$\Pi_2 \cap \Sigma_2 \equiv AFMC$

Orna Kupferman[1]* and Moshe Y. Vardi[2]**

[1] Hebrew University, School of Engineering and Computer Science, Jerusalem 91904, Israel
Email: orna@cs.huji.ac.il, URL: http://www.cs.huji.ac.il/~orna

[2] Rice University, Department of Computer Science, Houston, TX 77251-1892, U.S.A.
Email:vardi@cs.rice.edu, URL: http://www.cs.rice.edu/~vardi

Abstract. The μ-*calculus* is an expressive specification language in which modal logic is extended with fixpoint operators, subsuming many dynamic, temporal, and description logics. Formulas of μ-calculus are classified according to their *alternation depth*, which is the maximal length of a chain of nested alternating least and greatest fixpoint operators. Alternation depth is the major factor in the complexity of μ-calculus model-checking algorithms. A refined classification of μ-calculus formulas distinguishes between formulas in which the outermost fixpoint operator in the nested chain is a least fixpoint operator (Σ_i formulas, where i is the alternation depth) and formulas where it is a greatest fixpoint operator (Π_i formulas). The *alternation-free* μ-*calculus* (AFMC) consists of μ-calculus formulas with no alternation between least and greatest fixpoint operators. Thus, AFMC is a natural closure of $\Sigma_1 \cup \Pi_1$, which is contained in both Σ_2 and Π_2. In this work we show that $\Sigma_2 \cap \Pi_2 \equiv$ AFMC. In other words, if we can express a property ξ both as a least fixpoint nested inside a greatest fixpoint and as a greatest fixpoint nested inside a least fixpoint, then we can express ξ also with no alternation between greatest and least fixpoints. Our result refers to μ-calculus over arbitrary Kripke structures. A similar result, for directed μ-calculus formulas interpreted over trees with a fixed finite branching degree, follows from results by Arnold and Niwinski. Their proofs there cannot be easily extended to Kripke structures, and our extension involves *symmetric nondeterministic Büchi* tree automata, and new constructions for them.

1 Introduction

The μ-*calculus* is an expressive specification language in which formulas are built from Boolean operators, existential ($\Diamond$) and universal ($\Box$) next-time modalities, and least (μ) and greatest (ν) fixpoint operators [Koz83]. The discovery and use of *symbolic model-checking* methods [McM93] for verification of large systems has made the μ-calculus important also from a practical point of view: symbolic model-checking tools proceed by computing fixpoint expressions over the model's set of states. For example, to find the set of states from which a state satisfying some predicate p is reachable, the model checker starts with the set S of states in which p holds, and repeatedly add to S the set $\Diamond S$ of states that have a successor in S. Formally, the model checker calculates the set of states that satisfy the μ-calculus formula $\mu y.p \vee \Diamond y$.

Formulas of μ-calculus are classified according to their *alternation depth*, which is the maximal length of a chain of nested alternating least and greatest fixpoint operators. From a practical point of view, the classification is important, as the alternation depth is the major factor in the complexity of μ-calculus model-checking algorithms: the original algorithm for model checking a structure of size m with respect to a formula of length n and alternation depth d requires time $O(mn)^d$ [EL86], and more sophisticated algorithms can do the job in time roughly $O(mn)^{\lfloor \frac{d}{2} \rfloor + 1}$ [Jur00]. From a theoretical point of view, the classification naturally raises questions about the expressive power of the classes. In particular, the question whether the expressiveness hierarchy for the μ-calculus collapses (i.e., whether there is some $d \geq 1$ such

* Supported in part by by NSF grant CCR-9988172 and by a research grant from the Center for Pure and Applied Mathematics at the University of California, Berkeley

** Supported in part by NSF grants CCR-9988322, CCR-0124077, IIS-9908435, IIS-9978135, and EIA-0086264, by BSF grant 9800096, and by a grant from the Intel Corporation.

that all μ-calculus formulas can be translated to formulas of alternation depth d) has been answered to the negative [Bra98]. The alternation-depth hierarchy of μ-calculus and the model-checking problem for the various classes in the hierarchy are strongly related to the index hierarchy in *parity games* and to the problem of deciding such games [Jur00].

A more refined classification of μ-calculus formulas distinguishes between formulas in which the outermost fixpoint operator in the nested chain is a least fixpoint operator (Σ_i formulas, where i is the alternation depth) and formulas where it is a greatest fixpoint operator (Π_i formulas). For example, the formula $\mu y.p \vee \Diamond y$ is a Σ_1 formula, as it has alternating depth 1 and its outermost fixpoint operator is μ. Similarly, the formula $\nu y.\mu z.\Box[(p \wedge y) \vee z]$ is a Π_2 formula[1]. By duality of the least and greatest fixpoint operators, the classes Π_i and Σ_i are complementary, in the sense that a formula ψ is in Π_i iff the formula $\neg\psi$ (in positive normal form, where negation is applied to atomic propositions only) is in Σ_i.

Some fragments of μ-calculus are of special interest in computer science: *Modal Logic* (ML) consists of μ-calculus formulas with no fixpoint operators (that is, $\text{ML} = \Sigma_0 \cup \Pi_0$). It is actually more correct to say that μ-calculus is the extension of ML with fixpoint operators. Extending ML with fixpoint operators still retain some of its basic semantic properties, in particular the property of being invariant under bisimulation [Ben91]. The *alternation-free μ-calculus* (AFMC) consists of μ-calculus formulas with no alternation between least and greatest fixpoint operators. Thus, AFMC is a natural closure of $\Sigma_1 \cup \Pi_1$, which is contained in both Σ_2 and Π_2. AFMC subsumes the branching temporal logic CTL and the dynamic logic PDL [FL79]. Formulas of AFMC can be symbolically evaluated in time linear in the structure [CS91,KVW00]. While designers may prefer to use higher-level logics to specify properties, model-checking tools often proceed by evaluating the corresponding AFMC formulas [BRS99]. Finally, it is hard to produce an understandable formula with more than one alternation. Thus, $\Pi_2 \cup \Sigma_2$ subsumes almost all formulas one may wish to specify in practice. Formally, $\Pi_2 \cup \Sigma_2$ subsumes the branching temporal logic CTL*, and in fact, until [Bra98], the strictness of the expressiveness hierarchy of μ-calculus was known only for Π_i and Σ_i with $i \leq 2$ [AN90]. Also, the symbolic evaluation of linear properties is reduced to calculating a Π_2 formula [VW86,EL85].

For several hierarchies in computer science, even strict ones, it is possible to show local *coalescence*, where membership in some class of the hierarchy and in its complementary class implies membership in a lower class. For example, RE $\cap$ co-RE = Rec describes coalescence at the bottom of the arithmetical hierarchy [Rog67]. On the other hand, the analogous coalescence for the polynomial hierarchy is not known; it is a major open question whether NP $\cap$ co-NP = P [GJ79]. In [KV01], we showed that the bottom levels of the μ-calculus expressiveness hierarchy coalesce: $\Sigma_1 \cap \Pi_1 \equiv ML$. In other words, if we can express a property ξ both as a least fixpoint and as a greatest fixpoint, then we can express ξ without fixpoints. The proof uses the fact that μ-calculus formulas in $\Sigma_1 \cap \Pi_1$ correspond to languages that are both safety and co-safety. Consequently, for every property $\xi \in \Sigma_1 \cap \Pi_1$, we can construct two nondeterministic *looping* tree automata $\mathcal{U}$ and $\mathcal{U}'$ such that $\mathcal{U}$ and $\mathcal{U}'$ accept exactly all the trees that satisfy ξ and its complement, respectively (the fact that $\mathcal{U}$ and $\mathcal{U}'$ are looping means that they have trivial acceptance conditions – every infinite run is accepting). We showed in [KV01] how $\mathcal{U}$ and $\mathcal{U}'$ can be combined to a *cycle-free* automaton and then translate to an ML formula expressing ξ.

In this paper we show coalescence in higher classes of the hierarchy, namely $\Sigma_2 \cap \Pi_2 \equiv$ AFMC.[2] In other words, if we can specify a property ξ both as a least fixpoint nested inside a greatest fixpoint and as a greatest fixpoint nested inside a least fixpoint, then we can express ξ also with no alternation between greatest and least fixpoints. Unfortunately, the technique of [KV01] is too weak to be helpful here. Indeed, formulas in Π_2 cannot be expressed by looping automata. As we explain below, the known automata-theoretic characterizations of Σ_2 and Π_2, and their relation to AFMC, cannot help us either.

[1] An exact definition of the classes Σ_i and Π_i refers to the scope of the fixpoint operators. As we discuss in Section 4, several different definitions are studied in the literature, and we follow here the definition of [Niw86].

