Generalised Quantifiers Based on Rabin-Mostowski Index

Denis Kuperberg
CNRS, LIP, Plume, ENS Lyon, France

Damian Niwiński
Institute of Informatics, University of Warsaw, Poland

Paweł Parys
Institute of Informatics, University of Warsaw, Poland

Michał Skrzypczak
Institute of Informatics, University of Warsaw, Poland

Abstract

In this work we introduce new generalised quantifiers which allow us to express the Rabin-Mostowski index of automata. Our main results study expressive power and decidability of the monadic second-order (MSO) logic extended with these quantifiers. We study these problems in the realm of both ω-words and infinite trees. As it turns out, the pictures in these two cases are very different. In the case of ω-words the new quantifiers can be effectively expressed in pure MSO logic. In contrast, in the case of infinite trees, addition of these quantifiers leads to an undecidable formalism.

To realise index-quantifier elimination, we consider the extension of MSO by game quantifiers. As a tool, we provide a specific quantifier-elimination procedure for them. Moreover, we introduce a novel construction of transducers realising strategies in ω-regular games with monadic parameters.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects; Theory of computation → Logic and verification; Theory of computation → Tree languages

Keywords and phrases monadic quantifiers, decidability, quantifier elimination, parity automata, game quantifier, Rabin-Mostowski index

Digital Object Identifier 10.4230/LIPIcs.STACS.2026.63

Related Version *Full Version*: https://arxiv.org/abs/2601.04739 [23]

Funding *Denis Kuperberg*: ANR ReCiProg.
Damian Niwiński: National Science Centre, Poland (grant no. 2024/55/B/ST6/00318).
Paweł Parys: National Science Centre, Poland (grant no. 2024/55/B/ST6/00318).
Michał Skrzypczak: National Science Centre, Poland (grant no. 2024/55/B/ST6/00318).

1 Introduction

Monadic second-order logic (MSO) considered over ω-words or infinite trees sets a golden standard in the theory of verification as a robust, expressive, yet still decidable formalism. The research surrounding this logic often takes two paths.

One focuses on properties of the MSO-definable languages of ω-words or trees, with an emphasis on decidability issues, aiming in effective characterisations. Another path, maybe more challenging, attempts to extend the expressive power of MSO while still maintaining decidability. These two paths often interplay, an archetypal example being the study of cardinality. First, Niwiński [35] showed that the cardinality of a regular language of infinite trees can be effectively computed. Then, Bárány, Kaiser, and Rabinovich [1] (see also [20]) studied an extension of the MSO logic (over the binary tree) by *cardinality quantifiers*, like $\exists^{\geq\kappa} X.\, \varphi(\vec{W}, X)$, stating that there are at least κ distinct sets X satisfying $\varphi(\vec{W}, X)$. The

43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026).
Editors: Meena Mahajan, Florin Manea, Annabelle McIver, and Nguyễn Kim Thắng
Article No. 63; pp. 63:1–63:22
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

extension turned out to admit an elimination procedure for cardinality quantifiers: the authors effectively translated MSO with cardinality quantifiers into pure MSO, rendering the considered formalism decidable [1].

In contrast, the *unboundedness* quantifier $\mathsf{U}X.\,\varphi(\vec{W}, X)$ introduced by Bojańczyk [2], stating that the formula $\varphi(\vec{W}, X)$ is satisfied by finite sets X of unbounded size, leads to a proper extension of MSO. After exhaustive investigation it was shown that MSO+U is undecidable even over ω-words [4]. However, the unboundedness property of a given regular language is easily decidable (due to an application of the pumping lemma); a related property called diagonality was shown to be decidable even for tree languages on all levels of the Caucal hierarchy [13].

The results of Niwiński, Parys, and Skrzypczak [38] fall into a similar category: the authors show that the ranks of MSO-definable well-founded relations satisfy a certain dichotomy and can be effectively bounded, although the rank itself is not directly expressible in MSO.

A general pattern behind these situations consists of several levels. On the basic level, we wish to decide if a language of ω-words or trees satisfies a specific property, usually related to some *difficulty*: uncountability, unboundedness, ordinal rank ω_1, etc. Then we ask if the property can be generalised to a type of quantifier, and whether the extension of MSO is proper, and eventually decidable.

The study in the present paper is motivated by the Rabin-Mostowski index problem, which is a pertinent open problem in automata theory. In terms of parity automata (see below), the question is to find an equivalent automaton of a given type (deterministic, non-deterministic, or alternating) with a minimal number of priorities. For technical reasons, we also take into account the minimal priority, so that an *index* is defined as a pair (i, j) (where i can be assumed to be 0 or 1). Recall that the index hierarchy over ω-words is strict only for deterministic automata, and collapses to the second level for non-deterministic and alternating ones. For infinite trees, both non-deterministic and alternating hierarchies are strict [6, 34]; the deterministic hierarchy is strict as well, but less interesting because deterministic tree automata do not capture all regular tree languages. The problem of computing the index is generally decidable for automata over ω-words [48], and open for automata over infinite trees. Several special cases have been shown decidable, in particular if an input tree automaton is a deterministic automaton [36, 37]; a *game automaton* [17]; or a Büchi automaton [14, 45]. Colcombet and Löding [15] reduced the non-deterministic index problem to a question on asymptotic behaviour of counter automata; their paper brought a bunch of interesting ideas (in particular, guidability), but the original problem has remained unsolved.

In the current paper, we approach the index problem "from above", that is, we introduce a class of quantifiers corresponding to the index property. Using the correspondence between sets (or tuples thereof) and their characteristic functions (i.e., labelled infinite words or trees), a general form of the new quantifier is

$$\mathtt{I}^{D}_{\mathcal{R}}X.\,\varphi(W_1, \ldots, W_k, X)$$

where D refers to the type of involved automata (deterministic or non-deterministic), and $\mathcal{R}$ determines the index. Such a formula holds for a valuation $\bar{w}_1, \ldots, \bar{w}_k$ if there exists an automaton $\mathcal{A}$ of type D and index $\mathcal{R}$, such that for every $\bar{x}$ the formula $\varphi(\bar{w}_1, \ldots, \bar{w}_k, \bar{x})$ holds if and only if $\mathcal{A}$ accepts $\langle \bar{w}_1, \ldots, \bar{w}_k, \bar{x} \rangle$. Note that in the above only $\bar{x}$ varies while the $\bar{w}_i$'s remain fixed, playing the role of parameters.

Our main results are twofold. First, we show that MSO+$\mathtt{I}$ effectively reduces to pure MSO over ω-words. Second, we prove that MSO+$\mathtt{I}$ is undecidable over infinite trees. To the best of our knowledge, this is the first negative decidability result for index-related problems over infinite trees. In fact, we establish undecidability already for the quantifier $\mathtt{I}_{\mathrm{safety}}$, which refers to automata that merely avoid some designated rejecting states. This stands in sharp contrast to the fact that deciding whether a regular tree language can be recognised by a safety automaton is straightforward, as it amounts to checking closedness in the standard topology on infinite trees (see, e.g., [15, 25]).

To achieve the positive part of our results, namely index-quantifier elimination over ω-words, we rely on a variant of Wadge games for the index hierarchy [26, 47]. These games can naturally be expressed in MSO equipped with *game quantifier* $\mathfrak{D}$ (see, e.g., the monograph by Moschovakis [31]). The fact that MSO+$\mathfrak{D}$ reduces to pure MSO follows from Kaiser [19] (we provide a direct proof adapted to our setup for the sake of completeness); nevertheless, we need a stronger property, allowing us to construct finite memory strategies (relating Büchi-Landweber construction [11] with uniformisations [24, 42]). This falls in similar lines as results by Winter and Zimmermann [51] and others on sequential uniformisation and functions realised by transducers. To achieve our goal, we show a novel fact, which can be seen as a parametrised version of Büchi-Landweber construction (for the case when the variables are in some sense separated). We believe that both game quantifiers in general, and this new fact are of independent interest and applications.

One can ask if the new quantifiers of our paper align with the concept of *generalised quantifiers* introduced by Mostowski [32] (see [49] for a survey). The idea there is that a formula $\mathtt{Q}x.\, \varphi(\vec{w}, x)$ expresses the fact that the x's satisfying $\varphi(\vec{w}, x)$ (for fixed parameters $\vec{w}$) fall into a specified family of subsets of the universe (e.g., all non-empty sets for $\exists$, and the singleton of the whole universe for $\forall$). More generally, a quantifier can bind k variables ($\mathtt{Q}x_1 \ldots x_k.\, \varphi(\vec{w}, \vec{x})$) and relate to a family of k-ary relations. These concepts can be adapted to MSO, where in the semantics of a quantifier $\mathtt{Q}X$ (or $\mathtt{Q}\vec{X}$), the universe is replaced by its powerset. The examples mentioned above, namely cardinality quantifiers and the unboundedness quantifier, can be easily presented in this way. The newly introduced index quantifiers and game quantifiers can as well be presented as generalised quantifiers. For an interested reader, we discuss this issue in more detail in Section 4.

2 Preliminaries

An alphabet A is a finite non-empty set of symbols. As usual, by A^* we denote the set of finite words over A, by A^+ the set of non-empty finite words over A, and by A^ω the set of ω-words over A, that is, functions from $\mathbb{N} = \{0, 1, 2, \ldots\}$ to A. The empty word is denoted $\varepsilon \in A^*$ and concatenation of two words u, v is denoted by $u \cdot v$. Given $n \in \mathbb{N}$ and either a finite word with at least n symbols or an ω-word $\bar{w} = w_0 w_1 w_2 \cdots$ by $\bar{w}{\restriction}_n$ we denote the finite word $w_0 w_1 \cdots w_{n-1}$, that is, $\bar{w}$ restricted to the first n symbols. An ω-word of the form $x \cdot y \cdot y \cdot y \cdot \ldots$ for some finite words $x, y \in A^+$ is called *ultimately periodic*. The prefix order on words is denoted by $\preceq$, with $\bar{w} \preceq \bar{w}'$ if there exists $n \in \mathbb{N}$ such that $\bar{w} = \bar{w}'{\restriction}_n$.

A (full, infinite, binary) tree over an alphabet A is any function $t\colon \{\mathtt{L}, \mathtt{R}\}^* \to A$; here a word in $\{\mathtt{L}, \mathtt{R}\}^*$ describes a path from the root ε to a node $x \in \{\mathtt{L}, \mathtt{R}\}^*$, with $\mathtt{L}$ being the left child and $\mathtt{R}$ the right child. The *label* of such a node is $t(x) \in A$. The set of all such trees is denoted Tr_A.

We use the standard terms to navigate within a tree, in particular x is a *descendant* of y if $x \succeq y$. In an analogous way we use the terms *ascendant*, *parent*, and *sibling*.

Subsets $L \subseteq A^*$, $L \subseteq A^\omega$, or $L \subseteq \mathrm{Tr}_A$ are called *languages*.

Transducers. In this work we use (sequential, deterministic, finite-memory) transducers from one alphabet to another. Assume that A_W, A_Y are some alphabets. A transducer τ from A_W to A_Y (denoted $\tau\colon A_W \twoheadrightarrow A_Y$) is a tuple $\tau = \langle A_W, A_Y, Q_\tau, \iota_\tau, \delta_\tau\rangle$, where:

- Q_τ is a finite set of *states*,
- $\iota_\tau \in Q_\tau$ is the *initial state*,
- $\delta_\tau\colon Q_\tau \times A_W \to A_Y \times Q_\tau$ is the *transition function.*

Given an *input ω-word* $\bar{w} = w_0w_1w_2\cdots \in (A_W)^\omega$ we inductively define the *run* $\bar{\rho} \stackrel{\text{def}}{=} \rho_0\rho_1\rho_2\cdots \in Q^\omega$ and the *output ω-word* $\tau(\bar{w}) \stackrel{\text{def}}{=} y_0y_1y_2\cdots \in (A_Y)^\omega$ taking $\rho_0 \stackrel{\text{def}}{=} \iota_\tau$ and $(y_n, \rho_{n+1}) \stackrel{\text{def}}{=} \delta_\tau(\rho_n, w_n)$ for all $n \in \mathbb{N}$.

Given two transducers $\tau\colon A_W \twoheadrightarrow A_Y$ and $\tau'\colon A_Y \twoheadrightarrow A_Z$ it is easy to construct the *composition* of the two, namely a transducer $\theta\colon A_W \twoheadrightarrow A_Z$ such that for every $\bar{w} \in (A_W)^\omega$ we have $\theta(\bar{w}) = \tau'\big(\tau(\bar{w})\big)$.

