L} \cap [\vartheta_{\overline{Y}}]$, where $[\vartheta_{\overline{Y}}]$ is the set of all infinite labeled paths through $\mathfrak{T}(2)$ with labelling $\vartheta_{\overline{Y}}$ defined below and $\mathcal{L}$ is an ω-regular language. Observe that it follows that $\mathcal{X}$ is analytic, since $[\vartheta_{\overline{Y}}]$ is closed by definition and $\mathcal{L}$ is in $\mathbf{\Sigma}^0_3$ by McNaughton's theorem, thus their intersection is also $\mathbf{\Sigma}^0_3$.

The idea is to consider infinite sequences over the alphabet $\{0,1\} \times \Theta$, where $\Theta = \mathrm{Tp}(N, M+1)$ with N equal to the quantifier rank of ψ. We define $\vartheta_{\overline{Y}}$ as the labelling on the binary tree mapping each node v to the N-type of $(\varnothing, \overline{Y})$ restricted to the subtree rooted at v.

By the effective version of the composition theorem (Theorem 1.12), we know that for every N-type τ in $|\overline{Y}|+1$ variables there is an ω-regular language L_τ of sequences $(\pi, \bar{1}, \tau) \in (\{0,1\} \times \{0,1\} \times \Theta)^\omega$ of paths, labels and N-types such that the N-type of the binary tree decorated with subsets $P = \mathrm{Pref}(\pi)$ and $\overline{Y}$ is τ.

We set $\mathcal{L}$ to be the union of those L_τ such that $\tau(P, \overline{Y})$ implies $\psi(P, \overline{Y})$. Then $\mathcal{L}$ is ω-regular as well. By construction, $\psi(P, \overline{Y})$ holds precisely if the corresponding triplet of sequences $(\pi, \bar{1}, \tau)$ is in $\mathcal{L}$. However, $\mathcal{L}$ typically also contains sequences that do not correspond to any path of the tree decorated with parameters $\overline{Y}$. But there is a bijection between paths P satisfying $\psi(P, \overline{Y})$ and sequences belonging to the first component of $\mathcal{L} \cap [\vartheta_{\overline{Y}}]$, which shows that $\mathcal{X}$ indeed has the required form and is thus analytic. □

6.3.2 U-D coloring and infinite independent D-sets

To eliminate the uncountability quantifier from $\exists^{\aleph_1} X\, \varphi(X, \overline{Y})$ over the binary tree we will, again, consider U-D colorings of the tree. We again first fix the parameters $\overline{Y}$, N as the quantifier rank of φ, M as the length of $\overline{Y}$, and K as the number of N-types in $M+1$ variables.

Let us remark that on trees labeled in a regular way, there is an intuitive correspondence between types and states of the automaton recognizing φ. The type of a subtree $\mathfrak{T}_v$ with labels $X, \overline{Y}$ corresponds to the set of states from which the automaton for φ can start in v and accept the tree $\mathfrak{T}_v$ with labels $X, \overline{Y}$. Thus a D-subtree corresponds intuitively to the existence of

another set $X' \neq X$ such that the automaton accepts $\mathfrak{T}_v$ with labels $X', \overline{Y}$ from the same set of states.

Definition 6.16. For a fixed set X satisfying $\varphi(X, \overline{Y})$ and a node $v \in \mathfrak{T}$ we say that $\mathfrak{T}_v$ is a *U-tree* when for every X'

$$\mathrm{Th}^N(\mathfrak{T}_v, \overline{Y}, X) = \mathrm{Th}^N(\mathfrak{T}_v, \overline{Y}, X') \implies X \cap \mathfrak{T}_v = X' \cap \mathfrak{T}_v,$$

and we say that $\mathfrak{T}_v$ is a *D-tree* otherwise. A path P is called a *D-path* when every subtree along P is a D-tree.

Again, we will be interested in the set $D(X)$ of all D-trees for X:

$$v \in D(X) \iff \mathfrak{T}_v \text{ is a D-tree for } X.$$

Observe that over the binary tree, the set $D(X)$ is prefix-closed since if for some $u = vw$ the tree $\mathfrak{T}_u$ is a D-tree, then $\mathfrak{T}_v$ is a D-tree as well, using X' on $\mathfrak{T}_u$ and X elsewhere. That is definable in MSO, and the following basic condition guarantees uncountably many X satisfying φ.

Condition A (trees)

$$\exists X\ \exists A\ \varphi(X, \overline{Y}) \text{ and } A \subseteq D(X) \text{ is an infinite antichain.}$$

Note that this is the only place where we use the predicate "A is infinite" and that the above condition is expressible in MSO with this predicate over arbitrary trees, and just in MSO over finitely branching trees (by König's Lemma).