[2] The analogous complexity-theoretic result would be $\Sigma_2^p \cap \Pi_2^p = \mathrm{P}^{\mathrm{NP}}$, where Σ_2^p and Π_2^p form the second level of the polynomial hierarchy and P^{NP} is the polynomial closure of NP [GJ79].

One such known characterization [Niw86,AN92] refers to the expressive power of the μ-calculus over trees with fixed finite branching degrees. Over such trees, the existential next-time modality of the μ-calculus can be parameterized with *directions*. A modality parameterized with direction d means that the corresponding existential requirement should be satisfied in the d-th child of the current state. For example, for a binary tree in which each node has a left child and a right child, the formula $\Diamond_{\mathbf{l}}\, p$ means that the left child of the root satisfies p, and the formula $\mu y.p \vee \Diamond_{\mathbf{r}} y$ means that some node in the rightmost path of the tree satisfies p. The ability of *directed μ-calculus* to distinguish between the various children of a node makes it convenient to translate formulas to tree automata and vice versa. In particular, it is known that directed-Π_2 is as expressive as *nondeterministic Büchi tree automata* [AN90,Kai95]. Our interest in this paper is in the expressive power of the μ-calculus over arbitrary *Kripke structures*, possibly with an infinite branching degree, which means that we cannot restrict attention to trees of fixed branching degrees.

An automata-theoretic framework for μ-calculus without directions is suggested in [JW95], by means of *μ-automata*, which are essentially symmetric alternating tree automata in a certain normal form. A related approach, in which alternation is more explicit, is presented in [Wil99]. Alternation allows the automaton to send several requirements to the same child. Symmetry means that the automaton does not distinguish between the different children of a node, and it sends copies to child nodes only in either a universal or an existential manner. It also means that the automaton can handle trees with a variable and even infinite branching degree. Formulas of μ-calculus in Π_i and Σ_i can be linearly translated to symmetric alternating parity/co-parity automata of index i. While it is possible to translate μ-calculus formulas to symmetric alternating automata, it is not immediately clear how such a translation can help in a translating of $\Sigma_2 \cap \Pi_2$ into the AFMC. By [AN92,KV99], formulas that are members of both directed-Π_2 and directed-Σ_2 can be translated to directed-AFMC. The proofs in [AN92,KV99] shows that given a formula $\psi \in \Sigma_2 \cap \Pi_2$, we can construct two nondeterministic Büchi tree automata $\mathcal{U}$ and $\mathcal{U}'$, for ψ and $\neg\psi$, and then combine the automata to a weak alternating automaton equivalent to ψ. The combination of $\mathcal{U}$ and $\mathcal{U}'$, however, crucially depends on the fact that the automata are nondeterministic (rather than alternating) and the fact that the automata can refer to particular directions in the tree.

The key to the results in [KV01] and here is a development of a theory of *symmetric nondeterministic* tree automata. In [KV01], we defined symmetric nondeterministic *looping* automata, and showed how to construct such automata for formulas in Π_1. In order to handle Σ_2 and Π_2, we define here symmetric nondeterministic *Büchi* automata, and translate Π_2 formulas to such automata. From a technical point of view, symmetric nondeterministic tree automata are essentially symmetric alternating automata with transitions in disjunctive normal form. Our main contribution is the development of various constructions for symmetric nondeterministic tree automata and their application to the study of the expressive power of the μ-calculus. Since removal of alternation in Büchi automata should take into an account the acceptance condition of the automaton and keep track of the states visited in each path of the run tree, the symmetry of the automaton poses real technical challenges. We then extend the construction in [KV99] to symmetric automata and combine the symmetric nondeterministic Büchi tree automata for ψ and $\neg\psi$ to a symmetric weak alternating automaton for ψ. Again, symmetry poses real technical challenges. (In fact, while the construction in [KV99] for the directed case is quadratic, here we end up with quadratically many states but exponentially many transitions.) Once we have a weak symmetric alternating automaton for ψ, it is possible to generate from it an equivalent AFMC formula [KV98].

2 Preliminaries

For a set $D \subseteq \mathbb{N}$ of directions, a *D-tree* is a nonempty set $T \subseteq D^*$, where for every $x \cdot d \in T$ with $x \in D^*$ and $d \in D$, we have $x \in T$. The elements of T are called *nodes*, and the empty word ε is the *root* of T. For every $x \in T$, the nodes $x \cdot d$, for $d \in D$, are the *children* of x. A node with no children is a *leaf*. The *degree* of a node x is the number of children x has. Note that the degree of x is bounded by $|D|$. For technical convenience, we assume that the set D is finite[3]. A D-tree is *leafless* if it has no leafs. Note that a leafless tree is infinite. A *path* π of a tree T is a set $\pi \subseteq T$ such that $\varepsilon \in \pi$ and for every $x \in \pi$, either x is a leaf or exactly one child of x is in π. For two nodes x_1 and x_2 of T, we say that $x_1 \leq x_2$ iff x_1 is a prefix of x_2; i.e., there exists $z \in D^*$ such that $x_2 = x_1 \cdot z$. We say that $x_1 < x_2$ iff $x_1 \leq x_2$ and $x_1 \neq x_2$. A *frontier* of a leafless tree is a set $E \subset T$ of nodes such that for every path $\pi \subseteq T$, we have $|\pi \cap E| = 1$. For example, the set $E = \{0, 100, 101, 11\}$ is a frontier of the $\{0,1\}$-tree $\{0,1\}^*$. For two frontiers E_1 and E_2, we say that $E_1 \leq E_2$ iff for every node $x_2 \in E_2$, there exists a node $x_1 \in E_1$ such that $x_1 \leq x_2$. We say that $E_1 < E_2$ iff for every node $x_2 \in E_2$, there exists a node $x_1 \in E_1$ such that $x_1 < x_2$. Note that while $E_1 < E_2$ implies that $E_1 \leq E_2$ and $E_1 \neq E_2$, the other direction does not necessarily hold. Given an alphabet Σ, a *Σ-labeled D-tree* is a pair $\langle T, V \rangle$ where T is a D-tree and $V : T \rightarrow \Sigma$ maps each node of T to a letter in Σ. We extend V to paths in a straightforward way. For a Σ-labeled D-tree $\langle T, V \rangle$ and a set $A \subseteq \Sigma$, we say that E is an A-frontier iff E is a frontier and for every node $x \in E$, we have $V(x) \in A$. We denote by $trees(D, \Sigma)$ the set of all Σ-labeled D-trees, and denote by $trees(\Sigma)$ the set of all Σ-labeled D-trees, for some D. For a set $\mathcal{T} \subseteq trees(\Sigma)$, we denote by $comp(\mathcal{T})$ the set of Σ-labeled trees that are not in $\mathcal{T}$; thus $comp(\mathcal{T}) = trees(\Sigma) \setminus \mathcal{T}$.