Parity indices. Assume that $i, j \in \mathbb{N}$ are natural numbers with $i \leq j$. The *(strong) parity index* $P_{i,j}$ and the *weak parity index* $W_{i,j}$ are defined by the languages

$$P_{i,j} \stackrel{\text{def}}{=} \{k_0k_1k_2\ldots \in \{i, i{+}1, \ldots, j\}^\omega \mid \limsup_{n\to\infty} k_n \equiv 0 \bmod 2\},$$

$$W_{i,j} \stackrel{\text{def}}{=} \{k_0k_1k_2\ldots \in \{i, i{+}1, \ldots, j\}^\omega \mid \sup_{n\in\mathbb{N}} k_n \equiv 0 \bmod 2\}.$$

An *index* is a pair $\mathcal{R} = \langle A_\mathcal{R}, L_\mathcal{R}\rangle$ that is either $\mathcal{P}_{i,j} = \langle\{i, i{+}1, \ldots, j\}, P_{i,j}\rangle$ or $\mathcal{W}_{i,j} = \langle\{i, i{+}1, \ldots, j\}, W_{i,j}\rangle$ for some $i, j \in \mathbb{N}$ with $i \leq j$.

The typical names for indices are: *Büchi* for $\mathcal{P}_{1,2}$ (infinitely many times priority 2), *co-Büchi* for $\mathcal{P}_{0,1}$ (finitely many times priority 1), *safety* for $\mathcal{W}_{0,1}$ (reaching priority 1 implies that we reject), and *reachability* for $\mathcal{W}_{1,2}$ (reaching priority 2 implies that we accept).

Automata over ω-words. A *non-deterministic parity ω-word automaton* over an alphabet A and of index $\mathcal{R} = \langle A_\mathcal{R}, L_\mathcal{R}\rangle$ is a tuple $\mathcal{D} = \langle A, \mathcal{R}, Q_\mathcal{D}, \iota_\mathcal{D}, \delta_\mathcal{D}\rangle$, where:

- $Q_\mathcal{D}$ is a finite set of *states*,
- $\iota_\mathcal{D} \subseteq Q_\mathcal{D}$ is the set of *initial states*,
- $\delta_\mathcal{D} \subseteq Q_\mathcal{D} \times A \times A_\mathcal{R} \times Q_\mathcal{D}$ is the *transition relation*,

and moreover the automaton is *complete*[1] in the sense that for every $q \in Q_\mathcal{D}$ and $a \in A$ there is at least one transition of the form $(q, a, k, q') \in \delta_\mathcal{D}$.

A *run* of an automaton $\mathcal{D}$ over an *input ω-word* $\bar{w} = w_0w_1w_2\cdots \in A^\omega$ producing *output ω-word* $\bar{k} = k_0k_1k_2\cdots \in (A_\mathcal{R})^\omega$ is a sequence of states $\bar{\rho} = \rho_0\rho_1\rho_2\cdots \in (Q_\mathcal{D})^\omega$ such that $\rho_0 \in \iota_\mathcal{D}$ and for every $n \in \mathbb{N}$ we have $(\rho_n, w_n, k_n, \rho_{n+1}) \in \delta_\mathcal{D}$. The ω-word $\bar{w}$ is *accepted* by $\mathcal{D}$ if there exists a run of $\mathcal{D}$ over $\bar{w}$ producing an ω-word $\bar{k}$ that belongs to $L_\mathcal{R}$.

The *language* of such an automaton, denoted $\mathrm{L}(\mathcal{D}) \subseteq A^\omega$, is the set of ω-words $\bar{w} \in A^\omega$ that are accepted by $\mathcal{D}$. A language $L \subseteq A^\omega$ is *ω-regular* if it is the language of some automaton.

An automaton is *deterministic* if $\iota_\mathcal{D}$ is a singleton and the transition relation $\delta_\mathcal{D}$ is in fact a function $\delta_\mathcal{D}\colon Q_\mathcal{D} \times A \to A_\mathcal{R} \times Q_\mathcal{D}$, in which case there is a unique run of $\mathcal{D}$ over every input ω-word $\bar{w} \in A^\omega$.

▶ Remark 2.1. If the index $\mathcal{R} = \langle A_\mathcal{R}, L_\mathcal{R}\rangle$ is fixed, then deterministic automata $\mathcal{D}$ over A and of index $\mathcal{R}$ are in natural bijection with transducers $\tau\colon A \twoheadrightarrow A_\mathcal{R}$ in such a way that $\mathrm{L}(\mathcal{D}) = \{\bar{w} \in A^\omega \mid \tau(\bar{w}) \in L_\mathcal{R}\}$.

[1] This technical assumption plays a role when considering weak indices of automata.

Ramsey theorem. Let C be a finite set of *colours*. An *edge labelling* of a set X is a function that to each edge $\{i,j\} \subseteq X$ (where $i \neq j$) assigns a colour from C. Given an edge labelling, we say that a set $I \subseteq X$ is *monochromatic* if all edges $\{i,j\} \subseteq I$ have the same colour.

▶ **Theorem 2.2** (Ramsey). *Let C be a finite set and let $k \in \mathbb{N}$. Then, there exists a computable constant $r \in \mathbb{N}$ such that for every edge labelling of $\{0,1,\ldots,r-1\}$ by colours from C there exists a monochromatic set $I \subseteq \{0,1,\ldots,r-1\}$ of size k.*

Moreover, for every edge labelling of $\mathbb{N}$ by colours from C there exists an infinite monochromatic set $I \subseteq \mathbb{N}$.

Semigroups and monoids. An algebraic structure $\langle S, (\cdot)\rangle$ with an associative binary operation $(\cdot)$ is called a *semigroup*. A *monoid* is a semigroup S which contains a *neutral element* $\varepsilon \in S$ such that $\varepsilon \cdot s = s \cdot \varepsilon = s$ for every $s \in S$. Every semigroup S can be extended into a monoid $S_{+\varepsilon} = S \cup \{\varepsilon\}$ by adding a formal neutral element ε with product defined appropriately. An *idempotent* is an element $e \in S$ such that $e \cdot e = e$.

The following fact is a standard application of Ramsey theorem (cf. Theorem 2.2).

▶ **Fact 2.3.** *For every finite semigroup S there exists a computable constant $r \in \mathbb{N}$ such that for every word $s_0 s_1 \cdots s_{r-1} \in S^r$ there exists a pair of positions $0 \leq i < j < r$ such that $e \stackrel{def}{=} s_{i+1} \cdot s_{i+2} \cdot \ldots \cdot s_j$ is an idempotent.*

In particular, putting $c \stackrel{def}{=} s_0 \cdot s_1 \cdot \ldots \cdot s_j$ we have

$$c \cdot e = s_0 \cdot s_1 \cdot \ldots \cdot s_{i-1} \cdot e \cdot e = s_0 \cdot s_1 \cdot \ldots \cdot s_{i-1} \cdot e = c.$$

Wilke algebras. In this work we use Wilke algebras as representations of ω-semigroups, as in Perrin and Pin [39]. A *Wilke algebra* S consists of two sets $(S^{\mathrm{fin}}, S^{\mathrm{inf}})$, two product operations

$$S^{\mathrm{fin}} \times S^{\mathrm{fin}} \to S^{\mathrm{fin}} \qquad \text{and} \qquad S^{\mathrm{fin}} \times S^{\mathrm{inf}} \to S^{\mathrm{inf}}$$

denoted $s \cdot s'$ for operands s, s', and an operation $S^{\mathrm{fin}} \to S^{\mathrm{inf}}$ denoted s^ω for an operand $s \in S^{\mathrm{fin}}$. Moreover, the operations are required to satisfy natural associativity axioms, in particular S^{fin} needs to be a semigroup. Each finite Wilke algebra S uniquely determines the *infinite product* operation $\odot\colon (S^{\mathrm{fin}})^\omega \to S^{\mathrm{inf}}$, which is associative. In particular $\odot(sss\cdots) = s^\omega$ and $\odot(s_0 s_1 \cdots) = s_0 \cdot \odot(s_1 s_2 \cdots)$.

A *homomorphism* α between two Wilke algebras S and T is a pair of functions $\alpha^{\mathrm{fin}}\colon S^{\mathrm{fin}} \to T^{\mathrm{fin}}$ and $\alpha^{\mathrm{inf}}\colon S^{\mathrm{inf}} \to T^{\mathrm{inf}}$ that commute with all the operations of the algebras and with the infinite product $\odot$.

Recognition. A canonical example of a Wilke algebra is $A^{\mathsf{W}} \stackrel{\mathrm{def}}{=} \langle A^+, A^\omega\rangle$, where A is an alphabet. The operations of this Wilke algebra are the concatenation $\cdot$, the infinite repetition $v^\omega \stackrel{\mathrm{def}}{=} v \cdot v \cdot v \cdot \ldots \in A^\omega$ for $v \in A^+$, and the infinite product $\odot(v_0 v_1 v_2 \cdots) \stackrel{\mathrm{def}}{=} v_0 \cdot v_1 \cdot v_2 \cdot \ldots \in A^\omega$ for $v_0, v_1, v_2, \ldots \in A^+$.

Associativity properties imply that if $\alpha\colon A^{\mathsf{W}} \to S$ is a homomorphism into a finite Wilke algebra then for every sequence of finite words $v_0, v_1, \ldots \in A^+$ we have

$$\alpha(v_0 \cdot v_1 \cdot v_2 \cdot \ldots) = \odot\big(\alpha(v_0)\alpha(v_1)\alpha(v_2)\cdots\big). \tag{2.1}$$

Note that if a Wilke algebra is finite then it can be represented as an input to an algorithm by providing its list of elements and “multiplication tables” for all the operations. The crucial fact about Wilke algebras is their ability to recognise ω-regular languages, as stated by the following theorem.

▶ **Theorem 2.4** ([50]). *Given a tuple of ω-regular languages $(L_0, \dots, L_{k-1})$ with $L_i \subseteq A^\omega$ for all $i < k$, one can effectively compute a finite Wilke algebra S together with a homomorphism $\alpha \colon A^{\mathsf{W}} \to S$ and a tuple of sets $(F_0, \dots, F_{k-1})$, where for every $i < k$ the set $F_i \subseteq S^{\mathrm{inf}}$ is such that $L_i = \alpha^{-1}(F_i)$. We say that α* recognises *$(L_0, \dots, L_{k-1})$ with $(F_0, \dots, F_{k-1})$.*

Moreover, one can require α to be onto *in the sense that $\alpha(A^+) = S^{\mathrm{fin}}$ and $\alpha(A^\omega) = S^{\mathrm{inf}}$.*

Let $\bar{z} = z_0 z_1 z_2 \cdots \in (S^{\mathrm{fin}}_{+\varepsilon})^\omega$ (recall that $S^{\mathrm{fin}}_{+\varepsilon}$ is S^{fin} extended with a formal neutral element ε). We say that $\bar{z}$ is *saturated* if it contains infinitely many symbols from S^{fin}, that is, symbols different than ε. In this case $\odot(\bar{z})$ is well-defined: we can erase all symbols ε from $\bar{z}$ obtaining an ω-word $\bar{z}' \in (S^{\mathrm{fin}})^\omega$ and put $\odot(\bar{z}) \stackrel{\mathrm{def}}{=} \odot(\bar{z}')$. This definition again satisfies the associativity properties as in Formula (2.1).

Lookahead and composition. Assume that $\alpha \colon A^{\mathsf{W}} \to S$ is a homomorphism into a finite Wilke algebra $S = (S^{\mathrm{fin}}, S^{\mathrm{inf}})$. For every $\bar{w} = w_0 w_1 w_2 \cdots \in A^\omega$ this homomorphism defines the *lookahead* $\mathsf{lk}_\alpha(\bar{w}) \in (S^{\mathrm{inf}})^\omega$ defined for each position $n \in \mathbb{N}$ as

$$\big(\mathsf{lk}_\alpha(\bar{w})\big)_n \stackrel{\mathrm{def}}{=} \alpha\big(w_{n+1} w_{n+2} w_{n+3} \cdots\big) \in S^{\mathrm{inf}}.$$

Note that, while producing a letter on a position $n \in \mathbb{N}$, a transducer uses letters on positions $0, 1, \dots, n$. On the other hand, a lookahead at position n depends on positions $n{+}1, n{+}2, n{+}3, \dots$ To create an output ω-word whose output letters in A_Y depend on both the past and the future of input ω-words, we consider transducers whose output letters are functions $(S^{\mathrm{inf}} \to A_Y)$, and then we apply these functions to letters in S^{inf} produced by a lookahead.

