Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 354 (2006) 173 – 186
Theoretical Computer Science
www.elsevier.com/locate/tcs

On the universal and existential fragments of the μ-calculus☆

Thomas A. Henzinger[a,b], Orna Kupferman[c], Rupak Majumdar[d,*]

[a]Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720, USA
[b]EPFL, Switzerland
[c]School of Computer Science and Engineering, Hebrew University, Jerusalem 91904, Israel
[d]Department of Computer Science, University of California, Los Angeles, CA 90095, USA

Abstract

One source of complexity in the μ-calculus is its ability to specify an unbounded number of switches between universal (*AX*) and existential (*EX*) branching modes. We therefore study the problems of satisfiability, validity, model checking, and implication for the universal and existential fragments of the μ-calculus, in which only one branching mode is allowed. The universal fragment is rich enough to express most specifications of interest, and therefore improved algorithms are of practical importance. We show that while the satisfiability and validity problems become indeed simpler for the existential and universal fragments, this is, unfortunately, not the case for model checking and implication. We also show the corresponding results for the alternation-free fragment of the μ-calculus, where no alternations between least and greatest fixed points are allowed. Our results imply that efforts to find a polynomial-time model-checking algorithm for the μ-calculus can be replaced by efforts to find such an algorithm for the universal or existential fragment.

Keywords: Automata and logic; μ-Calculus; Model checking; Satisfiability

1. Introduction

In model checking, we reason about systems and their properties by reasoning about formal models of systems and formal specifications of the properties [8]. The algorithmic nature of model checking makes it fully automatic, convenient to use, and attractive to practitioners. At the same time, model checking is very sensitive to the size of the formal model of the system and the formal specification. Commercial verification tools need to cope with the exceedingly large state spaces that are present in real-life designs. One of the most important developments in this area is the discovery of symbolic methods [5,33]. Typically, symbolic model-checking tools proceed by computing fixed-point 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 y of states in which p holds, and repeatedly adds

☆ A preliminary version of this paper appeared in the Proceedings of the 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2003), Lecture Notes in Computer Science, vol. 2619, Springer, Berlin, 2003, pp. 49–64. This work was supported in part by NSF Grant CCR-9988172, the AFOSR MURI Grant F49620-00-1-0327, and a Microsoft Research Fellowship.
* Corresponding author.
E-mail address: rupak@cs.ucla.edu (R. Majumdar).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2005.11.015

to y the set EXy of states that have a successor in y. Formally, the model checker calculates the least fixed point of the expression $y = (p \vee EXy)$.

Such fixed-point computations are described naturally in the *μ-calculus* [25,1], which is a logic that contains the existential and universal next modalities EX and AX, and the least and greatest fixed-point quantifiers μ and ν. The μ-calculus is an extremely general modal logic. It is as expressive as automata on infinite trees, and it subsumes most known specification formalisms, including dynamic logics such as PDL [16] and temporal logics such as LTL and CTL⋆ [10,11] (see [22] for a general result). The *alternation-free* fragment of the μ-calculus (AFMC, for short) [15] has a restricted syntax that does not allow the nesting of alternating least and greatest fixed-point quantifiers, which makes the evaluation of expressions very simple [9]. The alternation-free fragment subsumes the temporal logic CTL.

Four decision problems arise naturally for every specification formalism: the *satisfiability* problem (given a formula φ, is there a model that satisfies φ?) checks whether a specification can be implemented, and algorithms for deciding the satisfiability problem are the basis for program synthesis and control [6,36,37]; the *validity* problem (given φ, do all models satisfy φ?) checks whether the specification is trivially satisfied, and is used as a sanity check for requirements [29]; the *model-checking* problem (given a formula φ and a model M, does M satisfy φ?) is the basic verification problem; and the *implication* problem (given two formulas φ and ψ, is $\varphi \rightarrow \psi$ valid?) arises naturally in the context of modular verification, where it must be shown that a module satisfies a property under an assumption about the environment [27,35].

The satisfiability, validity, and implication problems for the μ-calculus are all EXPTIME-complete [2,16] (since the μ-calculus is closed under negation, it is easy to get EXPTIME completeness for the validity and implication problems by reductions to and from the satisfiability problem). The model-checking problem for the μ-calculus was first considered in [15], which described an algorithm with complexity $O((mn)^{l+1})$, where m is the size of M, n is the size of φ, and l is the number of alternations between least and greatest fixed-point quantifiers in φ. In [14], the problem was shown to be equivalent to the nonemptiness problem for parity tree automata, and thus to lie in NP ∩ co-NP. Today, it is known that the problem is in UP ∩ co-UP [23], [1] and the best known algorithm for μ-calculus model checking has a time complexity of roughly $O(mn^{l/2})$ [24,31,38], which is still exponential in the number of alternations. The precise complexity of the problem, and in particular, the question whether a polynomial time solution exists, is a long-standing open problem.

