Choice Functions and Well-Orderings over the Infinite Binary Tree

Arnaud Carayol
Laboratoire d'Informatique Gaspard Monge
Université Paris-Est & CNRS

Christof Löding
Lehrstuhl Informatik 7
RWTH Aachen

Damian Niwiński*
Institute of Informatics
University of Warsaw

Igor Walukiewicz
LaBRI
Bordeaux University & CNRS

Abstract

We give a new proof showing that it is not possible to define in monadic second-order logic (MSO) a choice function on the infinite binary tree. This result was first obtained by Gurevich and Shelah using set theoretical arguments. Our proof is much simpler and only uses basic tools from automata theory. We show how the result can be used to prove the inherent ambiguity of languages of infinite trees. In a second part we strengthen the result of the non-existence of an MSO-definable well-founded order on the infinite binary tree by showing that every infinite binary tree with a well-founded order has an undecidable MSO-theory.

1 Introduction

The main goal of this paper is to present a simple proof for the fact (first shown by Gurevich and Shelah in [11]) that there is no MSO-definable choice function on the infinite binary tree $\mathfrak{t}_2$. A choice function on $\mathfrak{t}_2$ is a mapping assigning to each nonempty set of nodes of $\mathfrak{t}_2$ one element from this set, i.e., the function chooses for each set one of its elements. Such a function is MSO-definable if there is an MSO-formula with one free set variable X and one free element variable x such that when X is interpreted as a nonempty set U then there is exactly one possible interpretation $u \in U$ for x making the formula satisfied.

*Supported by the Polish government Grant N206 008 32/0810.

The question of the existence of an MSO-definable choice function over the infinite binary tree can be seen as a special instance of the more general uniformization problem, which asks, given a relation that is defined by a formula with free variables, whether it is possible to define by another formula a function that is compatible with this relation. More precisely, given a formula $\phi(\overline{X}, \overline{Y})$ with vectors $\overline{X}, \overline{Y}$ of free variables, such that the formula $\forall \overline{X} \exists \overline{Y} \phi(\overline{X}, \overline{Y})$ is true over the infinite binary tree, uniformization asks for a formula $\phi^*(\overline{X}, \overline{Y})$ such that

1. ϕ^* implies ϕ (each interpretation of $\overline{X}, \overline{Y}$ making ϕ^* true also makes ϕ true),
2. and ϕ^* defines a function in the sense that for each interpretation of $\overline{X}$ there is exactly one interpretation of $\overline{Y}$ making ϕ^* true.

The question of the existence of a choice function is the uniformization problem for the formula $\phi(X, Y) := X \neq \emptyset \rightarrow (Y \text{ is a singleton} \wedge Y \subseteq X)$.

The infinite line, i.e., the structure consisting of natural numbers equipped with the successor relation, is known to have the uniformization property for MSO [25], even when adding unary predicates [21]. On the infinite binary tree MSO is known to be decidable [2] but it does not have the uniformization property. This was conjectured in [25] and proved in [11] where it is shown that there is no MSO-definable choice function on the infinite binary tree. The proof in [11] uses complex set theoretical arguments, whereas it appears that the result can be obtained by much more basic techniques. We show that this is indeed true and present a proof that only relies on the equivalence of MSO and automata over infinite trees and otherwise only uses basic techniques from automata theory. Besides its simplicity, another advantage of the proof is that we provide a concrete family of sets (parametrized by natural numbers) such that each formula fails to make a choice for those sets with the parameters chosen big enough. We use this fact when we discuss two applications of the result.

At first, we show an example of a tree game with an MSO-definable winning condition, where the winner has no MSO-definable winning strategy. The second application concerns the existence of inherently ambiguous tree languages. An unambiguous automaton is an automaton which has exactly one accepting run on every tree it accepts. On finite words and trees it is clear that every regular language can be accepted by an unambiguous automaton because deterministic automata are unambiguous. The regular languages of infinite words are accepted by unambiguous Büchi automata [1]. For regular languages of infinite trees, we show that the language of infinite trees labeled by $\{0, 1\}$ having at least one node labeled by 1 is not accepted by any unambiguous parity tree automaton. Using the counter-example sets introduced previously, we strengthen this result by giving a regular tree language and a tree in this language such that any automaton accepting the language has at least two accepting runs on the tree.

The subject of MSO-definability of choice functions on trees has been studied in more depth in [13], where the authors consider more general trees and not only the infinite binary tree. They show the following dichotomy: for a tree it is either not possible to define a choice function in MSO even with additional predicates, or it is possible to define a well-ordering on the domain of the tree with additional predicates.

Note that it is very easy to define a choice function if one has access to a well-ordering of the domain: for each set one chooses the minimal element according to the well-ordering. The same result remains true if we only have access to a partial well-ordering: we pick the left-most element of the finite set of minimal elements instead of the minimal element. The non-existence of an MSO-definable choice function on the infinite binary tree implies that there is no MSO-definable partial well-ordering on the infinite binary tree. We strengthen this result by showing that extending the infinite binary tree by any partial well-ordering leads to a structure with an undecidable MSO-theory. As a consequence we obtain that each structure in which we can MSO-define a partial well-ordering and MSO-interpret the infinite binary tree must have an undecidable MSO-theory.

The paper is structured as follows. In Section 2 we introduce notations and basic results for automata on infinite trees and monadic second-order logic. The proof of the undefinability of choice functions over the infinite binary tree is presented in Section 3, where we also discuss the first application concerning the definability of winning strategies in infinite games. In Section 4 we prove that there are regular languages of infinite trees that are inherently unambiguous. Section 5 is on the undecidability of the monadic second-order theory of the infinite binary tree extended with a partial well-ordering. We conclude in Section 6 and give some possible directions of future research.

This paper is a full version of the conference publication [4].

2 Preliminaries

In this section we first introduce some general notations and then give some background on automata and logic.

The set of natural numbers (non-negative integers) is denoted by $\mathbb{N}$. A finite interval $\{i, \ldots, j\}$ of natural numbers is written as $[i, j]$.

An *alphabet* is a finite set of symbols, called *letters*. Usually, alphabets are denoted by Σ. The set of finite words over Σ is written as Σ^* and the set of infinite words as Σ^ω. The length of a finite word $w \in \Sigma^*$ is denoted by $|w|$ and ε is the empty word.

For all words $w_1, w_2 \in \Sigma^*$, w_1 is a *prefix* of w_2 (written $w_1 \sqsubseteq w_2$) if there exists $w \in \Sigma^*$ such that $w_2 = w_1 w$. If $w \in \Sigma^+$ then w_1 is a *strict prefix* of w_2 (written $w_1 \sqsubset w_2$). The *greatest common prefix* of two words w_1 and w_2 (written $w_1 \wedge w_2$) is the longest word which is a prefix of w_1 and w_2.

2.1 Automata on infinite trees

We view the set $\{0,1\}^*$ of finite words over $\{0,1\}$ as the *domain* of an infinite binary tree. The *root* is the empty word ε, and for a node $u \in \{0,1\}^*$ we call $u0$ its left successor (or 0-successor), and $u1$ its right successor (or 1-successor). A *branch* π is an infinite sequence $u_0, u_1, u_2 \dots$ of successive nodes starting with the root, i.e., $u_0 = \varepsilon$ and $u_{i+1} = u_i0$ or $u_{i+1} = u_i1$ for all $i \geq 0$.

An (infinite binary) *tree* labeled by a finite alphabet Σ is a mapping $t : \{0,1\}^* \to \Sigma$. We denote by $\mathcal{T}^\omega_\Sigma$ the set of all trees labeled by Σ. In a tree t, a branch π induces an infinite sequence of labels in Σ^ω. We sometimes identify a branch with this infinite word. It should always be clear from the context to which meaning of a branch we are referring to.

In connection with logic, the labels of the infinite tree are used to represent interpretations of set variables. This motivates the following definitions. For a set $U \subseteq \{0,1\}^*$, we write $t[U] \in \mathcal{T}^\omega_{\{0,1\}}$ for the *characteristic tree* of U, i.e., the tree which labels all nodes in U with 1 and all the other nodes with 0. This notation is extended to the case of several sets. The characteristic tree of $U_1, \dots, U_n \subseteq \{0,1\}^*$ is the tree labeled by $\{0,1\}^n$ written $t[U_1, \dots, U_n]$ and defined for all $u \in \{0,1\}^*$ by $t[U_1, \dots, U_n](u) := (b_1, \dots, b_n)$ where for all $i \in [1,n]$, $b_i = 1$ if $u \in U_i$ and $b_i = 0$ otherwise. For a singleton set $U = \{u\}$ we simplify notation and write $t[u]$ instead of $t[\{u\}]$.

We now turn to the definition of automata for infinite trees. A *parity tree automaton* (PTA) on Σ-labeled trees is a tuple $\mathcal{A} = (Q, \Sigma, q^{\mathrm{i}}, \Delta, c)$ with a finite set Q of states, an initial state $q^{\mathrm{i}} \in Q$, a transition relation $\Delta \subseteq Q \times \Sigma \times Q \times Q$, and a priority function $c : Q \to \mathbb{N}$. A *run* of $\mathcal{A}$ on a tree $t \in \mathcal{T}^\omega_\Sigma$ from a state $q \in Q$ is a tree $\rho \in \mathcal{T}^\omega_Q$ such that $\rho(\varepsilon) = q$, and for each $u \in \{0,1\}^*$ we have $(\rho(u), t(u), \rho(u0), \rho(u1)) \in \Delta$. We say that ρ is *accepting* if the parity condition is satisfied on each branch of the run, i.e., on each branch the minimal priority appearing infinitely often is even. If we only speak of a run of $\mathcal{A}$ without specifying the state at the root, we implicitly refer to a run from q^{i}.

A tree t is accepted by $\mathcal{A}$ if there is an accepting run of $\mathcal{A}$ on t. The *language recognized* by $\mathcal{A}$ is the set of all accepted trees:

$$T(\mathcal{A}) = \{t \in \mathcal{T}^\omega_\Sigma \mid t \text{ accepted by } \mathcal{A}\}.$$

A tree language is called *regular* if it is recognized by some PTA.

For two trees t and t' we say that they are $\mathcal{A}$-*equivalent*, written as $t \equiv_{\mathcal{A}} t'$, if for each state q of $\mathcal{A}$ there is an accepting run from q on t if and only if there is an accepting run from q on t'. Intuitively, this means that $\mathcal{A}$ cannot distinguish the two trees.

2.2 Monadic second-order logic

A *signature* is a ranked set τ of symbols, where for all $R \in \tau$, $|R|$ denotes the arity (which is ≥ 1) of the symbol R. A *relational structure* $\mathcal{S}$ over τ is given by a tuple $(D, (R^{\mathcal{S}})_{R\in\tau})$ where D is the *domain* (or the *universe*) of $\mathcal{S}$ and where for all $R \in \tau$, $R^{\mathcal{S}}$ (which is also called the interpretation of R in $\mathcal{S}$) is a subset of $D^{|R|}$. When $\mathcal{S}$ is clear from the context, we simply write R instead of $R^{\mathcal{S}}$.

We use $\cong$ to denote that two structures are isomorphic. Usually, we do not distinguish isomorphic structures but sometimes we refer to different representations of structures over specific domains, and in these cases we use $\cong$ instead of $=$.