To simplify the notation, we use the following shorthand: if $\bar{f} = f_0 f_1 f_2 \cdots \in (A_X \to A_Y)^\omega$ and $\bar{x} = x_0 x_1 x_2 \cdots \in (A_X)^\omega$, then $\bar{f} \bullet \bar{x} \in (A_Y)^\omega$ is defined for each position $n \in \mathbb{N}$ as $(\bar{f} \bullet \bar{x})_n = f_n(x_n)$.

Automata over infinite trees. A *non-deterministic parity tree automaton* over an alphabet A and of index $\mathcal{R} = \langle A_{\mathcal{R}}, L_{\mathcal{R}} \rangle$ is a tuple $\mathcal{A} = \langle A, \mathcal{R}, Q_{\mathcal{A}}, \iota_{\mathcal{A}}, \Delta_{\mathcal{A}} \rangle$, where $Q_{\mathcal{A}}$ is a finite set of *states*, $\iota_{\mathcal{A}} \subseteq Q_{\mathcal{A}}$ a set of *initial states*, and $\Delta_{\mathcal{A}} \subseteq Q \times A \times A_{\mathcal{R}} \times Q \times Q$ a *transition relation*. Again we require the automaton to be complete, that is, for every $q \in Q_{\mathcal{A}}$ and $a \in A$ it needs to contain at least one transition $(q, a, k, q_{\mathtt{L}}, q_{\mathtt{R}}) \in \Delta_{\mathcal{A}}$.

A *run* of $\mathcal{A}$ over a tree $\tilde{t} \in \mathrm{Tr}_A$ producing an output tree $\tilde{\eta} \in \mathrm{Tr}_{A_{\mathcal{R}}}$ is a tree $\tilde{\rho} \in \mathrm{Tr}_{Q_{\mathcal{A}}}$ such that $\tilde{\rho}(\varepsilon) \in \iota_{\mathcal{A}}$ and $(\tilde{\rho}(v), \tilde{t}(v), \tilde{\eta}(v), \tilde{\rho}(v\mathtt{L}), \tilde{\rho}(v\mathtt{R})) \in \Delta_{\mathcal{A}}$ for all nodes $v \in \{\mathtt{L}, \mathtt{R}\}^*$. A tree $\tilde{t} \in \mathrm{Tr}_A$ is *accepted* by $\mathcal{A}$ if there exists a run of $\mathcal{A}$ over $\tilde{t}$ producing a tree $\tilde{\eta}$ such that for every branch $\bar{w} \in \{\mathtt{L}, \mathtt{R}\}^\omega$, the sequence $\tilde{\eta}(\bar{w}\restriction_0)\tilde{\eta}(\bar{w}\restriction_1)\tilde{\eta}(\bar{w}\restriction_2) \cdots \in (A_{\mathcal{R}})^\omega$ belongs to $L_{\mathcal{R}}$. The *language* of an automaton $\mathcal{A}$ is the set of trees which it accepts. A language $L \subseteq \mathrm{Tr}_A$ is a *regular tree language* if it is the language of some automaton $\mathcal{A}$.

A tree automaton is (top-down) *deterministic* if $\iota_{\mathcal{A}}$ is a singleton and $\Delta_{\mathcal{A}} \colon Q \times A \to A_{\mathcal{R}} \times Q \times Q$ is a function.

Monadic second-order logic. Formulae of the MSO logic are evaluated in an appropriate structure, which in our case is $\mathbb{N}$ with the successor relation (in the case of ω-words) or $\{\mathtt{L}, \mathtt{R}\}^*$ with the left-child and right-child relations (in the case of trees). Elements of the structure are called positions or nodes. Usually, a monadic variable in MSO represents a set of positions, which can be also seen as a word or a tree over the alphabet $\{0, 1\}$, with 1 indicating positions that are in the set. In this paper, we employ a seemingly more general setting, where each monadic variable X represents a word or a tree over some alphabet A_X, possibly larger than $\{0,1\}$. In the sequel, we usually assume a fixed alphabet A_X associated to each variable X, but sometimes we explicitly specify the alphabet next to a quantifier (writing e.g., $\exists X \in (A_X)^\omega.\, \varphi(X)$). Then, for a letter $x \in A_X$ and for a first-order variable v we have an atomic formula $X(v) = x$ checking whether the letter of X at the position v is x. This way of seeing monadic variables does not increase the expressive power of MSO, since a variable with values in A_X can be represented by a tuple of $|A_X|$ usual set variables, which should be forced to partition the domain (even $\lceil \log |A_X| \rceil$ set variables suffice).

By equivalence between MSO and regular languages [10, 28, 40], we know that for every MSO formula $\varphi(X_1, \ldots, X_n)$ we can construct a deterministic parity ω-word automaton (in the case of ω-words) or a non-deterministic parity tree automaton (in the case of trees) over the alphabet $A_{X_1} \times \ldots \times A_{X_n}$ which accepts exactly those ω-words / trees $\bar{w}$ over this alphabet for which $\varphi\big(\pi_1(\bar{w}), \ldots, \pi_n(\bar{w})\big)$ holds, where each $\pi_i(\bar{w})$ is obtained from $\bar{w}$ by projecting labels of all positions to their i-th coordinate. Note that the index of the constructed automaton depends on the formula φ and in general cannot be bounded [5, 34, 48].

To simplify the notation, we identify a structure $\bar{w}$ over such a product alphabet $A_{X_1} \times \ldots \times A_{X_n}$ with the tuple of structures $\langle \pi_1(\bar{w}), \ldots, \pi_n(\bar{w}) \rangle$ over respective alphabets. In particular, for a formula $\varphi(X_1, \ldots, X_n)$ we can speak about the *language* of a formula which is defined as the set of structures $\bar{w}$ over $A_{X_1} \times \ldots \times A_{X_n}$ that satisfy $\varphi\big(\pi_1(\bar{w}), \ldots, \pi_n(\bar{w})\big)$. Due to the ability of translating formulae into automata, these languages are always regular.

Games. We use the general framework of perfect information games of infinite duration played between two players (typically called Player I and Player II). Such a game is given by a tuple $\mathcal{G} = \langle A, L_\mathcal{G}, V_\mathcal{G} = V_\mathcal{G}^{(\mathrm{I})} \sqcup V_\mathcal{G}^{(\mathrm{II})}, \iota_\mathcal{G}, \delta_\mathcal{G} \rangle$ where A is an alphabet, $L_\mathcal{G} \subseteq A^\omega$ is a *winning condition*, $V_\mathcal{G}$ is a (possibly infinite) set of *positions*, partitioned into the positions of the respective players, $\iota_\mathcal{G} \in V_\mathcal{G}$ is an *initial position*, and $\delta_\mathcal{G} \subseteq V_\mathcal{G} \times A \times V_\mathcal{G}$ is an *edge relation* (again satisfying completeness property that each $v \in V_\mathcal{G}$ admits at least one edge $(v, a, v') \in \delta_\mathcal{G}$). The letter $a \in A$ is called the *label* of an edge $(v, a, v') \in \delta_\mathcal{G}$.

A play of such a game is played in rounds, with the initial position $v_0 = \iota_\mathcal{G}$. In round number $n \in \mathbb{N}$ the player P such that $v_n \in V_\mathcal{G}^{(P)}$ chooses an edge $(v_n, k_n, v_{n+1}) \in \delta_\mathcal{G}$ moving to the next position v_{n+1}. After an infinite play, Player II wins if and only if $\bar{k} \stackrel{\mathrm{def}}{=} k_0 k_1 k_2 \cdots$ belongs to $L_\mathcal{G}$. Classical theorems [27] imply that if $L_\mathcal{G}$ is sufficiently simple, then one of the players can ensure to win this game, that is, has a *winning strategy*. In general such a strategy for a player P is a tree-shaped object but we mostly work with *positional strategies*, that is, functions $\sigma^{(P)} \colon V_\mathcal{G}^{(P)} \to \delta_\mathcal{G}$ such that for every $v \in V_\mathcal{G}^{(P)}$ we have $\sigma^{(P)}(v) = (v, k, v')$ for some $k \in A$ and $v' \in V_\mathcal{G}$.

A *parity game* of index $\mathcal{R} = \langle A_\mathcal{R}, L_\mathcal{R} \rangle$ is a game $\mathcal{G}$ as above where $A = A_\mathcal{R}$ and $L_\mathcal{G} = L_\mathcal{R}$.

▶ **Theorem 2.5** ([16, 33]). *If $\mathcal{G}$ is a parity game then some player P has a positional winning strategy $\sigma^{(P)}$ in $\mathcal{G}$.*

3 New quantifiers

In this section we introduce the two types of quantifiers which are studied in this work. When doing so, we follow the convention to assume that in a formula $\mathtt{Q}X.\, \varphi(W_1, \ldots, W_k, X)$ all the parameter variables $W_1, \ldots, W_k$ are combined into a single free variable W over a product alphabet, as explained above. Thus, we focus on formulae of the form $\mathtt{Q}X.\, \varphi(W, X)$, even if the respective coordinates of W come from different outer quantifiers.

Index quantifiers. Consider a new quantifier $\mathtt{I}^{D}_{\mathcal{R}} X.\, \varphi(W, X)$ where $D \in \{\mathsf{dt}, \mathsf{nd}\}$ determines the type of involved automata and $\mathcal{R}$ is an index (either a strong parity index $\mathcal{P}_{i,j}$ or a weak parity index $\mathcal{W}_{i,j}$). Such a formula holds for a parameter $\bar{w} \in (A_W)^\omega$ if there exists an automaton $\mathcal{A}_{\bar{w}}$ of index $\mathcal{R}$, which is either deterministic ($D = \mathsf{dt}$) or non-deterministic ($D = \mathsf{nd}$), such that for every $\bar{x} \in (A_X)^\omega$ the formula $\varphi(\bar{w}, \bar{x})$ holds if and only if $\mathcal{A}_{\bar{w}}$ accepts $\langle \bar{w}, \bar{x} \rangle$.

Note that the parameter $\bar{w}$ occurs in the above definition in two roles. First, the automaton $\mathcal{A}_{\bar{w}}$ may depend on the parameter $\bar{w}$. Second, the automaton, when verifying whether the given $\bar{x}$ makes $\varphi(\bar{w}, \bar{x})$ true, has access not only to $\bar{x}$ but also to the parameter $\bar{w}$ (in particular, the automaton is over the alphabet $A_W \times A_X$).

► Remark 3.1. One may ask what changes if we consider another semantics of the index quantifier, where the hypothetical automaton does not have access to the parameters $\bar{w}$ but only reads the quantified ω-word $\bar{x}$. In this case the formalism becomes immediately undecidable. Indeed, consider the simplest possible formula $\mathtt{I}^{\mathsf{dt}}_{\mathcal{W}_{0,1}} X.\, (X = W)$, which involves the deterministic safety index quantifier. Then, for a given $\bar{w}$ the set of ω-words $\bar{x}$ that satisfy $\bar{x} = \bar{w}$ is $\{\bar{w}\} \subseteq (A_W)^\omega$. This language is recognised by a deterministic safety automaton if and only if $\bar{w}$ is ultimately periodic. Due to Bojańczyk et al. [3], this extended logic is undecidable.

Note that both deterministic and non-deterministic index quantifiers make sense for both ω-words and trees. Let MSO+$\mathtt{I}$ denote the extension of monadic second-order logic by index quantifiers.

Game quantifiers. As a natural way to study the index quantifier, we need to formalise within MSO the concept of the *game quantifier* $\Game$ (see [22, § 20.D] and [7, 8, 18, 21, 29, 31]). This quantifier, written $\Game X{\to}Y$ (alternatively, in some papers the symbol ⅁ is used), binds two monadic variables X and Y. A formula

$$\Game X{\to}Y.\, \varphi(W, X, Y)$$

holds, given a parameter $\bar{w} \in (A_W)^\omega$, if Player II has a winning strategy in the game $\mathcal{G}(\bar{w}, \varphi)$, defined as follows.[2] The game consists of infinitely many rounds. In a round $n \in \mathbb{N}$, Player I proposes a letter $x_n \in A_X$ and Player II answers with a letter $y_n \in A_Y$. At the end, Player II wins if and only if $\varphi(\bar{w}, \bar{x}, \bar{y})$ holds for $\bar{x} \stackrel{\text{def}}{=} x_0x_1x_2 \cdots \in (A_X)^\omega$ and $\bar{y} \stackrel{\text{def}}{=} y_0y_1y_2 \cdots \in (A_Y)^\omega$. This game can easily be represented by a formal game $\mathcal{G}$ with positions $V^{(\mathrm{I})}_{\mathcal{G}} \stackrel{\text{def}}{=} \mathbb{N}$ and $V^{(\mathrm{II})}_{\mathcal{G}} \stackrel{\text{def}}{=} \mathbb{N} \times A_X$ and $L_{\mathcal{G}}$ given by φ; however we do not need to study the exact structure of this game.