In this paper we study the complexity of the four decision problems for the *existential* and *universal* fragments of the μ-calculus. The existential fragment consists of formulas where the only allowed next modality is the existential one (EX), and the universal fragment consists of formulas where the only allowed next modality is the universal one (AX). We consider μ-calculus in positive normal form, thus the strict syntactic fragments are also semantic fragments—there is no way of specifying an existential next in the universal fragment without negation, and vice versa. Both sublogics induce the state equivalence *similarity* (mutual simulation) [18,30], as opposed to bisimilarity, which is induced by the full μ-calculus [19]. The existential and universal fragments of the μ-calculus subsume the existential and universal fragments of the branching-time logics CTL and CTL*. For temporal logics, the universal and existential fragments have been studied (see, e.g., [27]). As we specify in the table in Fig. 1, the satisfiability, validity, and implication problems for the universal and existential fragments of CTL and CTL* are all easier than the corresponding problems for the full logics [12,16,27,40]. On the other hand, the model-checking complexities for the universal and existential fragments of CTL and CTL⋆ coincide with the complexities of the full logics, and the same holds for the system complexities of model checking (i.e., the complexities in terms of the size of the model, assuming the specification is fixed. Since the model is typically much bigger than the specification, system complexity is important) [7,28].

In contrast to CTL and CTL⋆, it is possible to express in the μ-calculus *unbounded switching* of AX and EX modalities. Such an unbounded switching is an apparent source of complexity. For example, the μ-calculus can express the reachability problem on And-Or graphs, which is PTIME-complete, while the reachability problem on plain graphs (existential reachability), and its universal counterpart, are NL-complete. Accordingly, the system complexity of the model-checking problem for the μ-calculus is PTIME-complete, whereas the one for CTL and CTL⋆ is only NL-complete [15,20,28]. By removing the switching of modalities from the μ-calculus, one may hope that the algorithms for the four decision problems, and model checking in particular, will become simpler. Since most specifications assert what a system must or must not do in *all* possible futures, the universal fragment of the μ-calculus is expressive enough to capture most specifications of interest. Also, the problem of checking symbolically whether a model contains a

[1] The class UP is a subset of NP, where each word accepted by the Turing machine has a unique accepting run.

	Satisfiability	Validity	Implication	Model checking	system complexity
CTL*	2EXPTIME	2EXPTIME	2EXPTIME	PSPACE	NL
∀CTL*	PSPACE	PSPACE	EXPSPACE	PSPACE	NL
∃CTL*	PSPACE	PSPACE	EXPSPACE	PSPACE	NL
CTL	EXPTIME	EXPTIME	EXPTIME	PTIME (linear)	NL
∀CTL	PSPACE	co-NP	PSPACE	PTIME (linear)	NL
∃CTL	NP	PSPACE	PSPACE	PTIME (linear)	NL
MC	EXPTIME	EXPTIME	EXPTIME	NP ∩ co-NP	PTIME
∀*MC*	*PSPACE*	*co-NP*	*EXPTIME*	*NP ∩ co-NP*	*PTIME*
∃*MC*	*NP*	*PSPACE*	*EXPTIME*	NP ∩ *co-NP*	PTIME
AFMC	EXPTIME	EXPTIME	EXPTIME	PTIME (linear)	PTIME
∀*AFMC*	*PSPACE*	*co-NP*	*EXPTIME*	*PTIME* (linear)	*PTIME*
∃*AFMC*	*NP*	*PSPACE*	*EXPTIME*	*PTIME* (linear)	*PTIME*

Fig. 1. Summary of known and new (in italics) results.

computation that satisfies an LTL formula is reduced to model checking of an existential μ-calculus formula. Hence, our study is not only of theoretical interest—efficient algorithms for the universal and existential fragments of the μ-calculus are of practical interest.

We determine the complexities of the four decision problems for the universal and existential fragments of the μ-calculus, as well as for the corresponding alternation-free fragments. Our results are summarized in Fig. 1. All the complexities in the figure except for the NP ∩ co-NP result for MC, $\exists MC$, and $\forall MC$ model checking are tight. It turns out that the hope to obtain simpler algorithms for the universal and existential fragments is only partially fulfilled. We show that while the satisfiability and validity problems become easier for the existential and universal fragments, both the model-checking and implication problems stay as hard as for the full μ-calculus (or its alternation-free fragment). In particular, our results imply that efforts to find a polynomial time model-checking algorithm for the μ-calculus can be replaced by efforts to find polynomial time model-checking algorithms for the universal or existential fragment. Note that the picture we obtain for the μ-calculus and its alternation-free fragment does not coincide with the picture obtained in the study of the universal and existential fragments of CTL and CTL*, where the restriction to the universal or existential fragments makes also the implication problem easier.

One key insight concerns the size of models for the existential and universal fragments of the μ-calculus. We prove that the satisfiability problem for the existential fragment of μ-calculus is in NP via a *linear-size model property*. This is in contrast to the full μ-calculus, which has only an exponential-size model property [26]. This shows that extending propositional logic by the *EX* modality and fixed-point quantifiers does not make the satisfiability problem harder. On the other hand, a similar extension with *AX* results in a logic for which the linear-size model property does not hold, and whose satisfiability problem is PSPACE-complete.