To every tree t labeled by $\Sigma = \{a_1, \ldots, a_n\}$, we associate a canonical structure over the signature $\{E_0, E_1, P_{a_1}, \ldots, P_{a_n}\}$ where E_0 and E_1 are binary symbols and the P_{a_i} are *predicates*, i.e., unary symbols. The universe of this structure is $\{0,1\}^*$. The symbols E_0 and E_1 are respectively interpreted as $\{(w, w0) \mid w \in \{0,1\}^*\}$ and $\{(w, w1) \mid w \in \{0,1\}^*\}$. Finally for all $i \in [1, n]$, P_{a_i} is interpreted as $\{u \in \Sigma^* \mid t(u) = a_i\}$. In the following, we do not distinguish between a tree and its canonical relational structure.

The unlabeled binary tree, i.e., the structure $(\{0,1\}^*, \{E_0, E_1\})$ with the above interpretation of E_0 and E_1 is denoted by $\mathfrak{t}_2$. The structure with $\mathbb{N}$ as domain and the successor relation on $\mathbb{N}$ can be viewed as a tree with unary branching. Hence we denote it by $\mathfrak{t}_1$.

We are mainly interested in *monadic second-order logic* (MSO) over relational structures with the standard syntax and semantics (see e.g. [9] for a detailed presentation). MSO-formulas use first-order variables, which are interpreted by elements of the structure, and monadic second-order variables, which are interpreted as sets of elements. First-order variables are usually denoted by lower case letters (e.g. x, y), and monadic second-order variables are denoted by capital letters (e.g. X, Y). First-order logic (FO) is the fragment of MSO that does not use quantifications over set variables.
Atomic MSO-formulas are of the form

- $R(x_1, \ldots, x_{|R|})$ for a relation symbol R from the signature and first-order variables $x_1, \ldots, x_{|R|}$, or
- $x = y$, $X = Y$, or $x \in X$ for first-order variables x, y, and monadic second-order variables X, Y

with the obvious semantics. Complex formulas are built as usual from atomic ones by the use of boolean connectives, and quantification.

We write $\phi(X_1, \ldots, X_n, y_1, \ldots, y_m)$ to indicate that the free variables of the formula ϕ are among $X_1, \ldots, X_n$ (monadic second-order) and $y_1, \ldots, y_m$ (first-order) respectively. A formula without free variables is called a *sentence*.

For a relational structure $\mathcal{S}$ and a sentence ϕ, we write $\mathcal{S} \models \phi$ if $\mathcal{S}$ *satisfies* the formula ϕ. The *MSO-theory* of $\mathcal{S}$ is the set of sentences satisfied by $\mathcal{S}$. For every formula $\phi(X_1, \ldots, X_n, y_1, \ldots, y_m)$, all subsets $U_1, \ldots, U_n$ of the universe of $\mathcal{S}$ and all elements $v_1, \ldots, v_m$ of the universe of $\mathcal{S}$, we write $\mathcal{S} \models \phi[U_1, \ldots, U_n, v_1, \ldots, v_m]$ to express that ϕ holds in $\mathcal{S}$ when X_i is interpreted as U_i for all $i \in [1, n]$ and y_j is interpreted as v_j for all $j \in [1, m]$.

Given a structure $\mathcal{S}$, we calla relation $R \subseteq D^n$, for some $n \geq 1$, *MSO-definable* in $\mathcal{S}$ if there is an MSO-formula $\phi(x_1, \ldots, x_n)$ with n free first-order variables such that $(u_1, \ldots, u_n) \in R$ iff $\mathcal{S} \models \phi[u_1, \ldots, u_n]$.

A single element u of a structure is MSO-definable if so is the unary relation $\{u\}$, i.e., if there is a formula $\phi(x)$ such that u is the only element with $\mathcal{S} \models \phi[u]$.

We are particularly interested in MSO logic over the infinite binary tree $\mathfrak{t}_2$, which we refer to as MSO[$\mathfrak{t}_2$]. The MSO-theory of the infinite binary tree $\mathfrak{t}_2$ is also referred to as S2S, the second-order theory of two successors [19]. Correspondingly, we refer to the MSO-theory of $\mathfrak{t}_1$, the natural numbers with successor, as S1S [2].

An MSO[$\mathfrak{t}_2$]-formula $\phi(X_1, \dots, X_n)$ with n free variables defines a tree language $T(\phi) \subseteq \mathcal{T}^{\omega}_{\{0,1\}^n}$ as follows:

$$T(\phi) = \{t[U_1, \dots, U_n] \mid \mathfrak{t}_2 \models \phi[U_1, \dots, U_n]\}.$$

It is often convenient to consider the binary tree $\mathfrak{t}_2$ along with a sequence of sets $U_1, \dots, U_n \subseteq \{0,1\}^*$, as a new logical structure over the signature extended by n fresh predicate symbols interpreted by $U_1, \dots, U_n$. We denote this structure by $\mathfrak{t}_2[U_1, \dots, U_n]$. Given a formula $\phi(X_1, \dots, X_n)$, we clearly have

$$\mathfrak{t}_2 \models \phi[U_1, \dots, U_n] \quad \text{iff} \quad \mathfrak{t}_2[U_1, \dots, U_n] \models \phi,$$

where in the latter case ϕ is considered as a sentence with the predicate symbols $X_1, \dots, X_n$ interpreted by $U_1, \dots, U_n$, respectively. This correspondence extends to formulas with additional free variables, say $\phi(X_1, \dots, X_n, Z_1, \dots, Z_k, y_1, \dots, y_m)$. That is, $\mathfrak{t}_2 \models \phi[U_1, \dots, U_n, W_1, \dots, W_k, v_1, \dots, v_m]$ if and only if $\mathfrak{t}_2[U_1, \dots, U_n] \models \phi[W_1, \dots, W_k, v_1, \dots, v_m]$, for any $W_1, \dots, W_k \subseteq \{0,1\}^*$, and $v_1, \dots, v_m \in \{0,1\}^*$. We will often use this correspondence implicitly in the sequel. Note that in general there may be more relations definable in $\mathfrak{t}_2[U_1, \dots, U_n]$ than in $\mathfrak{t}_2$.

A theorem of Rabin states that the tree languages definable by MSO[$\mathfrak{t}_2$]-formulas are precisely those that can be accepted by tree automata [19]. An analogous fact for *MSO*[$\mathfrak{t}_1$]-definable languages of infinite words (also called ω-regular) was proved previously by Büchi [2]. The acceptance condition used by Rabin in [19] differs from the parity condition but it can be transformed into a parity condition (see e.g. [29, 10]). Parity tree automata have first been used in [18] where they are shown to be equivalent to the automata used in [19].

We state here the difficult direction of the theorem, the proof of which is based on the closure properties of PTAs.

Theorem 2.1 ([19, 18]) *For every MSO*[$\mathfrak{t}_2$]*-formula $\phi(X_1, \dots, X_n)$ there is a parity tree automaton $\mathcal{A}_\phi$ such that $T(\mathcal{A}_\phi) = T(\phi)$.*

This result can be used to show the decidability of the satisfiability problem for MSO[$\mathfrak{t}_2$], i.e., the question whether for a given MSO[$\mathfrak{t}_2$] formula there is an interpretation of the free variables such that the formula is satisfied in $\mathfrak{t}_2$. Furthermore, one can even construct a satisfying assignment:

Theorem 2.2 ([20]) *Let $\phi(X_1, \dots, X_n)$ be a satisfiable MSO*[$\mathfrak{t}_2$]*-formula. There are regular sets $U_1, \dots, U_n$ such that $\mathfrak{t}_2[U_1, \dots, U_n] \models \phi$.*

Above we have introduced the notion of MSO-definability. We conclude from Theorem 2.2 that on $\mathfrak{t}_2$ each MSO-definable set is regular: Starting from a formula $\phi(x)$ defining a set, we construct the formula $\phi'(X) = \forall x. x \in X \leftrightarrow \phi(x)$. Then there is exactly one set satisfying ϕ' and this set is regular according to Theorem 2.2. Conversely, each regular set can be defined in MSO by describing an accepting run of the DFA defining the set.

PROPOSITION 2.3 *A set $U \subseteq \{0,1\}^*$ is definable in MSO$[\mathfrak{t}_2]$ iff it is regular.*

As a consequence we can add regular predicates to $\mathfrak{t}_2$ without affecting the decidability result for MSO. An MSO-formula having access to regular predicates $P_1, \dots, P_n$ can be turned in a standard MSO-formula over $\mathfrak{t}_2$ by using formulas $\phi_{P_1}, \dots, \phi_{P_n}$ defining these predicates.

The notion of interpretation that is introduced in the following generalizes this idea.

Let τ and $\hat{\tau}$ be two signatures. An *MSO-interpretation* from τ-structures to $\hat{\tau}$-structures is given as a list $\mathcal{I} = \langle \phi_{dom}, (\phi_{\hat{R}})_{\hat{R} \in \hat{\tau}} \rangle$ of MSO-formulas over the signature τ. The formula ϕ_{dom} has one free first-order variable and is called the *domain formula*. For each $\hat{R} \in \hat{\tau}$ the formula $\phi_{\hat{R}}$ has $|\hat{R}|$ many free first-order variables.

Applied to a τ-structure $\mathcal{S} = (D, (R^{\mathcal{S}})_{R \in \tau})$, the interpretation defines a $\hat{\tau}$-structure $\mathcal{I}(\mathcal{S}) = (\hat{D}, (\hat{R}^{\mathcal{I}(\mathcal{S})})_{\hat{R} \in \hat{\tau}})$, where the domain formula defines the domain of the new structure (all elements of $\mathcal{S}$ satisfying ϕ_{dom}), and the formulas $\phi_{\hat{R}}$ define the relations of $\mathcal{I}(\mathcal{S})$:

- $\hat{D} = \{u \in D \mid \mathcal{S} \models \phi_{dom}[u]\}$,
- $\hat{R}^{\mathcal{I}(\mathcal{S})} = \{(u_1, \dots, u_{|\hat{R}|}) \in \hat{D}^{|\hat{R}|} \mid \mathcal{S} \models \phi_{\hat{R}}[u_1, \dots, u_n]\}$.

Now assume that we are given an MSO-formula over $\mathcal{I}(\mathcal{S})$ and we want to know whether it is satisfiable in $\mathcal{I}(\mathcal{S})$. We can replace each atomic formula $\hat{R}(x_1, \dots, x_{|\hat{R}|})$ by its definition $\phi_{\hat{R}}(x_1, \dots, x_n)$, and restrict the range of all variables to the set $\hat{D}$ defined by ϕ_{dom}. In this way we obtain an MSO-formula over $\mathcal{S}$ that is satisfiable in $\mathcal{S}$ iff the original formula is satisfiable in $\mathcal{I}(\mathcal{S})$. Applying this technique to MSO-sentences (without free variables), we can transfer the decidability of the MSO-theory as stated in the following proposition.

PROPOSITION 2.4 *If $\mathcal{I}$ is an MSO-interpretation and if the MSO-theory of $\mathcal{S}$ is decidable, then the MSO-theory of $\mathcal{I}(\mathcal{S})$ is decidable.*

3 MSO-definable choice functions

As already described in the introduction, an MSO-definable choice function is given by an MSO-formula $\phi(X, x)$ such that:

> for every non-emptyset U, there is precisely one $u \in U$ such that $\mathfrak{t}_2 \models \phi[U, u]$.

This section is devoted to the proof of the following theorem of Gurevich and Shelah.

THEOREM 3.1 ([11]) *There is no MSO-definable choice function on the infinite binary tree.*

The technical formulation of the result we prove is given in Theorem 3.5 (on page 12), where we concretely provide counter examples for which a given formula cannot choose a unique element. As a machinery for the proof we use tree automata, that are easier to manipulate (at least for our purpose) than formulas. In the following we present a slight modification of the standard automaton model that we use in the proof.