Automata on infinite trees (tree automata, for short) run on leafless Σ-labeled trees. *Alternating tree automata* generalize nondeterministic tree automata and were first introduced in [MS87]. *Symmetric* alternating tree automata [JW95,Wil99] are capable of reading trees with variable branching degrees. When a symmetric automaton reads a node of the input tree it sends copies to all successors of that node or to some successor. Formally, for a given set X, let $\mathcal{B}^+(X)$ be the set of positive Boolean formulas over X. For a set $Y \subseteq X$ and a formula $\theta \in \mathcal{B}^+(X)$, we say that Y *satisfies* θ iff assigning $\mathbf{true}$ to elements in Y and assigning $\mathbf{false}$ to elements in $X \setminus Y$ satisfies θ. A symmetric alterating Büchi tree automaton (symmetric ABT, for short) is a tuple $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$ where Σ is the input alphabet, Q is a finite set of states, $\delta : Q \times \Sigma \rightarrow \mathcal{B}^+(\{\Box, \Diamond\} \times Q)$ is a transition function, $q_0 \in Q$ is an initial state, and $F \subseteq Q$ is a Büchi acceptance condition. Intuitively, an atom $\langle \Box, q \rangle$ in $\delta(q, \sigma)$ denotes a universal requirement to send a copy of the automaton in state q to all the children of the current node. An atom $\langle \Diamond, q \rangle$ denotes an existential requirement to send a copy of the automaton in state q to some child of the current node. When, for instance, the automaton is in state q, reads a node x with k children $x \cdot 1, \ldots, x \cdot k$, and $\delta(q, V(x)) = (\Box, q_1) \wedge (\Diamond, q_2) \vee (\Diamond, q_3) \wedge (\Diamond, q_4)$, it can either send k copies in state q_1 to the nodes $x \cdot 1, \ldots, x \cdot k$ and send a copy in state q_2 to some node in $x \cdot 1, \ldots, x \cdot k$ or send one copy in state q_3 to some node in $x \cdot 1, \ldots, x \cdot k$ and send one copy in state q_4 to some node in $x \cdot 1, \ldots, x \cdot k$. So, while nondeterministic tree automata send exactly one copy to each child, symmetric alternating automata can send several copies to the same child. On the other hand, symmetric alternating automata cannot distinguish between the different successors and can send copies to child nodes only in either a universal or an existential manner. Formally, a *run* of $\mathcal{A}$ on an input Σ-labeled D-tree $\langle T, V \rangle$, for some set D of directions, is an $(D^* \times Q)$-labeled $\mathbb{N}$-tree $\langle T_r, r \rangle$ such that $\varepsilon \in T_r$ and $r(\varepsilon) = (\varepsilon, q_0)$, and for all $y \in T_r$ with $r(y) = (x, q)$ and $\delta(q, V(x)) = \theta$, there is a (possibly empty) set $S \subseteq \{\Box, \Diamond\} \times Q$, such that S satisfies θ, and for all $(c, s) \in S$, the following hold: (1) If $c = \Box$, then for each $d \in D$, there is $j \in \mathbb{N}$ such that $y \cdot j \in T_r$ and $r(y \cdot j) = (x \cdot d, s)$. (2) If $c = \Diamond$, then for some $d \in D$, there is $j \in \mathbb{N}$ such that $y \cdot j \in T_r$ and $r(y \cdot j) = (x \cdot d, s)$. Note that if $\theta = \mathbf{true}$, then y need not have children. This is the reason why T_r may have leaves. Also, since there exists no set S as required for $\theta = \mathbf{false}$, we cannot have a run that takes a transition with $\theta = \mathbf{false}$. For a run $\langle T_r, r \rangle$ and an infinite path $\pi \subseteq T_r$, we define $inf(\pi)$ to be the set of states that are visited infinitely often in π, thus $q \in inf(\pi)$ if and only if there are infinitely many $y \in \pi$ for which $r(y) \in T \times \{q\}$. A run $\langle T_r, r \rangle$ is accepting if all its infinite paths satisfy the Büchi acceptance condition; thus $inf(\pi) \cap F \neq \emptyset$. A tree $\langle T, V \rangle$ is accepted by $\mathcal{A}$ iff there exists an accepting run of $\mathcal{A}$ on

[3] As we detail in the proof of Theorem 6, due to the bounded-tree-model property for μ-calculus, this technical assumption does not prevent us from proving our main result also for general structures with an infinite branching degree.

$\langle T, V\rangle$, in which case $\langle T, V\rangle$ belongs to $\mathcal{L}(\mathcal{A})$. A tree $\langle T, V\rangle$ is accepted by $\mathcal{U}$ iff there exists an accepting run of $\mathcal{A}$ on $\langle T, V\rangle$, in which case $\langle T, V\rangle$ belongs to the language, $\mathcal{L}(\mathcal{A})$, of $\mathcal{A}$.

The transition function of an ABT $\mathcal{A}$ induces a graph $G_{\mathcal{A}} = \langle Q, E\rangle$ where $E(q, q')$ if there is $\sigma \in \Sigma$ such that $(\Box, q')$ or $(\Diamond, q')$ appears in $\delta(q, \sigma)$. An ABT is a *weak alternating tree automaton* (AWT, for short) if for each strongly connected component $C \subseteq Q$ of $G_{\mathcal{A}}$, either $C \subseteq F$ or $C \cap F = \emptyset$ [MSS86]. Note that every infinite path of a run of an AWT ultimately gets "trapped" within some strongly connected component C of $G_{\mathcal{A}}$. The path then satisfies the acceptance condition if and only if $C \subseteq F$.

The symmetry condition can also be applied to nondeterministic tree automata. In a *symmetric nondeterministic Büchi tree automaton* (symmetric NBT, for short) $\mathcal{U} = \langle \Sigma, Q, \delta, q_0, F\rangle$, the state space is $Q = 2^S$ for some set S of *micro-states*, and the transition function $\delta : Q \times \Sigma \to 2^{2^S \times 2^S}$ maps a state and a letter to sets of pairs $\langle U, E\rangle$ of subsets of S. The set $U \subseteq S$ is the *universal set* and it describes the micro-states that should be members in all the child states. The set $E \subseteq S$ is the *existential set* and it describes micro-states each of which has to be a member in at least one child state. Formally, given $k \geq 1$, a k-tuple $\langle S_1, \ldots, S_k\rangle$ is induced by $\delta(q, \sigma)$ if there is $\langle U, E\rangle$ in $\delta(q, \sigma)$ such that for all $1 \leq i \leq k$ we have $U \subseteq S_i$, and for all $s \in E$ there is $1 \leq i \leq k$ such that $s \in S_i$. Intuitively, when the automaton reads a node x labeled σ that has k children, and it proceeds from the state q, it has to take two choices. First, the automaton chooses a pair $\langle U, E\rangle \in \delta(q, \sigma)$. Then, it chooses a way to deliver E among the k children. Thus, we can describe the two choices of the automaton by a pair $\langle U, \langle E_1, \ldots, E_k\rangle\rangle$, where $\langle U, \bigcup_{1 \leq i \leq k} E_z\rangle \in \delta(q, \sigma)$. Note that E_z may be empty. We denote by $\delta_k(q, \sigma)$ the set of such pairs. A *run* of $\mathcal{U}$ on an input tree $\langle T, V\rangle$ is a Q-labeled tree $\langle T, r\rangle$, such that $r(\varepsilon) = q_0$, and for every $x \in T$ with $r(x) = q$, there exists $\langle q_1, \ldots, q_k\rangle \in \delta_k(q, V(x))$ such that for all $1 \leq i \leq k$, we have $r(x \cdot i) = q_i$. Note that each node of the input tree corresponds to exactly one node in the run tree. A run $\langle T, r\rangle$ is accepting if all its paths satisfy the Büchi acceptance condition. Thus, for all paths π, we have $inf(\pi) \cap F = \emptyset$, where $q \in inf(\pi)$ if and only if there are infinitely many $x \in \pi$ for which $r(x) = q$. Equivalently, $\langle T, r\rangle$ is accepting iff $\langle T, r\rangle$ contains infinitely many F-frontiers $G_0 < G_1 < \ldots$. For a state $q \in Q$, let $\mathcal{U}^q$ be $\mathcal{U}$ with initial state q. We say that a symmetric NBT is *monotonic* if for every two states q and p such that $q \subseteq p$, we have that $\mathcal{L}(\mathcal{U}^p) \subseteq \mathcal{L}(\mathcal{U}^q)$, and $p \in F$ implies $q \in F$. In other words, the smaller the state is, the easier it is to accept from it. Note that symmetric nondeterministic tree automata are essentially symmetric alternating automata with transitions in disjunctive normal form (DNF); if we write the transition functions in DNF, then each disjunct is a conjunction of universal and existential requirements, corresponding to a pair $\langle U, E\rangle$.

3 From symmetric NBT and co-NBT to symmetric AWT

Let $\mathcal{U} = \langle \Sigma, \mathcal{D}, Q, q_0, M, F\rangle$ and $\mathcal{U}' = \langle \Sigma, \mathcal{D}, Q', q_0', M', F'\rangle$ be two NBT, and let $|Q| \cdot |Q'| = m$. In [Rab70], Rabin studies the joint behavior of a run of $\mathcal{U}$ with a run of $\mathcal{U}'$. Recall that an accepting run of $\mathcal{U}$ contains infinitely many F-frontiers $G_0 < G_1 < \ldots$, and an accepting run of $\mathcal{U}'$ contains infinitely many F'-frontiers $G_0' < G_1' < \ldots$. It follows that for every labeled tree $\langle T, V\rangle \in \mathcal{L}(\mathcal{U}) \cap \mathcal{L}(\mathcal{U}')$ and accepting runs $\langle T, r\rangle$ and $\langle T, r'\rangle$ of $\mathcal{U}$ and $\mathcal{U}'$ on $\langle T, V\rangle$, the joint behavior of $\langle T, r\rangle$ and $\langle T, r'\rangle$ contains infinitely many frontiers $E_i \subset T$, with $E_i < E_{i+1}$, such that $\langle T, r\rangle$ reaches an F-frontier and $\langle T, r'\rangle$ reaches an F'-frontier between E_i and E_{i+1}. Rabin shows that the existence of m such frontiers, in the joint behavior of some runs of $\mathcal{U}$ and $\mathcal{U}'$, is sufficient to imply that the intersection $\mathcal{L}(\mathcal{U}) \cap \mathcal{L}(\mathcal{U}')$ is not empty. We now extend Rabin's result to symmetric automata.