Lemma 6.17. *When Condition A is satisfied over the binary tree then there exist uncountably many sets X satisfying $\varphi(X, \overline{Y})$.*

Proof. If Condition A is satisfied over the tree, one can modify X below every node v in the infinite antichain A without changing the theory below that node. By composition and the fact that A is an antichain, these modifications are independent and thus give $2^{\aleph_0}$ possible sets, all satisfying φ. □

6.3.3 Infinite dependent D-set on the tree

Over linear orders Condition A is, as proved in the previous section, sufficient to guarantee that the mapping $X \mapsto D(X)$ is countable-to-one. This is not the case over the binary tree, as there may exist an infinite set $D(X)$ containing no infinite antichain. Still, as each $D(X)$ is prefix-closed, it can consist of at most a finite number of infinite paths in addition to a finite part (or else include an infinite antichain). We focus on one such infinite path π starting in a node v_0 and construct an MSO formula

$$\varphi_\tau^{2^{\aleph_0}}(\pi, v_0, \overline{Y})$$

that holds exactly if there are uncountably many X with $D(X) \cap \mathfrak{T}_{v_0} = \pi$ and having type τ on $\mathfrak{T}_{v_0}$.

The construction of $\varphi_\tau^{2^{\aleph_0}}$ is discussed later, but we can already formulate another condition for uncountability that, together with Condition A, is sufficient to guarantee that the mapping $X \mapsto D(X)$ is countable-to-one.

Condition B (trees)

$$\exists X \textit{ satisfying } \varphi\ \exists \textit{ path } \pi \subseteq D(X)\ \exists v_0 \in \pi \\ \bigvee_{\tau \in \mathrm{Tp}(N,M+1)} \left(\tau = \mathrm{Th}^N(\mathfrak{T}_{v_0}, \overline{Y}, X) \wedge \varphi_\tau^{2^{\aleph_0}}(\pi, v_0, \overline{Y})\right).$$

Lemma 6.18. *If Condition B is satisfied then there exist uncountably many sets X satisfying φ. If neither Condition A nor Condition B is satisfied then the mapping $X \mapsto D(X)$ is countable-to-one.*

Proof. If Condition B is satisfied then by the definition of $\varphi_\tau^{2^{\aleph_0}}$ there exist uncountably many sets Z with type τ on $\mathfrak{T}_{v_0}$. As there is an X satisfying φ with the same type on $\mathfrak{T}_{v_0}$, we can, by composition, swap the part of X below v_0 for any of the sets Z and the result still satisfies φ.

If neither Condition A nor Condition B holds, every set $D(X)$ contains only finitely many infinite paths and, by definition of $\varphi_\tau^{2^{\aleph_0}}$ and finiteness of the set of types, there are only countably many sets X sharing each infinite path. As each of the finitely many infinite paths is shared by only

countably many X, and every U-tree can be shared only by as many X as there are types, we get that for each set D there are only countably many sets X with $D = D(X)$. □

We now focus on the construction of $\varphi_\tau^{2^{\aleph_0}}$. Let us fix π as one of the finitely many infinite paths in D and a node v_0 on π so that no branching of D occurs below v_0. We will reduce the question about the existence of uncountably many X with type τ and $D(X) \cap \mathfrak{T}_{v_0} = \pi$ to the uncountability quantifier case over ω, which was solved in [50] and reduces to Condition A as shown in Corollary 6.9.

For the reduction, we construct a bijection between the sets X that have $D(X) \cap \mathfrak{T}_{v_0} = \pi$ and ω-words over an alphabet Θ. We choose Θ to be the set of tuples $(d \in \{0,1\}, l \in \{0,1\}, \tau_X, \tau_Y)$ where τ_X is an $N+2$-type in $M+1$ variables and τ_Y is an $N+3$-type in M variables.

We put a special requirement on Θ with respect to the τ_X component. We require that there is an N-type in $M+1$ variables τ'_X such that $\tau_X \equiv \tau'_X \wedge \exists! X \tau'_X$. Intuitively, we use this requirement to assure that X is a U-set wherever its theory is τ_X, and for that reason we use $N+2$-types for τ_X.

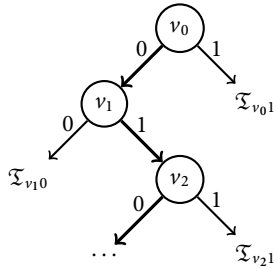
Let us assign to every set X with $D(X) \cap \mathfrak{T}_{v_0} = \pi$ the following word w_X.

$$\begin{aligned} w_x[i] = (&\pi[i], \\ &v_0\pi|_i \in X, \\ &\mathrm{Th}^{N+2}(\mathfrak{T}_{v_0\pi|_i(1-\pi[i])}, \overline{Y}, X), \\ &\mathrm{Th}^{N+3}(\mathfrak{T}_{v_0\pi|_i(1-\pi[i])}, \overline{Y})\ ), \end{aligned}$$

as represented in Figure 6.1, where $\mathrm{Th}^N_{\overline{Y}}(\mathfrak{T}, X)$ stands for $\mathrm{Th}^N(\mathfrak{T}, \overline{Y}, X)$.