An MSO[t_2]-formula with free variables defines a relation between subsets of $\{0,1\}^*$. In this section we consider formulas with two free variables, and a natural view in the context of choice functions is that the first variable represents the input, and the second variable represents the output. A choice function is the special case that the second variable is an element and not a set, and that for each nonempty input set there is a unique output which is inside the input set. To adapt this general view of inputs and outputs on the automata theoretic level we introduce automata with output, called transducers. These transducers are very simple: An output symbol is associated to each transition, which gives a natural correspondence between runs and output trees.

A *parity tree automaton with output* is a tuple $\mathcal{A} = (Q, \Sigma, q^{\mathrm{i}}, \Delta, c, \lambda)$, where $(Q, \Sigma, q^{\mathrm{i}}, \Delta, c)$ is a standard PTA, and $\lambda : \Delta \to \Gamma$ is an output function assigning to each transition a symbol from the output alphabet Γ. When ignoring the output function we can use the terminology of standard PTAs, in particular the notion of accepting run and the notion of $\mathcal{A}$-equivalence (see page 4).

In general, a PTA with output defines for each input tree $t \in \mathcal{T}^\omega_\Sigma$ a set of output trees $\mathcal{A}(t) \subseteq \mathcal{T}^\omega_\Gamma$ defined by

$$\mathcal{A}(t) = \{\lambda(\rho) \mid \rho \text{ is an accepting run of } \mathcal{A} \text{ on } t\},$$

where $\lambda(\rho)$ denotes the tree in $\mathcal{T}^\omega_\Gamma$ that is obtained by taking at each node u the output produced by λ applied to the transition used at u in ρ.

As already indicated above we are interested in such transducers for representing formulas with two free variables, i.e., the input and the output alphabets are both equal to $\{0,1\}$. The following theorem can easily be shown by a simple argument based on Theorem 2.1.

THEOREM 3.2 *For each MSO*[t_2]*-formula* $\phi(X, Y)$ *there is a PTA with output* $\mathcal{A}_\phi$ *such that* $\mathsf{t}_2[U, U'] \models \phi$ *iff* $\mathsf{t}_2[U'] \in \mathcal{A}(\mathsf{t}_2[U])$.

We are interested in PTAs with output that represent possible candidates for choice formulas. i.e., the case where the second free variable of the formula represents a single element. This motivates the following definition.

A *weak choice automaton* is a PTA with output such that the input and the output alphabet are equal to $\{0,1\}$, and for each $U \subseteq \{0,1\}^*$ there is at most one output tree in $\mathcal{A}(\mathfrak{t}_2[U])$, and this output tree is of the form $\mathfrak{t}_2[u]$ for some $u \in U$. We say that $\mathcal{A}$ chooses u in U. A weak choice automaton chooses for each set U at most one element from this set. It is called weak because it does not have to choose an element. A weak choice automaton that chooses one element from each nonempty set represents a choice formula. Our goal is to show that such an automaton cannot exist. Note that there are weak choice automata: for example the automaton that rejects all inputs is a weak choice automaton because it does not choose an element for any set.

We show in Lemma 3.4 that for each weak choice automaton $\mathcal{A}$ we can find a set that is complex enough such that $\mathcal{A}$ cannot choose an element of U. If $\mathcal{A}$ had a run choosing an element of U, we could construct another run choosing a different element, contradicting the property of a weak choice automaton. More precisely, we define a family $(U_{M,N})_{M,N\in\mathbb{N}}$ of sets such that for each weak choice automaton $\mathcal{A}$ we can find M and N such that $\mathcal{A}$ cannot choose an element of $U_{M,N}$. To achieve this, we "hide" the elements of the set very deep in the tree so that weak choice automata up to a certain size are not able to uniquely choose an element.

For $M, N \in \mathbb{N}$ the set $U_{M,N} \subseteq \{0,1\}^*$ is defined by the following regular expression

$$U_{M,N} = \{0,1\}^*(0^N 0^* 1)^M \{0,1\}^*.$$

In Figure 1 a deterministic finite automaton for the set $U_{M,N}$ is shown, where the dashed edges represent transitions for input 1 that lead back to the initial state x_M. The chains of 0-edges between x_{k+1} and x_k have length N. The final state is x_0. It is easy to verify that x_0 is reachable from x_M by exactly those paths whose sequence of edge labels is in the set $U_{M,N}$. Let $t_{M,N} = t[U_{M,N}]$ and let $t_{k,M,N} = \mathfrak{t}_2[U_{k,M,N}]$, where $U_{k,N,M}$ is the set that we obtain by making x_k the initial state.

We now fix a weak choice automaton $\mathcal{A} = (Q, \{0,1\}, q_0, \Delta, c, \lambda)$ on $\{0,1\}$-labeled trees and take $M = 2^{|Q|}+1$ and $N = |Q|+1$. For these fixed parameters we simplify the notation by letting $t_k = t_{k,M,N}$. In particular, $t_M = t_{M,M,N} = t_{M,N}$. We say that a subtree (of some tree t) that is isomorphic to t_k for some k is of type t_k.

Our aim is to trick the automaton $\mathcal{A}$ to show that it cannot choose a unique element from the set $U_{M,N}$. This is done by modifying a run that outputs an element u of $U_{M,N}$ such that we obtain another run outputting a different element from the same set.

To understand the general idea, consider the path between x_{k+1} and x_k for some k. If we take a 1-edge before having reached the end of the 0-chain, i.e., one of the dashed edges leading back to the initial state x_M, then we reach a subtree of type t_M. But if we walk to the end of the 0-chain and then move to the right using a 1-edge, then we arrive at a subtree of type t_k. If we show that there is $\ell < M$ such that t_M and t_ℓ are $\mathcal{A}$-equivalent, then this means that $\mathcal{A}$ has no means to identify when it enters the part where taking a 1-edge leads to a subtree of type t_ℓ. We then exploit this fact by pumping the run on this part of the tree such that we obtain another run with a different output.

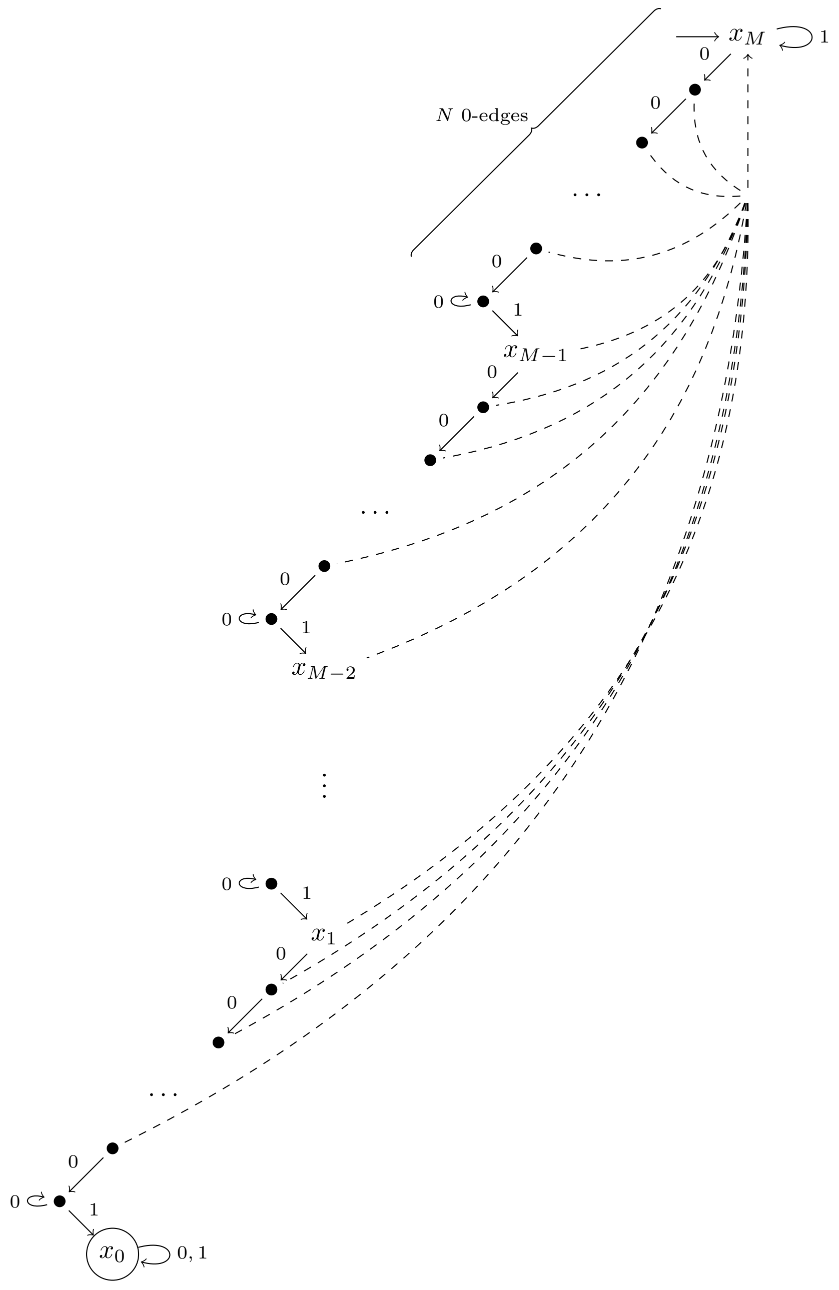

Figure 1: A DFA accepting $U_{M,N}$. Dashed edges are labeled by 1. The tree $t_{M,N}$ is obtained by unraveling this DFA.

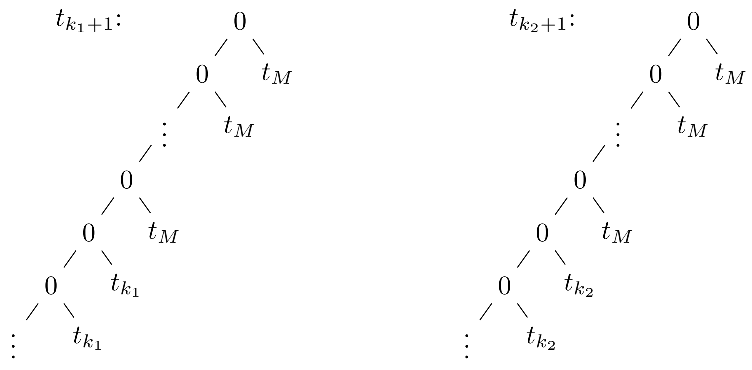

Figure 2: The shape of the trees t_{k_1+1} and t_{k_2+1} in the proof of Lemma 3.3

Lemma 3.3 *There exists $\ell < M$ such that $t_M \equiv_{\mathcal{A}} t_\ell$.*

Proof. We consider for each tree $t \in \mathcal{T}^{\omega}_{\{0,1\}}$ the function $f_t : Q \to \{a, r\}$ with $f_t(q) = a$ if there is an accepting run of $\mathcal{A}$ from q on t, and $f_t(q) = r$ otherwise. By definition, two trees t, t' are $\mathcal{A}$-equivalent if $f_t = f_{t'}$. There are at most $2^{|Q|}$ different such functions. By the choice of M there are $1 \leq k_1 < k_2 \leq M$ such that $t_{k_1} \equiv_{\mathcal{A}} t_{k_2}$.

We now show that $t_{k_1} \equiv_{\mathcal{A}} t_{k_2}$ implies $t_{k_1+1} \equiv_{\mathcal{A}} t_{k_2+1}$ (in case $k_2 < M$, otherwise we choose $\ell = k_1$). This allows us to lift the equivalence step by step to t_ℓ and t_M for $\ell = k_1 + (M - k_2)$.