Assume that $\mathcal{U}$ and $\mathcal{U}'$ above are symmetric NBT. We say that a sequence $E_0, \ldots, E_m$ of frontiers of T is a *trap for $\mathcal{U}$ and $\mathcal{U}'$* iff $E_0 = \{\varepsilon\}$ and there exists a tree $\langle T, V\rangle$ and (not necessarily accepting) runs $\langle T, r\rangle$ and $\langle T, r'\rangle$ of $\mathcal{U}$ and $\mathcal{U}'$ on $\langle T, V\rangle$, such that for every $0 \leq i \leq m-1$, we have that $\langle T, r\rangle$ contains an F-frontier G_i such that $E_i \leq G_i < E_{i+1}$, and $\langle T, r'\rangle$ contains an F'-frontier G_i' such that $E_i \leq G_i' < E_{i+1}$. We say that $\langle T, r\rangle$ and $\langle T, r'\rangle$ *witness* the trap for $\mathcal{U}$ and $\mathcal{U}'$.

Theorem 1. *Consider two symmetric nondeterministic Büchi tree automata $\mathcal{U}$ and $\mathcal{U}'$. If there exists a trap for $\mathcal{U}$ and $\mathcal{U}'$, then $\mathcal{L}(\mathcal{U}) \cap \mathcal{L}(\mathcal{U}')$ is not empty.*

Proof. The proof follows the same line of reasoning as in [Rab70]. For a state $q \in Q$, let $\mathcal{U}^q$ be $\mathcal{U}$ with initial state q, and similarly for $q' \in Q'$ and $\mathcal{U}'^{q'}$. We define a sequence of relations over $Q \times Q'$. Let $H_0 = Q \times Q'$. Then, $\langle q, q' \rangle \in H_{i+1}$ iff $\langle q, q' \rangle \in H_i$ and there is a nonempty Σ-labeled D-tree $\langle T, V \rangle$, a frontier $E \subseteq T$, and runs $\langle T, r \rangle$ and $\langle T, r' \rangle$ of $\mathcal{U}^q$ and $\mathcal{U}'^{q'}$ on $\langle T, V \rangle$, such that there is an F-frontier $G < E$ and an F'-frontier $G' < E$, such that for all $x \in E$, we have $\langle r(x), r'(x) \rangle \in H_i$. It is easy to see that $H_0 \supseteq H_1 \supseteq H_2 \supseteq \ldots$. Also, if $H_i = H_{i+1}$, then $H_i = H_{i+k}$ for all $k \geq 0$. In particular, since $|Q| \times |Q'| = m$, it must be that $H_m = H_{m+k}$ for all $k \geq 0$. As in [Rab70], it can now be shown that $\mathcal{L}(\mathcal{U}) \cap \mathcal{L}(\mathcal{U}') \neq \emptyset$ iff $H_m(q_0, q_0')$, and the result follows.

Theorem 1 is the key to the construction described in Theorem 2 below.

Theorem 2. *Let $\mathcal{U}$ and $\mathcal{U}'$ be two symmetric monotonic NBT with $\mathcal{L}(\mathcal{U}') = comp(\mathcal{L}(\mathcal{U}))$. There exists a symmetric AWT $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{U})$.*

Proof. Let $\mathcal{U} = \langle \Sigma, Q, q_0, M, F \rangle$ and $\mathcal{U}' = \langle \Sigma, Q', q_0', M', F' \rangle$, and let $|Q| \cdot |Q'| = m$. Also, let S and S' be the micro-states of $\mathcal{U}$ and $\mathcal{U}'$, respectively, thus $Q = 2^S$ and $Q' = 2^{S'}$. We define the symmetric AWT $\mathcal{A} = \langle \Sigma, P, p_0, \delta, \alpha \rangle$ as follows.

- $P = Q \times Q' \times \{0, \ldots, 2m-1\}$ and $p_0 = \langle q_0, q_0', 0 \rangle$. Intuitively, a copy of $\mathcal{A}$ that visits the state $\langle q, q', i \rangle$ as it reads the node x of the input tree corresponds to runs r and r' of $\mathcal{U}$ and $\mathcal{U}'$ that visit the states q and q', respectively, as they read the node x of the input tree. Let $\rho = y_0, y_1, \ldots, y_{|x|}$ be the path from ε to x. Consider the joint behavior of r and r' on ρ. We can represent this behavior by a sequence $\tau_\rho = \langle t_0, t_0' \rangle, \langle t_1, t_1' \rangle, \ldots, \langle t_{|x|}, t_{|x|}' \rangle$ of pairs in $Q \times Q'$ where $t_j = r(y_j)$ and $t_j' = r'(y_j)$. We say that a pair $\langle t, t' \rangle \in Q \times Q'$ is an *F-pair* iff $t \in F$ and is an *F'-pair* iff $t' \in F'$. We can partition the sequence τ_ρ to blocks $\beta_0, \beta_1, \ldots, \beta_i$ such that we close block β_b and open block β_{b+1} whenever we reach the first F'-pair that is preceded by an F-pair in β_b. In other words, whenever we open a block, we first look for an F-pair, ignoring F'-pairness. Once an F-pair is detected, we look for an F'-pair, ignoring F-pairness. Once an F'-pair is detected, we close the current block and we open a new block. Note that a block may contain a single pair that is both an F-pair and an F'-pair. The third element of a state keeps track of the visits to blocks. When we visit $\langle q, q', i \rangle$, the index of the last block in τ_ρ is $\lfloor \frac{i}{2} \rfloor$, and this block already contains an F-pair iff i is odd. We refer to i as the *status* of the state $\langle q, q, i \rangle$. For a status $i \in \{0, \ldots, 2m-1\}$, let $P_i = Q \times Q \times \{i\}$ be the set of states with status i.
- In order to define the transition function δ, we first define a function $next : P \to \{0, \ldots, 2m-1\}$ that updates the status of states. For that, we first define the function $next' : P \to \{0, \ldots, 2m\}$ as follows.

$$next'(\langle q, q', i \rangle) = \begin{cases} i & \text{If } (i \text{ is even and } q \notin F) \text{ or } (i \text{ is odd and } q' \notin F') \\ i+1 & \text{If } (i \text{ is even and } q \in F \text{ and } q' \notin F') \text{ or } (i \text{ is odd and } q' \in F') \\ i+2 & \text{If } i \text{ is even and } q \in F \text{ and } q' \in F'. \end{cases}$$

  Now, $next(\langle q, q', i \rangle) = \min\{next'(\langle q, q', i \rangle), 2m-1\}$.
  Intuitively, $next$ updates the status of states by recording and tracking of blocks. Recall that the status i indicates in which block we are and whether an F-pair in the current block has already been detected. The conditions for not changing i or for increasing it to $i+1$ and $i+2$ follow directly from the definition of the status. For example, the new status stays i if the current i is even and $\langle q, q' \rangle$ is not an F-block, or if i is odd and $\langle q, q' \rangle$ is not an F'-block. When i reaches or exceeds $2m-1$, we no longer increase it, even if $q' \in F'$.
  The automaton $\mathcal{A}$ proceeds as follows. Essentially, for every run $\langle T, r' \rangle$ of $\mathcal{U}'$, the automaton $\mathcal{A}$ guesses a run $\langle T, r \rangle$ of $\mathcal{U}$ such that for every path ρ of T, the run $\langle T, r \rangle$ visits F along ρ at least as many times as $\langle T, r' \rangle$ visits F' along ρ. Thus, when we record blocks along ρ, we do not want to get stuck in an even status. Since $\mathcal{L}(\mathcal{U}) \cap \mathcal{L}(\mathcal{U}') = \emptyset$, then, by Theorem 1, no run $\langle T, r \rangle$ can witness with $\langle T, r' \rangle$ a trap for $\mathcal{U}$ and $\mathcal{U}'$. Consequently, recording of visits to F and F' along ρ can be completed once $\mathcal{A}$ detects that τ_ρ contains m blocks as above.

Recall that $Q = 2^S$ and $Q' = 2^{S'}$. For a set $E \subseteq S$, a *partition* of E is a set $\{E_1, \ldots, E_l\}$ with $E_i \subseteq E$ such that $E = \bigcup_{1 \leq i \leq l} E_i$, and for all $1 \leq i \neq j \leq n$, we have $E_i \cap E_j = \emptyset$. Let $par(E)$ be the set of partitions of E. Consider a set $E' \subseteq S'$ and a partition $\gamma' \in par(E')$. For a set $E \subseteq S$, we say that a partition η of $E \cup E'$ *agrees with* γ' if for all s'_1 and s'_2 in E', we have that s'_1 and s'_2 are in the same set in η iff they are in the same set in γ'. Let $agree(E, \gamma')$ be the set of partitions of $E \cup E'$ that agree with γ'. For example, if $E = \{s_1\}$ and $E' = \{s_2, s_3\}$, then the two possible partitions of E' are $\gamma'_1 = \{\{s_2, s_3\}\}$ and $\gamma'_2 = \{\{s_2\}, \{s_3\}\}$. Then, $agree(E, \gamma'_1)$ contains the two partitions $\{\{s_1, s_2, s_3\}\}$ and $\{\{s_1\}, \{s_2, s_3\}\}$, and $agree(E, \gamma'_2)$ contains the three partitions $\{\{s_1, s_2\}, \{s_3\}\}$, $\{\{s_1, s_3\}, \{s_2\}\}$, and $\{\{s_1\}, \{s_2\}, \{s_3\}\}$.
Now, let $p = \langle q, q', i\rangle$ be a state in P such that $M(q, \sigma) = \{\langle U_1, E_1\rangle, \ldots, \langle U_n, E_n\rangle\}$ and $M'(q', \sigma) = \{\langle U'_1, E'_1\rangle, \ldots, \langle U'_{n'}, E'_{n'}\rangle\}$. We distinguish between two cases.