Typically, one applies the game quantifiers in the context where the involved games are determined, although the definition makes sense even without this assumption.

Let MSO+$\Game$ denote the extension of monadic second-order logic by game quantifiers. Note that as it is defined, the game quantifier makes sense only for ω-words, because the shape of the time-structure of a game of infinite duration is ω.

[2] Classically, in the works of Moschovakis and Kechris [22, 31] the "game quantifier" requires Player I to win the game, however in automata-theoretic context (e.g., the Church synthesis problem [11, 41]) or Wadge games [47], it is more customary to focus on Player II.

4 Generalised quantifiers

In this section we relate the quantifiers introduced in this paper to the general concept of *generalised quantifiers*. They were proposed by Mostowski [32] as an abstract logical construct that generalises the classical quantifiers $\exists$ and $\forall$. Since then, they became an important tool in various applications of logic (see, e.g., [49] for a survey).

At the syntactic level, a quantifier $\mathtt{Q}$ extends the language by a construction $\mathtt{Q}x.\,\varphi(\vec{w}, x)$, for an arbitrary formula φ. Here, a variable x is *bound* by $\mathtt{Q}$, whereas the variables in $\vec{w} = (w_1, \ldots, w_k)$ remain *free*. At the semantic level, the quantifier is associated with an operator, which, for any structure $\mathcal{M}$ (with universe M) defines a family of sets $\mathtt{Q}^{\mathcal{M}} \subseteq \mathsf{P}(M)$. Then, given a valuation $\vec{w} \mapsto \vec{a} \in M^k$, the formula $\mathtt{Q}x.\,\varphi(\vec{a}, x)$ holds in $\mathcal{M}$ if the set $\{b \in M \mid \varphi(\vec{a}, b) \text{ holds in } \mathcal{M}\}$ belongs to $\mathtt{Q}^{\mathcal{M}}$. In this setting, $\exists^{\mathcal{M}}$ is the family of all non-empty subsets of M, whereas $\forall^{\mathcal{M}} = \{M\}$. As a less standard example, one can express the property that the cardinality of the set of x's satisfying $\varphi(\vec{w}, x)$ belongs to some specified class of cardinals (i.e., $\exists^{\infty}$ says that the set is infinite), or that the set of x's that do satisfy φ and those that do not, have the same cardinality. It is usually assumed that the family $\mathtt{Q}^{\mathcal{M}}$ is invariant under permutations of M, but a weakening of this requirement is sometimes justified.

More generally, one can consider n-ary quantifiers, where a quantifier $\mathtt{Q}$ bounds simultaneously n variables and, respectively, $\mathtt{Q}^{\mathcal{M}}$ is a family of n-ary relations over M. For example, if $n = 2$ and $\mathtt{Q}^{\mathcal{M}}$ is the class of rectangles, that is, $\mathtt{Q}^{\mathcal{M}} = \{X \times Y \mid X, Y \in \mathsf{P}(M)\}$ then $\mathtt{Q}xy.\,\varphi(\vec{w}, x, y)$ expresses the fact that whenever $\varphi(\vec{a}, b_1, c_1)$ and $\varphi(\vec{a}, b_2, c_2)$ hold in $\mathcal{M}$ then $\varphi(\vec{a}, b_1, c_2)$ and $\varphi(\vec{a}, b_2, c_1)$ hold as well.

One can adapt the above concepts to monadic second-order logic (MSO), with $\mathtt{Q}^{\mathcal{M}} \subseteq \mathsf{P}(\mathsf{P}(M))$ in the unary case, and in general $\mathtt{Q}^{\mathcal{M}} \subseteq \mathsf{P}\left((\mathsf{P}(M))^n\right)$. Indeed, several generalised quantifiers of this kind have been considered in the literature, the eminent example being the *weak* quantifiers, that is, the quantifiers $\exists$ and $\forall$ restricted to *finite* sets. The cardinality quantifiers and unboundedness quantifiers mentioned in the introduction can also be presented in this framework.

Game quantifiers. We begin by discussing how game quantifiers introduced above can be viewed as generalised MSO quantifiers over the structure $\mathbb{N}$. To explain the idea, let us first take a simple example in first-order logic. Consider a formula

$$\forall x.\,\exists y.\,\forall x'.\,\exists y'.\,\varphi(\vec{w}, x, y, x', y').$$

Clearly, its meaning in a structure $\mathcal{M}$ can be viewed as a game of two players, say $\exists$ and $\forall$, consisting of 4 rounds. Now the block of 4 quantifiers can be replaced by a single 4-ary quantifier, so that the formula becomes $\mathtt{Q}xyx'y'.\,\varphi(\vec{w}, x, y, x', y')$. The semantics of $\mathtt{Q}$ is specified by a property that a 4-ary relation r in $\mathtt{Q}^{\mathcal{M}}$ should possess. In terms of a game, in which Players I and II select in alternation elements of M, Player II should have a strategy to force the selected quadruple into r.

Now consider a formula $\varphi(W, X, Y)$ interpreted in the structure $\mathbb{N}$, where W, X, Y are set variables (more generally, they could be some tuples of set variables). Consider an infinite game, in which Players I and II select in alternation bits in $\{0, 1\}$, so that the result is an infinite sequence

$$x_0, y_0, x_1, y_1, x_2, y_2, \ldots, x_n, y_n, \ldots$$

The sequences $x_0, x_1, x_2, \ldots$ and $y_0, y_1, y_2, \ldots$ constitute characteristic functions of some subsets $\bar{x}$ and $\bar{y}$ of $\mathbb{N}$, respectively. Now, for a valuation $W \mapsto \bar{w}$, a formula defined with the game quantifier

$$\mathfrak{D}X\text{-}Y.\, \varphi(\bar{w}, X, Y)$$

holds if Player II has a strategy to force that the formula $\varphi(\bar{w}, \bar{x}, \bar{y})$ holds in $\mathbb{N}$. The game quantifier $\mathfrak{D}$ can be defined as a *binary* generalised MSO quantifier. Its semantics $\mathfrak{D}^{\mathbb{N}}$ is defined by a family of binary relations over $\mathsf{P}(\mathbb{N})$ that comprises *all* relations $R \subseteq \mathsf{P}(\mathbb{N}) \times \mathsf{P}(\mathbb{N})$, such that in the game described above, Player II has a strategy to force the resulting pair $(\bar{x}, \bar{y})$ into R. Then, indeed, the formula $\mathfrak{D}X\text{-}Y.\, \varphi(\bar{w}, X, Y)$ holds precisely when the relation $\{(\bar{x}, \bar{y}) \mid \varphi(\bar{w}, \bar{x}, \bar{y}) \text{ holds in } \mathbb{N}\}$ belongs to $\mathfrak{D}^{\mathbb{N}}$.

Index quantifiers. To present our new index quantifier $\mathtt{I}^{D}_{\mathcal{R}}X.\, \varphi(\vec{W}, X)$ as a generalised MSO quantifier, let us, for concreteness, focus on the MSO theory of the full binary tree, whose domain is $\{\mathtt{L}, \mathtt{R}\}^*$. As we have assumed that our automaton reads the values of both $\vec{W}$ and X, the construction does not fit into the unary case, but, like the game quantifier, it can be expressed as a *binary* quantifier, or more generally, $(k{+}\ell)$-ary quantifier (if $\vec{W}$ is a k-vector and X an ℓ-vector).

For simplicity, let us consider $k = \ell = 1$; an extension to higher k, ℓ is straightforward. The key point is to choose a class of binary relations over $\mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)$ that would serve as the intended semantics of the quantifier. For a binary relation $r \subseteq \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*) \times \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)$, and a set $K \in \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)$, we define the *cut* of r by K as the binary relation

$$r_K = r \cap \big(\{K\} \times \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)\big) = \big\{(K, L) \in r \mid L \in \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)\big\}.$$

Recall that in our quantifier we are interested in automata of type D and index $\mathcal{R}$. A pair of sets (K, L) is accepted by an automaton (over the alphabet $\{0, 1\}^2$) if so is its characteristic function, and a relation $r \subseteq \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*) \times \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*)$ is recognised by an automaton if it consists precisely of pairs that the automaton accepts. Now consider the class of relations

$$\mathcal{C}^{D}_{\mathcal{R}} = \big\{r_K \mid K \in \mathsf{P}(\{\mathtt{L}, \mathtt{R}\}^*) \ \wedge\ r \text{ is recognised by an automaton of type } D \text{ and index } \mathcal{R}\big\}.$$

Then it is straightforward to see that the formula $\mathtt{I}^{D}_{\mathcal{R}}X.\, \varphi(W, X)$ is equivalent to

$$\mathtt{Q}^{D}_{\mathcal{R}}ZX.\, \varphi(Z, X) \ \wedge\ Z = W,$$

where the semantics of the quantifier $\mathtt{Q}^{D}_{\mathcal{R}}$ over trees is given by the class $\mathcal{C}^{D}_{\mathcal{R}}$.

Clearly, the variable Z above plays only a technical role; therefore, for clarity of notation, in our paper we use the notation $\mathtt{Q}^{D}_{\mathcal{R}}X.\, \varphi(W, X)$, without Z.

Let us also remark that our proposal is not the only possible approach. One could also consider a unary quantifier $\mathtt{Q}^{D}_{\mathcal{R}}X$, where a formula $\mathtt{Q}^{D}_{\mathcal{R}}X.\, \varphi(W, X)$ holds for a valuation $W \mapsto \tilde{w}$ if the language of all sets $\tilde{x}$ such that $\varphi(\tilde{w}, \tilde{x})$ holds is accepted by an automaton (of appropriate kind), without reading the parameter $\tilde{w}$, as discussed in Remark 3.1. That is, the semantics is given simply by a class of all languages accepted by automata of type D and index $\mathcal{R}$.

While this may appear quite natural, we believe that such an extension would be less interesting. Not only it brings an undecidable formalism over ω-words as indicated in Remark 3.1 but it additionally restricts available correlation between the involved variables. Indeed, if such a formula is satisfied by some $\tilde{w}$ which is not regular, then it follows from general properties of MSO (namely Regular Tree Theorem) that there is a regular $\tilde{w}'$, such that the languages $\{\tilde{x} \mid \varphi(\tilde{w}, \tilde{x})\}$ and $\{\tilde{x} \mid \varphi(\tilde{w}', \tilde{x})\}$ coincide. Thus the relation defined by the formula $\varphi(W, X)$, in some sense, necessarily weakly correlates its arguments. These issues require further investigation.

5 Game quantifiers over ω-words

The first part of our results concerns the *game quantifier* $\Game$. We start by showing that the extended formalism of MSO+$\Game$ can be reduced back to pure MSO, that is, the game quantifiers can be eliminated. However, our goal is to obtain a stronger property, stated in Theorem 5.12: under appropriate assumptions on the formula, games described by quantifiers $\Game$ admit strategies that can be realised by finite-memory transducers.

Consider an instance of a game quantifier $\Game X{\to}Y.\, \varphi(W, X, Y)$, where the internal formula $\varphi(W, X, Y)$ is in MSO.

▶ **Lemma 5.1** (Folklore). *For every formula of the form $\Game X{\to}Y.\, \varphi(W, X, Y)$, where φ is in MSO, one can effectively construct an equivalent formula of pure MSO.*

This construction can be found in a work by Kaiser [19]. We include a proof for the sake of completeness. The concepts introduced in this proof will be useful later on in the paper.

Proof. Let $\mathcal{D}$ be a deterministic parity automaton over the alphabet $A_W \times A_X \times A_Y$ of a strong parity index $\mathcal{R} = \langle \{i, i{+}1, \ldots, j\}, P_{i,j}\rangle$ that is equivalent to φ, that is, the automaton accepts an ω-word $\langle \bar{w}, \bar{x}, \bar{y}\rangle$ if and only if $\varphi(\bar{w}, \bar{x}, \bar{y})$ holds.

Given an ω-word $\bar{w} = w_0 w_1 w_2 \cdots \in (A_W)^\omega$, we can consider a parity game $\mathcal{G}(\bar{w}, \mathcal{D})$ obtained as a product of $\mathcal{G}(\bar{w}, \varphi)$ with the automaton $\mathcal{D}$, defined as follows.