Let us consider words $w = (\overline{d}, \overline{l}, \overline{\tau_X}, \overline{\tau_Y})$ for which on all positions i it holds that $\overline{\tau_Y}[i] \vDash \exists X \overline{\tau_X}[i]$. Note that τ_Y are $N+3$ types, with τ_X being $N+2$-types, exactly to make this requirement an easily checkable syntactic condition. We say that such words w are *well-composed*. Observe that for every X the word w_X is well-composed by the existence of X.

Claim 6.19. *For a fixed v_0, path π and parameters $\overline{Y}$, the mapping $X \mapsto w_X$ is a bijection between all X with $D(X) \cap \mathfrak{T}_{v_0} = \pi$ and well-composed ω-*



$$w_X = \begin{bmatrix} 0 \\ v_0 \in X \\ \mathrm{Th}^{N+2}_{\overline{Y}}(\mathfrak{T}_{v_0 1}, X) \\ \mathrm{Th}^{N+3}_{\overline{Y}}(\mathfrak{T}_{v_0 1}) \end{bmatrix} \begin{bmatrix} 1 \\ v_1 \in X \\ \mathrm{Th}^{N+2}_{\overline{Y}}(\mathfrak{T}_{v_1 0}, X) \\ \mathrm{Th}^{N+3}_{\overline{Y}}(\mathfrak{T}_{v_1 0}) \end{bmatrix} \begin{bmatrix} 0 \\ v_2 \in X \\ \mathrm{Th}^{N+2}_{\overline{Y}}(\mathfrak{T}_{v_2 1}, X) \\ \mathrm{Th}^{N+3}_{\overline{Y}}(\mathfrak{T}_{v_2 1}) \end{bmatrix} \dots$$

Figure 6.1: Constructing the word w_X.

words over Θ with the first component of the word equal to π and the last to the $N+3$-types of $\overline{Y}$ on the nodes below π.

Proof. If there were two distinct sets X and X' with $w_X = w_{X'}$ then these sets would have to differ on some tree $\mathfrak{T}_u$ for u not on the path π. As the words w_X and $w_{X'}$ coincide, the sets X and X' must have the same type on $\mathfrak{T}_u$. But the requirement put on $\Theta \ni \tau_X$ guarantees that $\mathfrak{T}_u$ is a U-tree for both X and X', which leads to a contradiction.

It remains to be shown that for each word w with the first component equal to π and the last to the types of $\overline{Y}$ below the path π, there exists a set X with $D(X) \cap \mathfrak{T}_{v_0} = \pi$ for which $w_X = w$. Indeed, for each node $u = v_0 \pi|_i (1 - \pi[i])$ below π, the fact that w is well-composed guarantees that there is a labelling X_i to put below u so that $\mathrm{Th}^{N+2}(\mathfrak{T}_u, X_i, \overline{Y}) = \tau_X$. Denoting by P the set of positions i for which the second component of $w[i]$ is 1, i.e. the labelling along π given by w, we can construct X by putting $X = \sum_{\pi,P} X_i$, and indeed $w_X = w$ holds. □

The bijection between sets and words allows us to reduce the case of

uncountably many X sharing a path in $D(X)$ to the word case, in fact to the case of Corollary 6.10.

To observe this, we use the coding of types introduced in section 1.6. From the constructive version of the composition theorem for trees (Theorem 1.12) it follows that for each type τ there exists an MSO formula $\psi_\tau(\pi, \nu_0, P, T^N(X))$ that guarantees that the sum along π, P of sets with N-types $T^N(X)$ below π has type τ in ν_0.

We assume in the theorem that the description of types $T^N(X)$ used as argument of ψ_τ is not below π but in fact shifted, i.e. $T^N(X) \subseteq \pi$. Moreover, both the uniqueness condition that we require from the $N+2$-types in Θ and the well-foundedness relation between $N+2$-types of $X, \overline{Y}$ and $N+3$-types of $\overline{Y}$ are syntactic and can be checked by MSO formulas, which we name 'unique' and 'well-founded' respectively. Thus, we can write the MSO formula

$$\begin{aligned}\theta_\tau\big(\pi, \nu_0, P, T^{N+2}(X, \overline{Y}), T^{N+3}(\overline{Y})\big) = \\ \text{unique}(T^{N+2}(X, \overline{Y})) \wedge \\ \text{well-composed}(T^{N+2}(X, \overline{Y}), T^{N+3}(\overline{Y}), \pi) \wedge \\ \psi_\tau(\pi, \nu_0, P, T^{N+2}(X, \overline{Y}))\end{aligned}$$

that holds if and only if the arguments code a well-composed word w along π so that $w = w_X$ and X has type τ in ν_0. As all the arguments of this formula are subsets of π, we can, by Corollary 6.10, write in MSO the formula

$$\begin{aligned}\varphi_\tau^{2^{\aleph_0}}(\pi, \nu_0, \overline{Y}) \; = \\ \exists^{2^{\aleph_0}}(P, T^{N+2}(X, \overline{Y})) \;\; \theta_\tau(\pi, \nu_0, P, T^{N+2}(X, \overline{Y}), T^{N+3}(\overline{Y})).\end{aligned}$$

Observe that $\varphi_\tau^{2^{\aleph_0}}(\pi, \nu_0, \overline{Y})$ holds if and only if, from ν_0 on along π, there are uncountably many well-formed (coded) words w with the $N+3$-types of $\overline{Y}$ as the last component. By Claim 6.19 this means that there are uncountably many X with $D(X) \cap \mathfrak{T}_{\nu_0} = \pi$ and having type τ on $\mathfrak{T}_{\nu_0}$, with the first condition guaranteed by the restriction on $\tau_X \in \Theta$ enforced by the 'unique' formula. This is exactly what we requested from $\varphi_\tau^{2^{\aleph_0}}$, which ends the construction.

