Deterministic and Game Separability for Regular Languages of Infinite Trees

Lorenzo Clemente
University of Warsaw, Poland

Michał Skrzypczak
University of Warsaw, Poland

Abstract

We show that it is decidable whether two regular languages of infinite trees are separable by a deterministic language, resp., a game language. We consider two variants of separability, depending on whether the set of priorities of the separator is fixed, or not. In each case, we show that separability can be decided in EXPTIME, and that separating automata of exponential size suffice. We obtain our results by reducing to infinite duration games with ω-regular winning conditions and applying the finite-memory determinacy theorem of Büchi and Landweber.

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

Keywords and phrases separation, infinite trees, regular languages, deterministic automata, game automata

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.126

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version *Full Version*: `https://arxiv.org/abs/2105.01137` [15]

Funding *Lorenzo Clemente*: Partially supported by the Polish NCN grant 2017/26/D/ST6/00201.
Michal Skrzypczak: Partially supported by the Polish NCN grant 2017/26/D/ST6/00201.

1 Introduction

One of the most intriguing and motivating problems in the field of automata theory is the *membership problem*. For two fixed classes of languages $\mathcal{C}$ (*input class*) and $\mathcal{D}$ (*output class*), the $(\mathcal{C}, \mathcal{D})$-membership problem asks, given a representation of a language in $\mathcal{C}$, whether this language belongs to $\mathcal{D}$. Among the first results of this type is the famous theorem by Schutzenberger [40] and McNaughton-Papert [30], characterising, among all regular languages of finite words, the subclass of languages that can be defined in first-order logic.

In this paper we consider the class $\mathcal{C}$ of regular languages of infinite trees. While there are many semantically equivalent automata models for this class – e.g., Muller, Rabin, and Street automata [27] – *parity automata* are without doubt the most established such model [25]. The most important descriptional complexity measure of a parity automaton is the set of priorities $C \subseteq \mathbb{N}$ it is allowed to use, which is called its *index*. Not only a larger index allows the automaton to recognise more languages [32], but the computational complexity of known procedures for the emptiness problem crucially depends on the index (the current best bound is quasi-polynomial [8]). The most famous open problem in the area of regular languages of infinite trees is the *nondeterministic index membership problem*, which is the $(\mathcal{C}, \mathcal{D})$-membership problem for $\mathcal{D}$ the class of languages recognised by some nondeterministic parity automaton of a fixed index C (cf. [18]). In many cases, the solution of the membership problem relies either on algebraic representations or determinisation, however algebraic structures for regular languages of infinite trees are of limited availability (cf. [2])

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).
Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 126; pp. 126:1–126:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and deterministic automata do not capture all regular languages. While on infinite words this problem was essentially solved by Wagner already at the end of the '70s [43], its solution for infinite trees seems still far away.

Known decidability results abound if we restrict either the input class $\mathcal{C}$ or the output class $\mathcal{D}$. Results of the first kind are known for $\mathcal{C}$ being the class of deterministic [35] and, more generally, game automata [26, Theorem 1.2]. Results of the second kind (i.e., when the input class $\mathcal{C}$ is the full class of regular languages) exist for the output class $\mathcal{D}$ being the lower levels of the index hierarchy [29, 44] and of the Borel hierarchy [4], the class of deterministic languages [33], and Boolean combinations of open sets [6]. Other variants of the index membership problem are known to be decidable, including the early result of Urbański showing that it is decidable whether a given deterministic parity tree automaton is equivalent to some nondeterministic Büchi one [42], the weak alternating index problems for the class of deterministic automata [31] and Büchi automata [17, 41], and deciding whether a given parity automaton is equivalent to some nondeterministic co-Büchi automaton [17].

Another problem closely related to membership is separability. The $(\mathcal{C}, \mathcal{D})$-*separability* problem asks, given a pair of languages L, M in $\mathcal{C}$, whether there exists a language S in $\mathcal{D}$ (called a *separator*) s.t. $L \subseteq S$ and[1] $S \perp M$. Intuitively, a separator S provides a certificate of disjointness, yielding information on the structure of L, M up to some chosen granularity. The separability problem is a generalisation of the membership problem if the class $\mathcal{C}$ is closed under complement, since we can always take M to be the complement of L, in which case the only candidate for the separator is L itself. There are many elegant results in computer science, formal logic, and mathematics showing that separators always exist. Instances include Lusin's separation theorem in topology (two disjoint analytic sets are always separable by a Borel set; cf. [28, Theorem 14.7]), a folklore result in computability theory (two disjoint co-recursively enumerable sets are separable by a recursive set), Craig's theorems in logic (jointly contradictory first-order formulas can be separated by a formula containing only symbols in the shared vocabulary [19]) and model theory (two disjoint projective classes are separable by an elementary class [19]); in formal language theory, a generalisation of a theorem suggested by Tarski and proved by Rabin [38, Theorem 29] states that two disjoint Büchi languages of infinite trees are separable by a weak language (cf. [39]).

In this work we study the $(\mathcal{C}, \mathcal{D})$-separability problems where $\mathcal{C}$ is the full class of regular languages of infinite trees, and $\mathcal{D}$ is one of four kinds of sub-classes thereof, depending on whether the automaton is deterministic or game, and depending on whether we fix a finite index $C \subseteq \mathbb{N}$ or we leave it unrestricted $C = \mathbb{N}$. Our main result is that all four kinds of the separability problems above are decidable and in EXPTIME. Moreover, we show that if a separator exists, then there is one of exponential size.

▶ **Theorem 1.** *The deterministic and game separability problems can be solved in EXPTIME, both for a fixed finite index $C \subseteq \mathbb{N}$, and an unrestricted one $C = \mathbb{N}$. Moreover, separators with exponentially many states and polynomially many priorities suffice.*

Our work is permeated by the observation that the separability problem for two languages L, M can be phrased in terms of a game of infinite duration with an ω-regular winning condition. In such a *separability game* there are two players, Separator trying to prove that L, M are separable, and Input with the opposite objective. In the simple case of $(\mathcal{C}, \mathcal{D})$-separability where $\mathcal{C}$ is the class of regular languages of ω-words and $\mathcal{D}$ the subclass induced by deterministic parity automata of finite index C, the i-th round of the game is as follows:

[1] We write $S \perp M$ for $S \cap M = \varnothing$.

- Separator plays a priority $c_i \in C$.
- Input plays a letter a_i from the finite alphabet Σ.

The resulting infinite play $(c_0, a_0)(c_1, a_1)\cdots$ is won by Separator if 1) $a_0 a_1 \cdots \in L$ implies $c_0 c_1 \cdots$ is accepting and 2) $a_0 a_1 \cdots \notin L$ implies $c_0 c_1 \cdots$ is rejecting. Since the winning condition is ω-regular, by the result of Büchi and Landweber [7] we can decide who wins the game and moreover finite-memory strategies for Separator suffice. Thanks to a correspondence between such strategies and deterministic separators, Separator wins such a game iff there exists a deterministic automaton with priorities in C separating L, M. This provides both decidability of the separability problem and an upper-bound on the size of separators. We design analogous games with ω-regular winning conditions for the more involved case of infinite trees for the separability problems mentioned above and apply [7].

The separability problems we consider have been open so far and generalise the corresponding membership problems. A solution for deterministic separability can easily be derived from [34], however our techniques based on games are novel and provide a unified view on all problems. When instantiated to the specific case of membership, our decidability results generalise the deterministic case (for both fixed and unconstrained index) [34, 33] and the game membership case for unconstrained index [26, Theorem 7.12]. We believe the game approach is much more direct than the combinatorial and pattern-based techniques used in the previous solutions, cf. [26, Section 7, pp. 29–37]. The game membership problem for a fixed index C has been open so far.

We are not aware of computation complexity results for separability problems over regular languages of infinite trees, neither of an analysis of the size of separators. Regarding deterministic membership, EXPTIME-completeness is known [34, Corollary 11], as well as EXPTIME upper [33, end of page 12] and lower bounds [44, Theorem 4.1] (cf., also [29]) for computing the optimal deterministic index. Devising non-trivial complexity lower bounds for the separability problem is left for future work, as well as extending our approach to other classes of separators.

Related works. Over finite words, variants of the $(\mathcal{C}, \mathcal{D})$-separability problem have been studied for classes $\mathcal{C}$ both more general than the regular languages, such as the context free languages [23, 45] and higher-order languages [14] (later extended to safe schemes over finite trees [1]), and for classes $\mathcal{D}$ more restrictive than the regular languages, such as in [36, 37]. The separability and membership problems have also been studied for several classes of infinite-state systems, such as vector addition systems [11, 10, 24], well-structured transition systems [22], one-counter automata [21], and timed automata [13, 12]. Recent developments on efficient algorithms solving parity games are based on the ability to find a simple separator, yielding both upper bounds on the problem, and lower bounds for a wide family of algorithms [5, 20, Chapter 3]. Finally, it is worth mentioning that games have already been successfully used to provide several characterisation results, such as in [18, 17, 16, 3, 41, 9].

Outline. In Section 2 we introduce automata and other mathematical preliminaries. In Sections 3–6 we present the game-theoretic characterisations of the separability problems we consider. We believe this is the most interesting aspect of this work. A technical report is available [15] where a detailed complexity analysis is performed and full proofs are provided.