If we only look at the left-most branches and the types of the trees going off to the right, then t_{k_1+1} and t_{k_2+1} look as shown in Figure 2. From this picture it should be clear that $t_{k_1+1} \equiv_{\mathcal{A}} t_{k_2+1}$ because an accepting run on t_{k_1+1} can easily be turned into an accepting run on t_{k_2+1}, and vice versa, using the equivalence of t_{k_1} and t_{k_2}. $\square$

The following lemma states that it is impossible for $\mathcal{A}$ to choose an element of $U_{M,N}$ (and hence the set of outputs computed by $\mathcal{A}$ on $t_{M,N}$ is empty).

Lemma 3.4 *The weak choice automaton $\mathcal{A}$ cannot choose an element from $U_{M,N}$, i.e., $\mathcal{A}([t_{M,N}]) = \emptyset$.*

Proof. Assume that there is an accepting run ρ of $\mathcal{A}$ on $t_M = t_{M,N}$. Since $\mathcal{A}$ is a weak choice automaton this means that the output of this run must be an element of $U_{M,N}$, i.e., $\lambda(\rho) = t[u]$ for some $u \in U_{M,N}$.

From ρ we construct another accepting run with a different output tree, contradicting the definition of weak choice automaton.

Since u is in $U_{M,N}$, we know that the path from the root to u must end with the pattern as defined by the automaton in Figure 1. Hence we can make the following definitions. For $0 \leq k \leq M$ let u_k denote the maximal prefix of u such that the subtree at u_k is of type t_k. Let ℓ be as in Lemma 3.3. For $i \geq 0$ let $v_i = u_{\ell+1}0^i$ and $v'_i = v_i 1$. Note that $v_0 = u_{\ell+1}$ and that for $0 \leq i < N$ the subtree at v'_i is of type t_M, and for $i \geq N$ the subtree at v'_i is of type t_ℓ.

From Lemma 3.3 we know that $t_\ell \equiv_{\mathcal{A}} t_M$. Hence, for each accepting run ρ_q of $\mathcal{A}$ from q on t_M we can pick an accepting run ρ'_q of $\mathcal{A}$ from q on t_ℓ.

By the choice of N (recall that $N = |Q| + 1$) there are $0 \leq j < j' < N$ such that $\rho(v_j) = \rho(v_{j'})$. For the moment, consider only the transitions taken in ρ on the sequence $v_0, v_1, \ldots$, i.e., on the infinite branch to the left starting from v_0. We now simply repeat the part of the run between v_j and $v_{j'}$ once. The effect is that some of the states that were at a node v'_i for $i < N$ are pushed to nodes v'_i for $i \geq N$, i.e., the states are moved from subtrees of type t_M to subtrees of type t_ℓ. But for those states q we can simply plug the runs ρ'_q that we have chosen above.

More formally, we define the new run ρ' of $\mathcal{A}$ on t_M as follows. On the part that is not in the subtree below v_0 the run ρ' corresponds to ρ. In the subtree at v_0 we make the following definitions, where $h = j' - j$.

- For $i < j'$ let $\rho'(v_i) = \rho(v_i)$ and $\rho'(v'_i) = \rho(v'_i)$.
- For $i \geq j'$ let $\rho'(v_i) = \rho(v_{i-h})$ and $\rho'(v'_i) = \rho(v'_{i-h})$.
- For the subtrees at v'_i for $i < j'$ we take the subrun of ρ at v'_i.
- For the subtrees at v'_i for $j' \leq i < N$ or $i \geq N + h$ we take the subrun of ρ at v'_{i-h}. This is justified because in these cases $\rho'(v'_i) = \rho(v'_{i-h})$ and the subtrees at v'_i and v'_{i-h} are of the same type (both of type t_M or both of type t_ℓ).
- For the subtrees at v'_i for $N \leq i < N+h$ we take the runs ρ'_{q_i} for $q_i = \rho'(v'_i)$. This is justified as follows. From $q_i = \rho'(v'_i)$ and the definition of ρ' we know that $\rho(v'_{i-h}) = q_i$. Hence, there is an accepting run of $\mathcal{A}$ from q_i on t_M. Thus, ρ'_{q_i} as chosen above is an accepting run of $\mathcal{A}$ from q_i on t_ℓ.

This run ρ' is accepting. Furthermore, the transition producing output 1 in ρ has been moved to another subtree: There are $n \geq N$ and $w \in \{0,1\}^*$ such that $u = u_{\ell+1}0^n w$. In ρ' the transition that is used at u in ρ is used at $u' = u_{\ell+1}0^{n+h}w$. Hence, we have constructed an accepting run whose output tree is different from $t[u]$, a contradiction. $\square$

Of course, the statement is also true if we increase the value of M or N, e.g., if we take $N = M = 2^{|Q|+1}$. Thus, combining Theorem 3.2 and Lemma 3.4 we obtain the following.

THEOREM 3.5 *Let $\phi(X,x)$ be an MSO-formula. There exists $m \in \mathbb{N}$ such that for all $n \geq m$ and for each $u \in U_{n,n}$ with $\mathfrak{t}_2 \models \phi[U_{n,n}, u]$ there is $u' \neq u$ with $\mathfrak{t}_2 \models \phi[U_{n,n}, u']$.*

Proof. Consider the formula

$$\phi^*(X,x) := \phi(X,x) \wedge x \in X \wedge \neg\exists y(y \neq x \wedge y \in X \wedge \phi(X,y)).$$

Applying Theorem 3.2 to this formula yields a weak choice automaton $\mathcal{A}$. Choose $n = 2^{|Q|+1}$ and assume that there is $u \in U_{n,n}$ such that $\mathfrak{t}_2 \models \phi[U_{n,n}, u]$. According to Lemma 3.4 we have $\mathcal{A}(t_{n,n}) = \emptyset$. Hence, because $\mathcal{A}$ and ϕ^* are equivalent, $\mathfrak{t}_2 \not\models \phi^*[U_{n,n}, u]$. By definition of ϕ^* this means that there must be $u' \neq u$ such that $\mathfrak{t}_2 \models \phi[U_{n,n}, u']$. □

A direct consequence is the theorem of Gurevichand Shelah (Theorem 3.1). The advantage of our proof is that we obtain a rather simple family of counter examples (the sets $U_{M,N}$).

An easy reduction allows us to extend the non-existence of an MSO-definable choice function to the case where we allow a finite number of fixed predicates as parameters. This result has already been shown in [14] in an even more general context, but again relying on the methods employed in [11].

Corollary 3.6 *Let $P_1, \ldots, P_n \subseteq \{0,1\}^*$ be arbitrary predicates. There is no MSO-formula $\phi(X,x)$ such that for each nonempty set U there is exactly one $u \in U$ with $\mathfrak{t}_2[P_1, \ldots, P_n] \models \phi[U,u]$.*

Proof. Suppose that there are $P_1, \ldots, P_n \subseteq \{0,1\}^*$ and an MSO[$\mathfrak{t}_2$]-formula $\phi(X_1, \ldots, X_n, X, x)$ such that for each nonempty set U there is exactly one $u \in U$ with $\mathfrak{t}_2[P_1, \ldots, P_n] \models \phi[U,u]$, where the symbols $X_1, \ldots, X_n$ are interpreted by $P_1, \ldots, P_n$, respectively. Then the formula

$$\begin{aligned}\exists X_1, \ldots, X_n \,\forall X \neq \emptyset\ \exists x \in X : \phi(X_1, \ldots, X_n, X, x) \wedge \\ \forall y\ \phi(X_1, \ldots, X_n, X, y) \rightarrow x = y\end{aligned}$$

holds in $\mathfrak{t}_2$. Hence, there are regular predicates $P'_1, \ldots, P'_n$ (see Theorem 2.2 on page 6) such that

$$\begin{aligned}\mathfrak{t}_2[P'_1, \ldots, P'_n] \models \forall X \neq \emptyset\ \exists x \in X : \phi(X_1, \ldots, X_n, X, x) \wedge \\ \forall y\ \phi(X_1, \ldots, X_n, X, y) \rightarrow x = y.\end{aligned}$$

Regular predicates are MSO-definable (see Proposition 2.3), and therefore we can find formulas $\psi_1(z), \ldots, \psi_n(z)$ defining $P'_1, \ldots, P'_n$, respectively. Then the formula $\phi'(X,x)$ defined as

$$\exists X_1, \ldots, X_n :\ \phi(X_1, \ldots, X_n, X, x) \wedge \bigwedge_{i=1}^{n} (\forall z : z \in X_i \leftrightarrow \psi_i(z))$$

describes a choice function, contradicting Theorem 3.5. □

We point out here that this method only relies on the fact that the property of being a choice function is MSO-definable. This way of reducing the case with parameters to the parameter free case can be applied whenever the properties of the object under consideration are MSO-definable (in Proposition 5.7 on page 25 we also use this technique).

MSO-definable winning strategies

In this section we use Theorem 3.1 to study the definability of winning strategies in infinite games. For a general introduction to games of infinite duration we refer the reader to [10]. The logical definability of winning regions in parity games has been studied in [8]. To express that a vertex of a game graph belongs to the winning region of one player it is sufficient to express that there exists a winning strategy from this vertex. Here we study the question whether it is possible to define a single winning strategy. In the following we show that there exist tree games (games with a tree as underlying game graph) that do not admit MSO-definable winning strategies.

A *game tree* is a tuple $\Gamma = (U_1, U_2, \Omega)$ where $U_1, U_2 \subseteq \{0,1\}^*$ form a partition of $\{0,1\}^*$, and $\Omega : \{0,1\}^* \to \{0,\dots,n\}$ maps each node to a number in a finite initial segment of $\mathbb{N}$. A *tree game* (Γ, W) is given by a game tree Γ and a winning condition $W \subseteq \{0,\dots,n\}^\omega$. A play in this game starts in ε. If the play is currently in $u \in \{0,1\}^*$, then Player 1 or Player 2, depending on whether $u \in U_1$ or $u \in U_2$, chooses $b \in \{0,1\}$, and the next game position is ub. In the limit, such a play forms an infinite word in $\{0,1\}^\omega$; we identify the play with this word. This infinite word corresponds to an infinite sequence over $\{0,\dots,n\}$ by applying Ω to each prefix. If this sequence is in W then Player 1 wins, and otherwise Player 2 wins.

A strategy for Player i is a function $f_i : U_i \to \{0,1\}$. A play γ is played according to f_i if, for each prefix $u \in U_i$ of γ, Player i uses the strategy to determine the next move, i.e., $uf_i(u)$ is again a prefix of γ. A strategy f_i is winning for Player i if each play γ played according to f_i is won by Player i.

If $W \subseteq \{0,\dots,n\}^\omega$ is a regular ω-language, then for each tree game (Γ, W) one of the players has a winning strategy [3].

A strategy f_i for Player i can be represented by two sets of nodes S_0 and S_1 of the game tree: the set of nodes where the value of f_i is 0 and 1, respectively. We are interested in MSO-definability of strategies in the game tree. To this end, we represent a game tree $\Gamma = (U_1, U_2, \Omega)$ as the binary tree $\mathfrak{t}_2$ extended by monadic predicates $U_1, U_2, \Omega_0, \dots, \Omega_n$, where each $\Omega_i \subseteq \{0,1\}^*$ corresponds to the set of nodes mapped to i by Ω. Let us abbreviate $\mathfrak{t}_2[U_1, U_2, \Omega_0, \dots, \Omega_n] = \mathfrak{t}_2[\Gamma]$.