A second insight is that, in model-checking as well as implication problems, the switching of *EX* and *AX* modalities can be encoded by the boolean connectives $\vee$ and $\wedge$ in combination with either one of the two modalities and fixed-point quantifiers. Let us be more precise. The model-checking problem for the μ-calculus is closely related to the problem of determining the winner in games on And-Or graphs. The system complexity of μ-calculus model checking is PTIME-hard, because a μ-calculus formula of a fixed size can specify an unbounded number of switches between universal and existential branching modes. In particular, the formula $\mu y.(t \vee EXAX\, y)$ specifies winning for And-Or reachability games, and formulas with alternations between least and greatest fixed-point quantifiers can specify winning for And-Or parity games. One would therefore suspect that the universal and existential fragments of the μ-calculus, in which no switching between branching modes is possible, might not be sufficiently strong to specify And-Or reachability.

Indeed, in [14] the authors define a fragment L_2 of the μ-calculus which explicitly bounds the number of switches between both AX and EX modalities and $\wedge$ and $\vee$ boolean operators. This fragment is as expressive as extended CTL* [14], and it cannot specify reachability in And-Or graphs (the system complexity of model checking is NL-complete). However, in model checking as well as implication problems, we can consider models in which the successors of a state are labeled in a way that enables the specification to directly refer to them. Then, it is possible to replace the existential next modality by a disjunction over all successors, and it is possible to replace the universal next modality by a conjunction that refers to each successor. More specifically, if we can guarantee that the successors of a state with branching degree two are labeled by l (left) and r (right), then the existential next formula EXy can be replaced by $AX(l \rightarrow y) \vee AX(r \rightarrow y)$, and the universal next formula AXy can be replaced by $EX(l \wedge y) \wedge EX(r \wedge y)$. While these observations are technically simple, they enable us to solve the open problems regarding the complexity of the universal and existential fragments of the μ-calculus.

2. Propositional μ-calculus

The *propositional μ-calculus* (*MC*, for short) is a propositional modal logic augmented with least and greatest fixed-point quantifiers [25]. Specifically, we consider a μ-calculus where formulas are constructed from boolean propositions with boolean connectives, the temporal modalities *EX* and *AX*, as well as least (μ) and greatest (ν) fixed-point quantifiers. We assume without loss of generality that μ-calculus formulas are written in positive normal form (negation is applied only to atomic propositions). Formally, given a set AP of atomic propositions and a set V of variables, a μ-calculus formula is either:

- *true*, *false*, p, or $\neg p$, for $p \in AP$;
- y, for $y \in V$;
- $\varphi_1 \wedge \varphi_2$ or $\varphi_1 \vee \varphi_2$, where φ_1 and φ_2 are μ-calculus formulas;
- $AX\varphi$ or $EX\varphi$, where φ is a μ-calculus formula;
- $\mu y.\varphi$ or $\nu y.\varphi$, where $y \in V$ and φ is a μ-calculus formula.

We say that the variable y is *bound* in $\mu y.\varphi$ and $\nu y.\varphi$. A variable is *free* if it is not bound. A *sentence* is a formula that contains no free variables. We refer to *AX* and *EX* as the *universal* and *existential* next modalities, respectively. For a μ-calculus formula φ, define the *size* $|\varphi|$ of φ as the size of the DAG representation of φ. Note that this is always smaller than or equal to the number of syntactic symbols occurring in the formula.

The *universal μ-calculus* ($\forall$*MC*, for short) is the fragment of the μ-calculus in which the only next modality allowed is the universal one. Dually, the *existential μ-calculus* ($\exists$*MC*, for short) is the fragment in which the only next modality allowed is the existential one. Note that since μ-calculus formulas are written in positive normal form, there is no way to specify existential next in $\forall$*MC* by negating universal next.

A μ-calculus formula is *alternation-free* if, for all $y \in V$, there are respectively no occurrences of $\nu(\mu)$ on any syntactic path from an occurrence of $\mu y(\nu y)$ to an occurrence of y. For example, the formula $\mu x.(p \vee \mu y.(x \vee EXy))$ is alternation-free, and the formula $\nu x.\mu y.((p \wedge x) \vee EXy)$ is not. The *alternation-free μ-calculus* (*AFMC*, for short) is the subset of the μ-calculus that contains only the alternation-free formulas. We also refer to the universal and existential fragments of *AFMC*, and denote them by $\forall$*AFMC* and $\exists$*AFMC*, respectively.

A μ-calculus formula is *guarded* if for all $y \in V$, all occurrences of y that are in a scope of a fixed-point quantifier $\lambda \in \{\mu, \nu\}$ are also in a scope of a next modality which is itself in the scope of λ. For example, the formula $\mu y.(p \vee EXy)$ is guarded, and the formula $EX\mu y.(p \vee y)$ is not. We assume that all μ-calculus formulas are guarded. As proved in [28], every μ-calculus formula can be linearly translated to an equivalent guarded one, thus we do not lose generality with our assumption.

The semantics of μ-calculus formulas is defined with respect to Kripke structures. A *Kripke structure* $\mathcal{K} = \langle AP, W, R, w_0, L\rangle$ consists of a set AP of atomic propositions, a set W of states, a total transition relation $R \subseteq W \times W$, an initial state $w_0 \in W$, and a labeling $L : W \rightarrow 2^{AP}$ that maps each state to the set of atomic propositions true in that state.