2 Preliminaries

A nonempty finite set Σ of *letters* $a \in \Sigma$ is called an *alphabet*. A (Σ-*labelled*) *tree* is a function $t\colon \{\mathsf{L},\mathsf{R}\}^* \to \Sigma$ assigning to each *node* $u \in \{\mathsf{L},\mathsf{R}\}^*$ a *label* $t(a) \in \Sigma$. The *root* of a tree is denoted ϵ. The set of all Σ-labelled trees is denoted Tr_Σ. The symbols L, R are called *directions* and a *branch* is an infinite sequence thereof $d_0 d_1 \cdots \in \{\mathsf{L},\mathsf{R}\}^\omega$. A tree t is uniquely defined by the set of its *paths* $\mathsf{Path}(t) = \{(a_0, d_0)(a_1, d_1) \cdots \in (\Sigma \times \{\mathsf{L},\mathsf{R}\})^\omega \mid \forall i.\ a_i = t(d_0 d_1 \cdots d_{i-1})\}$, which is extended to languages pointwise as $\mathsf{Path}(L) = \{\mathsf{Path}(t) \mid t \in L\}$.

Automata. Fix a nonempty finite set of *priorities* $C \subseteq \mathbb{N}$. A *(top-down, nondeterministic, parity, tree) automaton* is a tuple $\mathcal{A} = (\Sigma, Q, q_0, \Omega, \Delta)$, where Σ is a finite alphabet, Q is a finite set of *states*, amongst which $q_0 \in Q$ is the *initial state*, $\Omega\colon Q \to C$ assigns a priority to every state, and $\Delta \subseteq Q \times \Sigma \times Q \times Q$ is a set of *transitions*. The priority function Ω is extended to a transition $\delta = (q, \_, \_, \_)$ as $\Omega(\delta) := \Omega(q)$, pointwise to an infinite sequence of states $\Omega(q_0 q_1 \cdots) := \Omega(q_0)\Omega(q_1) \cdots \in C^\omega$ and transitions $\Omega(\delta_0 \delta_1 \cdots) = \Omega(\delta_0)\Omega(\delta_1) \cdots \in C^\omega$. An infinite sequence of priorities $c_0 c_1 \cdots \in C^\omega$ is *accepting* if the maximal priority occurring infinitely often is even. Similarly, an infinite sequence of states $\rho = q_0 q_1 \cdots \in Q^\omega$ or of transitions $\rho = \delta_0 \delta_1 \cdots \in \Delta^\omega$ is accepting whenever $\Omega(\rho)$ is accepting. We write $\Delta(q, a) = \{(q, a, q_\mathsf{L}, q_\mathsf{R}) \in \Delta\}$ for the set of transitions from a state $q \in Q$ over a letter $a \in \Sigma$, and $\Delta(a) = \bigcup\{\Delta(q, a) \mid q \in Q\}$ for all transitions over a. We extend the notation above to an infinite path $b = (a_0, d_0)(a_1, d_1) \cdots \in (\Sigma \times \{\mathsf{L},\mathsf{R}\})^\omega$ by writing $\Delta(b)$ for the set of infinite sequences of transitions $\vec{\delta} = \delta_0 \delta_1 \cdots \in \Delta^\omega$ of the form $\delta_i = (q_i, a_i, q_{\mathsf{L},i}, q_{\mathsf{R},i})$ for every i, which are *conform to* b in the sense that q_0 is the initial state of the automaton and $q_{i+1} = q_{d_i,i}$.

A *run* of an automaton $\mathcal{A}$ as above over a tree $t \in \mathrm{Tr}_\Sigma$ is a Q-labelled tree $\rho \in \mathrm{Tr}_Q$ s.t. $\rho(\epsilon) = q_0$ is the initial state and for every node in the tree $u \in \{\mathsf{L},\mathsf{R}\}^*$ the quadruple $\big(\rho(u), t(u), \rho(u\mathsf{L}), \rho(u\mathsf{R})\big)$ belongs to Δ. Such a run is *accepting* if for every branch $d_0 d_1 \cdots \in \{\mathsf{L},\mathsf{R}\}^\omega$ the sequence of states $\big(\rho(d_0 \cdots d_{i-1})\big)_{i \in \omega}$ is accepting. The set of all trees $t \in \mathrm{Tr}_\Sigma$ s.t. $\mathcal{A}$ has an accepting run over t is denoted $\mathrm{L}(\mathcal{A})$ and is called the *language* recognised by $\mathcal{A}$. The corresponding *path language* is $\mathrm{L}^{\mathsf{path}}(\mathcal{A}) := \mathsf{Path}(\mathrm{L}(\mathcal{A})) \subseteq (\Sigma \times \{\mathsf{L},\mathsf{R}\})^\omega$. If $q \in Q$ is a state of an automaton $\mathcal{A}$ then by $\mathcal{A}_q$ we denote the same automaton as $\mathcal{A}$ but with the initial state q_0 changed to q. Thus, $\mathrm{L}(\mathcal{A}_q)$ is the set of trees over which $\mathcal{A}$ has an accepting run ρ starting at $\rho(\epsilon) = q$. In the rest of the paper we assume that all states q in an automaton are *productive* in the sense that $\mathrm{L}(\mathcal{A}_q) \neq \varnothing$.

Deterministic and game automata. We say that $\mathcal{A}$ is a *game automaton* if, for every $q \in Q$ and $a \in \Sigma$, either we have a *conjunctive transition* $\Delta(q, a) = \{(q, a, q_\mathsf{L}, q_\mathsf{R})\}$ or two *disjunctive transitions* $\Delta(q, a) = \{(q, a, q_\mathsf{L}, \top), (q, a, \top, q_\mathsf{R})\}$ (cf. [26, Definition 3.2]), where $\top \neq q_0$ represents a distinguished state in Q accepting every tree (i.e., $\mathrm{L}(\mathcal{A}_\top) = \mathrm{Tr}_\Sigma$) and $q_\mathsf{L}, q_\mathsf{R} \neq \top$. An automaton $\mathcal{A}$ is *deterministic* if it is a game automaton with only conjunctive transitions and in this case for every tree $t \in \mathrm{Tr}_\Sigma$ there exists a unique run ρ of $\mathcal{A}$ over t. A tree language L is *deterministic*, resp., *game*, if it can be recognised by some deterministic, resp., game automaton. Game automata can be complemented with very low complexity by just increasing every priority by one and by swapping conjunctive and disjunctive transitions.

▶ **Lemma 2.** *If $\mathcal{A}$ is a game parity tree automaton, then $\mathrm{Tr}_\Sigma \setminus \mathrm{L}(\mathcal{A})$ can be recognised by a game parity tree automaton with the same number of states and priorities.*

Acceptance games. We present a game-theoretic view on accepting runs. This will serve both as an example of the kind of games that we consider throughout paper, and as a technical tool in the proofs from Sections 5 and 6. Let $t \in \mathrm{Tr}_\Sigma$ be a tree. The *acceptance game* $G^{\mathrm{acc}}(\mathcal{A}, t)$ is played in rounds by two players, Automaton and Pathfinder. The goal of Automaton is to show that $t \in \mathrm{L}(\mathcal{A})$; Pathfinder has the complementary objective $t \notin \mathrm{L}(\mathcal{A})$.

Acceptance game $G^{\mathrm{acc}}(\mathcal{A}, t)$

At the i-th round starting at a *position* $v_i = (u_i, q_i) \in V := \{\mathsf{L}, \mathsf{R}\}^* \times Q$:
[A: δ] Automaton plays a transition $\delta_i = (q_i, t(u_i), q_{\mathsf{L},i}, q_{\mathsf{R},i}) \in \Delta(q_i, t(u_i))$.
[P: d] Pathfinder plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.
The next position is $v_{i+1} := (u_i d_i, q_{d_i,i})$.

The initial position is $v_0 := (\epsilon, q_0)$. Automaton *wins* the resulting infinite play $\pi = (\delta_0, d_0)(\delta_1, d_1)\cdots$ if the sequence of transitions $\delta_0\delta_1\cdots$ is accepting. Automaton's moves in the acceptance game $G^{\mathrm{acc}}(\mathcal{A}, t)$ are performed according to a *strategy* for Automaton. This is a tuple $\mathcal{M} = (M, \ell_0, \overline{\delta}, \tau)$, where M is a set of *memory states*, of which $\ell_0 \in M$ is the initial memory state, $\overline{\delta}\colon V \times M \to \Delta$ is an output function which in a position (u, q) and a memory state ℓ selects a transition $\overline{\delta}\big((u, q), \ell\big) \in \Delta(q, t(u))$ of $\mathcal{A}$, and $\tau\colon V \times M \times \{\mathsf{L}, R\} \to M$ is a memory update function which, in a given position v, memory state ℓ, and direction d selects the next memory state $\tau(v, \ell, d) \in M$. An infinite play π as above is *conform* to a strategy $\mathcal{M}$ if during the play π Automaton keeps track of the current position v_i and memory state ℓ_i, updating them after each round (i.e., $\ell_{i+1} := \tau(v_i, \ell_i, d_i)$) and her consecutive choices of transitions δ_i are done according to $\overline{\delta}(v_i, \ell_i)$. A strategy $\mathcal{M}$ is *winning* if every play conform to it is winning for Automaton. Automaton wins the acceptance game if she has a winning strategy. The following proposition is folklore.