6.3.4 Counting paths on the tree

We formulated conditions A and B that guarantee on the binary tree the existence of uncountably many X satisfying $\varphi(X, \overline{Y})$. Moreover, we know by Lemma 6.18 that if neither of these conditions holds, then for every X the set $D(X)$ has finitely many paths and the mapping $X \mapsto D(X)$ is countable-to-one. Thus, the existence of uncountably many X satisfying φ when neither Condition A nor Condition B holds can be reduced to the existence of uncountably many paths in all sets $D(X)$.

$$\begin{aligned}&\exists^{\aleph_1} X\ \varphi(X, \overline{Y}) \equiv \\ &\qquad \text{Condition A or Condition B or} \\ &\qquad\qquad \exists^{\aleph_1} P\ \big(\mathrm{path}(P)\ \wedge\ \exists X\ \big(\varphi(X, \overline{Y})\ \wedge\ P \subseteq D(X)\big)\big),\end{aligned}$$

For this reason, we now consider the following formula

$$\psi(P, \overline{Y})\ =\ \exists X\ \big(\varphi(X, \overline{Y}) \wedge P \subseteq D(X)\big).$$

We show that the question whether $\exists^{\aleph_1} P\,\big(\mathrm{path}(P)\ \wedge\ \psi(P, \overline{Y})\big)$ is indeed simpler as the general question and can be expressed in MSO. The main difference between this and the general case is that the set of paths satisfying an MSO formula is analytic, as shown in Lemma 6.15. Thus, it contains a perfect subset if and only if it is uncountable, and a perfect subset can in turn be represented by an infinite set of branching points, as in the following condition.

Condition C (trees) *There exists a set $B \subseteq \mathfrak{T}(2)$ such that each node in B has two incomparable successors in B and every path P passing through infinitely many nodes of B satisfies $\psi(P, \overline{Y})$.*

Lemma 6.20. *The formula $\exists^{\aleph_1} P\,\big(\mathrm{path}(P)\ \wedge\ \psi(P, \overline{Y})\big)$ is equivalent to Condition C.*

Proof.

($\Leftarrow$) The structure of the set B guaranteed to exist in Condition C is isomorphic to the complete binary tree. In particular, there are $2^{\aleph_0}$ many

paths passing through infinitely many nodes of B, and, by Condition C, all of them satisfy $\psi(P, \overline{Y})$.

($\Rightarrow$) By Lemma 6.15, if the set of paths satisfying $\psi(P, \overline{Y})$ is uncountable, then it contains a perfect subset $\mathcal{P}$. Consider the restriction of the tree to the set B' of nodes where two paths belonging to $\mathcal{P}$ are branching away from each other. Since $\mathcal{P}$ is closed, any path passing through infinitely many nodes of B' is contained in $\mathcal{P}$. Furthermore, every node in B' has at least two incomparable descendants in B', for otherwise there would be an isolated path in $\mathcal{P}$ contradicting its being perfect. Thus there is a subset $B \subseteq B'$ satisfying Condition C. □

Since Condition C is monadic second-order definable, this completes the treatment of the uncountability quantifier over trees and leads to the following theorem.

Theorem 6.21. *Over the class of labeled binary trees, the following equivalence holds.*

$$\begin{aligned} \exists^{\aleph_1} X\, \varphi(X, \overline{Y}) &\equiv \exists^{2^{\aleph_0}} X\, \varphi(X, \overline{Y}) \\ &\equiv \textit{Condition A or Condition B or Condition C.} \end{aligned}$$

7 Outlook

We considered the connection between logic and games underlying model-checking procedures on finite structures and extended it to the class of automatic structures. Thus, we defined a new class of hierarchical games suitable for model-checking first-order logic over automatic structures. These games can be used for model-checking not only first-order logic, but also formulas with the regular game quantifier. Moreover, cardinality and counting quantifiers can be reduced to first-order logic on automatic structures. Thus, hierarchical games provide a way to model-check first-order logic extended with cardinality, counting and game quantification on automatic presentations.

In our basic model of hierarchical games, two coalitions with strictly opposing objectives play a game with a particular kind of imperfect information. In general multiplayer games, strictly opposing objectives of players are not common. Moreover, the hierarchical constraint is a technical limitation introduced to keep the problem of establishing the winners decidable. Therefore we ask which other classes or representations of games can be used for model-checking first-order logic on automatic structures. One natural way to define such games is by departing from the standard abstraction of a token moved on a state graph and allowing the players to play with more complex objects.