Given a Kripke structure $\mathcal{K} = \langle AP, W, R, w_0, L\rangle$ and a set $\{y_1, \dots, y_n\}$ of free variables, a *valuation* $\mathcal{V} : \{y_1, \dots, y_n\} \rightarrow 2^W$ is an assignment of subsets of W to the variables in $\{y_1, \dots, y_n\}$. For a valuation $\mathcal{V}$, a variable y, and a set $W' \subseteq W$, denote by $\mathcal{V}[y \leftarrow W']$ the valuation mapping y to W', and y' to $\mathcal{V}(y')$ for all $y' \neq y$. A formula φ with atomic propositions from AP and free variables $\{y_1, \ldots, y_n\}$ is interpreted over the structure $\mathcal{K}$ as a mapping $\varphi^{\mathcal{K}}$ from valuations to 2^W. Thus, $\varphi^{\mathcal{K}}(\mathcal{V})$ denotes the set of states that satisfy φ under the valuation $\mathcal{V}$. The mapping $\varphi^{\mathcal{K}}$ is defined inductively as follows:

- $true^{\mathcal{K}}(\mathcal{V}) = W$ and $false^{\mathcal{K}}(\mathcal{V}) = \emptyset$.
- For $p \in AP$, let $p^{\mathcal{K}}(\mathcal{V}) = \{w \in W \mid p \in L(w)\}$ and $(\neg p)^{\mathcal{K}}(\mathcal{V}) = \{w \in W \mid p \notin L(w)\}$.
- $(\varphi_1 \wedge \varphi_2)^{\mathcal{K}}(\mathcal{V}) = \varphi_1^{\mathcal{K}}(\mathcal{V}) \cap \varphi_2^{\mathcal{K}}(\mathcal{V})$.
- $(\varphi_1 \vee \varphi_2)^{\mathcal{K}}(\mathcal{V}) = \varphi_1^{\mathcal{K}}(\mathcal{V}) \cup \varphi_2^{\mathcal{K}}(\mathcal{V})$.
- $(AX\varphi)^{\mathcal{K}}(\mathcal{V}) = \{w \in W \mid \forall w'.\text{ if } (w, w') \in R \text{ then } w' \in \varphi^{\mathcal{K}}(\mathcal{V})\}$.
- $(EX\varphi)^{\mathcal{K}}(\mathcal{V}) = \{w \in W \mid \exists w'.(w, w') \in R \text{ and } w' \in \varphi^{\mathcal{K}}(\mathcal{V})\}$.
- $(\mu x.\varphi)^{\mathcal{K}}(\mathcal{V}) = \bigcap\{W' \subseteq W \mid \varphi^{\mathcal{K}}(\mathcal{V}[x \leftarrow W']) \subseteq W'\}$.
- $(\nu x.\varphi)^{\mathcal{K}}(\mathcal{V}) = \bigcup\{W' \subseteq W \mid W' \subseteq \varphi^{\mathcal{K}}(\mathcal{V}[x \leftarrow W'])\}$.

By the Knaster–Tarski theorem, the required fixed-points always exist. For a sentence, no valuation is required. For a state $w \in W$ of the Kripke structure $\mathcal{K}$, and a sentence φ, we write $\mathcal{K}, w \models \varphi$ iff $w \in \varphi^{\mathcal{K}}$.

The μ-calculus is a very expressive logic, and most temporal logics used in specification and verification can be translated to the μ-calculus. Moreover, the temporal logics LTL and universal CTL can be translated to the universal fragment of the μ-calculus [8,10].

3. Satisfiability and validity

The satisfiability problem for a μ-calculus sentence φ is to decide whether there is a Kripke structure $\mathcal{K}$ and a state w in it such that $\mathcal{K}, w \models \varphi$. The validity problem is to decide whether $\mathcal{K}, w \models \varphi$ for all $\mathcal{K}$ and w. Note that φ is satisfiable iff $\neg\varphi$ is not valid. The satisfiability and validity problems for μ-calculus and its alternation-free fragment are EXPTIME-complete [2,16]. In this section we study the satisfiability and validity problems for the universal and existential fragments.

Theorem 1. *The satisfiability problem for* $\forall MC$ *and* $\forall AFMC$ *is PSPACE-complete.*

Proof. For a $\forall MC$ formula φ, let $[\varphi]$ denote the linear-time μ-calculus formula [25] obtained from φ by omitting all its universal path quantifiers. It is easy to see that φ is satisfiable iff $[\varphi]$ is satisfiable. Indeed, a model for $[\varphi]$ is also a model for φ, and each path in a model for φ is a model for $[\varphi]$. Since the satisfiability problem for the linear-time μ-calculus and its alternation-free fragment is PSPACE-complete [39], so is the satisfiability problem for $\forall MC$ and $\forall AFMC$. $\square$

Since both $\exists MC$ and $\exists AFMC$ subsume propositional logic, the satisfiability problem for these logics is clearly hard for NP. We show that the satisfiability problem is in fact NP-complete. To show membership in NP, we prove a *linear-size model property* for $\exists MC$.