▶ **Proposition 3.** *Let $t \in \mathrm{Tr}_\Sigma$ and $\mathcal{A}$ be an automaton over the alphabet Σ. Automaton wins the acceptance game $G^{\mathrm{acc}}(\mathcal{A}, t)$ if, and only if, $t \in \mathrm{L}(\mathcal{A})$.*

Disjointness games. Let $\mathcal{A}$ and $\mathcal{B}$ be two nondeterministic automata. We recall a standard game used to characterise whether $\mathrm{L}(\mathcal{A}) \perp \mathrm{L}(\mathcal{B})$. This will be crucial in the correctness proofs throughout Sections 3–6. The *disjointness game* $G^{\mathrm{dis}}(\mathcal{A}, \mathcal{B})$ is played by two players, Automaton and Pathfinder. Automaton's aim is to incrementally build a tree accepted by both $\mathcal{A}$ and $\mathcal{B}$, witnessing $\mathrm{L}(\mathcal{A}) \cap \mathrm{L}(\mathcal{B}) \neq \varnothing$, while Pathfinder has the opposite objective.[2] The set of positions of the game is $Q^{\mathcal{A}} \times Q^{\mathcal{B}}$, and the initial position is $(q_0^{\mathcal{A}}, q_0^{\mathcal{B}})$.

Disjointness game $G^{\mathrm{dis}}(\mathcal{A}, \mathcal{B})$

At the i-th round starting at a position $(q_i^{\mathcal{A}}, q_i^{\mathcal{B}})$:
[A: a] Automaton plays a letter $a_i \in \Sigma$.
[A: $\delta^{\mathcal{A}}$] Automaton plays a transition $\delta_i^{\mathcal{A}} = (q_i^{\mathcal{A}}, a_i, q_{\mathsf{L},i}^{\mathcal{A}}, q_{\mathsf{R},i}^{\mathcal{A}}) \in \Delta^{\mathcal{A}}(q_i^{\mathcal{A}}, a_i)$.
[A: $\delta^{\mathcal{B}}$] Automaton plays a transition $\delta_i^{\mathcal{B}} = (q_i^{\mathcal{B}}, a_i, q_{\mathsf{L},i}^{\mathcal{B}}, q_{\mathsf{R},i}^{\mathcal{B}}) \in \Delta^{\mathcal{B}}(q_i^{\mathcal{B}}, a_i)$.
[P: d] Pathfinder plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.
The next position is $(q_{d_i,i}^{\mathcal{A}}, q_{d_i,i}^{\mathcal{B}})$.

[2] The disjointness game could equivalently be phrased as a nonemptiness game for the product automaton $\mathcal{A} \times \mathcal{B}$ recognising $\mathrm{L}(\mathcal{A}) \cap \mathrm{L}(\mathcal{B})$. However, in our technical development it will be more direct to use the disjointness game.

Let the resulting infinite play be $\pi = (a_0, \delta_0^{\mathcal{A}}, \delta_0^{\mathcal{B}}, d_0)(a_1, \delta_1^{\mathcal{A}}, \delta_1^{\mathcal{B}}, d_1)\cdots$. Such a play induces an infinite path $b = (a_0, d_0)(a_1, d_1)\cdots$ and two sequences of transitions $\vec{\delta}^{\mathcal{A}} := \delta_0^{\mathcal{A}}\delta_1^{\mathcal{A}}\cdots$ and $\vec{\delta}^{\mathcal{B}} := \delta_0^{\mathcal{B}}\delta_1^{\mathcal{B}}\cdots$. The rules of the game guarantee that $\vec{\delta}^{\mathcal{A}} \in \Delta^{\mathcal{A}}(b)$ and $\vec{\delta}^{\mathcal{B}} \in \Delta^{\mathcal{B}}(b)$. Automaton wins the play π if both sequences $\vec{\delta}^{\mathcal{A}}$ and $\vec{\delta}^{\mathcal{B}}$ are accepting.

In the rest of the paper it will be more useful to consider Pathfinder's point of view. Since her winning condition can be presented as Rabin condition, whenever she wins, she has a *memoryless* (i.e., $M = \{\ell_0\}$) winning strategy. Such a memoryless strategy for Pathfinder in the disjointness game can be represented by a function $\mathcal{P}\colon \left(\bigcup_{a\in\Sigma} \Delta^{\mathcal{A}}(a) \times \Delta^{\mathcal{B}}(a)\right) \to \{\mathsf{L}, \mathsf{R}\}$, which we call a *pathfinder*.

▶ **Lemma 4.** *If* $\mathrm{L}(\mathcal{A}) \perp \mathrm{L}(\mathcal{B})$ *then there is a pathfinder* $\mathcal{P}$ *which is winning for* Pathfinder *in the disjointness game* $G^{\mathrm{dis}}(\mathcal{A}, \mathcal{B})$*.*

▶ **Corollary 5.** *Assume that* $\mathrm{L}(\mathcal{A}) \perp \mathrm{L}(\mathcal{B})$ *and let* $\mathcal{P}$ *be a pathfinder as above. Let* $b = (a_0, d_0)(a_1, d_1)\cdots \in (\Sigma \times \{\mathsf{L}, \mathsf{R}\})^\omega$ *be a path and* $\vec{\delta}^{\mathcal{A}} = \delta_0^{\mathcal{A}}\delta_1^{\mathcal{A}}\cdots \in \Delta^{\mathcal{A}}(b)$, $\vec{\delta}^{\mathcal{B}} = \delta_0^{\mathcal{B}}\delta_1^{\mathcal{B}}\cdots \in \Delta^{\mathcal{B}}(b)$ *be two sequences of transitions of these automata that are conform to* b*. If for every* $i \in \omega$ *we have* $\mathcal{P}(\delta_i^{\mathcal{A}}, \delta_i^{B}) = d_i$ *then at least one of the sequences* $\vec{\delta}^{\mathcal{A}}$ *and* $\vec{\delta}^{\mathcal{B}}$ *is rejecting.*

The construction above has a specific property if one of the involved automata (e.g., $\mathcal{A}$) is a game automaton. Since we assume that every state is productive, positions of the form $(\top, q^{\mathcal{B}})$ are losing for Pathfinder in $G^{\mathrm{dis}}(\mathcal{A}, \mathcal{B})$. Therefore, without loss of generality we can assume that the pathfinder $\mathcal{P}$ satisfies the following observation.

▶ Remark 6. Consider a transition $\delta^{\mathcal{A}} = (q^{\mathcal{A}}, a, q_{\mathsf{L}}^{\mathcal{A}}, \top)$ (resp., $\delta^{\mathcal{A}} = (q^{\mathcal{A}}, a, \top, q_{\mathsf{R}}^{\mathcal{A}})$) in a game automaton $\mathcal{A}$. Then, $\mathcal{P}(\delta^{\mathcal{A}}, \_)$ is constantly equal to L (resp., R).

3 Separability by deterministic automata with priorities in C

In this section we present a game-theoretic characterisation of separability by deterministic automata over a fixed finite set of priorities $C \subseteq \mathbb{N}$. Let $\mathcal{A}$, $\mathcal{B}$ be two nondeterministic automata over infinite trees. We extend the game from the introduction over ω-words with two additional actions, a *selector* for Separator and a direction for Input.

C-deterministic-separability game $G_{\mathrm{det}}^{\mathrm{sep}}(\mathcal{A}, \mathcal{B}, C)$

At the i-th round:
- [S: c] Separator plays a priority $c_i \in C$.
- [I: a] Input plays a letter $a_i \in \Sigma$.
- [S: f] Separator plays a *selector* $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$.
- [I: d] Input plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

Intuitively, a selector encodes a direction for each (relevant) transition of $\mathcal{B}$ and this is used for the correctness of the separator. Assume that the resulting infinite play is $\pi = (c_0, a_0, f_0, d_0)(c_1, a_1, f_1, d_1)\cdots$, with the induced infinite path $b := (a_0, d_0)(a_1, d_1)\cdots$. Separator wins the play π if the following two conditions are satisfied:

1. $\pi \in \mathbf{W}_{\mathcal{A}}$: If there exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{A}} = \delta_0^{\mathcal{A}}\delta_1^{\mathcal{A}}\cdots \in \Delta^{\mathcal{A}}(b)$, then $c_0c_1\cdots$ is accepting.
2. $\pi \in \mathbf{W}_{\mathcal{B}}$: If there exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{B}} = \delta_0^{\mathcal{B}}\delta_1^{\mathcal{B}}\cdots \in \Delta^{\mathcal{B}}(b)$ s.t. for every $i \in \omega$ we have $f_i(\delta_i^{\mathcal{B}}) = d_i$, then $c_0c_1\cdots$ is rejecting.

The following lemma states that the separability game correctly characterises the deterministic separability problem.

▶ **Lemma 7.** *Separator wins* $G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A},\mathcal{B},C)$ *if, and only if,* $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$ *can be separated by a deterministic parity tree automaton with priorities in* C.

We present a full proof in order to show the rôle of Separator's selectors.

Soundness. Assume that Separator wins the separability game $G := G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A},\mathcal{B},C)$ by a finite-memory winning strategy $\mathcal{M} = (M, \ell_0, (\overline{c}, \overline{f}), \tau)$. Strategy $\mathcal{M}$ has two decision functions: $\overline{c}$ assigns to each $\ell \in M$ a priority $\overline{c}(\ell) \in C$, and $\overline{f}$ assigns to each $\ell \in M$ and $a \in \Sigma$ a selector $\overline{f}(\ell, a) \in \{\mathsf{L},\mathsf{R}\}^{\Delta^{\mathcal{B}}(a)}$. Moreover, the type of the memory update function is $\tau\colon M \times \Sigma \times \{\mathsf{L},\mathsf{R}\} \to M$. Consider a deterministic parity tree automaton $\mathcal{S} := (\Sigma, M, \ell_0, \Omega^{\mathcal{S}}, \Delta^{\mathcal{S}})$ which has the same set of states M and initial state ℓ_0 as $\mathcal{M}$, priorities are induced by the decision function $\overline{c}$ of $\mathcal{M}$ as $\Omega^{\mathcal{S}}(\ell) := \overline{c}(\ell)$, and transitions are of the form $\Delta^{\mathcal{S}} = \{(\ell, a, \tau(\ell, a, \mathsf{L}), \tau(\ell, a, \mathsf{R})) \mid \ell \in M, a \in \Sigma\}$.