▶ **Definition 5.2.** *Let Q be the set of states of $\mathcal{D}$, and δ its transition function. The set of positions of $\mathcal{G}(\bar{w}, \mathcal{D})$ is then given by $V^{(\mathrm{I})} \stackrel{def}{=} \mathbb{N} \times Q$ and $V^{(\mathrm{II})} \stackrel{def}{=} \mathbb{N} \times Q \times A_X$. From a position $(n, q) \in \mathbb{N} \times Q$ first Player* I *proposes $x_n \in A_X$ and the game moves to the position (n, q, x_n). Then Player* II *proposes $y_n \in A_Y$ and the game moves to the position $(n{+}1, q')$ where $\delta(q, (w_n, x_n, y_n)) = (k_n, q')$. The label of the former edge equals the lowest priority i (i.e., is irrelevant), while the label of the latter edge equals k_n.*

It is easy to see that $\mathcal{G}(\bar{w}, \mathcal{D})$ is equivalent to $\mathcal{G}(\bar{w}, \varphi)$ in the sense that a player P wins one game if and only if she wins another: the automaton $\mathcal{D}$ is deterministic, so there is a one-to-one correspondence between choices in $\mathcal{G}(\bar{w}, \varphi)$ and choices in $\mathcal{G}(\bar{w}, \mathcal{D})$, so that strategies from one game can be directly transferred to the other game. Moreover, due to positional determinacy of parity games (see Theorem 2.5), Player II wins $\mathcal{G}(\bar{w}, \varphi)$ if and only if Player II has a positional winning strategy in $\mathcal{G}(\bar{w}, \mathcal{D})$.

A positional strategy $\sigma^{(\mathrm{II})}$ of Player II in $\mathcal{G}(\bar{w}, \mathcal{D})$ can be represented by an ω-word $\bar{\sigma} = \sigma_0 \sigma_1 \sigma_2 \cdots \in \big(Q {\times} A_X \to A_Y\big)^\omega$, where Q is the set of states of $\mathcal{D}$: in this ω-word, the letter σ_n satisfies $\sigma_n(q, x) = y$ where $\sigma^{(\mathrm{II})}(n, q, x) = \big((n, q, x), k, (n{+}1, q')\big)$ with $\delta_{\mathcal{D}}\big(q, (w_n, x, y)\big) = (k, q')$. The following claim is straightforward, as MSO allows us to quantify over infinite plays in $\mathcal{G}(\bar{w}, \mathcal{D})$ and can express the parity condition $\mathcal{P}_{i,j}$.

▷ Claim 5.3. There exists an MSO formula $\psi^{(\mathrm{II})}(W, \Sigma)$ such that $\psi^{(\mathrm{II})}(\bar{w}, \bar{\sigma})$ holds for $\bar{w} \in (A_W)^\omega$ and $\bar{\sigma} \in \big(Q \times A_X \to A_Y\big)^\omega$ if and only if $\bar{\sigma}$ encodes a positional winning strategy $\sigma^{(\mathrm{II})}$ of Player II in $\mathcal{G}(\bar{w}, \mathcal{D})$.

It follows that the formula $\eth X{\to}Y.\,\varphi(W,X,Y)$ is equivalent to

$$\exists\Sigma \in \big(Q \times A_X \to A_Y\big)^\omega.\, \psi^{(\mathrm{II})}(W,\Sigma),$$

where the set $\big(Q \times A_X \to A_Y\big)$ is finite and therefore one can treat it as an alphabet. Consequently, this formula belongs to pure MSO. ◀

Using the above lemma to inductively eliminate an innermost game quantifier, we immediately obtain the following corollary.

▶ **Corollary 5.4.** *The expressive power of MSO+$\eth$ is equal to that of MSO. Moreover, there exists an effective procedure that eliminates the game quantifiers.*

▶ Remark 5.5. Consider formulae without the parameter W, that is, $\eth X{\to}Y.\,\varphi(X,Y)$, where φ is in MSO. In this case the game $\mathcal{G}(\bar{w},\mathcal{D})$ can be played over the arena Q instead of $\mathbb{N}\times Q$. Thus, it is a finite parity game, which can be solved directly. The resulting strategy $\sigma^{(\mathrm{II})}$ takes the shape of a transducer $\tau\colon A_X \twoheadrightarrow A_Y$ (its set of states is just Q) such that for every $\bar{x} \in (A_X)^\omega$ we have $\varphi\big(\bar{x},\tau(\bar{x})\big)$.

The above remark can be seen as a modern version of a proof of the Büchi-Landweber theorem [11], based on determinacy of parity games. This means that the proposed procedure of elimination of a game quantifier can be seen as a parametrised version of the construction of Büchi and Landweber, where we search for a strategy that may depend on the parameter $\bar{w} \in (A_W)^\omega$.

5.1 Sequential strategies

One may ask if it is possible to recover some version of Remark 5.5 in the presence of external parameters W, namely represent the strategy $\sigma^{(\mathrm{II})}$ as a transducer. Of course the exact strategy may depend on the global properties of W, so one cannot expect to have a single transducer $\tau\colon A_W \times A_X \twoheadrightarrow A_Y$ that would realise the strategy. However, what happens if we allow the transducer to depend on a given ω-word $\bar{w}$?

▶ **Question 5.6.** *Assume that for some parameter $\bar{w} \in (A_W)^\omega$ a formula $\eth X{\to}Y.\,\varphi(\bar{w},X,Y)$ holds. Does it mean that there exists a transducer $\tau_{\bar{w}}\colon A_W \times A_X \twoheadrightarrow A_Y$ that realises a winning strategy of Player* II *in $\mathcal{G}(\bar{w},\varphi)$? In other words, we ask if we can ensure that*

$$\textit{for every } \bar{x} \in (A_X)^\omega \textit{ we have } \varphi\big(\bar{w},\bar{x},\tau_{\bar{w}}(\bar{w},\bar{x})\big). \tag{5.1}$$

It turns out that the answer is negative – the strategies used by Player II may not be made finite-memory, even if $\bar{w}$ is known in advance. Intuitively, this boils down to the fact that $\bar{w}$ may not be ultimately periodic, while φ may require some position-to-position correspondence between Y and W. More precisely, we have the following fact.

▶ **Fact 5.7.** *There exists a formula $\varphi(W,X,Y)$ in MSO such that for some concrete ω-word $\bar{w} \in (A_W)^\omega$ we have $\eth X{\to}Y.\,\varphi(\bar{w},X,Y)$ while no transducer $\tau\colon (A_W \times A_X) \twoheadrightarrow A_Y$ satisfies Formula* (5.1)*.*

Proof. Let $A_W = A_Y = \{0,1\}$ and let $\varphi(\bar{w},\bar{x},\bar{y})$ for $\bar{w} = w_0w_1w_2\cdots$ and $\bar{y} = y_0y_1y_2\cdots$ say that for every $n \in \mathbb{N}$ we have $y_n = w_{n+1}$. Notice that for all $\bar{w} \in (A_W)^\omega$ we have $\eth X{\to}Y.\,\varphi(\bar{w},X,Y)$ because X plays no role in φ and it is enough for Player II to play consecutive values $y_0 \stackrel{\text{def}}{=} w_1$, $y_1 \stackrel{\text{def}}{=} w_2$, and so on.

Let $\bar{w}$ be defined as $0^1 10^2 10^3 1 \cdots \in (A_W)^\omega$. It remains to show that no transducer $\tau_{\bar{w}} \colon (A_W \times A_X) \twoheadrightarrow A_Y$ satisfies Formula (5.1). Assume to the contrary that $\tau_{\bar{w}}$ is such a transducer with a set of states Q and a transition function δ. Fix any letter $x \in A_X$, and consider the unique run of τ over the ω-word $\bar{w}$ defined above and over $\bar{x} = xxx \cdots \in (A_X)^\omega$. Take any $n \geq |Q|$, and concentrate on the fragment of this run reading the infix $10^{n+1}1$ of $\bar{w}$. The transducer should produce 0's while reading the first n zeroes of the input fragment (because the next input letter is 0), and 1 over the last zero (because the next input letter is 1). Let $\rho_0, \rho_1, \ldots, \rho_{n+1}$ be the states of τ visited over this fragment, with ρ_0 before the first 0, and ρ_{n+1} after the last 0. By the pigeonhole principle, we have $\rho_k = \rho_\ell$ for some k, ℓ with $0 \leq k < \ell \leq n$. For $i \in \{0, \ldots, n-1\}$ we have $\delta(\rho_i, (0, x)) = (0, \rho_{i+1})$, which applied to consecutive positions after k and ℓ implies $\rho_{k'} = \rho_n$, where $k' = k + (n - \ell) < n$. But then $(0, \rho_{k'+1}) = \delta(\rho_{k'}, (0, x)) = \delta(\rho_n, (0, x)) = (1, \rho_{n+1})$. In other words, the transducer has no way of counting where to produce a 1, if the number of zeroes exceeds the number of its states. ◀

This negative answer can be explained from two perspectives. One, directly suggested by the above example, focuses on the need of a lookahead – if $\tau_{\bar{w}}$ was able to perform some lookahead to the future of the parameter word $\bar{w}$, then it could easily realise the respective strategy. This observation is formalised in Lemma 5.9, where the lookahead is allowed. This approach follows similar lines as the results of Winter and Zimmermann [51], where the authors study games with lookahead.

Another point of view is that in contrast to the construction by Büchi and Landweber [11] (see also [9]), the arena of $\partial(\bar{w}, \mathcal{D})$ is infinite. Thus, some subtle synchronisation between the variables may go on indefinitely. To avoid this problem, we consider the notion of a formula that *depends separately* on one variable (see Section 5.4). It turns out that in this case the infiniteness of the arena stops being a problem and the strategies can again be realised by transducers, as stated in Theorem 5.12.

5.2 Uniformisation by transducers with lookahead

Before we move on, we need to first show how *uniformised* relations can be realised by transducers with lookahead. We say that an MSO formula $\psi(W, Y)$ is *uniformised* if for every $\bar{w}$ there exists at most one $\bar{y}$ such that $\psi(\bar{w}, \bar{y})$ holds. The next fact states that a partial function described by a uniformised MSO-formula can be realised by a transducer composed with a lookahead. This fact is rather general and almost folklore; it relies on the composition method for MSO [43] (expressed by Wilke algebras in our setup).

▶ **Fact 5.8.** *Assume that $\psi(W, Y)$ is uniformised. Then, one can effectively construct a homomorphism $\alpha \colon (A_W)^{\mathsf{W}} \to S$ onto a finite Wilke algebra S together with a transducer $\tau \colon A_W \twoheadrightarrow (S^{\inf} \to A_Y)$ such that for every $\bar{w} \in (A_W)^\omega$ for which $\exists Y.\, \psi(\bar{w}, Y)$ holds we have*

$$\psi\big(\bar{w}, \tau(\bar{w}) \bullet \mathsf{lk}_\alpha(\bar{w})\big).$$

In other words, for an input ω-word $\bar{w} = w_0 w_1 w_2 \cdots$ and $f_0 f_1 f_2 \cdots \stackrel{\text{def}}{=} \tau(\bar{w})$ we consider $h_n \stackrel{\text{def}}{=} \alpha(w_{n+1} w_{n+2} \cdots)$ and $y_n \stackrel{\text{def}}{=} f_n(h_n)$ defined for $n = 0, 1, 2, \ldots$, and claim that $\psi(\bar{w}, y_0 y_1 y_2 \cdots)$ holds.

Proof. For $n \in \mathbb{N}$ let $\chi_n \in \{0,1\}^\omega$ denote the ω-word having 1 at the position n, and zeroes everywhere else. For each $y \in A_Y$ consider a formula $\psi_y(W, Z)$ such that, assuming $\exists Y.\, \psi(\bar{w}, Y)$, we have $\psi_y(\bar{w}, \chi_n)$ if the letter at position n of the unique $\bar{y}$ such that $\psi(\bar{w}, \bar{y})$ holds equals y; such a formula can be easily constructed out of ψ.

Apply Theorem 2.4 to the tuple of languages defined by formulae $\big(\psi_y(W,Z)\big)_{y\in A_Y}$ to obtain a homomorphism $\beta\colon (A_W\times\{0,1\})^{\mathsf{W}}\to S$ onto a finite Wilke algebra S together with a tuple of sets $(F_y)_{y\in A_Y}$ such that $\psi_y(\bar{w},\bar{z})$ holds if and only if $\beta\big(\langle\bar{w},\bar{z}\rangle\big)\in F_y$.

Let $\mathsf{add}_0\colon (A_W)^{\mathsf{W}}\to(A_W\times\{0,1\})^{\mathsf{W}}$ be the homomorphism adding 0 on the second coordinate of all letters in a given word. Then as $\alpha\colon (A_W)^{\mathsf{W}}\to S$ we take $\mathsf{add}_0\circ\beta$.