We say that a strategy represented by S_0 and S_1 is *MSO-definable* in the game tree Γ if these sets are definable in $\mathfrak{t}_2[\Gamma]$, i.e., if there exists two MSO-formulas $\phi_0(x)$ and $\phi_1(x)$ determining the sets S_0 and S_1:

$$\begin{aligned} S_0 =& \{u : \mathfrak{t}_2[\Gamma] \models \phi_0[u]\} \\ S_1 =& \{u : \mathfrak{t}_2[\Gamma] \models \phi_1[u]\}. \end{aligned}$$

Finite-state strategies (i.e. strategies implementable by a finite-state automaton) are a particular case of MSO-definable strategies.

Note that the above formulas $\phi_0(x)$ and $\phi_1(x)$ may depend on the game tree Γ. The question whether such formulas exist for a particular game tree Γ and a player i should not be confused with the fact that the family of all winning strategies for player i is definable in MSO in a uniform way. More precisely, whenever the condition W is ω-regular then it is not difficult to write an MSO-formula $\psi_W(X_1, X_2, Y_0, \ldots, Y_n, Z_0, Z_1)$, such that, for any game tree $\Gamma = (U_1, U_2, \Omega)$ with $\Omega : \{0,1\}^* \to \{0, \ldots, n\}$, and any $S_0, S_1 \subseteq \{0,1\}^*$,

$$\mathfrak{t}_2 \models \psi_W[U_1, U_2, \Omega_0, \ldots, \Omega_n, S_0, S_1]$$

iff S_0 and S_1 represent a winning strategy for Player i in (Γ, W). Thus, we can express the *existence* of a winning strategy for Player i in the game (Γ, W) by an MSO-formula interpreted in $\mathfrak{t}_2[\Gamma]$. If $\mathfrak{t}_2[\Gamma]$ has decidable *MSO*-theory, we can therefore decide who wins the game (Γ, W) for any ω-regular winning condition W. This however does not imply the existence of an MSO-definable winning strategy. In fact, we show in the following that there is a fixed tree game (with a decidable MSO-theory) for which there are no MSO-definable winning strategies.

It is well-known that there exists a recursive tree game where Player 1 has a winning strategy but no recursive one. In [28], an example of such a tree game is given that has moreover an MSO-definable winning condition. Nevertheless this example cannot be used to prove the following theorem as both players admit MSO-definable strategies.

THEOREM 3.7 *There is a game tree $\Gamma = (U_1, U_2, \Omega)$ with decidable MSO-theory, and an MSO-definable winning condition W, such that Player 1 has a winning strategy in the game (Γ, W), but no winning strategy for Player 1 is MSO-definable in $\mathfrak{t}_2[\Gamma]$.*

Proof. Consider the following $\{0,1,2\}$-labeled tree t such that for each $n \in \mathbb{N}$ the subtree at the node 1^n0 is isomorphic to $t_{n,n}$, and all nodes of the form 1^n are labeled by 2. The labeling of t defines the mapping Ω, i.e., $\Omega(u) = t(u)$. We let $U_2 = \{1^n \mid n \in \mathbb{N}\}$ and $U_1 = \{0,1\}^* \setminus U_2$.

Consider $\Gamma = (U_1, U_2, \Omega)$ and let the winning condition W (for Player 1) contain all infinite words over $\{0,1,2\}$ that do not contain 0 or that contain a 1.

Intuitively, Player 2 can move along the right branch of the game tree. If he continues like this forever then he loses because only nodes labeled 2 are visited during the play. Otherwise, he moves to the left at some position, that is, to the root of a subtree $t_{n,n}$. Now Player 1 chooses all the following moves and wins if a node labeled 1 is reached eventually. As each subtree $t_{n,n}$ contains a node labeled 1 it is obvious that Player 1 has a winning strategy.

Assume that there is a winning strategy f_1 for Player 1 that is MSO-definable in $\mathfrak{t}_2[\Gamma]$ by formulas $\phi_0(x)$ and $\phi_1(x)$. Let S_{f_1} be the set of all nodes that occur in a play that is played according to f_1. (This set can be seen as an alternative way of coding the strategy f_1.) From ϕ_0 and ϕ_1 we can easily derive an MSO[$\mathfrak{t}_2$]-formula $\phi(X_1, X_2, Y_0, Y_1, Y_2, X)$, such that

$$\mathfrak{t}_2 \models \phi[U_1, U_2, \Omega_0, \Omega_1, \Omega_2, S]$$

holds precisely when $S = S_{f_1}$.

The formula ϕ is equivalent to a parity automaton over alphabet $\{0,1\}^6$, but using the definition of our Γ (constructed from the tree t), we can simplify it to an automaton $\mathcal{A}$ over the alphabet $\{0,1,2\} \times \{0,1\}$, where the 1 on the second component should be read as "marked". This automaton reads trees over $\{0,1,2\}$, where some nodes are marked, and whenever it reads our tree t, it accepts it only in the case when the marking coincides with S_{f_1}.

Using $\mathcal{A}$ we can construct a formula $\phi^*(X,x)$ that chooses exactly one $u \in U_{n,n}$ for each $n \in \mathbb{N}$, contradicting Theorem 3.5. For this we fix an arbitrary order on the states of $\mathcal{A}$. For each n there is at least one state q of $\mathcal{A}$ such that $\mathcal{A}$ accepts from q exactly one winning strategy on the subtree $t_{n,n}$ of t (namely the state assumed at the root of the subtree $t_{n,n}$ in an accepting run for the unique winning strategy on Γ that is accepted by $\mathcal{A}$). The formula ϕ^* is designed to select the smallest state q with this property and then chooses the element of $U_{n,n}$ that is described by the unique winning strategy accepted by $\mathcal{A}$ from q on $t_{n,n}$. It is not difficult to verify that the formalization is indeed possible in MSO. $\square$

One should note here that the tree constructed in the proof of Theorem 3.7 (and also the one from Theorem 4.3) is not too complicated: it belongs to the Caucal hierarchy[1] [6]. This means that it can be obtained from a regular tree by a finite number of applications of MSO-interpretations and unfoldings, or equivalently, it is the transition graph of a higher-order pushdown automaton [5]. In [15] it is shown that the tree belongs to the fourth level of the hierarchy. The possibility of constructing such rather simple examples is one of the advantages of our new proof of Theorem 3.1.

In the tree games that we consider we do not require that there is a strict alternation between the moves of the two players. However, by inserting additional nodes in the tree games, it is always possible to obtain a game with strict alternation.

4 Unambiguous automata

In this section, we show that unambiguous parity tree automata do not accept all regular tree languages. For the proof we use the non-existence of a MSO-definable choice function.

Unambiguous automata have been introduced on finite words because they allow more efficient algorithms for equivalence and inclusion testing [26]. This has been generalized to automata on finite trees in [23]. On finite words and trees it is clear that every regular language can be accepted by an unambiguous automaton because deterministic automata are unambiguous.

For Büchi automata on infinite words the situation is more difficult. Although determinization is possible ([16, 22]), it requires the model of deterministic parity (or Muller) automata with acceptance conditions that are more powerful than Büchi conditions. However, from the determinization result one

[1]In particular, as all graphs in the Caucal hierarchy have a decidable MSO-theory, the tree constructed in the proof of Theorem 3.7 also has a decidable MSO-theory.

can infer that all regular languages of infinite words can be accepted by an unambiguous Büchi automaton [1]. In [12] a construction for unambiguous automata is presented that does not rely on deterministic automata.

In this section we show that the situation is different for infinite trees: There are regular languages that cannot be accepted by an unambiguous automaton. The proof is based on the main result from the previous section: the undefinability of a choice function in MSO. The underlying idea is rather simple. We consider the language

$$T_{\exists 1} = \{t \in \mathcal{T}^{\omega}_{\{0,1\}} \mid \exists u \in \{0,1\}^* \ : \ t(u) = 1\}$$

of trees with at least one node labeled 1. Each tree in $T_{\exists 1}$ can be viewed as the set of nodes labeled 1. Intuitively, an automaton accepting $T_{\exists 1}$ has to verify that the input tree contains a 1, i.e., an accepting run has to identify some position labeled by 1. If the automaton is unambiguous, then there is exactly one run for each tree in $T_{\exists 1}$, and hence the automaton identifies exactly one position from the set coded by the 1-positions. This can be turned into a formula defining a choice function.

We show the following more technical result because it can be used to prove different statements on the ambiguity of automata.

Lemma 4.1 *Let $\mathcal{A}$ be a parity tree automaton over the alphabet $\{0,1\}$ not accepting the tree that is completely labeled by 0. Then there is a formula $\psi_{\mathcal{A}}(X,x)$ such that for each $U \subseteq \{0,1\}^*$ for which there is a unique accepting run of $\mathcal{A}$ on $\mathfrak{t}_2[U]$, there is a unique element $u \in U$ such that $\mathfrak{t}_2 \models \psi_{\mathcal{A}}[U,u]$.*

Proof. We consider the following game which can be seen as the emptiness game[2] for the automaton $\mathcal{A}$ intersected with the trivial automaton that accepts only the tree $t[\emptyset]$ (i.e., the tree labeled 0 everywhere). This intersection corresponds to removing all transitions that use a letter different from 0 from the standard emptiness game for $\mathcal{A}$.

Formally, the game is played between two players Eva and Adam who move in alternation. The positions of Eva are the states of $\mathcal{A}$, and the positions of Adam are the transitions of $\mathcal{A}$ that use input letter 0. The initial position is the initial state of $\mathcal{A}$. Then the players move in alternation as follows:

- From a position q Eva chooses a transition $(q,0,q_0,q_1)$ of $\mathcal{A}$.
- From position $(q,0,q_0,q_1)$ Adam chooses a state q_0 or q_1.

Eva wins a play if she can always choose a transition and if the resulting sequence of states chosen in the play satisfies the acceptance condition of $\mathcal{A}$. Since the acceptance condition of $\mathcal{A}$ is a parity condition, we obtain a parity game (for basic facts on parity games see [29]).

We have already mentioned that the presented game is an emptiness game for the subautomaton of $\mathcal{A}$ in which all transitions with label 1 have been removed.

[2]For a description of the emptiness game for parity tree automata see [29].

A winning strategy of Eva in this game would yield an accepting run of $\mathcal{A}$ on the tree $t[\emptyset]$, contradicting the assumption $t[\emptyset] \notin T(\mathcal{A})$. We can conclude that Adam has a winning strategy, and since we are working with a parity game, Adam even has a positional[3] winning strategy f that picks for each transition of the form $(q, 0, q_0, q_1)$ one of the states q_0, q_1 (see [29] for a proof of the positional determinacy of parity games).

We now construct the formula $\psi_{\mathcal{A}}(X, x)$ based on this strategy f. Evaluated on $t[U, u]$ the formula states that there exists an accepting run of $\mathcal{A}$ on $t[U]$ such that the path leading to node u corresponds to the choices of Adam according to the strategy f. Note that if we apply the strategy of Adam to the transitions in an accepting run, then this results in a path ending in a node labeled 1 because otherwise the resulting infinite play would be winning for Eva.