For example, we imagine games where players build a new graph during the game by choosing moves labeled by some simple graph rewriting rules. One promising candidate for such rules are the separated handle hypergraph rewriting rules introduced in [20]. Graphs constructed using these rules have bounded clique-width and are MSO-interpretable in the binary tree, and thus have a decidable MSO theory (see [9] for an overview). Following this approach, a model-checking game for a formula $\exists x \, \forall y \, R(x, y)$, with x and y represented by finite words, would start

with the Verifier building an arbitrary large graph that represents the game $\forall y\, R(u, y)$ for some word u that he chooses. Then, the Falsifier continues the construction for some word w of his choice. Finally, a regular condition is checked on the graph constructed for $R(u, w)$ to determine the winner.

Such a description seems more intuitive than the definition of hierarchical games as it involves only two players with perfect information. On the other hand, it is not clear what kind of construction rules should be allowed and how to define a natural class of such games where establishing the winner is decidable. Still, it is an interesting subject for future work to find other classes of games for model-checking first-order logic on automatic, or other finitely presented structures.

Another direction is to extend hierarchical games and to use them for model-checking on larger classes of structures. One question is whether we can obtain model-checking games for tree-automatic structures in this way. We conjecture that the answer is positive and that the necessary extension is to add two new players that transcend information levels. More precisely, the moves of the new players would be visible to all other players in the game and conversely, the new players would be able to see moves of all other players as well. The intuition is that the moves of the new players correspond to the choice of a branch in the tree when an alternating tree automaton is running. This conjecture leads to another question, namely how can such games be further extended to larger classes of structures, for example to generalized-automatic ones.

Aiming at model-checking games for larger classes of structures, we see two main directions to follow. On the one hand, games on certain infinite graphs, for example on pushdown graphs, can still be solved algorithmically. Thus, one can try to use such games for model-checking. On the other hand, one may combine the games played in the syntactic setting, like dialogue games, with model-checking games played on graphs. In this way, one views a winning strategy of the Verifier in a hierarchical game for a formula φ as a description of the choices needed to build a proof of φ by induction on the structure of words used in the presentation.

The game itself is then a compact description of possible choices in the proofs for both φ and $\neg\varphi$.

For these considerations to be useful, it is necessary to find efficient algorithms for establishing the winner in the particular class of games. This is a difficult task and the procedures used to prove decidability of the games are often not efficient enough for practical applications. For example, it is unclear how to solve alternating hierarchical parity games without the complex step of determinizing alternating parity automata. Still, there are reasons to hope that it is feasible to solve even complex games and that representing problems as games helps to find efficient solutions. For example, the antichain method introduced in [17] for games with imperfect information turned out to be successful in improving model-checking algorithms based on automata [21]. We believe that further work in this direction will confirm that games can both give us better understanding of the expressive power of various logics and lead to efficient algorithms with practical applications.

Bibliography

[1] Samson Abramsky, Dan R. Ghica, Andrzej S. Murawski and Luke C.-H. Ong. Applying game semantics to compositional software modeling and verification. In *Proceedings of the 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'04* (2004), vol. 2988 of Lecture Notes in Computer Science, pp. 421–435. (Cited on page iii.)

[2] Rajeev Alur, Thomas A. Henzinger and Orna Kupferman. Alternating-time temporal logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science, FOCS '97* (1997), IEEE, pp. 100–109. (Cited on page 43.)

[3] Andre Arnold, Jacques Duparc, Filip Murlak and Damian Niwiński. On the topological complexity of tree languages. In *Logic and Automata: History and Perspectives*, vol. 2 of Texts in Logic and Games. Amsterdam University Press, 2007, pp. 9–29. (Cited on page 119.)

[4] Salman Azhar, Gary Peterson and John H. Reif. On multiplayer non-cooperative games of incomplete information: Part 2 - lower bounds. Technical report DUKE–TR–1991–38, Duke University, 1991. (Cited on page 43.)

[5] Vince Bárány. Invariants of automatic presentations and semi-synchronous transductions. In *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science, STACS'06* (2006), vol. 3884 of Lecture Notes in Computer Science, pp. 289–300. (Cited on page 14.)

[6] Vince Bárány. *Automatic Presentations of Infinite Structures*. Dissertation, RWTH Aachen, 2007. (Cited on page 14.)

[7] Vince Bárány, Łukasz Kaiser and Sasha Rubin. Cardinality and counting quantifiers on omega-automatic structures. In *Proceedings of the 25th International Symposium on Theoretical Aspects of Computer Science, STACS'08* (2008), S. Albers and P. Weil, Eds. (Cited on page 89.)

[8] Achim Blumensath. Automatic structures. Diploma thesis, RWTH Aachen, 1999. (Cited on pages vii, 16, 17, 89 and 102.)

[9] Achim Blumensath. Prefix-recognisable graphs and monadic second-order logic. Technical report AIB-2001-06, RWTH Aachen, 2001. (Cited on page 129.)