Next, we construct the transducer $\tau\colon A_W\twoheadrightarrow(S^{\mathrm{inf}}\to A_Y)$. It remembers the value under α of the prefix read so far. To this end, its set of states is the monoid $S^{\mathrm{fin}}_{+\varepsilon}$ obtained from S^{fin} by adding a formal neutral element ε. The initial state is ε. For $v\in S^{\mathrm{fin}}_{+\varepsilon}$ and $w\in A_W$ let

$$\delta\big(v,w\big)\stackrel{\mathrm{def}}{=}\big(f,v\cdot\alpha(w)\big),$$

where $f\colon S^{\mathrm{inf}}\to A_Y$ is defined for every $h\in S^{\mathrm{inf}}$ as follows: $f(h)$ is any fixed letter $y\in A_Y$ such that $v\cdot\beta(w,1)\cdot h\in F_y$, or just any element of A_Y if $v\cdot\beta(w,1)\cdot h\notin F_y$ for all $y\in A_Y$ (morally, one should think that there is a unique such y; however strictly speaking this needs not to be true, which is caused by words $\bar{w}$ for which $\exists Y.\,\psi(\bar{w},Y)$ does not hold).

Fix now an input ω-word $\bar{w}=w_0w_1w_2\cdots\in(A_W)^\omega$ such that $\exists Y.\,\psi(\bar{w},Y)$ holds. After reading a prefix $w_0w_1\cdots w_{n-1}$, the state of τ is $\alpha(w_0w_1\cdots w_{n-1})$ (or just ε if $n=0$). It follows that the n-th letter of $\tau(\bar{w})\bullet\mathsf{lk}_\alpha(\bar{w})$ is a letter y that satisfies

$$\alpha(w_0w_1\cdots w_{n-1})\cdot\beta(w_n,1)\cdot\alpha(w_{n+1}w_{n+2}w_{n+3}\cdots)\in F_y.$$

This is the case precisely when $\psi_y(\bar{w},\chi_n)$ holds, and because ψ is uniformised, this holds for precisely one y, which is the letter at position n in the unique $\bar{y}$ such that $\psi(\bar{w},\bar{y})$ holds. We thus obtain that $\psi\big(\bar{w},\tau(\bar{w})\bullet\mathsf{lk}_\alpha(\bar{w})\big)$ holds, as required. ◀

5.3 Allow lookahead

Using Fact 5.8 we now show that a winning strategy of Player II for a game quantifier can be realised by a transducer composed with a lookahead.

▶ **Lemma 5.9.** *Given a formula $\varphi(W,X,Y)$, one can effectively construct a homomorphism $\alpha\colon (A_W)^{\mathsf{W}}\to S$ onto a finite Wilke algebra S together with a transducer $\tau\colon (A_W\times A_X)\twoheadrightarrow(S^{\mathrm{inf}}\to A_Y)$ such that for every ω-word $\bar{w}\in(A_W)^\omega$ satisfying $\mathfrak{D}X{\to}Y.\,\varphi(\bar{w},X,Y)$, and for every $\bar{x}\in(A_X)^\omega$ we have*

$$\varphi\big(\bar{w},\bar{x},\tau(\bar{w},\bar{x})\bullet\mathsf{lk}_\alpha(\bar{w})\big).$$

In other words, if for every $n\in\mathbb{N}$ as $f_n\in(S^{\mathrm{inf}}\to A_Y)$ we take the output letter produced by τ after reading the prefixes $w_0w_1\ldots w_n$ of $\bar{w}$ and $x_0x_1\ldots x_n$ of $\bar{x}$ (so that $f_0f_1f_2\cdots=\tau(\bar{w},\bar{x})$), and we consider $h_n\stackrel{\mathrm{def}}{=}\alpha(w_{n+1}w_{n+2}w_{n+3}\ldots)$ and $y_n\stackrel{\mathrm{def}}{=}f_n(h_n)$, then $\varphi(\bar{w},\bar{x},y_0y_1y_2\ldots)$ holds. Intuitively, the above lemma says that one can construct the resulting ω-word $\bar{y}$ by a transducer, assuming that we allow a lookahead over the whole ω-word $\bar{w}$ (note that there is no lookahead over $\bar{x}$: moves of Player II cannot be allowed to depend on future moves of Player I).

This lemma is essentially a composition of Fact 5.8 with the following lemma. The only technical difficulty lies in the fact that the lookahead is given after the transducer has read the whole input ω-word.

▶ **Lemma 5.10** ([24, 42, 44]). *For every MSO formula $\psi(W,Y)$ one can construct a uniformised formula $\psi_{\mathsf{u}}(W,Y)$ such that*
- *for all ω-words $\bar{w}$ we have $\big(\exists Y.\, \psi(\bar{w},Y)\big) \Rightarrow \big(\exists Y.\, \psi_{\mathsf{u}}(\bar{w},Y)\big)$, and*
- *for all ω-words $\bar{w}$, $\bar{y}$ we have $\psi(\bar{w},\bar{y}) \Leftarrow \psi_{\mathsf{u}}(\bar{w},\bar{y})$.*

It may be worth mentioning that the original proof of the above lemma as given by Lifsches and Shelah [24, Theorem 6.3] says "By [1].", where "[1]" is the work of Büchi and Landweber on synthesis [11]. This is incorrect, because Büchi and Landweber show how to win games using finite-state strategies, while a uniformisation may in general depend on the future. More precisely, it is always possible to uniformise a formula $\psi(X,Y)$ even if $\mathfrak{D}X{\rightarrow}Y.\, \psi(X,Y)$ does not hold (imagine $\psi(X,Y)$ saying that the first letter of Y is 1 if and only if infinitely many letters of X are 1). However, the mistake made by Lifsches and Shelah may not be a coincidence: Lemma 5.9 shows that uniformisation is indeed possible using transducers with lookahead, while Theorem 5.12 proved later shows how to eliminate the lookahead when $\psi(W,X,Y)$ depends separately on the involved variable Y (see Definition 5.11).

5.4 Separate coordinates

Fact 5.7 tells us that in general the lookahead lk_α in Lemma 5.9 is necessary when we want to realise a winning strategy by a transducer. The example from Fact 5.7 needed a lookahead only to check the letter on the next position of the parameter $\bar{w}$. One can imagine another example: in order to win, Player II should output y_n that equals the first letter in $\{a,b\}$ among $w_{n+1}, w_{n+2}, w_{n+3}, \dots$ (skipping all letters c before it). Here the future interval checked by the lookahead needs to be unbounded, but still finite. On the other hand, the lookahead never needs to check "the whole infinite future" of $\bar{w}$, since $\bar{w}$ is known in advance. To see the intuitions for this, assume that the value of y_n needed to win depends on whether letter a belongs to the set $\{w_{n+1}, w_{n+2}, w_{n+3}, \dots\}$. Here a winning strategy seems to depend on the whole future of $\bar{w}$, but since we know $\bar{w}$ in advance, we may avoid this: either $\bar{w}$ contains infinitely many a (and then there is always some a in the future), or the last a occurs on some position n (and then the transducer may count to the fixed number n). This suggests that it should be possible to create a transducer which does not use a lookahead, but produces letters of $\bar{y}$ with some delay, needed to check future properties of $\bar{w}$ (formally, this is impossible to realise, because there is also an issue of synchronisation between $\bar{y}$ and $\bar{x}$).

The main result of this section states that we may avoid the aforementioned need for a lookahead (or a delay) once we disallow φ to enforce any position-to-position correspondence between $\bar{w}$ and $\bar{y}$. More precisely, we introduce the following definition.

▶ **Definition 5.11.** *We say that a formula $\varphi(W,X,Y)$ depends* separately on Y *if it is a finite Boolean combination of formulae $\psi_i(W,X)$ and formulae $\gamma_i(Y)$.*

▶ **Theorem 5.12.** *Assume that $\varphi(W,X,Y)$ depends separately on Y and that $\bar{w} \in (A_W)^\omega$ is such that $\mathfrak{D}X{\rightarrow}Y.\, \varphi(\bar{w},X,Y)$ holds. Then, there exists a transducer $\tau_{\bar{w}}\colon A_W \times A_X \twoheadrightarrow A_Y$ such that Formula* (5.1) *holds, that is, for every $\bar{x} \in (A_X)^\omega$ we have $\varphi\big(\bar{w},\bar{x},\tau_{\bar{w}}(\bar{w},\bar{x})\big)$.*

It is worth mentioning that if $\varphi(W,X,Y)$ is a Boolean combination of formulae $\psi_i(W,X)$ and formulae $\gamma_i(X,Y)$ then one can still recover the example from Fact 5.7 by writing that either $(\bar{x} \neq \bar{w})$ or $y_n = x_{n+1}$ for all $n \in \mathbb{N}$, which is of the required shape and still no transducer can realise the strategy.

Since each transducer induces a strategy, Theorem 5.12 in fact provides an equivalence.

▶ **Corollary 5.13.** *Take $\varphi(W,X,Y)$ that depends separately on Y and any $\bar{w} \in (A_W)^\omega$. Then $\eth X{\to}Y.\,\varphi(\bar{w},X,Y)$ holds if and only if there exists a transducer $\tau_{\bar{w}}\colon A_W \times A_X \twoheadrightarrow A_Y$ such that for every $\bar{x} \in (A_X)^\omega$ we have $\varphi\big(\bar{w},\bar{x},\tau_{\bar{w}}(\bar{w},\bar{x})\big)$.*

Proof of Theorem 5.12 (sketch). We begin by applying Lemma 5.9 to construct a transducer which constructs the desired winning strategy. The transducer uses lookahead given by a homomorphism $\alpha\colon (A_W)^{\mathsf{W}} \to S$ into some finite Wilke algebra S. We fix the parameter $\bar{w}$ and apply Ramsey's theorem (cf. Theorem 2.2) to split the word $\bar{w}$ into a finite prefix and then infinitely many subwords whose image under α is the same idempotent $e \in S^{\mathrm{fin}}$. In all the split points the transducer can be sure that the lookahead (i.e., the value of the suffix under α) is $\alpha(e^\omega)$. But how can the transducer detect the split points? The first one can be hardcoded in the transducer. Then, knowing some split point, the transducer has to find a next one. A brave conjecture would be that every subword evaluating to e moves us from one splitting point to a next one; but this is false (maybe we should split after a different subword evaluating to e). However, a slightly stronger condition is sufficient: if the transducer encounters two consecutive subwords evaluating to e, then it can be sure that the position after the first of them can be chosen (that is, the suffix after this position can be split into infinitely many subwords evaluating to e). We remark that a similar technical trick occurs in a work of Thomas [46].

These infinitely many splitting points detected by the transducer (with some delay) are positions where we know the value of the lookahead, and thus we can produce at those points the fragments of an actual output ω-word $\bar{y}$. Since these positions are scattered in an arbitrary way, we need to be able to *pad* the output in-between these positions. This is where the assumption of separate dependency on Y comes into play: the satisfaction of $\varphi(\bar{w},\bar{x},\bar{y})$ depends separately on $\langle \bar{w},\bar{x}\rangle$ and on $\beta(\bar{y})$ for an appropriately chosen homomorphism $\beta\colon (A_Y)^{\mathsf{W}} \to T$ onto another finite Wilke algebra T. Now, using some standard techniques involving idempotents, we can ensure that we pad the output ω-word in such a way that there are no delays in its generation (i.e., we really produce some letter from A_Y in each step), while still we control the final value of $\beta(\bar{y})$, making sure that $\varphi(\bar{w},\bar{x},\bar{y})$ holds. ◀

6 Index quantifiers over ω-words

We now use the previous results to show quantifier elimination procedure for index quantifiers.