Assuming that the states of $\mathcal{A}$ are $\{1, \ldots, n\}$ the formula $\psi_{\mathcal{A}}(X, x)$ looks as follows:

$$\begin{array}{ll} \exists X_1, \ldots, X_n : & \mathit{AccRun}(X_1, \ldots, X_n, X) \wedge X(x) \wedge \\ & \forall y \sqsubset x : \mathit{strat}_f(X, x, X_1, \ldots, X_n, y) \end{array}$$

where

- the formula *AccRun* expresses that $X_1, \ldots, X_n$ describe an accepting run of $\mathcal{A}$ on the characteristic tree of X (the construction of such a formula is standard, see e.g. [29]), and
- strat_f states that the strategy f applied at node y in the run coded by $X_1, \ldots, X_n$ moves into the direction of x. In case the strategy allows to move to both directions (when the states q_0 and q_1 are the same), the formula picks the left move:

$$\neg X(y) \wedge \bigwedge_{q, q_0, q_1 \in \{1, \ldots, n\}} \begin{array}{l} X_q(y) \wedge X_{q_0}(y0) \wedge X_{q_1}(y1) \rightarrow \\ \big[f(q, 0, q_0, q_1) = q_0 \leftrightarrow (y0 \sqsubseteq x)\big] \end{array}$$

If we fix the interpretations for X and $X_1, \ldots, X_n$, then there is exactly one interpretation of x such that $\forall y \sqsubset x.\mathit{strat}_f(X, x, X_1, \ldots, X_n, y)$ is satisfied (if there are two positions, then we obtain a contradiction when interpreting y as the greatest common ancestor of these two positions).

Now assume that there is exactly one accepting run of $\mathcal{A}$ on $t[U]$. Then the interpretations of $X_1, \ldots, X_n$ are fixed by U and hence the formula $\psi_{\mathcal{A}}(X, x)$ has the claimed property. $\square$

Using this lemma it is easy to obtain the following theorem.

THEOREM 4.2 *There is no unambiguous parity tree automaton accepting the language $T_{\exists 1}$ consisting of exactly those $\{0, 1\}$-labeled trees in which at least one node is labeled* 1.

[3]The moves of a positional strategy only depend on the current vertex and not on the entire history of the play.

Proof. Assume $\mathcal{A}$ is an unambiguous automaton for $T_{\exists 1}$ and consider the formula $\psi_{\mathcal{A}}(X, x)$ from Lemma 4.1. As there is a unique accepting run of $\mathcal{A}$ on $t[U]$ for each non-empty set U, $\psi_{\mathcal{A}}(X, x)$ defines a choice function. This contradicts Theorem 3.1. $\square$

Using Lemma 4.1 we can also show that there are regular languages whose ambiguity is witnessed by a single tree.

THEOREM 4.3 *There is a regular language $T \subseteq \mathcal{T}^{\omega}_{\{0,1\}}$ and a tree $t \in T$ such that there is no parity automaton accepting T that has a unique accepting run for t.*

Proof. Consider the language T of trees with the property that each subtree rooted at a node of the form 1^*0 contains a node labeled 1. Obviously this is a regular language.

The tree t is defined to have all nodes of the form 1^* labeled 0, and for each n we plug the tree $t_{n,n}$ as the subtree rooted at the node 1^n0 (as in the proof of Theorem 3.7.) As each $t_{n,n}$ contains a node labeled 1, we have $t \in T$.

Assume that there is parity automaton $\mathcal{A}$ accepting T that has a unique run on t. Let q be a state of $\mathcal{A}$ that occurs at infinitely many nodes of the form 1^*0 in this run. Let $\mathcal{A}'$ be the automaton $\mathcal{A}$ with initial state q. As $\mathcal{A}$ accepts T it is clear that $\mathcal{A}'$ does not accept the tree $t[\emptyset]$. Furthermore, as the run of $\mathcal{A}$ on t is unique, there are infinitely many n such that $\mathcal{A}'$ has a unique run on $t_{n,n}$. For the formula $\psi_{\mathcal{A}'}(X, x)$ from Lemma 4.1 this yields a contradiction to Theorem 3.5. $\square$

5 Well-orderings

A *well-ordering* is a total order relation with the property that there are no infinite descending chains or, equivalently, every non-empty subset of the domain of the relation has a smallest element. A simple example is the natural ordering on the set of natural numbers $\mathbb{N}$. The natural ordering on the set of integers $\mathbb{Z}$ is not a well-ordering because, e.g., the whole set $\mathbb{Z}$ does not have a minimal element w.r.t. to this ordering. If we define an ordering $\preceq$ on $\mathbb{Z}$ by first comparing the absolute values of the numbers and in case of equality letting the negative one be smaller, i.e., $n \preceq m$ iff $|n| < |m|$, or $|n| = |m|$ and $n \leq m$, then we obtain a well-ordering.

A *partial well-ordering* is a partial order relation with no infinite descending chains and with no infinite sets of pairwise incomparable elements. Equivalently a partial well-ordering is a partial order in which any non-empty set admits a non-empty finite set of minimal elements. Another useful equivalent property is that for every infinite sequence $(a_i)_{i \in \mathbb{N}}$ there are two indices $k < l$ such that $a_k \leq a_l$. For two elements u and v of the partial well-ordering, we write $u \mid v$ if u and v are incomparable for the order.

A well-ordering is a particular case of partial well-ordering. If we partially order the integers in $\mathbb{Z}$ by comparing their absolute values if they have the same sign, we obtain a partial well-ordering on $\mathbb{Z}$.

In this section we are interested in well-orderings and partial well-orderings on the tree domain $\{0,1\}^*$. A typical ordering of $\{0,1\}^*$ is the *lexicographic ordering* $\leq_{\text{lex}}$ defined by $u_1 \leq_{\text{lex}} u_2$ if u_1 is a prefix of u_2, or $u_1 = u0u_1'$ and $u_2 = u1u_2'$ for some $u, u_1', u_2' \in \{0,1\}^*$. From the definition of $\leq_{\text{lex}}$ one can easily see that it is MSO-definable in $\mathfrak{t}_2$. But it is not a well-ordering because, e.g., the set of nodes of the form 0^*1 does not have a minimal element.

In a similar way as we changed the standard ordering on $\mathbb{Z}$ into a well-ordering, we can obtain a well-ordering on $\{0,1\}^*$ by first comparing the length of the elements and in case of equality take the lexicographic ordering. We obtain the *length-lexicographic ordering* $\leq_{\text{llex}}$ formally defined by $u_1 \leq_{\text{llex}} u_2$ iff $|u_1| < |u_2|$, or $|u_1| = |u_2|$ and $u_1 \leq_{\text{lex}} u_2$. This defines a well-ordering but its definition requires to compare the length of the elements. From the fact that the MSO-theory of $\mathfrak{t}_2$ extended with the equal length predicate is undecidable (see [27]) one can easily derive that $\leq_{\text{llex}}$ is not MSO-definable in $\mathfrak{t}_2$.

A partial well-ordering $<_{\text{len}}$ on $\{0,1\}^*$ is obtained by only comparing the length of the words (i.e. $u \leq_{\text{len}} v$ if $u = v$ or $|u| < |v|$). In fact, as there are only finitely many words of a given length, any non-empty subset of $\{0,1\}^*$ has a non-empty set of minimal elements for $<_{\text{len}}$. Again $<_{\text{len}}$ is not MSO-definable in $\mathfrak{t}_2$.

More generally, as a direct consequence of Theorem 3.1 we obtain that there exists no MSO-definable partial well-ordering on the nodes of the infinite binary tree. In fact, from an MSO-formula $\phi_{\leq}(x,y)$ defining a partial well-ordering in $\mathfrak{t}_2$, a choice function choosing x in X is easily defined by taking the smallest element for $\leq_{\text{lex}}$ of the finite set of minimal elements of X (for the partial well-ordering):

$$\begin{aligned}\psi_{\text{choice}}(X,x) := \exists Y \subseteq X.\ & x \in Y \wedge (\forall y \in Y.\ x \leq_{\text{lex}} y) \wedge \\ & \forall z \in X.\ \ z \in Y \leftrightarrow \neg(\exists z' \in X\ \phi_{<}(z',z))\end{aligned}$$

This naturally raises the question whether there is a partial well-ordering that we can add to $\mathfrak{t}_2$ while preserving the decidability of MSO. In this section, we prove the following negative result.

Theorem 5.1 *The MSO-theory of the full-binary tree together with an arbitrarypartial well-ordering is undecidable.*

In the particular case of t_{llex}, the infinite binary tree with length-lexicographic order, i.e., the structure $t_{\text{llex}} = (\{0,1\}^*, E_0, E_1, \leq_{\text{llex}})$, this result is well-known. It easily follows from the undecidability of $\mathfrak{t}_2$ extended with the equal length predicate (see [27]).

We show that t_{llex} can be MSO-interpreted in the infinite binary tree with any partial well-ordering.

Theorem 5.2 *There exists an MSO-interpretation $\mathcal{I}$ such that for every partially well-ordered infinite binary tree t, $\mathcal{I}(t)$ is isomorphic to t_{llex}.*

As MSO-interpretations preserve the decidability of MSO (see Proposition 2.4), Theorem 5.1 follows from the undecidability of the MSO-theory of t_{llex}. The rest of this section is dedicated to the proof of Theorem 5.2.

Partially well-ordered trees

We consider structures over the binary signature $\tau = \{E_0, E_1, \leq\}$. A *canonical well-ordered tree* has the universe $\{0,1\}^*$ where E_0 and E_1 are respectively interpreted as $\{(u, u0) \mid u \in \{0,1\}^*\}$ and $\{(u, u1) \mid u \in \{0,1\}^*\}$, and $\leq$ is interpreted as a well-ordering on $\{0,1\}^*$. We say that a τ-structure t is a *well-ordered (infinite binary) tree* if it is isomorphic to a canonical well-ordered tree. Similarly, if $\leq$ is a partial well-order we talk about *partially well-ordered trees.*

Up to isomorphism, a well-ordered tree is entirely characterized by the well-ordering on the set of words over $\{0,1\}$. Above, we have already defined t_{llex} to be the well-ordered tree with $\leq_{\text{llex}}$ as ordering relation.

The key property of t_{llex} is that it is MSO-definable (up to isomorphism) in the class of partially well-ordered trees[4].

PROPOSITION 5.3 *There exists an MSO-sentence ϕ_{llex} such that a partially well-ordered tree t is isomorphic to t_{llex} iff $t \models \phi_{llex}$.*

Proof. We describe the properties that we need to express by ϕ_{llex} to ensure that partially well-ordered t is isomorphic to t_{llex}. In the description, for a node $v1 \in 1^+$ we denote by Bubble$(v1)$ the set $\{u : v < u \leq v1\}$. Note that if $\leq$ equals $\leq_{\text{llex}}$, then Bubble$(v1)$ consists of all nodes that are on the same level as $v1$. The formula ϕ_{llex} expresses the following properties:

1. Bubble$(1) = \{0,1\}$, and for all $v \in 1^*$, Bubble$(v1)$ is the set of all the children of nodes in Bubble(v).

2. The root of the tree is the least element for the order $\leq$, and on Bubble(v) the relation $\leq$ agrees with the lexicographic order for all $v \in 1^+$.

These properties can be defined in MSO. It is easy to see that t_{llex} satisfies the formula ϕ_{llex}. It remains to show that for every well-ordered tree t satisfying ϕ_{llex} is isomorphic to t_{llex}.

Let t be a partially well-ordered tree satisfying ϕ_{llex}. First observe that for $v \in 1^*$ the first condition assures that Bubble$(v) = \{u : |u| = |v|\}$, i.e., the set of all the nodes that are on the same level as v.

We write Succ$_{\text{llex}}(u)$ for the successor of u for the order $\leq_{\text{llex}}$. To prove the proposition it is enough to show that every node u has exactly one successor in $<$-ordering, and it is Succ$_{\text{llex}}(u)$; in other terms, the set of minimal elements of $\{v : u < v\}$ is exactly $\{\text{Succ}_{\text{llex}}(u)\}$.

Assume by contradiction that this property is not satisfied. Let $\bar{u}$ be the smallest in the $\leq_{\text{llex}}$ ordering node violating this property. By the first condition,

[4]The class of well-ordered trees and the class of partially well-ordered trees are themselves MSO-definable in the class of τ-structures.