Lemma 2. *Let* φ *be a formula of* $\exists MC$. *If* φ *is satisfiable, then it has a model with at most* $O(|\varphi|)$ *states and* $O(|\varphi|)$ *transitions.*

Proof. The proof is similar to the one used in [27] to show a linear-size model property for $\exists$CTL. We proceed by induction on the structure of $\exists MC$ formulas. With each $\exists MC$ formula φ, we associate a set S_φ of models (Kripke structures) that satisfy φ. We define S_φ by structural induction. The states of the models in S_φ are labeled by both the atomic propositions in AP and the variables free in φ. We use $S_{\varphi_1} \rightarrow S_{\varphi_2}$ to denote the set of models obtained by taking a model M_1 from S_{φ_1}, a model M_2 from S_{φ_2}, adding a transition from the initial state of M_1 to the initial state of M_2, and fixing the initial state to be the one of M_1.

For models M_1 and M_2 such that M_1 and M_2 agree on the labeling of their initial states, let $M_1 \oplus M_2$ denote the model obtained by fixing the initial state to be the initial state of M_1, redirecting transitions to the initial state of M_2

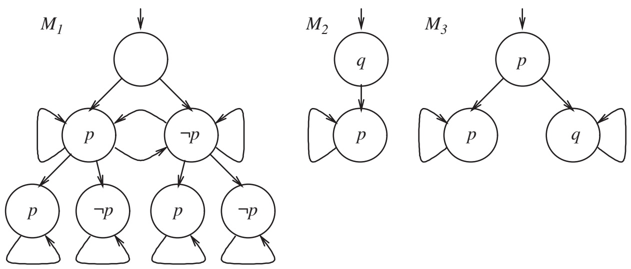

Fig. 2. Linear-size models.

into the initial state of M_1, and adding transitions from the initial state of M_1 to all the successors of the initial state of M_2. We use $S_{\varphi_1} \cap^* S_{\varphi_2}$ to denote the set of models

$$\{M_1 \oplus M_2 \mid M_1 \in S_{\varphi_1}, M_2 \in S_{\varphi_2}, \text{ and } M_1 \text{ and } M_2 \text{ agree on the labeling of their initial states}\}.$$

Finally, we use $S_{\varphi(\#)} \downarrow$, where # is an atomic proposition not in AP, to denote the set of models obtained from a model in $S_{\varphi(\#)}$ by adding transitions from states labeled by # to all the successors of the initial state, and removing # from the labels of states. We can now define S_φ as follows. Note that we do not consider the case where $\varphi = x$, for $x \in V$, as we assume that φ is a sentence.

- $S_{\mathbf{true}}$ is the set of all one-state models over AP.
- $S_{\mathbf{false}} = \emptyset$.
- S_p, for $p \in AP$, is the set of all one-state models over AP in which p holds.
- $S_{\neg p}$, for $\neg p \in AP$, is the set of all one-state models over AP in which p does not hold.
- $S_{\varphi_1 \vee \varphi_2} = S_{\varphi_1} \cup S_{\varphi_2}$.
- $S_{\varphi_1 \wedge \varphi_2} = S_{\varphi_1} \cap^* S_{\varphi_2}$.
- $S_{EX\varphi_1} = S_{\mathbf{true}} \to S_{\varphi_1}$.
- $S_{\mu x.\varphi_1(x)} = S_{\varphi_1(\varphi_1(\mathit{false}))}$.
- $S_{\nu x.\varphi_1(x)} = S_{\varphi_1(\# \wedge(\varphi_1(\mathit{true})))} \downarrow$.

For example, if $AP = \{p\}$, and $\varphi = \nu x.\varphi_1(x)$ with $\varphi_1(x) = EX(p \wedge x) \wedge EX(\neg p \wedge x)$, then $\varphi_1(\# \wedge (\varphi_1(\mathit{true}))) = EX(p \wedge \# \wedge EX p \wedge EX\neg p) \wedge EX(\neg p \wedge \# \wedge EX p \wedge EX\neg p)$, and S_φ contains the two models obtained from the model M_1 described in Fig. 2 by labeling the initial state by either p or $\neg p$. Also, if $AP = \{p, q\}$ and $\varphi = EX p \wedge (\mu y.q \vee (p \wedge EX y))$, then S_φ contains the models obtained from the models M_2 and M_3 described in Fig. 2 by completing labels of p or q that are left unspecified with all possible valuations.

The models in S_φ are "economical" with respect to states that are required for satisfaction of formulas that refer to the strict future. For example, the initial state of models in $S_{EX\varphi_1}$ has a single successor that satisfies φ_1, and models in $S_{\mu y.\varphi(y)}$ that do not satisfy $\varphi(\mathit{false})$ in the initial state are required to satisfy $\varphi(\mathit{false})$ in a successor state.

It is not hard to prove, by induction on the structure of φ, that each model in S_φ has $O(|\varphi|)$ states and $O(|\varphi|)$ transitions. Recall that we are using a dag representation of the formulas, so the sizes of the models remain linear.

We now prove, by an induction on the structure of φ, the following three claims.

(1) For every model $M \in S_\varphi$, we have that M satisfies φ.
(2) For every model M that satisfies φ, there is a model $M' \in S_\varphi$ such that M and M' agree on the labeling of their initial states.
(3) For every model $M \in S_\varphi$, and any Kripke structure M' that agrees with M on the labeling of the initial states, $M \oplus M'$ and $M' \oplus M$ are both models of φ.