▶ **Theorem 6.1.** *The logic MSO+$\mathtt{I}$ effectively reduces to the pure MSO over ω-words.*

First, one can observe that the non-deterministic index quantifiers $\mathtt{I}^{\mathsf{nd}}_{\mathcal{R}}$ are either trivial or equivalent to the deterministic ones $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}'}$, with an appropriate change of indices. More precisely, the following equivalences hold:

- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{R}}X.\,\varphi(W,X)$ is equivalent to $\forall X.\,\varphi(W,X)$ whenever $\mathcal{R}$ is either $\mathcal{P}_{0,0}$ or $\mathcal{W}_{0,0}$, because automata of these indices accept all ω-words.
- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{R}}X.\,\varphi(W,X)$ is equivalent to $\forall X.\,\neg\varphi(W,X)$ whenever $\mathcal{R}$ is either $\mathcal{P}_{1,1}$ or $\mathcal{W}_{1,1}$, because automata of these indices recognise empty languages.
- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{P}_{1,2}}X.\,\varphi(W,X)$ is always true, because non-deterministic Büchi automata recognise all regular languages of ω-words [10]. The same holds for $\mathcal{P}_{i,j}$ with $i \in \{0,1\}$ and $j \geq 2$.
- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{P}_{0,1}}X.\,\varphi(W,X)$ is equivalent to $\mathtt{I}^{\mathsf{dt}}_{\mathcal{P}_{0,1}}X.\,\varphi(W,X)$, because non-deterministic co-Büchi automata have the same expressive power as deterministic co-Büchi automata [30].
- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{R}}X.\,\varphi(W,X)$ is equivalent to $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}}X.\,\varphi(W,X)$ whenever $\mathcal{R}$ is either $\mathcal{W}_{0,1}$ (safety), $\mathcal{W}_{1,2}$ (reachability), or $\mathcal{W}_{0,2}$, again because of the ability to determinise these automata without change of index (simple powerset-like constructions suffice).

- $\mathtt{I}^{\mathsf{nd}}_{\mathcal{W}_{1,3}} X.\, \varphi(W,X)$ is equivalent to $\mathtt{I}^{\mathsf{dt}}_{\mathcal{W}_{0,1}} X.\, \varphi(W,X)$, because non-deterministic $\mathcal{W}_{1,3}$ automata have the same expressive power as deterministic co-Büchi automata. The same holds for all $\mathcal{W}_{i,j}$ with $i \in \{0,1\}$ and $j \geq 3$.

This covers all cases, because we can always shift the indices so that $i \in \{0,1\}$.

Thus, for the rest of this section we focus on the deterministic index quantifiers $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}}$. Similarly as in Corollary 5.4 we proceed inductively, that is, we eliminate index quantifiers starting from inside. Let $\mathcal{R} = \langle A_{\mathcal{R}}, L_{\mathcal{R}} \rangle$ be an index (either $\mathcal{R} = \mathcal{P}_{i,j}$, or $\mathcal{R} = \mathcal{W}_{i,j}$ for $i \leq j$). Consider a formula $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}} X.\, \varphi(W,X)$ with $\varphi(W,X)$ in MSO. Our goal is to construct a formula of pure MSO that is equivalent to $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}} X.\, \varphi(W,X)$. As in the previous section, we consider here only the case of a single parameter W (which can encode multiple parameters using a product alphabet).

Consider $\varphi_{\mathcal{R}}(W,X,K)$ which, for $\bar{w} \in (A_W)^\omega$, $\bar{x} \in (A_X)^\omega$, and $\bar{k} \in (A_{\mathcal{R}})^\omega$, says that $\bar{k} \in L_{\mathcal{R}}$ if and only if $\varphi(\bar{w},\bar{x})$ holds. Note that $\varphi_{\mathcal{R}}(W,X,K)$ depends separately on the variable K.

▶ **Lemma 6.2.** *The formulae $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}} X.\, \varphi(W,X)$ and $\mathfrak{D} X{\to}K.\, \varphi_{\mathcal{R}}(W,X,K)$ are equivalent.*

Note that this lemma concludes the proof of Theorem 6.1 because the game quantifier $\mathfrak{D}$ involved in the latter formula can be effectively eliminated due to Lemma 5.1.

Proof. Take any parameter $\bar{w} \in (A_W)^\omega$. The following conditions are equivalent:

- $\mathtt{I}^{\mathsf{dt}}_{\mathcal{R}} X.\, \varphi(\bar{w},X)$ holds;
- there exists a deterministic automaton $\mathcal{D}_{\bar{w}}$ of index $\mathcal{R}$ such that for every ω-word $\bar{x} \in (A_X)^\omega$ we have $\varphi(\bar{w},\bar{x})$ if and only if $(\bar{w},\bar{x}) \in \mathrm{L}(\mathcal{D}_{\bar{w}})$;
- there exists a transducer $\tau_{\bar{w}} \colon (A_W \times A_X) \twoheadrightarrow A_{\mathcal{R}}$ such that for every ω-word $\bar{x} \in (A_X)^\omega$ we have $\varphi(\bar{w},\bar{x})$ if and only if $\tau_{\bar{w}}(\bar{w},\bar{x}) \in L_{\mathcal{R}}$;
- $\mathfrak{D} X{\to}K.\, \varphi_{\mathcal{R}}(\bar{w},X,K)$ holds.

The first two items are equivalent from the definition of the index quantifier. The second two items are equivalent due to Remark 2.1. The last two items are equivalent by the choice of $\varphi_{\mathcal{R}}$ and Corollary 5.13. ◀

7 Index quantifiers over trees

In this section we prove the following theorem.

▶ **Theorem 7.1.** *The theory of MSO+*$\mathtt{I}$ *over trees is undecidable. This holds even if we allow only the simplest possible index, namely the weak parity index $\mathcal{W}_{0,1}$ (i.e., safety), and any type of determinism $D \in \{\mathsf{dt}, \mathsf{nd}\}$, thus we use only the index quantifier $\mathtt{I}^{D}_{\mathcal{W}_{0,1}}$.*

To simplify the notations, in this section we consider only variables over the alphabet $\{0,1\}$; thus their valuations can be seen as sets, rather than functions from tree nodes to $\{0,1\}$. Thus, we can write $\tilde{x} \cup \tilde{y}$, $\tilde{x} \subseteq \tilde{y}$, etc.

A set $I \subseteq \{\mathtt{L},\mathtt{R}\}^*$ of tree nodes is called an *interval* if it is of the form $\{u\mathtt{R}^i \mid 0 \leq i \leq n\}$ for some *topmost node* u (denoted *top*(I)), *bottommost node* $u\mathtt{R}^n$, and some *length* $n \in \mathbb{N}$ (denoted *len*(I)). Two intervals I_1, I_2 are *independent* if their topmost nodes u_1, u_2 are such that $u_1 \not\preceq u_2$ and $u_2 \not\preceq u_1$ (i.e., none of them is a descendant of the other). A *union of independent intervals* is a set $\tilde{z}$ that can be written as $\bigcup_{i \in J} I_i$, where $(I_i)_{i \in J}$ are pairwise independent intervals. Note that the decomposition of such $\tilde{z}$ into intervals is unique. Moreover, it can be accessed in MSO (we can write in MSO things like "I is one of the intervals in Z", etc.).

The crucial technical contribution is the following lemma.

▶ **Lemma 7.2.** *In MSO+$\mathtt{I}^{\mathsf{nd}}_{\mathcal{W}_{0,1}}$ and in MSO+$\mathtt{I}^{\mathsf{dt}}_{\mathcal{W}_{0,1}}$ over trees one can write a formula $\varphi_{\mathsf{b}}(Z)$ such that for each union of independent intervals $\tilde{z}$, the intervals in $\tilde{z}$ have lengths bounded by some $n \in \mathbb{N}$ if and only if $\varphi_{\mathsf{b}}(\tilde{z})$ holds.*

Proof (sketch). As $\varphi_{\mathsf{b}}(Z)$ we take

$$\forall W \subseteq Z.\ \forall Y \subseteq Z.\ \mathtt{I}^{D}_{\mathcal{W}_{0,1}} X.\ \underbrace{\big(X \subseteq W \wedge \forall x \in X.\, \exists y \in Y.\, x \preceq y\big)}_{\psi(W,Y,X)},$$

where D is either dt or nd (i.e., the index quantifier is either for deterministic or non-deterministic automata – both will work). The formula $\varphi_{\mathsf{b}}(Z)$, given a set $\tilde{z}$, expresses that for every choice of subsets $\tilde{w}, \tilde{y} \subseteq \tilde{z}$ there is a safety automaton $\mathcal{A}$ that checks whether a given subset $\tilde{x}$ of $\tilde{w}$ contains only points with a descendant in $\tilde{y}$.

Note that the subformula $\psi(W, Y, X)$ does not depend on $\tilde{z}$ so a hypothetical safety automaton $\mathcal{A}$ does not have access to $\tilde{z}$, even though the variable Z is formally available in the scope of the subformula $\psi(W, Y, X)$ (one may ensure that $\tilde{z}$ is not visible inside $\psi(W, Y, X)$ by artificially overshadowing the variable Z by another quantifier $\exists Z.$ in front of $\mathtt{I}^{D}_{\mathcal{W}_{0,1}}$).

Suppose first that the intervals in some set $\tilde{z}$ have lengths bounded by some $n \in \mathbb{N}$, and take any $\tilde{w}, \tilde{y} \subseteq \tilde{z}$. The only elements of $\tilde{y}$ that can be descendants of elements of $\tilde{w}$ are elements of the same interval, hence they are located at most n levels below. Thus the property in question is recognised by the following deterministic safety automaton: after every element of $\tilde{x}$ (check if it belongs to $\tilde{w}$ and) wait for n levels on the branch going only right; if no element of $\tilde{y}$ was found, reject. This shows that $\varphi_{\mathsf{b}}(\tilde{z})$ holds (no matter whether we used $\mathtt{I}^{\mathsf{nd}}_{\mathcal{W}_{0,1}}$ or $\mathtt{I}^{\mathsf{dt}}_{\mathcal{W}_{0,1}}$ in its definition). Note that the size of the constructed automaton depends on n.

Suppose now that intervals in $\tilde{z}$ have unbounded lengths. We want to prove that $\varphi_{\mathsf{b}}(\tilde{z})$ does not hold. As $\tilde{w}$ we choose the topmost points of all intervals in $\tilde{z}$. The set $\tilde{y}$ contains the bottommost points of carefully chosen intervals from $\tilde{z}$. Among other properties, the chosen intervals need to have growing lengths. First of all, the whole construction is done in a diagonal way, because the hypothetical safety automaton $\mathcal{A}$ is not known in advance, so we iterate over all such automata, ensuring that each of them is not a good witness for $\mathtt{I}^{D}_{\mathcal{W}_{0,1}} X.\ \psi(\tilde{w}, \tilde{y}, X)$ to hold.

For a fixed automaton $\mathcal{A}$, if the lengths of the intervals are unbounded, then some interval exceeds the counting capacity of the automaton $\mathcal{A}$. Thus, some interval whose bottommost point y belongs to $\tilde{y}$ needs to be sufficiently long, so that the automaton $\mathcal{A}$ admits a pumping pattern with two repeating states along the interval. This allows to repeat the pattern indefinitely (i.e., *pump*), which effectively removes the node y from $\tilde{y}$, making the formula $\psi(\tilde{w}, \tilde{y}, \tilde{x})$ false, while the hypothetical automaton still accepts $\langle \tilde{w}, \tilde{y}, \tilde{x} \rangle$. However, we are not allowed to change the parameter set $\tilde{y}$, so the actual pumping needs to be performed on another part of $\tilde{y}$, which requires to apply a Ramsey-like argument allowing us to shift the pumping place outside the considered set $\tilde{y}$ (this vaguely resembles the pumping scheme from Carayol and Löding [12]). ◀

Once we know that boundedness of sets of independent intervals is expressible in MSO+$\mathtt{I}^{D}_{\mathcal{W}_{0,1}}$, it remains to adjust the technical construction from Bojańczyk et al. [4] to express runs of Minsky machines in terms of boundedness of sets of intervals. This is a rather standard adjustment.

8 Conclusions

Our motivation in this work was to introduce and study index quantifiers $\mathtt{I}$, which try to incorporate the index problem into the syntax of the logic. From that perspective, an important message stems from Theorem 7.1 stating that the logic MSO+$\mathtt{I}$ is undecidable over trees, even in the simplest form of the quantifier, namely deterministic safety index quantifier. The problem whether a regular tree language can be recognised by an automaton of index $\mathcal{R}$ amounts to the satisfaction of a formula $\mathtt{I}^{D}_{\mathcal{R}}X.\,\varphi(X)$, in which the index quantifier is used only once and no parameters are allowed. While this fragment can still be decidable, Theorem 7.1 shows that the frontier is close, and this is (to our knowledge) the first undecidability result related to the index problem.

On the other hand, our study of index quantifiers $\mathtt{I}$ over ω-words provides a more optimistic view. We have introduced an effective index-quantifier elimination procedure, thus reducing MSO+$\mathtt{I}$ into pure MSO. Although the usefulness of index quantifiers in the realm of verification and model-checking is questionable, our approach provides a new and generic game-based way of proving that the index problem over ω-words is decidable [48]. This follows similar lines to Löding [26]. We believe that the most important consequence of this aspect of our study was a thorough analysis of *game quantifiers* $\mathfrak{D}$ – our proof of quantifier elimination for $\mathtt{I}$ goes through a translation to $\mathfrak{D}$ and then elimination of those.