We show that $\mathcal{S}$ separates $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$. We first show $\mathrm{L}(\mathcal{A}) \subseteq \mathrm{L}(\mathcal{S})$. Let $t \in \mathrm{L}(\mathcal{A})$ be a tree that is accepted by the automaton $\mathcal{A}$, as witnessed by an accepting run $\rho^{\mathcal{A}}$. Let $\rho^{\mathcal{S}}$ be the unique run of $\mathcal{S}$ over t. Consider any branch $d_0 d_1 \cdots \in \{\mathsf{L},\mathsf{R}\}^{\omega}$. We need to show that the sequence of priorities $\big(\Omega^{\mathcal{S}}(\rho^{\mathcal{S}}(d_0 \cdots d_{i-1}))\big)_{i\in\omega}$ is accepting. Consider a play π of G where at the i-th round Separator plays according to the strategy $\mathcal{M}$ with current memory state $\ell_i \in M$ and Input plays according to the letters from t and directions $d_0 d_1 \cdots$ fixed above:

[S: c] Separator plays the priority $c_i := \overline{c}(\ell_i) \in C$.
[I: a] Input plays the letter $a_i := t(u_i) \in \Sigma$, where $u_i := d_0 \cdots d_{i-1}$.
[S: f] Separator plays the selector $f_i := \overline{f}(\ell_i, a_i) \in \{\mathsf{L},\mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$ (the selector is irrelevant in this part of the proof).
[I: d] Input plays the direction $d_i \in \{\mathsf{L},\mathsf{R}\}$ as fixed above.

The next memory state is $\ell_{i+1} := \tau(\ell_i, a_i, d_i)$. Let the resulting infinite play be $\pi = (c_0, a_0, f_0, d_0)(c_1, a_1, f_1, d_1)\cdots$. By the construction of $\mathcal{S}$ we know that $\ell_i = \rho^{\mathcal{S}}(u_i)$ and therefore $c_i = \Omega^{\mathcal{S}}(\rho^{\mathcal{S}}(u_i))$. Since $t \in \mathrm{L}(\mathcal{A})$, there exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{A}} = \delta^{\mathcal{A}}_0 \delta^{\mathcal{A}}_1 \cdots \in \Delta^{\mathcal{A}}(b)$ along the path $b = (a_0, d_0)(a_1, d_1)\cdots$. Since Separator is winning, $\pi \in \mathbf{W}_{\mathcal{A}}$ and thus the sequence $c_0 c_1 \cdots$ is accepting, as required.

We now argue that $\mathrm{L}(\mathcal{S})$ and $\mathrm{L}(\mathcal{B})$ are disjoint. Towards reaching a contradiction, assume that $t \in \mathrm{L}(\mathcal{S}) \cap \mathrm{L}(\mathcal{B})$ belongs to their intersection. Let $\rho^{\mathcal{S}}$ be the unique run of $\mathcal{S}$ over t, and let $\rho^{\mathcal{B}}$ be an accepting run of $\mathcal{B}$ over t. Consider a play $\pi = (c_0, a_0, f_0, d_0)(c_1, a_1, f_1, d_1)\cdots$ of G where the i-th round is played as above except that Input plays the direction $d_i := f_i(\delta^{\mathcal{B}}_i)$, obtained by applying the selector f_i to the transition $\delta^{\mathcal{B}}_i := \big(\rho^{\mathcal{B}}(u_i), t(u_i), \rho^{\mathcal{B}}(u_i\mathsf{L}), \rho^{\mathcal{B}}(u_i\mathsf{R})\big)$ determined according to the run $\rho^{\mathcal{B}}$. By the choice of directions d_i's, the sequence of transitions $\vec{\delta}^{\mathcal{B}} = \delta^{\mathcal{B}}_0 \delta^{\mathcal{B}}_1 \cdots \in (\Delta^{\mathcal{B}})^{\omega}$ satisfies $f_i(\delta^{\mathcal{B}}_i) = d_i$ for every $i \in \omega$. Since the run $\rho^{\mathcal{B}}$ is accepting, $\vec{\delta}^{\mathcal{B}}$ is accepting. Since Separator is winning, $\pi \in \mathbf{W}_{\mathcal{B}}$ and thus the sequence of priorities $c_0 c_1 \cdots$ is rejecting. However, this is a contradiction, because for each $i \in \omega$ we have $\ell_i = \rho^{\mathcal{S}}(u_i)$ and $c_i = \Omega^{\mathcal{S}}(\ell_i)$ and we assumed that the run $\rho^{\mathcal{S}}$ is accepting. ◀

Completeness. Assume that $\mathcal{S} = (\Sigma, Q^{\mathcal{S}}, q^{\mathcal{S}}_0, \Delta^{\mathcal{S}}, \Omega^{\mathcal{S}})$ is a deterministic automaton with priorities in C separating $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$, and we show that Separator wins the separability game G. Since $\mathcal{S}$ is a separator, we have that $\mathrm{L}(\mathcal{S}) \perp \mathrm{L}(\mathcal{B})$, and by Lemma 4 there exists a pathfinder $\mathcal{P}$. Consider the following strategy of Separator, with memory structure $Q^{\mathcal{S}}$ and initial memory state $q^{\mathcal{S}}_0$. At the i-th round of G, starting with a memory state $q^{\mathcal{S}}_i$,

[S: c] Separator plays the priority $c_i := \Omega^{\mathcal{S}}(q^{\mathcal{S}}_i) \in C$.
[I: a] Input plays an arbitrary letter $a_i \in \Sigma$.
[S: f] Separator plays the selector $f_i := \mathcal{P}(\delta^{\mathcal{S}}_i, \_) \in \{\mathsf{L},\mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$, where $\Delta^{\mathcal{S}}(q^{\mathcal{S}}_i, a_i) = \{\delta^{\mathcal{S}}_i\}$.
[I: d] Input plays an arbitrary direction $d_i \in \{\mathsf{L},\mathsf{R}\}$.

The next memory state is $q^{\mathcal{S}}_{i+1} := q^{\mathcal{S}}_{d_i,i}$, where $\delta^{\mathcal{S}}_i = (q^{\mathcal{S}}_i, a_i, q^{\mathcal{S}}_{\mathsf{L},i}, q^{\mathcal{S}}_{\mathsf{R},i})$. This concludes the description of the i-th round of G. Let the resulting infinite play be $\pi = (c_0, a_0, f_0, d_0)(c_1, a_1, f_1, d_1)\cdots$, with induced infinite path $b := (a_0, d_0)(a_1, d_1)\cdots$. Let $\vec{\delta}^{\mathcal{S}} := \delta^{\mathcal{S}}_0\delta^{\mathcal{S}}_1\cdots$ be the sequence of transitions used to define the selectors f_i. Clearly $\vec{\delta}^{\mathcal{S}} \in \Delta^{\mathcal{S}}(b)$.

First, we argue that $\pi \in \mathbf{W}_{\mathcal{A}}$ holds. Let $\vec{\delta}^{\mathcal{A}} = \delta^{\mathcal{A}}_0\delta^{\mathcal{A}}_1\cdots \in \Delta^{\mathcal{A}}(b)$ be an accepting sequence of transitions of the automaton $\mathcal{A}$. Since each state of $\mathcal{A}$ is productive, one can construct a tree $t \in \mathrm{L}(\mathcal{A})$ s.t. $b \in \mathsf{Path}(t)$. Since $\mathrm{L}(\mathcal{A}) \subseteq \mathrm{L}(\mathcal{S})$ by the assumption, $t \in \mathrm{L}(\mathcal{S})$ as well, and since $\mathcal{S}$ is deterministic, the unique run of $\mathcal{S}$ over t is accepting. By the definition of Separator's strategy, the sequence of priorities along the branch $d_0d_1\cdots$ of this accepting run is precisely $c_0c_1\cdots$, which thus must be accepting, as required.

Regarding $\mathbf{W}_{\mathcal{B}}$, let $\vec{\delta}^{\mathcal{B}} := \delta^{\mathcal{B}}_0\delta^{\mathcal{B}}_1\cdots \in \Delta^{\mathcal{B}}(b)$ be an accepting sequence of transitions over the path b conform to the selectors f_i, i.e., for every $i \in \omega$ we have $f_i(\delta^{\mathcal{B}}_i) = d_i$. By the definition of f_i, for every $i \in \omega$ we have $d_i = \mathcal{P}(\delta^{\mathcal{S}}_i, \delta^{\mathcal{B}}_i)$. Thus, the assumptions of Corollary 5 are satisfied and at least one of the sequences $\vec{\delta}^{\mathcal{S}}$, $\vec{\delta}^{\mathcal{B}}$ must be rejecting. Since we assumed that $\vec{\delta}^{\mathcal{B}}$ is accepting, it means that $\vec{\delta}^{\mathcal{S}}$ is rejecting, and so is $c_0c_1\cdots$ since $c_i = \Omega^{\mathcal{S}}(\delta^{\mathcal{S}}_i)$. ◀

4 Separability by deterministic automata

In this section we present a game-theoretic characterisation of the deterministic separability problem. Notice that here we do not fix in advance a finite set of priorities C. The deterministic-separability game $G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A}, \mathcal{B})$ below is a variant of the game with fixed priorities C from Section 3.