[10] Achim Blumensath and Erich Grädel. Automatic Structures. In *Proceedings of 15th IEEE Symposium on Logic in Computer Science LICS'00* (2000), pp. 51–62. (Cited on page iv.)

[11] Achim Blumensath and Erich Grädel. Finite presentations of infinite structures: Automata and interpretations. *Theory of Computing Systems 37* (2004), 641 – 674. (Cited on pages 14, 16, 17 and 91.)

[12] V. Bruyère, G. Hansel, Ch. Michaux and R. Villemaire. Logic and p-recognizable sets of integers. *Bull. Belg. Math. Soc. 1* (1994), 191 – 238. (Cited on page 14.)

[13] Julius R. Büchi. On decision method in restricted second order arithmetic. In *Proceedings of the International Congress on Logic, Methodology and Philosophy of Science* (1962), pp. 1–11. (Cited on pages iv, 7, 10 and 18.)

[14] Julius R. Büchi. Decision methods in the theory of ordinals. *Bulletin of the American Mathematical Society 71* (1965), 767–770. (Cited on page 18.)

[15] Julius R. Büchi and Dirk Siefkes. *The Monadic Second Order Theory of All Countable Ordinals*, vol. 328 of Lecture Notes in Mathematics. Springer, 1973. (Cited on page 18.)

[16] Alan Bundy. The automation of proof by mathematical induction. In *Handbook of Automated Reasoning*. Elsevier and MIT Press, 2001, pp. 845–911. (Cited on page iv.)

[17] Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger and Jean-Francois Raskin. Algorithms for omega-regular games with imperfect information. In *Proceedings of the 15th Annual Conference of the European Association for Computer Science Logic, CSL '06* (2006), vol. 4207 of Lecture Notes in Computer Science, Springer, pp. 287–302. (Cited on pages 41 and 131.)

[18] Thomas Colcombet and Christof Löding. Transforming structures by set interpretations. *Logical Methods in Computer Science 3*, 2:4 (2007), 1–36. (Cited on pages vii, 16 and 17.)

[19] Thomas Colcombet and Damian Niwinski. On the positional determinacy of edge-labeled games. *Theoretical Computer Science 352*, 1-3 (2006), 190–196. (Cited on page 84.)

[20] Bruno Courcelle, Joost Engelfriet and Grzegorz Rozenberg. Context-free handle-rewriting hypergraph grammars. In *Proceedings of the 4th International Workshop on Graph-Grammars and Their Application to Computer Science* (1991), vol. 532 of Lecture Notes in Computer Science, Springer, pp. 253–268. (Cited on page 129.)

[21] L. Doyen and J.-F. Raskin. Improved algorithms for the automata based approach to model-checking. In *Proceedings of TACAS 2007: Tools and Algorithms for the Construction and Analysis of Systems*, vol. 4424 of Lecture Notes in Computer Science. Springer, 2007, pp. 451–465. (Cited on page 131.)

[22] Stefan Dziembowski, Marcin Jurdziński and Igor Walukiewicz. How much memory is needed to win infinite games? In *Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science, LICS '97* (1997), IEEE Computer Society Press, pp. 99–110. (Cited on page 66.)

[23] Heinz-Dieter Ebbinghaus and J. Flum. *Finite Model Theory*. Perspectives in Mathematical Logic. Springer, 1995. (Cited on page 1.)

[24] Heinz-Dieter Ebbinghaus, Jörg Flum and Wolfgang Thomas. *Mathematical Logic*. Undergraduate texts in mathematics. Spinger-Verlag, 1984. (Cited on page 1.)

[25] Andrzej Ehrenfeucht. Decidability of the theory of the linear ordering relation. *Notices of the American Mathematical Society 6* (1959), 268. (Cited on page 18.)

[26] Andrzej Ehrenfeucht. An application of games to the completeness problem for formalized theories. *Fundamenta Mathematicae 44* (1961), 241–248. (Cited on page 18.)

[27] Jürgen Ehrensberger and Claus Zinn. Dialog: A system for dialogue logic. In *Proceedings of the 14th International Conference on Automated Deduction, CADE'14* (1997), vol. 1249 of Lecture Notes in Computer Science, Springer, pp. 446–460. (Cited on page iii.)

[28] E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science, FoCS '91* (1991), IEEE, pp. 368–377. (Cited on pages 37 and 64.)

[29] David Gale and Frank M. Stewart. Infinite games with perfect information. In *Contributions to the Theory of Games II*, vol. 28 of Annals of Mathematical Studies. Princeton University Press, 1953, pp. 245–266. (Cited on page 24.)

[30] Georg Gottlob. Relativized logspace and generalized quantifiers over finite ordered structures. *Journal of Symbolic Logic 62*, 2 (1997), 545–574. (Cited on page vii.)

[31] Erich Grädel and Łukasz Kaiser. What kind of memory is needed to win infinitary Muller games? In *Proceedings of the 7th Augustus de Morgan Workshop on Interactive Logic, Games and Social Software* (2007), vol. 1 of Texts in Logic and Games, Amsterdam University Press, pp. 89–116. (Cited on page 61.)