Game quantifiers have been known and studied since 70s, however their applications were mostly limited to descriptive set theory. In a way similar to Kaiser [19], we explicitly introduce the notion of game quantifiers $\mathfrak{D}$ and study the expressive power of MSO+$\mathfrak{D}$. The quantifier-elimination procedure for $\mathfrak{D}$ is quite direct and resembles a parametrised version of the construction of Büchi and Landweber (see Remark 5.5). In contrast to index quantifiers, we believe that a direct use of game quantifiers in MSO may be quite useful when expressing certain game-related properties.

For the sake of applying our quantifier-elimination procedure to index quantifiers, we needed to develop a theory of transducers realising winning strategies in parametrised ω-regular games. To achieve it, we show a novel and apparently quite strong result (see Theorem 5.12) stating that if the involved variables are in some sense separate, then whenever a game quantifier holds, the existence of a respective strategy can be witnessed by a finite-memory transducer.

References

1 Vince Bárány, Łukasz Kaiser, and Alexander Rabinovich. Expressing cardinality quantifiers in monadic second-order logic over trees. *Fundam. Informaticae*, 100(1-4):1–17, 2010. `doi:10.3233/FI-2010-260`.

2 Mikołaj Bojańczyk. A bounding quantifier. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *Computer Science Logic, 18th International Workshop, CSL 2004, 13th Annual Conference of the EACSL, Karpacz, Poland, September 20-24, 2004, Proceedings*, volume 3210 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004. `doi:10.1007/978-3-540-30124-0_7`.

3 Mikołaj Bojańczyk, Laure Daviaud, Bruno Guillon, Vincent Penelle, and A. V. Sreejith. Undecidability of a weak version of MSO+U. *Log. Methods Comput. Sci.*, 16(1), 2020. `doi:10.23638/LMCS-16(1:12)2020`.

4 Mikołaj Bojańczyk, Paweł Parys, and Szymon Toruńczyk. The MSO+U theory of (N,<) is undecidable. In Nicolas Ollinger and Heribert Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*, pages 21:1–21:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. `doi:10.4230/LIPIcs.STACS.2016.21`.

5 Julian C. Bradfield. The modal μ-calculus alternation hierarchy is strict. *Theor. Comput. Sci.*, 195(2):133–153, 1998. doi:10.1016/S0304-3975(97)00217-X.

6 Julian C. Bradfield. Simplifying the modal mu-calculus alternation hierarchy. In Michel Morvan, Christoph Meinel, and Daniel Krob, editors, *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*, volume 1373 of *Lecture Notes in Computer Science*, pages 39–49. Springer, 1998. doi:10.1007/BFB0028547.

7 Julian C. Bradfield. Fixpoints, games and the difference hierarchy. *RAIRO Theor. Informatics Appl.*, 37(1):1–15, 2003. doi:10.1051/ITA:2003011.

8 Julian C. Bradfield, Jacques Duparc, and Sandra Quickert. Transfinite extension of the mu-calculus. In C.-H. Luke Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005, Proceedings*, volume 3634 of *Lecture Notes in Computer Science*, pages 384–396. Springer, 2005. doi:10.1007/11538363_27.

9 Benedikt Brütsch and Wolfgang Thomas. Solving infinite games in the Baire space. *Fundam. Informaticae*, 186(1-4):63–88, 2022. doi:10.3233/FI-222119.

10 J. Richard Büchi. On a decision method in restricted second-order arithmetic. In *Proc. 1960 Int. Congr. for Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

11 J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Trans. Amer. Math. Soc*, 138:295–311, 1969. doi:10.1090/S0002-9947-1969-0280205-0.

12 Arnaud Carayol and Christof Löding. MSO on the infinite binary tree: Choice and order. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2007. doi:10.1007/978-3-540-74915-8_15.

13 Lorenzo Clemente, Paweł Parys, Sylvain Salvati, and Igor Walukiewicz. The diagonal problem for higher-order recursion schemes is decidable. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 96–105. ACM, 2016. doi:10.1145/2933575.2934527.

14 Thomas Colcombet, Denis Kuperberg, Christof Löding, and Michael Vanden Boom. Deciding the weak definability of Büchi definable tree languages. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013), CSL 2013, September 2-5, 2013, Torino, Italy*, volume 23 of *LIPIcs*, pages 215–230. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013. doi:10.4230/LIPIcs.CSL.2013.215.

15 Thomas Colcombet and Christof Löding. The non-deterministic Mostowski hierarchy and distance-parity automata. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, volume 5126 of *Lecture Notes in Computer Science*, pages 398–409. Springer, 2008. doi:10.1007/978-3-540-70583-3_33.

16 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy. In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 368–377. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185392.

17 Alessandro Facchini, Filip Murlak, and Michał Skrzypczak. Rabin-Mostowski index problem: A step beyond deterministic automata. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2013, New Orleans, LA, USA, June 25-28, 2013*, pages 499–508. IEEE Computer Society, 2013. doi:10.1109/LICS.2013.56.

18 Olivier Finkel, Dominique Lecomte, and Pierre Simonnet. An upper bound on the complexity of recognizable tree languages. *RAIRO Theor. Informatics Appl.*, 49(2):121–137, 2015. doi:10.1051/ITA/2015002.

19 Łukasz Kaiser. Game quantification on automatic structures and hierarchical model checking games. In Zoltán Ésik, editor, *Computer Science Logic, 20th International Workshop, CSL 2006, 15th Annual Conference of the EACSL, Szeged, Hungary, September 25-29, 2006, Proceedings*, volume 4207 of *Lecture Notes in Computer Science*, pages 411–425. Springer, 2006. doi:10.1007/11874683_27.

20 Łukasz Kaiser. *Logic and Games on Automatic Structures - Playing with Quantifiers and Decompositions*, volume 6810 of *Lecture Notes in Computer Science*. Springer, 2011. doi:10.1007/978-3-642-22807-0.

21 Vladimir G. Kanoveĭ. Kolmogorov's ideas in the theory of operations on sets. *Russian Math. Surveys*, 43(6):111–155, 1988. doi:10.1070/RM1988v043n06ABEH001995.

22 Alexander Kechris. *Classical Descriptive Set Theory*. Springer-Verlag, New York, 1995. doi:10.1007/978-1-4612-4190-4.

23 Denis Kuperberg, Damian Niwiński, Paweł Parys, and Michał Skrzypczak. Generalised quantifiers based on Rabin-Mostowski index, 2026. arXiv:2601.04739.

24 Shmuel Lifsches and Saharon Shelah. Uniformization and Skolem functions in the class of trees. *J. Symb. Log.*, 63(1):103–127, 1998. doi:10.2307/2586591.

25 Christof Löding. Logic and automata over infinite trees. Habilitation thesis, RWTH Aachen University, 2009.

26 Christof Löding. Decision problems for deterministic pushdown automata on infinite words. In Zoltán Ésik and Zoltán Fülöp, editors, *Proceedings 14th International Conference on Automata and Formal Languages, AFL 2014, Szeged, Hungary, May 27-29, 2014*, volume 151 of *EPTCS*, pages 55–73, 2014. doi:10.4204/EPTCS.151.4.

27 Donald A. Martin. Borel determinacy. *Annals of Mathematics*, 102(2):363–371, 1975. doi:10.2307/1971035.

28 Robert McNaughton. Testing and generating infinite sequences by a finite automaton. *Inf. Control.*, 9(5):521–530, 1966. doi:10.1016/S0019-9958(66)80013-X.

29 Henryk Michalewski and Damian Niwiński. On topological completeness of regular tree languages. In Robert L. Constable and Alexandra Silva, editors, *Logic and Program Semantics - Essays Dedicated to Dexter Kozen on the Occasion of His 60th Birthday*, volume 7230 of *Lecture Notes in Computer Science*, pages 165–179. Springer, 2012. doi:10.1007/978-3-642-29485-3_11.

30 Satoru Miyano and Takeshi Hayashi. Alternating finite automata on omega-words. *Theor. Comput. Sci.*, 32:321–330, 1984. doi:10.1016/0304-3975(84)90049-5.

31 Yiannis N. Moschovakis. *Elementary Induction on Abstract Structures*. North-Holland Publishing Company, 1974.

32 Andrzej Mostowski. On a generalization of quantifiers. *Fundamenta Mathematicae*, 44(1):12–36, 1957. doi:10.4064/fm-44-1-12-36.

33 Andrzej W. Mostowski. Games with forbidden positions. Technical report, University of Gdańsk, 1991.

34 Damian Niwiński. On fixed-point clones. In Laurent Kott, editor, *Automata, Languages and Programming, 13th International Colloquium, ICALP86, Rennes, France, July 15-19, 1986, Proceedings*, volume 226 of *Lecture Notes in Computer Science*, pages 464–473. Springer, 1986. doi:10.1007/3-540-16761-7_96.

35 Damian Niwiński. On the cardinality of sets of infinite trees recognizable by finite automata. In Andrzej Tarlecki, editor, *Mathematical Foundations of Computer Science 1991, 16th International Symposium, MFCS'91, Kazimierz Dolny, Poland, September 9-13, 1991, Proceedings*, volume 520 of *Lecture Notes in Computer Science*, pages 367–376. Springer, 1991. doi:10.1007/3-540-54345-7_80.

36 Damian Niwiński and Igor Walukiewicz. A gap property of deterministic tree languages. *Theor. Comput. Sci.*, 1(303):215–231, 2003. doi:10.1016/S0304-3975(02)00452-8.

37 Damian Niwiński and Igor Walukiewicz. Deciding nondeterministic hierarchy of deterministic tree automata. In Ruy J. G. B. de Queiroz and Patrick Cégielski, editors, *Proceedings of the 11th Workshop on Logic, Language, Information and Computation, WoLLIC 2004, Fontainebleau, France, July 19-22, 2004*, volume 123 of *Electronic Notes in Theoretical Computer Science*, pages 195–208. Elsevier, 2004. doi:10.1016/J.ENTCS.2004.05.015.

38 Damian Niwiński, Paweł Parys, and Michał Skrzypczak. A dichotomy theorem for ordinal ranks in MSO. In Olaf Beyersdorff, Michal Pilipczuk, Elaine Pimentel, and Kim Thang Nguyen, editors, *42nd International Symposium on Theoretical Aspects of Computer Science, STACS 2025, Jena, Germany, March 4-7, 2025*, volume 327 of *LIPIcs*, pages 69:1–69:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.STACS.2025.69.

39 Dominique Perrin and Jean-Éric Pin. *Infinite Words: Automata, Semigroups, Logic and Games*. Pure and Applied Mathematics. Elsevier Morgan Kaufmann, 2004.

40 Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969. doi:10.1090/S0002-9947-1969-0246760-1.

41 Michael O. Rabin. *Automata on Infinite Objects and Church's Problem*. American Mathematical Society, Boston, MA, USA, 1972.

42 Alexander Rabinovich. On decidability of monadic logic of order over the naturals extended by monadic predicates. *Inf. Comput.*, 205(6):870–889, 2007. doi:10.1016/J.IC.2006.12.004.

43 Saharon Shelah. The monadic theory of order. *Annals of Mathematics*, 102(3):379–419, 1975. doi:10.2307/1971037.

44 Dirk Siefkes. The recursive sets in certain monadic second order fragments of arithmetic. *Arch. Math. Logik*, 17(1–2):71–80, 1975. doi:10.1007/BF02280817.

45 Michał Skrzypczak and Igor Walukiewicz. Deciding the topological complexity of Büchi languages. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 99:1–99:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.99.

46 Wolfgang Thomas. Star-free regular sets of omega-sequences. *Inf. Control.*, 42(2):148–156, 1979. doi:10.1016/S0019-9958(79)90629-6.

47 William Wadge. *Reducibility and Determinateness in the Baire Space*. PhD thesis, University of California, Berkeley, 1983.

48 Klaus W. Wagner. On omega-regular sets. *Inf. Control.*, 43(2):123–177, 1979. doi:10.1016/S0019-9958(79)90653-3.

49 Dag Westerståhl. Generalized quantifiers. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2024 edition, 2024. URL: https://plato.stanford.edu/archives/fall2024/entries/generalized-quantifiers/.

50 Thomas Wilke. An algebraic theory for regular languages of finite and infinite words. *Int. J. Algebra Comput.*, 3(4):447–490, 1993. doi:10.1142/S0218196793000287.

51 Sarah Winter and Martin Zimmermann. Finite-state strategies in delay games. *Inf. Comput.*, 272:104500, 2020. doi:10.1016/J.IC.2019.104500.