Deterministic-separability game $G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A}, \mathcal{B})$

At the i-th round:
- [I: a] Input plays a letter $a_i \in \Sigma$.
- [S: f] Separator plays a selector $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$.
- [I: d] Input plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

Separator wins the resulting infinite play $\pi = (a_0, f_0, d_0)(a_1, f_1, d_1)\cdots$, with induced infinite path $b := (a_0, d_0)(a_1, d_1)\cdots$, if at least one of the two conditions below fails:
1. $\pi \in \mathbf{W}_{\mathcal{A}}$: There exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{A}} = \delta^{\mathcal{A}}_0\delta^{\mathcal{A}}_1\cdots \in \Delta^{\mathcal{A}}(b)$.
2. $\pi \in \mathbf{W}_{\mathcal{B}}$: There exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{B}} = \delta^{\mathcal{B}}_0\delta^{\mathcal{B}}_1\cdots \in \Delta^{\mathcal{B}}(b)$ s.t. for every $i \in \omega$ we have $f_i(\delta^{\mathcal{B}}_i) = d_i$.

Before we prove the equivalence between the game and the existence of a separator, we define a separator candidate, namely the path-closure of $\mathrm{L}(\mathcal{A})$. This is important since it will turn out that if a separator exists, then the path-closure is itself a separator. Given a language of trees L, its *path-closure*, denoted $\forall\mathsf{Path}(L)$, is the set of all trees t s.t. for every path $b \in \mathsf{Path}(t)$ there exists some tree $t' \in L$ s.t. $b \in \mathsf{Path}(t')$ as well. The path-closure operator is directly connected with deterministic automata.

▶ **Lemma 8** (cf. [34, Proposition 1]). *Given a nondeterministic automaton $\mathcal{A}$ one can construct a deterministic automaton $\mathcal{A}^{\mathrm{path}}$ recognising the path closure of $\mathrm{L}(\mathcal{A})$, i.e., $\mathrm{L}(\mathcal{A}^{\mathrm{path}}) = \forall\mathsf{Path}(\mathrm{L}(\mathcal{A}))$. Moreover, $\mathrm{L}(\mathcal{A}^{\mathrm{path}})$ is the smallest deterministic language containing $\mathrm{L}(\mathcal{A})$.*

The following lemma binds together the game $G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A}, \mathcal{B})$, separability, and path-closures.

▶ **Lemma 9.** *The following three conditions are equivalent:*
1. *Separator wins the deterministic-separability game $G^{\mathrm{sep}}_{\mathrm{det}}(\mathcal{A}, \mathcal{B})$.*
2. *The automaton $\mathcal{A}^{\mathrm{path}}$ is a deterministic separator for $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$.*
3. *There exists a deterministic separator for $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$.*

Proof sketch. Consider the implication "1 ⇒ 2". Firstly, $L(\mathcal{A}) \subseteq L(\mathcal{A}^{\text{path}})$ because the operator $\forall\mathsf{Path}(\_)$ is non-decreasing. Moreover, the fact that $L(\mathcal{A}^{\text{path}}) \perp L(\mathcal{B})$ is witnessed by the choices of selectors f_i by a winning strategy of Separator in $G^{\text{sep}}_{\text{det}}(\mathcal{A}, \mathcal{B})$. The implication "2 ⇒ 3" is trivial. The proof of the implication "3 ⇒ 1" is similar to the proof of completeness in Lemma 7 – a separating automaton $\mathcal{S}$ can be used to construct a pathfinder $\mathcal{P}$, witnessing that $L(\mathcal{S})$ and $L(\mathcal{B})$ are disjoint. Now, one can construct a strategy of Separator in $G^{\text{sep}}_{\text{det}}(\mathcal{A}, \mathcal{B})$ by simulating $\mathcal{S}$ and using $\mathcal{P}$ to choose the selectors f_i. ◀

5 Separability by game automata

In this section we provide a game-theoretic characterisation for the game automata separability problem. Fix two automata $\mathcal{A}$ and $\mathcal{B}$ and consider the following separability game $G^{\text{sep}}_{\text{game}}(\mathcal{A}, \mathcal{B})$. The new ingredient is that Separator can choose a *mode* – a symbol from the set $\{\vee, \wedge\}$. It has two uses. First, in the construction of the separating game automaton, the mode dictates whether there will be a conjunctive or a disjunctive transition. Second, depending on the chosen mode, Separator will have to play a selector for the automaton $\mathcal{A}$ or $\mathcal{B}$, which will guarantee that the constructed automaton is a separator.

Game-separability game $G^{\text{sep}}_{\text{game}}(\mathcal{A}, \mathcal{B})$

At the i-th round:
- [I: a] Input plays a letter $a_i \in \Sigma$.
- [S: m] Separator plays a mode $m_i \in \{\vee, \wedge\}$.
- [S: f] Separator plays either
  - **a.** a selector $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{A}}(a_i)}$ for $\mathcal{A}$ if $m_i = \vee$ or
  - **b.** a selector $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$ for $\mathcal{B}$ if $m_i = \wedge$.
- [I: d] Input plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

Separator wins an infinite play $\pi = (a_0, m_0, f_0, d_0)(a_1, m_1, f_1, d_1)\cdots$ inducing a path $b = (a_0, d_0)(a_1, d_1)\cdots$ whenever at least one of the two conditions below fail:

1. $\pi \in \mathbf{W}_{\mathcal{A}}$: There exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{A}} = \delta^{\mathcal{A}}_0 \delta^{\mathcal{A}}_1 \cdots \in \Delta^{\mathcal{A}}(b)$ s.t. for all $i \in \mathbb{N}$ we have $(m_i = \vee) \Rightarrow f_i(\delta^{\mathcal{A}}_i) = d_i$.
2. $\pi \in \mathbf{W}_{\mathcal{B}}$: There exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{B}} = \delta^{\mathcal{B}}_0 \delta^{\mathcal{B}}_1 \cdots \in \Delta^{\mathcal{B}}(b)$ s.t. for all $i \in \mathbb{N}$ we have $(m_i = \wedge) \Rightarrow f_i(\delta^{\mathcal{B}}_i) = d_i$.

▶ **Lemma 10.** *Separator wins the separability game $G^{\text{sep}}_{\text{game}}(\mathcal{A}, \mathcal{B})$ if, and only if, there exists a game automaton $\mathcal{S}$ separating $L(\mathcal{A})$, $L(\mathcal{B})$.*

In the proof of this lemma we will build separating automata with a more general acceptance condition than the parity condition, which will simplify the technical details. A *generalised game automaton* $\mathcal{A} = (\Sigma, Q, q_0, \Delta, \mathcal{D})$ is just like a game automaton except that the priority mapping Ω is replaced by a deterministic ω-word parity automaton $\mathcal{D}$ over alphabet $\Sigma \times \{\mathsf{L}, \mathsf{R}\}$. A run $\rho \in \mathrm{Tr}_Q$ of such an automaton over a tree $t \in \mathrm{Tr}_\Sigma$ is *accepting* if for every path $b = (a_0, d_0)(a_1, d_1)\cdots \in \mathsf{Path}(t)$ either $\rho(d_0 \cdots d_{i-1}) = \top$ for some $i \in \omega$, or $b \in L(\mathcal{D})$. The acceptance game $G^{\text{acc}}(\mathcal{A}, t)$ can easily be adapted to the case of a generalised game automaton $\mathcal{A}$ by only modifying the winning condition.

▶ **Lemma 11.** *A generalised game automaton $\mathcal{A}$ with a generalised acceptance condition recognised by a deterministic parity automaton $\mathcal{D}$ can be transformed into an equivalent (ordinary) game automaton $\mathcal{B}$ of size polynomial in $\mathcal{A}$ and $\mathcal{D}$.*

We now prove Lemma 10. Its proof is given in full details because, unlike in Sections 3 and 4, it is not obvious how to construct a separator from a winning strategy for Separator.

Soundness. Assume that Separator wins the game-separability game $G := G^{\mathrm{sep}}_{\mathrm{game}}(\mathcal{A}, \mathcal{B})$ and we show that there exists a game automaton $\mathcal{S}$ separating $\mathrm{L}(\mathcal{A})$ from $\mathrm{L}(\mathcal{B})$. Let $\mathcal{M} = (M, \ell_0, (\overline{m}, \overline{f}), \tau)$ be a finite-memory winning strategy of Separator in G.