- If $i < 2m - 1$ or $q \notin F$, then

$$\delta(p, \sigma) = \bigwedge_{1 \leq j' \leq n'} \bigwedge_{\gamma' \in par(E'_{j'})} \left(\bigvee_{1 \leq j \leq n} \bigvee_{\eta \in agree(E_j, \gamma')} go(j, j', \eta, next(p)) \right), \text{ where}$$

$$go(j, j', \eta, l) = \Box\langle U_j, U'_{j'}, l\rangle \wedge \bigwedge_{X \in \eta} \Diamond\langle U_j \cup (X \cap E_j), U'_{j'} \cup (X \cap E'_{j'}), l\rangle.$$

  That is, for every choice of $\mathcal{U}'$ for a $1 \leq j' \leq n'$ and for the way the existential requirements in $E'_{j'}$ are partitioned, there is a choice of $\mathcal{U}$ for a $1 \leq j \leq n$ and for the way the existential requirements in E_j are partitioned and combined with these in $E'_{j'}$ to a partition of $E_j \cup E'_{j'}$, such that the universal requirements in U_j and $U'_{j'}$ are sent to all directions, and existential requirements that are in the same set in the joint partition of $E_j \cup E'_{j'}$ are sent to the same direction. Note that the sets U_j and $U'_{j'}$ are sent along with the existential requirements. This guarantees that the states that are sent in the existential mode correspond to the states that $\mathcal{U}$ and $\mathcal{U}'$ visit, and not to subsets of such states.
- If $i = 2m - 1$ and $q \in F$, then $\delta(p, \sigma) = \mathbf{true}$.

Note that $par(E')$ is exponential in $|E'|$, and the number of possible $\eta \in agree(E, \gamma')$ is exponential in $E \cup E'$. Thus, the size of δ is exponential in the sizes of M and M'.
– $\alpha = Q \times Q' \times \{i : i \text{ is odd}\}$. Thus, α makes sure that infinite paths of the run visits infinitely many states in which the status is odd, thus states in which we are in the second phase of blocks. moshe2:

Each set P_i is a strongly connected component, thus the automaton $\mathcal{A}$ is indeed an AWT. Note that, by the definition of α, a run is accepting iff no path of it gets trapped in a set of the form P_i, for an even i, namely a set in which $\mathcal{A}$ is waiting for a visit of $\mathcal{U}$ in a state in F. The number of states of $\mathcal{A}$ is $O(m^2)$. We prove that $\mathcal{L}(\mathcal{U}) = \mathcal{L}(\mathcal{A})$. We first prove that $\mathcal{L}(\mathcal{U}) \subseteq \mathcal{L}(\mathcal{A})$. Consider a D-tree $\langle T, V\rangle$. With every run $\langle T, r\rangle$ of $\mathcal{U}$ on $\langle T, V\rangle$ we can associate a run $\langle T_R, R\rangle$ of $\mathcal{A}$ on $\langle T, V\rangle$. Intuitively, the run $\langle T, r\rangle$ directs $\langle T_R, R\rangle$ in the nondeterminism in δ (that is, the choices of $1 \leq j \leq n$ and $\eta \in agree(E_j, \gamma')$). Formally, recall that a run of $\mathcal{A}$ on a D-tree $\langle T, V\rangle$ is a $(T \times P)$-labeled tree $\langle T_R, R\rangle$, where a node $y \in T_R$ with $R(y) = \langle x, p\rangle$ corresponds to a copy of $\mathcal{A}$ that reads the node $x \in T$ and visits the state p. We define $\langle T_R, R\rangle$ as follows.

– $\varepsilon \in T_R$ and $R(\varepsilon) = (\varepsilon, \langle q_0, q'_0, 0\rangle)$.
– Consider a node $y \in T_R$ with $R(y) = (x, \langle q, q', i\rangle)$. By the definition of $\langle T_R, R\rangle$ so far, we have $r(x) = t$ for $q \subseteq t$. Consider first the case that $t = q$. Let $\{x \cdot 1, \ldots, x \cdot k\}$ be the children of x in T, and let $\langle U, \langle E_1, \ldots, E_k\rangle\rangle \in M_k(q, V(x))$ describe the choice $\mathcal{U}$ makes when it proceeds from the node x. Thus, for each $1 \leq z \leq k$, we have $r(x \cdot z) = U \cup E_z$. Let $j = next(\langle q, q', i\rangle)$. Consider the set

$$Y = \bigcup_{\substack{\langle U', \langle E'_1, \ldots, E'_k\rangle\rangle \in \\ M'_k(q', V(x))}} \{(1, \langle U, U', j\rangle), (1, \langle U \cup E_1, U' \cup E'_1, j\rangle), \ldots (k, \langle U, U', j\rangle), (k, \langle U \cup E_k, U' \cup E'_k, j\rangle)\}.$$

By the definition of δ, the set Y satisfies $\delta(\langle q, q', i\rangle, V(x))$ [4]. Let $l = |M'_k(q', V(x))|$, and let $\langle U'^w, E'^w_1, \ldots E'^w_k\rangle$, for $1 \leq w \leq l$, be the w-th pair in $M'_k(q', V(x))$. For all $1 \leq w \leq l$ and $1 \leq z \leq k$, we have $\{y\cdot(2k(w-1)+z-1), y\cdot(2k(w-1)+z)\} \subseteq T_R$, with $R(y\cdot(2k(w-1)+z-1)) = (x\cdot z, \langle U, U'^w, j\rangle)$ and $R(y\cdot(2k(w-1)+z)) = (x\cdot z, \langle U \cup E_z, U'^w \cup E'^w_z, j\rangle)$. Note that the invariant that for all $y \in T_R$ with $R(y) = (x, \langle q, q', i\rangle)$, we have $r(x) = t$ for $q \subseteq t$, is maintained. If fact, we know that all the nodes $y \in T_R$ that correspond to copies of $\mathcal{A}$ that satisfy an existential requirement have $q = t$, and node $y \in T_R$ that correspond to copies of $\mathcal{A}$ that satisfy a universal requirement have $q = t$ iff the run r sends no existential requirement to the corresponding direction.

Consider now the case where $q \subset t$. Since $\mathcal{U}$ is monotonic, there is an accepting run $\langle T^x, r^x_q\rangle$ of $\mathcal{U}^q$ on the subtree of T with root x. We can proceed exactly as above, with $\langle T^x, r^x_q\rangle$ instead of $\langle T, r\rangle$.

Consider a tree $\langle T, V\rangle \in \mathcal{L}(\mathcal{U})$. Let $\langle T, r\rangle$ be an accepting run of $\mathcal{U}$ on $\langle T, V\rangle$, and let $\langle T_R, R\rangle$ be the run of $\mathcal{A}$ on $\langle T, V\rangle$ induced by $\langle T, r\rangle$ (and the "subtree runs", like $\langle T^x, r^x_q\rangle$ above). It can be shown that $\langle T_R, R\rangle$ is a legal accepting run. Indeed, since $\langle T, r\rangle$ and the subtree runs contains infinitely many F-frontiers, and since (by the definition of monotonic automaton) we do not lose visits to F when we switch to subset runs), no infinite paths of $\langle T_R, R\rangle$ can get trapped in a set P_i for an even i.