Note that Claim (2) implies that if φ is satisfiable, then S_φ is not empty. Thus, the two claims together imply that if φ is satisfiable, then it has a satisfying model in S_φ, which is guaranteed to be of size linear in $|\varphi|$. Further, once we prove Claim (1), we get Claim (3) since all formulas are existential, and therefore closed under embeddings of models.

The proof for φ of the form *true*, *false*, p, $\neg p$, and $\varphi_1 \vee \varphi_2$ are easy. For the other cases, we proceed as follows.

- Let $\varphi = \varphi_1 \wedge \varphi_2$. By induction hypothesis, all models in S_{φ_1} and S_{φ_2} satisfy φ_1 and φ_2, respectively. Take a model M from S_φ, and let $M = M_1 \oplus M_2$ where M_1 is in S_{φ_1} and M_2 is in S_{φ_2}. By claim (3), we know that $M_1 \oplus M_2$ is a model for φ_1 as well as φ_2. Hence, it is a model of φ.

  Claim (2) is straightforward: any model M of φ is also a model of φ_1. Therefore, there exists a model in S_{φ_1} that agrees with M on the labeling of the internal states, and thus there exists a model in S_φ with the same labeling.
- Let $\varphi = EX\varphi_1$. By the induction hypothesis, all models in S_{φ_1} satisfy φ_1. Hence, (1) follows immediately from the definition of S_φ. To see (2), consider a model M that satisfies φ. Since $S_{\mathbf{true}}$ is the set of all one-state models over AP, it contains a model M' that agrees with M on the labeling of their initial states.
- Let $\varphi = \mu x.\varphi_1(x)$. By the semantics of μ-calculus, a model that satisfies $\varphi_1(\varphi_1(\mathit{false}))$, satisfies φ as well. Hence, (1) follows immediately from the definition of S_φ. To see (2), consider a model M that satisfies φ. Since μ-calculus has a finite-model property [2,16], we can without loss of generality consider M to be finite. In this case, for some $i > 0$, the model M satisfies $\varphi_1^i(\mathit{false})$. We construct a model M' that satisfies $\varphi_1(\varphi_1(\mathit{false}))$ and agrees with M on the label of the initial state. Let # be a proposition not in AP, and consider the formula $\varphi_1(\# \wedge \varphi_1^{i-1}(\mathit{false}))$. By attributing by # exactly all the states of M that satisfy $(\varphi_1^{i-1}(\mathit{false}))$, the model M can be attributed by # to satisfy $\varphi_1(\# \wedge \varphi_1^{i-1}(\mathit{false}))$. Moreover, since φ is guarded, the initial state of M is attributed by # only if there is a self-loop in the initial state. Such a self-loop can be unwound, so we can assume that the initial state of M is not attributed by #.

Next, we use the observation that if φ is satisfiable, so is $\varphi_1(\mathit{false})$. If not, then for any Kripke structure $\mathcal{K}$, we claim that $\varphi^{\mathcal{K}} = \emptyset$, which is a contradiction to the satisfiability of φ. To see the claim, notice that by the semantics of the μ-calculus, $\varphi_1^{\mathcal{K}}(\mathcal{V}[x \leftarrow \emptyset) = \emptyset$, and therefore $\emptyset$ is a fixpoint (hence the least fixpoint) of φ_1.

Let N be a model of $\varphi_1(\mathit{false})$. The structure M' is obtained from M by replacing all states attributed by # with N (i.e., all transitions leading into a state attributed by # are redirected to the initial state of N). Then, M' is a model of $\varphi_1(\varphi_1(\mathit{false}))$, and agrees with M in the labeling of the initial states.

- Let $\varphi = \nu x.\varphi_1(x)$. By the semantics of μ-calculus, a model M satisfies φ iff M satisfies $\varphi_1^i(\mathit{true})$, for all $i \geqslant 0$. Consider a model $M \in S_\varphi$. By the definition of S_φ, the model M satisfies $\varphi_1(\mathit{true})$, and the states attributed # satisfy $\varphi_1(\mathit{true})$ as well. Since $\varphi_1(\mathit{true})$ is existential, the states attributed # continue to satisfy $\varphi_1(\mathit{true})$ after the new edges are added. In fact, it is not hard to see that after the new edges are added, the states attributed # also satisfy $\varphi_1(\#)$. Thus, for all $i \geqslant 1$, the model M can be unfolded $(i-1)$ times to show M satisfies $\varphi_1^i(\mathit{true})$, and we are done. To see (2), let M be a model of φ and let # be a proposition not in AP. Then, M satisfies $\varphi_1(\varphi_1(\mathit{true}))$, and by attributing by # exactly all the states that satisfy $\varphi_1(\mathit{true})$, it can be attributed by # to satisfy $\varphi_1(\# \wedge \varphi_1(\mathit{true}))$. As in the previous case, since φ is guarded, we can ensure that this leaves the labeling of the initial state unchanged, possibly after unwinding a self-loop in the initial state. In addition, adding transitions from states attributed by # to all successors of the initial state, leaves the label of the initial state unchanged, and thus results in a model in S_φ that agrees with M on the labeling of their initial states.