[32] Erich Grädel and Igor Walukiewicz. Positional determinacy of games with infinitely many priorities. *Logical Methods in Computer Science* (2006). (Cited on pages 67, 68 and 71.)

[33] Yuri Gurevich. Modest theory of short chains I. *Journal of Symbolic Logic 44* (1979), 481–490. (Cited on pages 18 and 19.)

[34] Yuri Gurevich. Monadic second-order theories. In *Model-Theoretical Logics*, J. Barwise and S. Feferman, Eds. Springer, 1985, pp. 479–506. (Cited on page 18.)

[35] Yuri Gurevich and Leo Harrington. Trees, automata and games. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, STOC '82* (1982), ACM, pp. 60–65. (Cited on pages vi, 61 and 64.)

[36] Yuri Gurevich, Menachem Magidor and Saharon Shelah. The monadic theory of ω_2. *Jounal of Symbolic Logic 48* (1983), 387–398. (Cited on pages 18 and 19.)

[37] Yuri Gurevich and Saharon Shelah. Modest theory of short chains II. *Journal of Symbolic Logic 44* (1979), 491–502. (Cited on pages 18 and 19.)

[38] Jakko Hintikka. Language-games for quantifiers. *Studies in Logical Theory, American Philosophical Quarterly Monograph Series 2* (1986), 46–72. (Cited on page iii.)

[39] Greg Hjörth, Bakhadyr Khoussainov, Antonio Montalbán and André Nies. Borel structures. Manuscript, 2007. (Cited on page 103.)

[40] B.R. Hodgson. Décidabilité par automate fini. *Ann. sc. math. Québec 7(1)* (1983), 39–57. (Cited on pages iv, 14 and 91.)

[41] David Janin and Igor Walukiewicz. On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic. In *Proceedings of the 7th International Conference on Concurrency Theory, CONCUR '96* (1996), vol. 1119 of Lecture Notes in Computer Science, Springer-Verlag, pp. 263–277. (Cited on page 39.)

[42] Marcin Jurdziński. Deciding the winner in parity games is in UP ∩ co-UP. *Information Processing Letters 68*, 3 (1998), 119–124. (Cited on pages iii and 40.)

[43] Marcin Jurdziński. Small progress measures for solving parity games. In *Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science, STACS'00* (2000), vol. 1770 of Lecture Notes in Computer Science, Springer, pp. 290–301. (Cited on pages iii and 40.)

[44] Marcin Jurdzinski, Mike Paterson and Uri Zwick. A deterministic subexponential algorithm for solving parity games. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'06* (2006), ACM Press, pp. 117–123. (Cited on pages iv and 40.)

[45] Łukasz Kaiser. Game quantification on automatic structures and hierarchical model checking games. In *Proceedings of the 15th Annual Conference on Computer Science Logic, CSL'06* (2006), Z. Esik, Ed., vol. 4207 of Lecture Notes in Computer Science, Springer-Verlag, pp. 411–425. (Cited on page 35.)

[46] Bakhadyr Khoussainov and Anil Nerode. Automatic presentations of structures. In *LCC '94* (1995), vol. 960 of Lecture Notes in Computer Science, Springer-Verlag, pp. 367–392. (Cited on pages iv, 14 and 91.)

[47] Bakhadyr Khoussainov, Sasha Rubin and Frank Stephan. Definability and regularity in automatic presentations of subsystems of arithmetic. Tech. rep., University of Auckland, 2003. (Cited on page 14.)

[48] Bakhadyr Khoussainov, Sasha Rubin and Frank Stephan. Definability and regularity in automatic structures. In *Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science, STACS'04* (2004), vol. 2996 of Lecture Notes in Computer Science, pp. 440–451. (Cited on pages 14, 91 and 101.)

[49] Phokion G. Kolaitis. Game quantification. In *Model-Theoretical Logics*, J. Barwise and S. Feferman, Eds. Springer, 1985, pp. 365–422. (Cited on pages 21 and 25.)

[50] Dietrich Kuske and Markus Lohrey. First-order and counting theories of ω-automatic structures. In *Proceedings of the 9th International Conference on Foundations of Software Science and Computation Structures, FOSSACS'06* (2006), pp. 322–336. (Cited on pages 91, 93, 101, 112 and 123.)

[51] Per Lindström. First order predicate logic with generalized quantifiers. *Theoria 32* (1966), 186–195. (Cited on pages vii and 89.)

[52] Paul Lorenzen. Logik und agon (reprinted in "Dialogische Logik"). In *Arti del XII Congresso Internationale de Filosofia* (1958), pp. 187–194. (Cited on page iii.)

[53] Paul Lorenzen. Ein dialogisches konstruktivitätskriterium. In *Infinitistic Methods, Proceedings of the Symposium on Foundations of Mathematics* (1961), Polskie Wydawnictwo Naukowe, pp. 193–200. (Cited on page iii.)

[54] Paul Lorenzen and Kuno Lorenz. *Dialogische Logik*. Wissenschaftliche Buchgeselschaft, 1978. (Cited on page iii.)