It is left to prove that $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{U})$. For that, we prove that $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{U}') = \emptyset$. Since $\mathcal{L}(\mathcal{U}) = comp(\mathcal{L}(\mathcal{U}'))$, it follows that every tree that is accepted by $\mathcal{A}$ is also accepted by $\mathcal{U}$. Consider a tree $\langle T, V\rangle$. With each run $\langle T_R, R\rangle$ of $\mathcal{A}$ on $\langle T, V\rangle$ and run $\langle T, r'\rangle$ of $\mathcal{U}'$ on $\langle T, V\rangle$, we associate a run $\langle T, r\rangle$ of $\mathcal{U}$ on $\langle T, V\rangle$. Intuitively, $\langle T, r\rangle$ makes the choices that $\langle T_R, R\rangle$ has made in its copies that correspond to the run $\langle T, r'\rangle$. Formally, $\langle T, r\rangle$ is such that $r(\varepsilon) = q_0$, and for all $x \in T$ with $r(x) = q$, we proceed as follows. Let $\{x\cdot 1, \ldots, x\cdot k\}$ be the children of x in T, and let $r'(x) = q'$. The run $\langle T, r'\rangle$ selects a pair $\langle U', \langle E'_1, \ldots, E'_k\rangle\rangle \in M'_k(q', V(x))$ that $\mathcal{U}'$ proceeds with when it reads the node x. Formally, for all $1 \leq z \leq k$, we have $r'(x\cdot z) = U' \cup E'_z$.[5] By the definition of $r(x)$ so far, the run $\langle T_R, R\rangle$ contains a node $y \in T_R$ with $R(y) = \langle x, \langle q, q', i\rangle\rangle$ for some status i. If $\delta(\langle q, q', i\rangle, V(x)) = \mathbf{true}$, we define the reminder of $\langle T, r\rangle$ arbitrarily. Otherwise, let $1 \leq j' \leq n'$ and $\gamma' \in par(E'_{j'})$ be such that $\langle U', \langle E'_1, \ldots, E'_k\rangle\rangle$ corresponds to j' and γ'. By the definition of δ, there are $1 \leq j \leq n$ and $\eta \in agree(E_j, \gamma')$ such that $go(j, j', \eta, next(\langle q, q', i\rangle))$ is satisfied and R proceeds according to j and η. Thus, if $\{E^1_j, \ldots, E^k_j\}$ is the partition of E_j that corresponds to η, then T_R contains at least k nodes $y\cdot c_z$, for $1 \leq z \leq k$, such that $R(y\cdot c_z) = \langle x\cdot z, \langle U_j \cup E^z_j, U' \cup E'_z, next(\langle q, q', i\rangle)\rangle\rangle$. For all $1 \leq z \leq k$, we define $r(x\cdot z) = U_j \cup E^z_j$. Note that the invariant about the runs $\langle T, r\rangle$ and $\langle T_R, R\rangle$ is maintained. Note also that if $E^z_j \cup E'_z = \emptyset$, then the existence of a node $y\cdot c_z$ as above is guaranteed from universal part of δ, and if $E^z_j \cup E'_z \neq \emptyset$, its existence is guaranteed from the existential part (in which case it is crucial that we sent the universal requirements along with the existential ones).

We can now prove that $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{U}') = \emptyset$. Assume, by way of contradiction, that there exists a tree $\langle T, V\rangle$ such that $\langle T, V\rangle$ is accepted by both $\mathcal{A}$ and $\mathcal{U}'$. Let $\langle T_R, R\rangle$ and $\langle T, r'\rangle$ be the accepting runs of $\mathcal{A}$ and $\mathcal{U}'$ on $\langle T, V\rangle$, respectively, and let $\langle T, r\rangle$ be the run of $\mathcal{U}$ on $\langle T, V\rangle$ induced by $\langle T_R, R\rangle$ and $\langle T, r'\rangle$. We claim that then, $\langle T, r\rangle$ and $\langle T, r'\rangle$ witness a trap for $\mathcal{U}$ and $\mathcal{U}'$. Since, however, $\mathcal{L}(\mathcal{U}) \cap \mathcal{L}(U') = \emptyset$, it follows from Theorem 1, that no such trap exists, and we reach a contradiction. To see that $\langle T, r\rangle$ and $\langle T, r'\rangle$ indeed witness a trap, define $E_0 = \{\varepsilon\}$, and define, for $0 \leq i \leq m-1$, the set E_{i+1} to contain exactly all nodes x for which there exists $y \in T_R$ such that either $R(y) = \langle x, \langle (r(x), r'(x), 2i+1\rangle\rangle$ and $r'(x) \in F'$ or $R(y) = \langle x, \langle (r(x), r'(x), 2i\rangle\rangle$ and $r(x) \in F$ and $r'(x) \in F'$. That is, for every path ρ of T, the set E_{i+1} consists of the nodes in which the i'th block is closed in τ_ρ. By the definition of δ, for all

[4] Note that $\delta(\langle q, q', i\rangle, V(x))$ is a formula in $\mathcal{B}^+(\{\Box, \Diamond\} \times P)$, whereas $Y \subseteq \{1, \ldots, k\} \times P$, but the extension of the satisfaction relation to this setting is straightforward: an atom $(\Diamond, p)$ is satisfied in Y if there is $1 \leq z \leq k$ with $(z, p) \in Y$, and an atom $(\Box, p)$ is satisfied in Y if for all $1 \leq z \leq k$, we have $(z, p) \in Y$.

[5] For a monotonic NBT, we assume that runs satisfy the requirements in transition function in an optimal way; thus when $\mathcal{A}$ chooses to proceed with $\langle U', \langle E'_1, \ldots, E'_k\rangle\rangle \in M'_k(q', V(x))$, it is indeed the case that $r'(x\cdot z) = U' \cup E'_z$. If $r'(x\cdot z) \supset U' \cup E'_z$, we can replace r' with a run for which the equation holds.

$0 \leq i \leq m-1$, the run $\langle T, r\rangle$ contains an F-frontier G_i such that $E_i \leq G_i < E_{i+1}$ and the run $\langle T, r'\rangle$ contains an F'-frontier G'_i such that $E_i \leq G'_i < E_{i+1}$. Hence, $E_0, \ldots, E_m$ is a trap for $\mathcal{U}$ and $\mathcal{U}'$.

4 From $\Pi_2 \cap \Sigma_2$ to the alternation-free μ-calculus

The *μ-calculus* is a propositional modal logic augmented with least and greatest fixpoint operators [Koz83]. Specifically, we consider a μ-calculus where formulas are constructed from Boolean propositions with Boolean connectives, the temporal operators $\exists\bigcirc$("exists next") and $\Box$ ("for all next"), as well as least (μ) and greatest (ν) fixpoint operators. We assume that μ-calculus formulas are written in positive normal form (negation only applied to atomic propositions constants and variables). Formally, given a set AP of atomic proposition constants and a set APV of atomic proposition variables, a μ-calculus formula is either:

– $\mathbf{true}$, $\mathbf{false}$, p or $\neg p$ for all $p \in AP$.
– y for all $y \in APV$;
– $\varphi \wedge \psi$, $\varphi \vee \psi$, $\Diamond\varphi$, or $\Box\varphi$, where φ and ψ are μ-calculus formulas;
– $\mu y.\varphi(y)$ or $\nu y.\varphi(y)$, where $y \in APV$ and $\varphi(y)$ is a μ-calculus formula containing y as a free variable.

We classify formulas to classes Σ_i and Π_i according to the nesting of fixpoint operators in them. Several versions to such a classification can be found in the literature [EL86,Niw86,Bra98]. We describe here the version defined in [Niw86]:

– A formula is in $\Sigma_0 = \Pi_0$ if it contains no fixpoint operators.
– A formula is in Σ_{i+1} if it is one of the following θ_i, $\theta_i \wedge \theta'_i$, $\theta_i \vee \theta'_i$, $\Diamond\theta_i$, $\Box\theta_i$, $\mu y.\varphi_{i+1}(y)$, $\varphi_{i+1}(Y)[y \leftarrow \varphi'_{i+1}]$, where θ_i and θ'_i are $\Sigma_i \cup \Pi_i$ formulas, φ_{i+1} and φ'_{i+1} are Σ_{i+1} formulas, $Y \subseteq APV$, $y \in Y$, and no free variable of φ'_{i+1} is in Y. In other words, to form Σ_{i+1}, we take $\Sigma_i \cup \Pi_i$ and close under Boolean and modal operations, $\mu y.\varphi(y)$ for $\varphi \in \Sigma_{i+1}$, and substitution of a free variable of $\varphi \in \Sigma_{i+1}$ by a formula $\varphi' \in \Sigma_{i+1}$ provided that no free variable of φ' is captured by φ.
– A formula is in Π_{i+1} if it is one of the following θ_i, $\theta_i \wedge \theta'_i$, $\theta_i \vee \theta'_i$, $\Diamond\theta_i$, $\Box\theta_i$, $\nu y.\psi_{i+1}(y)$, $\psi_{i+1}(Y)[y \leftarrow \psi'_{i+1}]$, where θ_i and θ'_i are $\Sigma_i \cup \Pi_i$ formulas, ψ_{i+1} and ψ'_{i+1} are Π_{i+1} formulas, $Y \subseteq APV$, $y \in Y$, and no free variable of ψ'_{i+1} is in Y.