Notice that the proof requires that the only allowed modalities are existential in order to prove Claim (3). Claim (3) is required in order to show the case $\varphi = \varphi_1 \wedge \varphi_2$. □

Note that the μ-calculus with both universal and existential next modalities has only an exponential-size model property (there is a μ-calculus sentence φ such that the smallest Kripke structure that satisfies φ is of size exponential in $|\varphi|$). Thus, the linear-size model property crucially depends on the fact that the only next modality that is allowed is the existential one. In fact, the construction for conjunction in the above proof works only because the allowed next modality is existential, and in general, the combination $M \oplus M'$ of a model for φ_1 and a model for φ_2 is not a model for $\varphi_1 \wedge \varphi_2$. The linear-size model theorem shows that the satisfiability problem for $\exists MC$ and $\exists AFMC$ is in NP.

Theorem 3. *The satisfiability problem for* $\exists MC$ *and* $\exists AFMC$ *is NP-complete.*

Since a formula φ is satisfiable iff $\neg\varphi$ is valid, and since negating an $\exists MC$ formula results in a $\forall MC$ formula and vice versa, the following theorem is an immediate corollary of Theorems 1 and 3.

Theorem 4. *The validity problem is co-NP-complete for* $\forall MC$ *and* $\forall AFMC$, *and is PSPACE-complete for* $\exists MC$ *and* $\exists AFMC$.

Since ECTL (respectively, ECTL*) formulas can be converted into $\exists MC$ formulas with a linear (respectively, exponential) blow-up [10], we get again the linear model theorem for ECTL [27] and an exponential size model theorem for ECTL*.

4. Model checking

The model-checking problem for the μ-calculus is to decide, given a Kripke structure $\mathcal{K}$ and a μ-calculus formula φ, the set of states in $\mathcal{K}$ that satisfy φ. In this section we study the model-checking problem for the universal and existential fragments of the μ-calculus. We show that in contrast to the case of satisfiability, the model-checking problem for the restricted fragments is not easier than the model-checking problem for the μ-calculus, and the same is true for the alternation-free fragments.

The model-checking problem for the μ-calculus is closely related to the problem of determining the winner in games on And-Or graphs. We first review here some definitions that will be used in the reduction of the model-checking problem for the full μ-calculus to the model-checking problem for the fragments. A *two-player game graph* is a directed graph $G = \langle V, E\rangle$, with a partition $V_e \cup V_u$ of V. The game is played between two players, player 1 and player 2. A position of the game is a vertex $v \in V$. At each step of the game, if the current position v is in V_e, then player 1 chooses the next position among the vertices in $\{w \mid \langle v, w\rangle \in E\}$. Similarly, if $v \in V_u$, then player 2 chooses the next position among the vertices in $\{w \mid \langle v, w\rangle \in E\}$. The game continues for an infinite number of steps, and induces an infinite path $\pi \in V^\omega$. The winner of the game depends on different conditions we can specify on words in V^ω. The simplest game is *reachability*. Then, the winning condition is some vertex $t \in V$, and player 1 wins the game if π eventually reaches the vertex t. Otherwise, player 2 wins. A richer game is *parity*. In parity games, there is a function $C : V \to \{0, \ldots, k-1\}$ that maps each vertex to a *color* in $\{0, \ldots, k-1\}$. Player 1 wins the parity game if the maximal color that repeats in π infinitely often is even.

A *strategy* for player 1 is a function $\xi_1 : V^* \times V_e \to V$ such that for all $u \in V^*$ and $v \in V_e$, we have $\xi_1(u \cdot v) \in \{w \mid \langle v, w\rangle \in E\}$. A strategy for player 2 is defined similarly, as $\xi_2 : V^* \times V_u \to V$. For a vertex $s \in V$, and strategies ξ_1 and ξ_2 for player 1 and player 2, respectively, the *outcome* of ξ_1 and ξ_2 from s, denoted $\pi(\xi_1, \xi_2)(s)$, is the trace $v_0, v_1, \ldots \in V^\omega$ such that $v_0 = s$ and for all $i \geqslant 0$, we have $v_{i+1} \in \xi_1(v_0 \ldots v_{i-1}, v_i)$ if $v_i \in V_e$, and $v_{i+1} \in \xi_2(v_0 \ldots v_{i-1}, v_i)$ if $v_i \in V_u$. Finally, a vertex $s \in V$ is winning for player 1 if there is a strategy ξ_1 of player 1 such that for all strategies ξ_2 of player 2, the outcome $\pi(\xi_1, \xi_2)(s)$ is winning for player 1. When G has an initial state s, we say that player 1 wins the game on G if s is winning for player 1 in G.

We start by considering the *system complexity* of the model-checking problem for the universal and existential fragments of the μ-calculus; that is, the complexity of the problem in terms of the model, assuming the formula is fixed. As discussed in Section 1, the system complexity of *AFMC* model checking is PTIME-complete, and hardness in PTIME [20] crucially depends on the fact that an *AFMC* formula of a fixed size can specify an unbounded number of switches between universal and existential branching modes. As we prove in Theorem 5 below, the setting of model checking enables us to trade an unbounded number of switches between universal and existential branching modes by an unbounded number of switches between disjunctions and conjunctions. The idea is that in model checking, unlike in satisfiability, we can consider models in which the successors of a state are labeled in a way that enables the formula to directly refer to them. Then, it is possible to replace the existential next modality by a disjunction over all successors, and it is possible to replace the universal next modality by a conjunction that refers to each successor.