Before we move to the construction of the separating automaton, we first define its generalised acceptance condition. Let $L_{\mathcal{A}}$ (resp., $L_{\mathcal{B}}$) be the set of those paths $b = (a_0, d_0)(a_1, d_1) \cdots \in (\Sigma \times \{\mathsf{L}, \mathsf{R}\})^\omega$ s.t. the unique play π of G in which Input plays consecutive letters and directions from b and Separator uses her winning strategy $\mathcal{M}$ satisfies the condition $\mathbf{W}_{\mathcal{A}}$ (resp., $\mathbf{W}_{\mathcal{B}}$). Since the strategy $\mathcal{M}$ is winning for Separator, the languages $L_{\mathcal{A}}$ and $L_{\mathcal{B}}$ are disjoint. Moreover, since the strategy $\mathcal{M}$ is finite memory and both $\mathbf{W}_{\mathcal{A}}$, $\mathbf{W}_{\mathcal{B}}$ are ω-regular, so are the languages $L_{\mathcal{A}}$ and $L_{\mathcal{B}}$. Let $\mathcal{D}$ be any deterministic automaton over ω-words that separates $L_{\mathcal{A}}$ from $L_{\mathcal{B}}$ (the simplest case is to take $\mathcal{D}$ recognising the language $L_{\mathcal{A}}$). We build a separating automaton as a generalised game automaton

$$\mathcal{S} := \alpha(\mathcal{M}, \mathcal{D}) := (\Sigma, M \cup \{\top\}, \ell_0, \Delta^{\mathcal{S}}, \mathcal{D}), \text{ where}$$
$$\Delta^{\mathcal{S}}(\ell, a) := \begin{cases} \{(\ell, a, \ell_{\mathsf{L}}, \top), (\ell, a, \top, \ell_{\mathsf{R}})\} & \text{if } \overline{m}(\ell, a) = \vee, \\ \{(\ell, a, \ell_{\mathsf{L}}, \ell_{\mathsf{R}})\} & \text{if } \overline{m}(\ell, a) = \wedge, \end{cases}$$

for every $\ell \in M$ and $a \in \Sigma$, where for $d \in \{\mathsf{L}, \mathsf{R}\}$ we have $\ell_d := \tau(\ell, a, d)$. We now show that $\mathcal{S}$ separates $\mathrm{L}(\mathcal{A})$ from $\mathrm{L}(\mathcal{B})$. In order to show $\mathrm{L}(\mathcal{A}) \subseteq \mathrm{L}(\mathcal{S})$, let $t \in \mathrm{L}(\mathcal{A})$ as witnessed by an accepting run $\rho^{\mathcal{A}}$. We show that Automaton wins the acceptance game $G_{\mathcal{S}} := G^{\mathrm{acc}}(\mathcal{S}, t)$. To show this we play in parallel the separability game G and the acceptance game $G_{\mathcal{S}}$, maintaining the following invariant: At the i-th round, the current finite path of the input tree t is $(a_0, d_0) \cdots (a_{i-1}, d_{i-1})$, Separator's winning strategy $\mathcal{M}$ in the separability game G is in memory state ℓ_i, the current state of the separating automaton $\mathcal{S}$ in the acceptance game $G_{\mathcal{S}}$ is also ℓ_i, and $\rho^{\mathcal{A}}(d_0 \cdots d_{i-1}) = q^{\mathcal{A}}_i$. The i-th round is then played as follows:

- $G.[\mathsf{I}\colon a]$ Input plays the letter $a_i := t(u_i)$ for $u_i := d_0 \cdots d_{i-1}$.
- $G.[\mathsf{S}\colon m]$ Separator plays the mode $m_i := \overline{m}(\ell_i, a_i) \in \{\vee, \wedge\}$.
- $G.[\mathsf{S}\colon f]$ Separator plays either
  - **a.** a selector $f_i := \overline{f}(\ell_i, a_i) \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{A}}(a_i)}$ for $\mathcal{A}$ if $m_i = \vee$ or
  - **b.** a selector $f_i := \overline{f}(\ell_i, a_i) \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$ for $\mathcal{B}$ if $m_i = \wedge$.
- $G_{\mathcal{S}}.[\mathsf{A}\colon \delta]$ Automaton plays the transition $\delta^{\mathcal{S}}_i \in \Delta^{\mathcal{S}}(\ell_i, a_i)$, defined as follows. Let $\delta^{\mathcal{A}}_i := \big(\rho^{\mathcal{A}}(u_i), t(u_i), \rho^{\mathcal{A}}(u_i\mathsf{L}), \rho^{\mathcal{A}}(u_i\mathsf{R})\big)$ be the $\mathcal{A}$-transition used in u_i by the run $\rho^{\mathcal{A}}$. We distinguish two cases.
  - **a.** In the first case, assume that Separator played $m_i = \vee$ and $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{A}}(a_i)}$. It means that $\Delta^{\mathcal{S}}\big((\ell_i, q_i), a_i\big)$ contains two disjunctive transitions, $\delta^{\mathcal{S}}_{\mathsf{L},i} := (\ell_i, a_i, \ell_{\mathsf{L},i}, \top)$ and $\delta^{\mathcal{S}}_{\mathsf{R},i} := (\ell_i, a_i, \top, \ell_{\mathsf{R},i})$. Let us put $\delta^{\mathcal{S}}_i := \delta^{\mathcal{S}}_{f_i(\delta^{\mathcal{A}}_i),i}$, i.e., the transition that sends a non-$\top$ state in the direction given by $f_i(\delta^{\mathcal{A}}_i)$.
  - **b.** In the second case, Separator played $m_i = \wedge$ and $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$. It means that $\Delta^{\mathcal{S}}\big(\ell_i, a_i\big)$ contains one conjunctive transition $\delta^{\mathcal{S}}_i := (\ell_i, a_i, \ell_{\mathsf{L},i}, \ell_{\mathsf{R},i})$.
- $G_{\mathcal{S}}.[\mathsf{I}\colon d]$ Input plays an arbitrary direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.
- $G.[\mathsf{I}\colon d]$ Input plays the direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

If $m_i = \vee$ and $d_i \neq f_i(\delta^{\mathcal{A}}_i)$ then the next position of the acceptance game $G_{\mathcal{S}}$ is $(u_i d_i, \top)$, which is a winning position for Automaton. Therefore, w.l.o.g. we assume that:

$$\forall i \in \omega.\ (m_i = \vee) \Rightarrow f_i(\delta^{\mathcal{A}}_i) = d_i. \tag{1}$$

Moreover, the new state of $\mathcal{S}$ in $G_{\mathcal{S}}$ is $\ell_{i+1} := \tau(\ell_i, a_i, d_i)$. Similarly, the new memory state of $\mathcal{M}$ in G is ℓ_{i+1}. This concludes the description of the i-th round of both games. Clearly the invariant is preserved. We argue that Automaton wins the resulting infinite play $(\delta_0^{\mathcal{S}}, d_0)(\delta_1^{\mathcal{S}}, d_1)\cdots$ of the acceptance game $G_{\mathcal{S}}$. Consider the infinite play $\pi = (a_0, m_0, f_0, d_0)(a_1, m_1, f_1, d_1)\cdots$ of the separability game G. Since the run $\rho^{\mathcal{A}}$ is accepting, the infinite sequence of $\mathcal{A}$-transitions $\delta_0^{\mathcal{A}}\delta_1^{\mathcal{A}}\cdots$ is accepting. Thus, (1) implies that $\pi \in \mathbf{W}_{\mathcal{A}}$. Therefore, the infinite path $b := (a_0, d_0)(a_1, d_1)\cdots$ belongs to $L_{\mathcal{A}} \subseteq \mathrm{L}(\mathcal{D})$ and thus the corresponding infinite play $(\delta_0^{\mathcal{S}}, d_0)(\delta_1^{\mathcal{S}}, d_1)\cdots$ of the acceptance game $G_{\mathcal{S}}$ is winning for Automaton, as required. This concludes the argument establishing $\mathrm{L}(\mathcal{A}) \subseteq \mathrm{L}(\mathcal{S})$.

It remains to show that $\mathrm{L}(\mathcal{S}) \perp \mathrm{L}(\mathcal{B})$, which is the same as $\mathrm{L}(\mathcal{B}) \subseteq \mathrm{L}(\mathcal{S}^{\mathrm{c}})$ for the complement game automaton. This follows directly from the construction above via the duality of the game G. ◀

Completeness. Assume that there exists a game automaton $\mathcal{S}$ that separates $\mathrm{L}(\mathcal{A})$ from $\mathrm{L}(\mathcal{B})$. We need to show that Separator wins the separability game $G := G^{\mathrm{sep}}_{\mathrm{game}}(\mathcal{A}, \mathcal{B})$. Let $\mathcal{R} := \mathcal{S}^{\mathrm{c}}$ be the syntactic dual of the game automaton $\mathcal{S}$ as in Lemma 2. Thus, the automata $\mathcal{S}$ and $\mathcal{R}$ share the same set of states. Also, their transitions are related: the conjunctive transitions of $\mathcal{S}$ correspond to disjunctive transitions of $\mathcal{R}$ and vice versa. By slightly rephrasing the separation condition, we have $\mathrm{L}(\mathcal{A}) \perp \mathrm{L}(\mathcal{R})$ and $\mathrm{L}(\mathcal{B}) \perp \mathrm{L}(\mathcal{S})$. This means that Pathfinder wins both disjointness games $G^{\mathrm{dis}}(\mathcal{R}, \mathcal{A})$ and $G^{\mathrm{dis}}(\mathcal{S}, \mathcal{B})$. Thus, we can apply Lemma 4 to obtain pathfinders $\mathcal{P}_{\mathcal{A}}\colon \left(\bigcup_{a\in\Sigma} \Delta^{\mathcal{R}}(a) \times \Delta^{\mathcal{A}}(a)\right) \to \{\mathsf{L}, \mathsf{R}\}$ and $\mathcal{P}_{\mathcal{B}}\colon \left(\bigcup_{a\in\Sigma} \Delta^{\mathcal{S}}(a) \times \Delta^{\mathcal{B}}(a)\right) \to \{\mathsf{L}, \mathsf{R}\}$.