Note that the "substitution step" suggests that the formula $\psi = \nu y.(\Diamond(y \wedge (\mu z.p \vee \Diamond z))$ is in both Π_2 and Σ_2. To see that ψ is in Σ_2 (it is easy to see that $\psi \in \Pi_2$), note that $\mu z.p \vee \Diamond z$ is in Σ_1, and hence also in Σ_2. In addition, the formula $\nu y.\Diamond(y \wedge x)$, for $x \in APV$, is in Π_1, and hence also in Σ_2. The formula $\mu z.p \vee \Diamond z$ has no free variables. Then, we can substitute x by it, get ψ, and stay in Σ_2. Note that for for classifications that do not allow such a substitution, the formula ψ is not in Σ_2. Note also that ψ is neither in Π_1 nor Σ_1.

Finally, we say that a formula is in Δ_i if it is one of the following θ_i, $\theta_i \wedge \theta'_i$, $\theta_i \vee \theta'_i$, $\Diamond\theta_i$, $\Box\theta_i$, $\theta(Y)[y \leftarrow \theta'_i]$, where θ_i and θ'_i are $\Sigma_i \cup \Pi_i$ formulas, $Y \subseteq APV$, $y \in Y$, and no variable of θ'_i is in Y. In other words, to form Δ_i, we take $\Sigma_i \cup \Pi_i$ and close under Boolean and modal operations, and under substitution that does not increase the alternation depth. Note that Δ_0 is ML and Δ_1 is AFMC.

Essentially, Σ_i contains all Boolean and modal combinations of formulas in which there are at most $i-1$ alternations of μ and ν, with the external fixpoint being a μ. Similarly, Π_i contains all Boolean and modal combinations of formulas in which there are at most i alternations of μ and ν, with the external fixpoint being a ν. A μ-calculus formula is *alternation free* if, for all atomic propositional variables y, there are no occurrences of ν (μ) on any syntactic path from an occurrence of μy (νy, respectively) to an occurrence of y. For example, the formula $\mu x.(p \vee \mu y.(x \vee EXy))$ is alternation free (and is in Σ_1) and the formula $\nu x.\mu y.((p \wedge x) \vee EXy)$ is not alternation free (and is in Π_2). The *alternation-free μ-calculus* is a subset of μ-calculus containing only alternation-free formulas. The alternation-free μ-calculus is a strict syntactic fragment of $\Pi_2 \cap \Sigma_2$. We now use Theorem 2 in order to show that $\Pi_2 \cap \Sigma_2$ is not more expressive than the alternation free μ-calculus. Thus, every formula in $\Pi_2 \cap \Sigma_2$ has an equivalent formula in AFMC.

For the alternation-free μ-calculus, an automata-theoretic characterization in terms of symmetric alternating weak automata is well known (a similar result is proven in [AN92] for directed trees):

Theorem 3. [KV98] *A set $\mathcal{T} \subseteq trees(\Sigma)$ can be expressed in AFMC iff $\mathcal{T}$ can be recognized by a symmetric weak alternating automaton.*

In [Kai95], Kaivola considered μ-calculus formulas in which the $\Diamond$ modality is parameterized with directions and translates Π_2 formulas to NBT. In order to apply Theorem 2, we should translate Π_2 formulas to symmetric monotonic NBT. For that, we first use a known translation of Π_2 formulas to symmetric ABT (Theorem 4; a similar translation for the directed case is described in [Niw86,Tak86]), and then remove alternation, with symmetry preserved (Theorem 5).

Theorem 4. [KVW00] *Given a Π_2 formula ψ, there is a symmetric alternating Büchi tree automaton $\mathcal{A}_\psi$ that accepts exactly all trees that satisfy ψ.*

Miyano and Hayashi described a translation of alternating Büchi word automata to equivalent nondeterministic Büchi word automata [MH84]. Mostowski extended the translation to tree automata [Mos84], and we extend it further to symmetric tree automata. Since the nondeterministic automaton needs to keep track of the states visited in each path of the run tree of the alternating automaton, the symmetry of the automaton poses real technical challenges.

Theorem 5. *Let $\mathcal{A}$ be a symmetric alternating Büchi tree automaton. There is a symmetric monotonic nondeterministic Büchi tree automaton $\mathcal{A}'$, with exponentially many states, such that $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.*

Proof. Let $\mathcal{A} = \langle \Sigma, S, s_{in}, \delta, \alpha\rangle$. Then $\mathcal{A}' = \langle \Sigma, Q, \{\langle s_{in}, 2\rangle\}, \delta', \alpha'\rangle$, where

- $Q = 2^{S\times\{1,2\}}$. For a state $q \in Q$, let $q[1] = \{s : \langle s, 1\rangle \in q\}$ and $q[2] = \{s : \langle s, 2\rangle \in q\}$. Intuitively, the automaton $\mathcal{A}'$ guesses a run of $\mathcal{A}$. At a given node x of a run of $\mathcal{A}'$, it keeps in its memory the set of all the states of $\mathcal{A}$ that visit x in the guessed run. As it reads the next input letter, it guesses the way in which an accepting run of $\mathcal{A}$ proceeds from all of these states. This guess induces the states that the run of $\mathcal{A}'$ visit in the children of x. In order to make sure that every infinite path visits states in α infinitely often, the states are tagged by 1 or 2. States tagged by 1 correspond to copies that have already visit α, and states tagged by 2 correspond to copies that owe a visit to α. When all the copies visit α (that is, all the states are tagged by 1), we change the tag of all states to 2.
- Given $S' \subseteq S$, $\sigma \in \Sigma$, and a pair $\langle U, E\rangle$ of subsets of S, we say that $\langle U, E\rangle$ *covers* S' *and* σ if the set $\{\Box s : s \in U\} \cup \{\Diamond s : s \in E\}$ satisfies $\bigwedge_{s'\in S'} \delta(s', \sigma)$.
  Now, $\delta' : Q \times \Sigma \to 2^{Q\times Q}$ is defined, for all $q \in Q$ and $\sigma \in \Sigma$, as follows.
  - If $q[2] \neq \emptyset$, then $\delta'(q, \sigma)$ contains all pairs $\langle U, E\rangle$ such that there is $\langle U_1, E_1\rangle$ that covers $q[1]$ and σ, and there is $\langle U_2, E_2\rangle$ that covers $q[2]$ and σ, and the following hold.
    - $U = \{\langle s, 1\rangle : s \in U_1 \cup (U_2 \cap \alpha)\} \cup \{\langle s, 2\rangle : s \in U_2 \setminus \alpha\}$.
    - $E = \{\langle s, 1\rangle : s \in E_1 \cup (E_2 \cap \alpha)\} \cup \{\langle s, 2\rangle : s \in E_2 \setminus \alpha\}$.
  - If $q[2] = \emptyset$, then $\delta'(q, \sigma)$ contains all pairs $\langle U, E\rangle$ such that there is $\langle U_1, E_1\rangle$ that covers $q[1]$ and σ and the following hold.
    - $U = \{\langle s, 1\rangle : s \in U_1 \cap \alpha\} \cup \{\langle s, 2\rangle : s \in U_1 \setminus \alpha)\}$.
    - $E = \{\langle s, 1\rangle : s \in E_1 \cap \alpha\} \cup \{\langle s, 2\rangle : s \in E_1 \setminus \alpha)\}$.
- $\alpha' = \{q : q[2] = \emptyset\}$. Note that a sequence of states of $\mathcal{A}$, which corresponds to the behavior of a copy of $\mathcal{A}$, changes the tag of its states from 2 to 1 when the copy visits a state in α. Also, once all the sequences change the tag of their states to 1, the attribution is changed back to 2. Thus, α' guarantees that all sequences visit α infinitely often.

It is easy to see that $\mathcal{A}$ is monotonic. Indeed, if $q \subseteq q'$, then $q[1] \subseteq q'[1]$ and $q[2] \subseteq q'[2]$. Thus, if a pair $\langle U, E\rangle$ covers $q'[1]$ and σ, then $\langle U, E\rangle$ also covers $q[1]$ and σ, and similarly for $q'[2]$ and $q[2]$. Hence, given an accepting run of $A^{q'}$, we can make it an accepting run of $\mathcal{A}^q$ by changing the labels of the root from (ε, q') to (ε, q). In addition, if $q'[2]$ is empty, so is $q[2]$.

Remark 1. A related approach for translating μ-calculus formulas into symmetric automata is taken in [JW95] (see also [AN01]). First, μ-calculus formulas are transformed into a disjunctive form. The removal of conjunctions described there is similar to the removal of universal branches in alternating tree automata (and indeed it involves the same determinization construction that is present in the automata-theoretic approach [MS87]). It is then shown that disjunctive μ-calculus formulas correspond to μ-automata. Our focus here is on the translation of Π_2 formulas to symmetric monotonic nondeterministic Büchi tree automata. It is possible to recast our proof in an extension of the framework of μ-automata [Wal03], but we find our notion of symmetric nondeterministic automata more transparent.