$\bar{u}$ is not the root. If $\bar{u} \in 1^+$ then the minimal elements of $\{v : \bar{u} < v\}$ are contained in Bubble($\bar{u}1$). By the second condition the only minimal element is $0^{|\bar{u}|+1} = \mathrm{Succ}_{\mathrm{llex}}(\bar{u})$. If $\bar{u} \notin 1^+$ then all its potential successors are in Bubble(v) where $v \in 1^*$ has the same length as $\bar{u}$. Once again the second condition allows us to conclude. $\square$

Interpreting t_{llex}

We now define the notion of *induced partially well-ordered tree*. Consider a canonical partially well-ordered tree t with partial well-ordering $\leq^t$ and a set $U \subseteq \{0,1\}^*$ of nodes that

- is closed under greatest common prefix ($u \in U \wedge v \in U \to u \wedge v \in U$),
- has the property that for all $u \in U$, $u0\{0,1\}^* \cap U \neq \emptyset$ and $u1\{0,1\}^* \cap U \neq \emptyset$, i.e., from every node we can go to the left and to the right and find another element of U below.

The partially well-ordered tree $t|_U$ induced by U in t has universe U.

To interpret its relations we use a formula $\psi_{\mathrm{greatest}}(X, x)$ stating that x is the greatest common prefix of the set X:

$$\begin{array}{rcl} E_0^{t|_U} & = & \{(u,v) \in U \times U \mid t \models \psi_{\mathrm{greatest}}[u0\{0,1\}^* \cap U, v]\} \\ E_1^{t|_U} & = & \{(u,v) \in U \times U \mid t \models \psi_{\mathrm{greatest}}[u1\{0,1\}^* \cap U, v]\} \\ \leq^{t|_U} & = & \{(u,v) \in U \times U \mid u \leq^t v\}. \end{array}$$

The plan is to show that in each partially well-ordered tree we can find a subset U inducing a well-ordered tree that is isomorphic to t_{llex}. As a first step we show that we can express each MSO-property ϕ of $t|_U$ by an MSO-formula ϕ^* on t that takes U as a parameter.

LEMMA 5.4 *For every MSO-formula ϕ over τ there exists a formula $\phi^*(X)$ such that for every canonical partially well-ordered tree t and set U, $t \models \phi^*[U]$ iff the set U induces a partially well-ordered tree $t|_U$ on t, and $t|_U \models \phi$.*

Proof. Consider an MSO-formula ϕ over τ. Let $\phi_{\mathrm{ind}}(X)$ be an τ-formula expressing that X satisfies the conditions to induce a full binary tree and let $\phi'(X)$ be the formula obtained from ϕ by relativizing the quantifications to X and by replacing $E_i(x,y)$ with $\psi_{\mathrm{choice}}(xi\{0,1\}^* \cap X, y)$, for $i \in \{0,1\}$. It is easy to check that the formula $\phi^*(X) := \phi_{\mathrm{ind}}(X) \wedge \phi'(X)$ satisfies the property stated in the lemma. $\square$

Applying this lemma to the formula ϕ_{llex} from Proposition 5.3 yields that $t \models \phi^*_{\mathrm{llex}}[U]$ iff U induces a well-ordered tree isomorphic to t_{llex}.

We now show that for every canonical partially well-ordered tree t there exists a subset $U \subseteq \{0,1\}^*$ such that $t|_U$ is isomorphic to t_{llex}. We even show a stronger result: there is in fact an MSO-definable such set (Lemma 5.6).

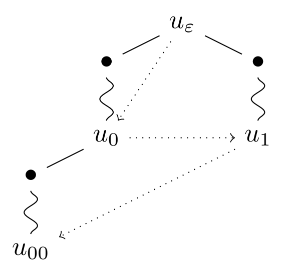

Figure 3: Construction of a set inducing t_{llex}

To construct such a set U we built up a sequence of nodes indexed by the elements from $\{0,1\}^*$. We start with a node u_ε representing the root of $t|_U$. Then we define u_0 to be a node in the left subtree of u_ε such that $u_\varepsilon < u_0$. We continue by finding a node u_1 in the right subtree of u_ε such that $u_0 < u_1$. Then we take a node u_{00} in the left subtree of u_0 such that $u_1 < u_{00}$, and so on. This construction is illustrated in Figure 3, where the dashed arrows represent the order relation $<$ on the sequence $u_\varepsilon, u_0, u_1, u_{00}, \ldots$ by always pointing to the next bigger element from this sequence.

For this construction to work we need to ensure that we always can find nodes as required, e.g., that there is a node u_{00} in the left subtree of u_0 such that $u_1 < u_{00}$. For this purpose we make the definition of a mixed node and later choose u_ε to be a node with this property. We call a node $u \in \{0,1\}^*$ of a canonical partially well-ordered tree t *mixed* if the ordering relation "jumps" between all different subtrees below u in the following sense: for all $v, v' \sqsupseteq u$ there exists $w \in \{0,1\}^*$ such that $v < v'w$. We show that we can indeed find a node with this property.

Lemma 5.5 *Every canonical partially well-ordered tree t contains a mixed node.*

Proof. Let t be a canonical partially well-ordered tree. Assume by contradiction that t does not have any mixed nodes. We are going to construct a sequence $(u_n)_{n\in\mathbb{N}}$ of nodes such that for no $i < j$, $u_i < u_j$. This is in contradiction with the fact that $<$ is a well partial ordering.

We construct the sequence $(u_n)_{n\in\mathbb{N}}$ by induction on n together with another sequence of nodes $(v_n)_{n\in\mathbb{N}}$ such that for all $n \geq 0$:

- $v_n \sqsubseteq u_{n+1}$ and $v_n \sqsubseteq v_{n+1}$,
- there is no $v > u_n$ in the subtree rooted in v_n.

Note that these conditions imply that for no $i < j$, $u_i < u_j$ because u_j is in the subtree rooted in v_i.

As ε is not mixed there exist two nodes u and v such that there is no node in the subtree of v that is greater (with respect to $<$) than u. We take $u_0 = u$ and

$v_0 = v$. If v_0 is not mixed then we can apply the same argument to the subtree rooted in v_0 obtaining u_1 and v_1. We get the desired sequence by induction. $\square$

Now we can formalize the construction of the set U inducing t_{llex} as indicated above and in Figure 3.

LEMMA 5.6 *For every canonical partially well-ordered tree t, there exists a set of nodes U inducing a well-ordered tree $t|_U$ isomorphic to t_{llex}.*
Moreover there exists an MSO-formula $\psi(X)$ which uniquely defines one such set (i.e. there exists a unique set U_0 s.t. $t[U_0] \models \psi$ and $t|_{U_0}$ is isomorphic to t_{llex}).

Proof. Let t be a canonical partially well-ordered tree. We construct a sequence of nodes $(u_w)_{w\in\{0,1\}^*}$ indexed by the set of words over $\{0,1\}^*$ such that:

- for all $w, w' \in \{0,1\}^*$, $w \leq_{\text{llex}} w'$ implies $u_w \leq u_{w'}$,
- and for all $w \in \{0,1\}^*$ and $i \in \{0,1\}$, $u_{wi} \in u_w i\{0,1\}^*$.

If we assume that this sequence has been constructed and we take $U := \{u_w \mid w \in \{0,1\}^*\}$, it is easy to check that U is closed by greatest common prefix and hence U induces a full binary tree on t. Furthermore the mapping from $\{0,1\}^*$ to U associating w to u_w is an isomorphism from t_{llex} to $t|_U$.

We now construct the sequence $(u_w)_{w\in\{0,1\}^*}$ by induction on the length-lexicographic order $\leq_{\text{llex}}$. By Lemma 5.5, the tree t has a mixed node. We take u_ε to be a mixed node of t.

Assume that the sequence has been constructed up to $w \in \{0,1\}^*$. Let w' be the successor of w in the length-lexicographic order. Note that as w' is not empty, it can uniquely be written as $w' = w''i$ for some $w'' \leq_{\text{llex}} w$ and $i \in 0, 1$. To construct $u_{w'}$, we need to find an element of $u_{w''}i\{0,1\}^*$ which greater than u_w. As u_ε is mixed and u_w and $u_{w''}i$ are below u_ε (below according to the prefix relation), there exists $z \in \{0,1\}^*$ such that $u_{w''}iz > u_w$. We take $u_{w'}$ equal to $u_{w''}iz$.

This establishes the first part of the lemma.

The construction of the set U presented above leaves several choices: namely the mixed node we take for u_ε and the element of $\{u_{w''}iz | z \in \{0,1\}^* \wedge u_{w''}iz > u_w\}$ we take for $u_{w'}$ for each $w' \in \{0,1\}^+$. We have already remarked (on page 20) that we can define a choice function on t by means of an MSO-formula $\psi(x, X)$. The formula states that x is the left-most minimal element of X (minimal with respect to $<$).

We will show that if for each of the choices in the construction of U we use the choice function defined by ψ we obtain a set U_0 which is MSO-definable in t.

Consider a set U_0 satisfying the following properties:

(1) U_0 induces a well-ordered tree which is isomorphic to t_{llex},

(2) the smallest element x_0 of U_0 is chosen by ψ in the set of mixed nodes of t,

(3) for all $x' \neq x_0 \in U_0$ with predecessor x in U_0 (according to $\leq$), father x'' in U_0 (according to the binary tree structure induced by U_0), and $i \in \{0,1\}$ such that $x''i \sqsubseteq x'$, we have that x' is the element chosen by ψ in the set of elements of U_0 which are below $x''i$ and greater than x (below according to the prefix relation, and greater according to $<$).

It follows from the first part of the proof that such a set exists. Moreover there exists at most one set satisfying properties (1), (2) and (3). It remains to verify that the above properties can be expressed in MSO-logic. For property (1), it is enough to take the formula $\phi^*_{\text{llex}}(X)$ obtained by applying Lemma 5.4 to the formula ϕ_{llex} of Proposition 5.3. For property (2), we simply need to remark that the property of being a mixed node can be expressed in MSO. For property (3) the translation is immediate. □

Note that we can now already derive Theorem 5.1: For every formula ϕ over τ, consider the formula $\phi^*(X)$ obtained from ϕ by Lemma 5.4 and $\phi^*_{\text{llex}}(X)$ obtained from the formula ϕ_{llex} of Proposition 5.3. By Lemma 5.6, for every partially well-ordered tree t:

$$t_{\text{llex}} \models \phi \quad \text{iff} \quad t \models \exists X \ : \ \phi^*_{\text{llex}}(X) \wedge \phi^*(X).$$

As the formula $\phi^*(X)$ can be effectively constructed from the formula ϕ, it follows that the MSO-theory of t_{llex} is recursive in the MSO-theory of any partially well-ordered tree t.

Using the previous results it is easy to prove Theorem 5.2.

Proof(of Theorem 5.2). The interpretation $\mathcal{I} = (\phi_{\text{dom}}, \phi_{E_0}, \phi_{E_1}, \phi_{\leq})$ such that $\mathcal{I}(t) \cong t_{\text{llex}}$ for each well-ordered tree t is defined as follows. As domain formula ϕ_{dom} we take the formula ψ defining the set U_0 from Lemma 5.6. The formulas ϕ_{E_0}, ϕ_{E_1}, and $\phi_{\leq}$ just define the induced successor relations and the induced ordering from the definition of induced well-ordered tree. □

An immediate consequence of this result is that the infinite binary tree cannot be MSO-interpreted in $\mathfrak{t}_1$ (the natural numbers with successor). Based on this we can use the same technique as in Corollary 3.6 to obtain the same result for $\mathfrak{t}_1$ extended with unary predicates.