We will now provide a strategy of Separator in G. The constructed strategy uses as its memory states the set of states of $\mathcal{S}$ that are distinct than $\top$. Let the initial memory state be q_0. Assume that the current memory state is q_i and consider the i-th round of the game.

[I: a] Input plays an arbitrary letter $a_i \in \Sigma$.

[S: m] Separator plays the mode $m_i \in \{\vee, \wedge\}$ defined as follows. We consider the following two cases for the mode of the transitions $\Delta^{\mathcal{S}}(q_i, a_i)$.

- **a.** If $\Delta^{\mathcal{S}}(q_i, a_i) = \{\delta_i^{\mathcal{S}}\}$ is a single conjunctive transition $\delta_i^{\mathcal{S}} = (q_i, a_i, q_{\mathsf{L},i}, q_{\mathsf{R},i})$ then we put $m_i := \wedge$ and $f_i := \mathcal{P}_{\mathcal{B}}(\delta_i^{\mathcal{S}}, \_)$ is a selector for $\mathcal{B}$.
- **b.** Otherwise, $\Delta^{\mathcal{S}}(q_i, a_i)$ is a pair of disjunctive transitions which means that $\Delta^{\mathcal{R}}(q_i, a_i)$ is a single conjunctive transition $\delta_i^{\mathcal{R}} = (q_i, a_i, q_{\mathsf{L},i}, q_{\mathsf{R},i})$. In this case we put $m_i := \vee$ and $f_i := \mathcal{P}_{\mathcal{A}}(\delta_i^{\mathcal{R}}, \_)$ is a selector for $\mathcal{A}$.

[S: f] Separator plays the selector f_i defined above (notice that f_i is either a selector for $\mathcal{A}$ or for $\mathcal{B}$, according to m_i).

[I: d] Input plays an arbitrary direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

The next memory state of our strategy is the state $q_{d_i,i}$ taken from one of the transitions $\delta_i^{\mathcal{S}}$ or $\delta_i^{\mathcal{R}}$, see above. We now argue that Separator wins the corresponding infinite play $\pi = (a_0, m_0, f_0, d_0)(a_1, m_1, f_1, d_1)\cdots$. Let $b = (a_0, d_0)(a_1, d_1)\cdots$ be the corresponding path. Consider a number $i \in \omega$. By the construction of the strategy above, we have two cases:

1. If $m_i = \wedge$, then a conjunctive transition $\delta_i^{\mathcal{S}} = (q_i, a_i, q_{\mathsf{L},i}, q_{\mathsf{R},i})$ of $\mathcal{S}$ was used to determine f_i. In this case, define $\delta_i^{\mathcal{R}}$ as the following disjunctive transition of $\mathcal{R}$: if $d_i = \mathsf{L}$ then $\delta_i^{\mathcal{R}} := (q_i, a_i, q_{\mathsf{L},i}, \top)$, otherwise $d_i = \mathsf{R}$ and $\delta_i^{\mathcal{R}} := (q_i, a_i, \top, q_{\mathsf{R},i})$.
2. If $m_i = \vee$, then a conjunctive transition $\delta_i^{\mathcal{R}} = (q_i, a_i, q_{\mathsf{L},i}, q_{\mathsf{R},i})$ of $\mathcal{R}$ was used to determine f_i. In this case, define $\delta_i^{\mathcal{S}}$ as the following disjunctive transition of $\mathcal{S}$: if $d_i = \mathsf{L}$ then $\delta_i^{\mathcal{S}} := (q_i, a_i, q_{\mathsf{L},i}, \top)$, otherwise $d_i = \mathsf{R}$ and $\delta_i^{\mathcal{S}} := (q_i, a_i, \top, q_{\mathsf{R},i})$.

The definitions above provide two sequences of transitions $\vec{\delta}^{\mathcal{S}} := \delta_0^{\mathcal{S}}\delta_1^{\mathcal{S}}\cdots \in \Delta^{\mathcal{S}}(b)$, $\vec{\delta}^{\mathcal{R}} := \delta_0^{\mathcal{R}}\delta_1^{\mathcal{R}}\cdots \in \Delta^{\mathcal{R}}(b)$. Since for every $i \in \omega$ the transitions $\delta_i^{\mathcal{S}}$ and $\delta_i^{\mathcal{R}}$ are from the same state $q_i \neq \top$, $\vec{\delta}^{\mathcal{S}}$ is accepting in $\mathcal{S}$ if, and only if, $\vec{\delta}^{\mathcal{R}}$ is rejecting in $\mathcal{R}$. Assume that $\vec{\delta}^{\mathcal{S}}$ is accepting (the other case is analogous). We will show that $\mathbf{W}_{\mathcal{B}}$ is violated (if $\vec{\delta}^{\mathcal{R}}$ is accepting then $\mathbf{W}_{\mathcal{A}}$ is violated). Assume for the sake of contradiction that $\mathbf{W}_{\mathcal{B}}$ holds, as witnessed by a sequence of $\mathcal{B}$-transitions $\vec{\delta}^{\mathcal{B}} = \delta_0^{\mathcal{B}}\delta_1^{\mathcal{B}}\cdots \in \Delta^{\mathcal{B}}(b)$. By Remark 6 we obtain that whenever $m_i = \vee$ and $\delta_i^{\mathcal{S}}$ is a disjunctive transition of $\mathcal{S}$ then $\mathcal{P}_{\mathcal{B}}(\delta_i^{\mathcal{S}}, \_)$ is constantly equal to d_i. By the assumption on $\vec{\delta}^{\mathcal{B}}$ from $\mathbf{W}_{\mathcal{B}}$ we know that whenever $m_i = \wedge$ then $f_i(\delta_i^{\mathcal{B}}) = d_i$. However, if $m_i = \wedge$ then $f_i(\delta_i^{\mathcal{B}}) = \mathcal{P}_{\mathcal{B}}(\delta_i^{\mathcal{S}}, \delta_i^{\mathcal{B}})$. Therefore, in both cases we know that $\mathcal{P}_{\mathcal{B}}(\delta_i^{\mathcal{S}}, \delta_i^{\mathcal{B}}) = d_i$. This means that the assumptions of Corollary 5 are met and at least one of the sequences $\vec{\delta}^{\mathcal{S}}$, $\vec{\delta}^{\mathcal{B}}$ is rejecting – a contradiction, since we assumed both these sequences to be accepting. ◀

6 Separability by game automata with priorities in C

In this section we present our last game-theoretic characterisation, namely game automata separability for a fixed finite set $C \subseteq \mathbb{N}$ of priorities. Fix two automata $\mathcal{A} = (\Sigma, Q^{\mathcal{A}}, q_0^{\mathcal{A}}, \Omega^{\mathcal{A}}, \Delta^{\mathcal{A}})$ and $\mathcal{B} = (\Sigma, Q^{\mathcal{B}}, q_0^{\mathcal{B}}, \Omega^{\mathcal{B}}, \Delta^{\mathcal{B}})$ over the same alphabet Σ. The game is a variation of $G^{\mathrm{sep}}_{\mathrm{game}}(\mathcal{A}, \mathcal{B})$ from Section 5 where Separator additionally plays priorities from C.

C-game-automata separability game $G^{\mathrm{sep}}_{\mathrm{game}}(\mathcal{A}, \mathcal{B}, C)$

At the i-th round:

- [S: c] Separator plays a priority $c_i \in C$.
- [I: a] Input plays a letter $a_i \in \Sigma$.
- [S: m] Separator plays a mode $m_i \in \{\vee, \wedge\}$.
- [S: f] Separator plays either
  - **a.** a selector $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{A}}(a_i)}$ for $\mathcal{A}$ if $m_i = \vee$, or
  - **b.** a selector $f_i \in \{\mathsf{L}, \mathsf{R}\}^{\Delta^{\mathcal{B}}(a_i)}$ for $\mathcal{B}$ if $m_i = \wedge$.
- [I: d] Input plays a direction $d_i \in \{\mathsf{L}, \mathsf{R}\}$.

Separator wins an infinite play $\pi = (c_0, a_0, m_0, f_0, d_0)(c_1, a_1, m_1, f_1, d_1)\cdots$ inducing a path $b = (a_0, d_0)(a_1, d_1)\cdots$ whenever both conditions below hold:

1. $\pi \in \mathbf{W}_{\mathcal{A}}$: If there exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{A}} = \delta_0^{\mathcal{A}}\delta_1^{\mathcal{A}}\cdots \in \Delta^{\mathcal{A}}(b)$ s.t. for all $i \in \omega$ we have $(m_i = \vee) \Rightarrow f_i(\delta_i^{\mathcal{A}}) = d_i$, then $c_0c_1\cdots$ is accepting.
2. $\pi \in \mathbf{W}_{\mathcal{B}}$: If there exists an accepting sequence of transitions $\vec{\delta}^{\mathcal{B}} = \delta_0^{\mathcal{B}}\delta_1^{\mathcal{B}}\cdots \in \Delta^{\mathcal{B}}(b)$ s.t. for all $i \in \omega$ we have $(m_i = \wedge) \Rightarrow f_i(\delta_i^{\mathcal{B}}) = d_i$, then $c_0c_1\cdots$ is rejecting.

▶ **Lemma 12.** *Separator wins $G^{\mathrm{sep}}_{\mathrm{game}}(\mathcal{A}, \mathcal{B}, C)$ if, and only if, there exists a game automaton $\mathcal{S}$ with priorities in C separating $\mathrm{L}(\mathcal{A})$, $\mathrm{L}(\mathcal{B})$.*

This lemma can be proved similarly as Lemma 10 except for the acceptance condition of the separator which is given by the priorities c_i's as in the proof of Lemma 7.