Theorem 6. $\Pi_2 \cap \Sigma_2 \equiv AFMC$.

Proof. Since AFMC is a syntactic fragment of $\Pi_2 \cap \Sigma_2$, one direction is trivial. Let ξ be a property expressible in $\Pi_2 \cap \Sigma_2$. Given $\theta \in \Pi_2$ expressing ξ, we can construct, by Theorems 4 and 5, a symmetric monotonic NBT $\mathcal{U}_\theta$ that accepts exactly all trees that satisfy θ. Also, $\xi \in \Sigma_2$ implies that there is $\psi \in \Pi_2$ that is equivalent to $\neg\theta$, so we can also construct a symmetric monotonic NBT $\mathcal{U}_\psi$ that accepts exactly all trees that do not satisfy θ. Clearly, $\mathcal{L}(\mathcal{U}_\psi) = comp(\mathcal{L}(\mathcal{U}_\theta))$. Hence, by Theorem 2, there is a symmetric alternating weak automaton $\mathcal{A}_\theta$ that is equivalent to $\mathcal{U}_\theta$. By Theorem 3, the automaton $\mathcal{A}_\theta$ can be translated to a formula φ in AFMC such that a tree satisfies φ iff it is accepted by $\mathcal{U}_\theta$ iff it is not accepted by $\mathcal{U}_\psi$. We claim that φ is logically equivalent to θ over arbitrary structures (in particular, structures with an infinite branching degree). To see this, assume, by way of contradiction, that φ is not logically equivalent to θ. Then, either $\theta \wedge \neg\varphi$ or $\varphi \wedge \psi$ is satisfiable in some general structure. But then, either $\theta \wedge \neg\varphi$ or $\varphi \wedge \psi$ is satisfiable by a tree model [SE84] of a finite branching degree, contradicting the fact that a tree satisfies φ iff it is accepted by $\mathcal{U}_\theta$ iff it is not accepted by $\mathcal{U}_\psi$.

Remark 2. Since it is also known that the μ-calculus has the *finite-model property* [KP84], it follows that Theorem 6 can also be relativized to finite Kripke structures.

5 Concluding Remarks

We showed that $\Sigma_2 \cap \Pi_2 \equiv$ AFMC. In other words, if we can specify a property ψ both as a least fixpoint nested inside a greatest fixpoint and as a greatest fixpoint nested inside a least fixpoint, we should be able to specify ψ also with no alternation between greatest and least fixpoints. This offers an elegant characterization of alternation freedom. The key to our results is a development of a theory of *symmetric nondeterministic Büchi* tree automata. A technical outcome of this theory is that the blow-up of our construction, i.e., going from formulas in $\Sigma_2 \cap \Pi_2$ to equivalent formulas in AFMC is doubly exponential. It would be interesting to try to improve this complexity or to prove its optimality.

Combining our result here with the result in [KV01] ($\Sigma_1 \cap \Pi_1 \equiv$ ML) suggests the possibility of a general coalescence result for the μ-calculus hierarchy. Recall the definition of Δ_i as the closure of $\Sigma_i \cap \Pi_i$ under Boolean and modal operations and under alternation-preserving substitutions. Then we have that $\Sigma_i \cap \Pi_i \equiv \Delta_{i-1}$ for $i = 1, 2$. It is tempting to conjecture that this holds for all $i > 0$, in analogy for such coalescence for the quantifier alternation hierarchy of first-order logic (cf. [Add62]). As is shown, however, in [AS03], this is not the case for $i > 2$.

Acknowledgements: We are grateful to J.W. Addison for valuable discussions regarding the first-order quantifier-alternation hierarchy and to I. Walukiewicz for discussions regarding μ-automata.

References

[Add62] J.W. Addision, The theory of hierarchies. *Proc. Internat. Congr. Logic, Method. and Philos. Sci. 1960*, pages 26–37, Stanford University Press, 1962.

[AN90] A. Arnold and D. Niwiński. Fixed point characterization of Büchi automata on infinite trees. *Information Processing and Cybernetics*, 8–9:451–459, 1990.

[AN92] A. Arnold and D. Niwiński. Fixed point characterization of weak monadic logic definable sets of trees. In *Tree Automata and Languages*, pages 159–188, Elsevier, 1992.

[AN01] A. Arnold and D. Niwiński. *Rediments of μ-calculus*. Elsevier, 2001.

[AS03] A. Arnold and L. Santocanale, On ambiguous classes in the μ-calculus hierarchy of tree languages, *Proc. Workshop on Fixed Points in Computer Science*, Warsaw, Poland, 2003.

[Ben91] J. Benthem. Languages in actions: categories, lambdas and dynamic logic. *Studies in Logic*, 130, 1991.

[Bra98] J.C. Bradfield. The modal μ-calculus alternation hierarchy is strict. *TCS*, 195(2):133–153, March 1998.

[BRS99] R. Bloem, K. Ravi, and F. Somenzi. Efficient decision procedures for model checking of linear time logic properties. In *Proc. 11th CAV*, LNCS 1633, pages 222–235. 1999.

[CS91] R. Cleaveland and B. Steffen. A linear-time model-checking algorithm for the alternation-free modal μ-calculus. In *Proc. 3rd CAV*, *LNCS* 575, pages 48–58, 1991.

[EL85] E.A. Emerson and C.-L. Lei. Temporal model checking under generalized fairness constraints. In *Proc. 18th Hawaii International Conference on System Sciences*, 1985.

[EL86] E.A. Emerson and C.-L. Lei. Efficient model checking in fragments of the propositional μ-calculus. In *Proc. 1st LICS*, pages 267–278, 1986.

[FL79] M.J. Fischer and R.E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and Systems Sciences*, 18:194–211, 1979.

[GJ79] M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. Freeman and Co., San Francisco, 1979.

[Jur00] M. Jurdzinski. Small progress measures for solving parity games. In *17th STACS*, LNCS 1770, pages 290–301. 2000.

[JW95] D. Janin and I. Walukiewicz. Automata for the modal μ-calculus and related results. In *Proc. 20th MFCS*, LNCS, pages 552–562. 1995.

[Kai95] R. Kaivola. On modal μ-calculus and Büchi tree automata. *IPL*, 54:17–22, 1995.

[Koz83] D. Kozen. Results on the propositional μ-calculus. *TCS*, 27:333–354, 1983.

[KP84] D. Kozen and R. Parikh. A decision procedure for the propositional μ-calculus. In *Logics of Programs*, LNCS 164, pages 313–325, 1984.

[KV98] O. Kupferman and M.Y. Vardi. Freedom, weakness, and determinism: from linear-time to branching-time. In *Proc. 13th LICS*, pages 81–92, June 1998.

[KV99] O. Kupferman and M.Y. Vardi. The weakness of self-complementation. In *Proc. 16th STACS*, *LNCS* 1563, pages 455–466. 1999.

[KV01] O. Kupferman and M.Y. Vardi. On clopen specifications. In *Proc. 8th LPAR*, LNCS 2250, pages 24–38. 2001.

[KVW00] O. Kupferman, M.Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, March 2000.

[McM93] K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, 1993.

[MH84] S. Miyano and T. Hayashi. Alternating finite automata on ω-words. *TCS*, 32:321–330, 1984.

[Mos84] A.W. Mostowski. Regular expressions for infinite trees and a standard form of automata. In *Computation Theory*, LNCS 208, pages 157–168. 1984.

[MS87] D.E. Muller and P.E. Schupp. Alternating automata on infinite trees. *TCS*, 54:267–276, 1987.

[MSS86] D.E. Muller, A. Saoudi, and P.E. Schupp. Alternating automata, the weak monadic theory of the tree and its complexity. In *Proc. 13th ICALP*, LNCS 226, 1986.

[Niw86] D. Niwiński. On fixed point clones. In *Proc. 13th ICALP*, LNCS 226, pages 464–473. 1986.

[Rab70] M.O. Rabin. Weakly definable relations and special automata. In *Proc. Symp. Math. Logic and Foundations of Set Theory*, pages 1–23, 1970.

[Rog67] H. Rogers, Theory of recursive functions and effective computability. McGraw-Hill, 1967.

[SE84] R.S. Street and E.A. Emerson. An elementary decision procedure for the μ-calculus. In *Proc. 11th ICALP*, LNCS 172, pages 465–472, 1984.

[Tak86] M. Takahashi. The greatest fixed-points and rational ω-tree languages. TCS 44, pp. 259–274, 1986.

[VW86] M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. 1st LICS*, pages 332–344, 1986.

[Wal03] I. Walukiewicz. Private communication, 2003.

[Wil99] T. Wilke. CTL^{+} is exponentially more succinct than CTL. In *Proc. 19th FST& TCS*, LNCS 1738, pages 110–121, 1999.