[55] Donald A. Martin. Borel determinacy. *Annals of Mathematics 102* (1975), 363–371. (Cited on pages 24 and 29.)

[56] Peter McBurney and Simon Parsons. Dialogue games in multi-agent systems. *Informal Logic (Special Issue on Applications of Argumentation in Computer Science) 22*, 3 (2002), 257–274. (Cited on page iii.)

[57] Satoru Miyano and Takeshi Hayashi. Alternating finite automata on ω-words. *Theoretical Computer Science 32* (1984), 321–330. (Cited on page 10.)

[58] Yiannis N. Moschovakis. *Descriptive Set Theory*, vol. 100 of Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Company, 1980. (Cited on pages 118 and 119.)

[59] Andrzej W. Mostowski. Games with forbidden positions. Tech. Rep. 78, Instytut Matematyki, Uniwersytet Gdański, Poland, 1991. (Cited on pages 37 and 64.)

[60] Damian Niwiński. On the cardinality of sets of infinite trees recognizable by finite automata. In *Proceedings of the 16th International Symposium on Mathematical Foundations of Computer Science, MFCS'91* (1991), vol. 520, Springer, pp. 367–376. (Cited on page 106.)

[61] Damian Niwinski and Igor Walukiewicz. A gap property of deterministic tree languages. *Theoretical Computer Science 1*, 303 (2003), 215–231. (Cited on page 119.)

[62] Dominique Perrin and Jean-Eric Pin. Semigroups and automata on infinite words. In *Semigroups, Formal Languages and Groups* (1995), J. Fountain, Ed., NATO Advanced Study Institute, Kluwer academic publishers, pp. 49–72. (Cited on pages 10 and 11.)

[63] Dominique Perrin and Jean-Eric Pin. *Infinite Words (Automata, Semigroups, Logic and Games)*. Elsevier, 2004. (Cited on page 10.)

[64] Sophie Pinchinat and Stéphane Riedweg. A decidable class of problems for control under partial observation. *Information Processing Letters 95*, 4 (2005), 454–460. (Cited on page 47.)

[65] Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society 141* (1969), 1–35. (Cited on page 7.)

[66] Alexander Rabinovich and Wolfgang Thomas. Decidable theories of the ordering of natural numbers with unary predicates. In *Proceedings of the 15th Annual Conference on Computer Science Logic, CSL'06* (2006), Z. Esik, Ed., vol. 4207 of Lecture Notes in Computer Science, Springer-Verlag, pp. 562–574. (Cited on page 18.)

[67] John H. Reif. Universal games of incomplete information. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing, STOC '79* (1979), ACM, pp. 288–308. (Cited on page 41.)

[68] John H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and Systems Sciences 29*, 2 (1984), 274–301. (Cited on page 41.)

[69] Sasha Rubin. Automatic structures. Ph.D. thesis, University of Auckland, New Zealand, 2004. (Cited on pages 16 and 17.)

[70] Sasha Rubin. Automata presenting structures: a survey of the finite string case. *Bulletin of Symbolic Logic* (2008). To appear. (Cited on page 91.)

[71] Saharon Shelah. The monadic theory of order. *Annals of Mathematics 102* (1975), 379–419. (Cited on pages 18 and 19.)

[72] Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics*,

J. van Leeuwen, Ed. Elsevier and MIT Press, 1990, pp. 133–192. (Cited on page 1.)

[73] Wolfgang Thomas. Ehrenfeucht games, the composition method, and the monadic theory of ordinal words. In *Structures in Logic and Computer Science*, vol. 1261 of Lecture Notes in Computer Science. Springer, 1997, pp. 118–143. (Cited on page 18.)

[74] Wolfgang Thomas. Languages, automata, and logic. In *Handbook of Formal Languages, volume III*, G. Rozenberg and A. Salomaa, Eds. Springer, New York, 1997, pp. 389–455. (Cited on page 1.)

[75] Wolfgang Thomas and Helmut Lescow. Logical specifications of infinite computations. In *REX School/Symposium* (1993), J. W. de Bakker, W. P. de Roever, and G. Rozenberg, Eds., vol. 803 of Lecture Notes in Computer Science, Springer, pp. 583–621. (Cited on page 119.)

[76] Jens Vöge and Marcin Jurdziński. A discrete strategy improvement algorithm for solving parity games. In *Proceedings of the 12th International Conference on Computer Aided Verification, CAV 2000* (2000), vol. 1855 of Lecture Notes in Computer Science, Springer, pp. 202–215. (Cited on pages iii and 40.)

[77] Thomas Wilke. An Eilenberg Theorem for ω-languages. In *Proceedings of the 18th International Collquium on Automata, Languages and Programming, ICALP'91* (1991), J. L. A. et al., Ed., vol. 510 of Lecture Notes in Computer Science, Springer, pp. 588–599. (Cited on page 10.)

[78] Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *TCS 200*, 1–2 (1998), 135–183. (Cited on pages vi, 62, 82 and 88.)

Index