Proposition 5.7 *Let $P_1, \ldots, P_n$ be unary predicates for $\mathfrak{t}_1$. There is no interpretation $\mathcal{I}$ such that $\mathcal{I}(\mathfrak{t}_1[P_1, \ldots, P_n]) \cong \mathfrak{t}_2$.*

Proof. As already mentioned there is no interpretation $\mathcal{I}$ such that $\mathcal{I}(\mathfrak{t}_1) \cong \mathfrak{t}_2$. Otherwise we could extend this interpretation by a formula transferring the ordering on $\mathfrak{t}_1$ to $\mathfrak{t}_2$, thus obtaining a well-ordered tree with decidable MSO-theory (contradicting Theorem 5.1). Now assume that there is an interpretation $\mathcal{I}$ such that $\mathcal{I}(\mathfrak{t}_1[P_1, \ldots, P_n]) \cong \mathfrak{t}_2$. We first construct a formula $\phi_{\text{tree}}(X_1, \ldots, X_n)$ such that

$$\mathfrak{t}_1 \models \phi_{\text{tree}}[U_1, \ldots, U_n] \text{ iff } \mathcal{I}(\mathfrak{t}_1[U_1, \ldots, U_n]) \cong \mathfrak{t}_2.$$

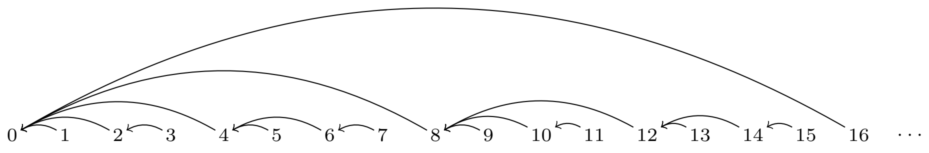

Figure 4: Graph of the flip function

Such a formula only needs to express that the formulas from the interpretation describe a structure in which all nodes have exactly one E_0 successor, exactly one E_1 successor, and exactly one predecessor except for one node which is the root. This can easily be done by an MSO-formula.

The formula $\exists X_1, \ldots, X_n : \phi_{\text{tree}}(X_1, \ldots, X_n)$ is true in $\mathfrak{t}_2$ (interpreting $X_1, \ldots, X_n$ by $P_1, \ldots, P_n$). Hence, there are regular interpretations $U_1, \ldots, U_n$ of $X_1, \ldots, X_n$ such that $\mathfrak{t}_1 \models \phi_{\text{tree}}[U_1, \ldots, U_n]$ (see Theorem 2.2). By construction of ϕ_{tree} this means that $\mathcal{I}(\mathfrak{t}_1[U_1, \ldots, U_n]) \cong \mathfrak{t}_2$. Since regular sets can be defined in MSO we can directly refer to $U_1, \ldots, U_n$ in the interpretation and obtain in interpretation $\mathcal{I}'$ such that $\mathcal{I}'(\mathfrak{t}_1) = \mathfrak{t}_2$, which is not possible. $\Box$

Note that Proposition 5.7 does not make any assumption on the predicates $P_1, \ldots, P_n$.

Another consequence of Theorem 5.1 is that $\mathfrak{t}_2$ is not MSO-interpretable in any structure that has a decidable MSO-theory and admits an MSO-definable well-ordering because otherwise one could also MSO-interpret $\mathfrak{t}_2$ extended with a well-ordering in a structure with decidable MSO-theory.

Consider, for example, the function *flip* on the natural numbers. It maps a natural number n to the number that is obtained by changing the least significant bit that is 1 in the binary representation of n to 0. The graph of the *flip* function is shown in Figure 4. The structure $(\mathfrak{t}_1, \mathit{flip})$ of the natural numbers with successor extended by the *flip* function has a decidable MSO-theory [17]. We conclude that $\mathfrak{t}_2$ cannot be MSO-interpretable in this structure because otherwise we could also interpret a well-ordered tree in it.

Proposition 5.8 *There is no MSO-interpretation $\mathcal{I}$ with $\mathcal{I}(\mathfrak{t}_1, \mathit{flip}) \cong \mathfrak{t}_2$.*

6 Conclusion

In this paper we have studied questions on MSO-definability in the binary tree and on decidability of MSO in extensions of the binary tree by well-orderings. As a result we obtain a rather simple and elementary proof of the theorem of Gurevich and Shelah stating that there is no MSO-definable choice function on the infinite binary tree. A simple consequence is that there is also no MSO-definable well-ordering on the domain of the infinite binary tree. We obtain the even stronger result that adding any partial well-ordering to the infinite binary tree yields a structure with undecidable MSO-theory. These two results can be used to derive non-definability results for MSO, as we have illustrated with some examples.

One natural question that remains open is whether there exists a choice function that can be added to the infinite binary tree such that the resulting structure has a decidable MSO-theory.

As mentioned in the introduction, MSO-definability of a choice function is a very special instance of the uniformization problem, namely for the formula $\phi(X, y) = y \in X$. The result from Section 3 shows that uniformization is not possible in general on the infinite binary tree. This leaves the question whether there are other types of formulas that allow uniformization. In [24] it is mentioned that uniformization is possible for formulas of the form $\phi(x, Y)$. Here, uniformization means that the relation between elements and sets defined by $\phi(x, Y)$ can be turned into an MSO-definable function associating to each element exactly one set from the relation.[5]

Another type of question related to this is the one of decidability, as for example in Church's synthesis problem [7] (see also [30]). An instance of this problem is given by an MSO-formula $\phi(\overline{X}, \overline{Y})$ over the infinite line $\mathfrak{t}_1$ such that for each input sequence $\overline{X}$ there is at least one output sequence $\overline{Y}$. A solution is a very specific function compatible with this relation: It should be implementable by a finite state automaton that reads the input sequence and produces in each step one element of the output sequence. The task is now to decide if such an automaton exists (and to construct one if possible). Similarly, one can study the decision variant of uniformization on the binary tree: Given an MSO-formula $\phi(\overline{X}, \overline{Y})$ over $\mathfrak{t}_2$, does there exist an MSO-formula $\phi^*(\overline{X}, \overline{Y})$ that defines a function compatible with $\phi(\overline{X}, \overline{Y})$. In its full generality this question seems to be too difficult but one could study specific instances of it for simple classes of formulas.

Acknowledgements. We would like to thank the anonymous referees for their helpful comments.

References

[1] A. Arnold. Rational ω-languages are non-ambiguous. *Theoret. Comput. Sci.*, 26:221–223, 1983.

[2] J. R. Büchi. On a decision method in restricted second order arithmetic. In Nagel E., Suppes P., and Tarski A., editors, *Proceedings of International*

[5]Such a result can be shown by transforming the formula into an equivalent automaton and then constructing a formula that selects for each x a unique run that accepts the tree annotated with x and some set Y. For this purpose it is enough to select a (say lexicographically) smallest run on the path to the element x that can be completed to an accepting run. In this finite part of the run we can plug for each 'dangling' state a fixed MSO-definable run. The details of this construction are left to the reader.

Congress on Logic, Methodology and Philosophy of Science, pages 1–11. Stanford University Press, 1962.

[3] J. R. Büchi and L. H. Landweber. Solving sequential conditions by finite-state strategies. *Trans. Amer. Math. Soc.*, 138:295–311, 1969.

[4] A. Carayol and C. Löding. MSO on the infinite binary tree: Choice and order. In J. Duparc and T. A. Henzinger, editors, *Proceedings of the 16th Annual Conference of the European Association for Computer Science Logic, CSL 2007*, volume 4646 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2007.

[5] A. Carayol and S. Wöhrle. The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In P. K. Pandya and J. Radhakrishnan, editors, *Proceedings of the 23rd Conference on Foundations of Software Technology and Theoretical Computer Science, FST TCS 2003*, volume 2914 of *Lecture Notes in Computer Science*, pages 112–123. Springer, 2003.

[6] D. Caucal. On infinite terms having a decidable monadic theory. In K. Diks and W. Rytter, editors, *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science, MFCS 2002*, volume 2420 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2002.

[7] A. Church. Logic, arithmetic and automata. In *Proceedings of the International Congress of Mathematicians*, pages 23–35, 1962.

[8] A. Dawar and E. Grädel. The descriptive complexity of parity games. In M. Kaminski and S. Martini, editors, *Proceedings of the 17th Annual Conference on Computer Science Logic, CSL 2008*, volume 5213 of *Lecture Notes in Computer Science*, pages 354–368. Springer, 2008.

[9] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Perspectives in Mathematical Logic. Springer, Berlin, 1995.

[10] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

[11] Y. Gurevich and S. Shelah. Rabin's uniformization problem. *J. Symbolic Logic*, 48(4):1105–1119, 1983.

[12] D. Kähler and T. Wilke. Complementation, disambiguation, and determinization of Büchi automata unified. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *Proceedings of the 35th International Colloquium on Automata, Languages and Programming, ICALP 2008, Part I*, volume 5125 of *Lecture Notes in Computer Science*, pages 724–735. Springer, 2008.

[13] S. Lifsches and S. Shelah. Uniformization, choice functions and well orders in the class of trees. *J. Symbolic Logic*, 61(4):1206–1227, 1996.

[14] S. Lifsches and S. Shelah. Uniformization and Skolem functions in the class of trees. *J. Symbolic Logic*, 63(1):103–127, 1998.

[15] C. Löding. *Automata and Logics over Infinite Trees.* Habilitationsschrift, RWTH Aachen, 2009.

[16] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9(5):521–530, 1966.

[17] A. Monti and A. Peron. Systolic tree omega-languages: the operational and the logical view. *Theoret. Comput. Sci*, 233(1–2):1–18, 2000.

[18] A. W. Mostowski. Regular expressions for infinite trees and a standard form of automata. In A. Skowron, editor, *Computation Theory*, volume 208 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 1984.

[19] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, July 1969.

[20] M. O. Rabin. *Automata on Infinite Objects and Church's Problem.* American Mathematical Society, Boston, MA, USA, 1972.

[21] A. Rabinovich. On decidability of monadic logic of order over the naturals extended by monadic predicates. *Information and Computation*, 205(6):870–889, 2007.

[22] S. Safra. On the complexity of omega-automata. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science, FoCS 1988*, pages 319–327, Los Alamitos, California, October 1988. IEEE Computer Society Press.

[23] H. Seidl. Deciding equivalence of finite tree automata. *SIAM J. Comput.*, 19(3):424–437, 1990.

[24] A. L. Semenov. Decidability of monadic theories. In P. Czechoslovakia, M. Chytil, and V. Koubek, editors, *Proceedings of the 11th International Symposium on Mathematical Foundations of Computer Science, MFCS 1984*, volume 176 of *Lecture Notes in Computer Science*, pages 162–175. Springer, June 1984.

[25] D. Siefkes. The recursive sets in certain monadic second order fragments of arithmetic. *Arch. für mat. Logik und Grundlagenforschung*, 17:71–80, 1975.

[26] R. E. Stearns and H. B. Hunt III. On the equivalence and containment problems for unambiguous regular expressions, regular grammars and finite automata. *SIAM J. Comput.*, 14(3):598–611, 1985.

[27] W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 133–192. Elsevier Science Publishers, Amsterdam, 1990.

[28] W. Thomas. On the synthesis of strategies in infinite games. In E. W. Mayr and C. Puech, editors, *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science, STACS '95*, volume 900 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 1995.

[29] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Language Theory*, volume III, pages 389–455. Springer, 1997.

[30] W. Thomas. Church's problem and a tour through automata theory. In *Pillars of Computer Science, Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, volume 4800 of *Lecture Notes in Computer Science*, pages 635–655. Springer, 2008.