References

1 David Barozzini, Lorenzo Clemente, Thomas Colcombet, and Paweł Parys. Cost Automata, Safe Schemes, and Downward Closures. In *Proc. of ICALP'20*, LIPIcs, pages 109:1–109:18, 2020. doi:10.4230/LIPIcs.ICALP.2020.109.

2 Achim Blumensath. Recognisability for algebras of infinite trees. *Theoretical Computer Science*, 412(29):3463–3486, 2011.

3 Mikołaj Bojańczyk. Star height via games. In *LICS*, pages 214–219, 2015.

4 Mikołaj Bojańczyk, Filippo Cavallari, Thomas Place, and Michał Skrzypczak. Regular tree languages in low levels of the Wadge Hierarchy. *Log. Meth. Comput. Sci.*, Volume 15, Issue 3, 2019. URL: `https://lmcs.episciences.org/5743`.

5 Mikołaj Bojańczyk and Wojciech Czerwiński. An automata toolbox, February 2018. URL: `https://www.mimuw.edu.pl/~bojan/paper/automata-toolbox-book`.

6 Mikołaj Bojańczyk and Thomas Place. Regular languages of infinite trees that are boolean combinations of open sets. In *Proc. of ICALP'12*, ICALP'12, pages 104—-115, Berlin, Heidelberg, 2012. Springer-Verlag. `doi:10.1007/978-3-642-31585-5_13`.

7 J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969. URL: `http://www.jstor.org/stable/1994916`.

8 Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *Proc. of STOC'17*, 2017.

9 Filippo Cavallari, Henryk Michalewski, and Michał Skrzypczak. A characterisation of Π^0_2 regular tree languages. In *Proc. of MFCS'17*, 2017.

10 Lorenzo Clemente, Wojciech Czerwiński, Sławomir Lasota, and Charles Paperman. Regular separability of parikh automata. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *Proc. of ICALP'17*, volume 80, pages 117:1–117:13, 2017. `doi:10.4230/LIPIcs.ICALP.2017.117`.

11 Lorenzo Clemente, Wojciech Czerwiński, Sławomir Lasota, and Charles Paperman. Separability of Reachability Sets of Vector Addition Systems. In *Proc. of STACS'17*, volume 66 of *LIPICs*, pages 24:1–24:14, 2017. `doi:10.4230/LIPIcs.STACS.2017.24`.

12 Lorenzo Clemente, Sławomir Lasota, and Radosław Piórkowski. Determinisability of One-Clock Timed Automata. In *Proc. of CONCUR'20*, volume 171 of *LIPIcs*, pages 42:1–42:17, 2020. `doi:10.4230/LIPIcs.CONCUR.2020.42`.

13 Lorenzo Clemente, Sławomir Lasota, and Radosław Piórkowski. Timed Games and Deterministic Separability. In *Proc. of ICALP'20*, volume 168 of *LIPIcs*, pages 121:1–121:16, 2020. `doi:10.4230/LIPIcs.ICALP.2020.121`.

14 Lorenzo Clemente, Paweł Parys, Sylvain Salvati, and Igor Walukiewicz. The diagonal problem for higher-order recursion schemes is decidable. In *Proc. of LICS'16*, 2016. `doi:10.1145/2933575.2934527`.

15 Lorenzo Clemente and Michał Skrzypczak. Deterministic and game separability for regular languages of infinite trees, May 2021. `arXiv:2105.01137`.

16 Thomas Colcombet. Fonctions régulières de coût. Habilitation thesis, Université Paris Diderot—Paris 7, 2013.

17 Thomas Colcombet, Denis Kuperberg, Christof Löding, and Michael Vanden Boom. Deciding the weak definability of Büchi definable tree languages. In *Proc. of CSL'13*, LIPIcs, pages 215–230, 2013. `doi:10.4230/LIPIcs.CSL.2013.215`.

18 Thomas Colcombet and Christof Löding. The Non-deterministic Mostowski Hierarchy and Distance-Parity Automata. In *Proc. of ICALP'08*, pages 398–409, 2008. `doi:10.1007/978-3-540-70583-3_33`.

19 William Craig. Three uses of the herbrand-gentzen theorem in relating model theory and proof theory. *The Journal of Symbolic Logic*, 22(3):269–285, 1957. URL: `http://www.jstor.org/stable/2963594`.

20 Wojciech Czerwiński, Laure Daviaud, Nathanaël Fijalkow, Marcin Jurdziński, Ranko Lazić, and Paweł Parys. Universal trees grow inside separating automata: Quasi-polynomial lower bounds for parity games. In *Proc. of SODA'19*, pages 2333–2349, USA, 2019. Society for Industrial and Applied Mathematics.

21 Wojciech Czerwiński and Sławomir Lasota. Regular Separability of One Counter Automata. *Logical Methods in Computer Science*, Volume 15, Issue 2, June 2019. URL: `https://lmcs.episciences.org/5563`.

22 Wojciech Czerwiński, Sławomir Lasota, Roland Meyer, Sebastian Muskalla, K. Narayan Kumar, and Prakash Saivasan. Regular Separability of Well-Structured Transition Systems. In *Proc. of CONCUR'18*, LIPIcs, pages 35:1–35:18, 2018. doi:10.4230/LIPIcs.CONCUR.2018.35.

23 Wojciech Czerwiński, Wim Martens, Lorijn van Rooijen, and Marc Zeitoun. A note on decidable separability by piecewise testable languages. In *Proc. of FCT'15*, 2015. doi:10.1007/978-3-319-22177-9_14.

24 Wojciech Czerwiński and Georg Zetzsche. An approach to regular separability in vector addition systems. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 341–354. ACM, 2020. doi:10.1145/3373718.3394776.

25 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. of SFCS'91*, pages 368–377. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185392.

26 Alessandro Facchini, Filip Murlak, and Michał Skrzypczak. Index problems for game automata. *ACM Trans. Comput. Logic*, 17(4):24:1–24:38, November 2016. doi:10.1145/2946800.

27 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games - A Guide to Current Research.* Springer, 2002.

28 Alexander Kechris. *Classical Descriptive Set Theory.* Springer-Verlag, 1995.

29 Ralf Küsters and Thomas Wilke. Deciding the first level of the μ-calculus alternation hierarchy. In *Proc. of FSTTCS'02*, pages 241–252, Berlin, 2002.

30 Robert McNaughton and Seymour Papert. *Counter-free automata.* M.I.T. Press research monographs. M.I.T. Press, 1971.

31 Filip Murlak. Weak index versus Borel rank. In *Proc. of STACS'08*, volume 1 of *LIPIcs*, pages 573–584, 2008. doi:10.4230/LIPIcs.STACS.2008.1318.

32 Damian Niwiński. On fixed-point clones (extended abstract). In *Proc. of ICALP'86*, pages 464–473, 1986. URL: http://dl.acm.org/citation.cfm?id=646240.683678.

33 Damian Niwiński and Igor Walukiewicz. Relating hierarchies of word and tree automata. In *Proc. of STACS'98*, pages 320–331, 1998.

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

35 Damian Niwiński and Igor Walukiewicz. Deciding nondeterministic hierarchy of deterministic tree automata. *Electronic Notes in Theoretical Computer Science*, 123:195–208, 2005. Proceedings of the 11th Workshop on Logic, Language, Information and Computation (WoLLIC 2004). doi:10.1016/j.entcs.2004.05.015.

36 Thomas Place and Marc Zeitoun. Separating regular languages with first-order logic. *Log. Methods Comput. Sci.*, 12(1), 2016. doi:10.2168/LMCS-12(1:5)2016.

37 Thomas Place and Marc Zeitoun. Adding successor: A transfer theorem for separation and covering. *ACM Trans. Comput. Logic*, 21(2), 2019. doi:10.1145/3356339.

38 Michael O. Rabin. Weakly definable relations and special automata. In *Mathematical Logic and Foundations of Set Theory*, volume 59 of *Studies in Logic and the Foundations of Mathematics*, pages 1–23. Elsevier, 1970. doi:10.1016/S0049-237X(08)71929-3.

39 Luigi Santocanale and André Arnold. Ambiguous classes in mu-calculi hierarchies. *Theoretical Computer Science*, 333(1):265–296, 2005. Foundations of Software Science and Computation Structures. doi:10.1016/j.tcs.2004.10.024.

40 Marcel Paul Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965.

41 Michał Skrzypczak and Igor Walukiewicz. Deciding the Topological Complexity of Büchi Languages. In *Proc. of ICALP'16*, volume 55 of *LIPIcs*, pages 99:1–99:13, 2016. doi:10.4230/LIPIcs.ICALP.2016.99.

42 Tomasz Fryderyk Urbanski. On Deciding if Deterministic Rabin Language Is in Büchi Class. In *Proc. of ICALP'00*, volume 1853 of *LNCS*, pages 663–674. Springer, 2000. doi:10.1007/3-540-45022-X_56.

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

44 Igor Walukiewicz. Deciding low levels of tree-automata hierarchy. *ENTCS*, 67:61–75, 2002. doi:10.1016/S1571-0661(04)80541-3.

45 Georg Zetzsche. An approach to computing downward closures. In *Proc. of ICALP'15*, volume 9135 of *LNCS*, pages 440–451, 2015. doi:10.1007/978-3-662-47666-6_35.