Theorem 5. *The complexity and system complexity of* $\forall AFMC$ (*so, also of* $\exists AFMC$) *model checking is PTIME-complete.*

Proof. Membership in PTIME follows from the linear time algorithm for *AFMC* [9]. For hardness, we reduce the problem of deciding a winner in a reachability game to model checking of a $\forall AFMC$ formula of a fixed size. Since one can model check a specification φ by checking $\neg\varphi$ and negating the result, the same lower bound holds for $\exists AFMC$.

Deciding reachability in two-player games is known to be PTIME-hard already for acyclic graphs with branching degree two, where universal and existential vertices alternate, and both s and t are in V_e [17]. Given a bipartite and acyclic game graph $G = \langle V, E\rangle$ with branching degree two, a partition of V to V_e and V_u, and two vertices s and t in V_e, we construct a Kripke structure $\mathcal{K} = \langle AP, W, R, w_0, L\rangle$ and a formula in $\forall AFMC$ such that $\mathcal{K}, w_0 \models \varphi$ iff player 1 wins the reachability game on G from state s and with target t. Notice that the $AFMC$ formula $\mu x.t \vee EXAXx$ (where the proposition t holds exactly at the state t) exactly encodes the set of states that can alternating-reach the target t in the graph G, we show how to encode this formula in the existential and universal fragments.

We do the proof in two steps. First, we transform the graph G to another graph G', with some helpful properties, and then we construct the Kripke structure $\mathcal{K}$ from G'. Essentially, in G' each universal vertex is a left or right successor of exactly one existential vertex. Formally, $G' = \langle V', E'\rangle$, where $V' = V_e \cup V'_u$, and V'_u and E' are defined as follows. Let $E_e = E \cap (V_e \times V_u)$ and $E_u = E \cap (V_u \times V_e)$. Recall that each vertex in V_e has two successors. Let $E_e = E_e^l \cup E_e^r$ be a partition of E_e so that for each $v \in V_e$, one successor v_l of v is such that $\langle v, v_l\rangle \in E_e^l$ and the other successor v_r of v is such that $\langle v, v_r\rangle \in E_e^r$. Note that a vertex u may be the left successor of some vertex w_1 and the right successor of some other vertex w_2; thus, $E_e^l(w_1, u)$ and $E_e^r(w_2, u)$. The goal of G' is to prevent such cases.

- $V'_u \subseteq V_u \times \{l, r\} \times V_e$ is such that $(v, l, w) \in V'_u$ iff $(w, v) \in E_e^l$ and $(v, r, w) \in V'_u$ iff $(w, v) \in E_e^r$. Thus, each edge $\langle w, v\rangle \in E_e$ contributes one vertex (v, l, w) or (v, r, w) to V'_u. Intuitively, visits to the vertex (v, l, w) correspond to visit to v in which it has been reached by following the left branch of w, and similarly for (v, r, w) and right.
- $E'_e = \{\langle v, (v_l, l, v)\rangle : \langle v, v_l\rangle \in E_e^l\} \cup \{\langle v, (v_r, r, v)\rangle : \langle v, v_r\rangle \in E_e^r\}$. Also, $E'_u = \{\langle (v, d, w), u\rangle : \langle v, u\rangle \in E_u\}$, and $E' = E'_e \cup E'_u$.

The size of G' is linear in the size of G. Indeed $|V'| = |V_e| + |E_e|$ and $|E'| = |E'_e| + |E'_u| \leqslant |E_e| + 2|E_u|$. It is not hard to see that player 1 can win the game in G iff he can win in G'. Note that the branching degree of G' remains two. The construction of G' ensures that the two successors of an existential vertex v can be referred to unambiguously as the left or the right successor of v.

The graph $G' = \langle V', E'\rangle$, together with s and t, induces the Kripke structure $\mathcal{K} = \langle AP, V', E', s, L\rangle$ described below. The set of atomic propositions $AP = \{t, l\}$. For readability, we also introduce the shorthand r for $\neg l$. The proposition t holds in (and only in) the state t, and the propositions l and r hold in the left and right successor respectively for an existential node. Thus, $l \in L(\langle v, l, w\rangle)$ and $r \in L(\langle v, r, w\rangle)$. Finally, let φ be the $\forall AFMC$ formula

$$\mu y.t \vee (AX(\neg l \vee AXy) \vee AX(\neg r \vee AXy)).$$

It is now easy to see that player 1 can win the reachability game for t from s in G' iff $\mathcal{K}, s \models \varphi$. This construction also proves PTIME-completeness for model checking $\exists AFMC$, since $\mathcal{K}, s \models \varphi$ iff $\mathcal{K}, s \not\models \neg\varphi$, and φ is in $\exists AFMC$ iff $\neg\varphi$ is in $\forall AFMC$. □

We now consider the model-checking problem for $\forall MC$ and $\exists MC$. We show that there is a polynomial time reduction from an instance of the μ-calculus model-checking problem to an instance of the $\forall MC$ (or $\exists MC$) model-checking problem. This shows that the model-checking problem for the restricted logics are polynomial time equivalent to the model-checking problem for the full μ-calculus: the existence of a polynomial time algorithm for the restricted logic implies the existence of a polynomial time algorithm for the full logic. We say a problem P is *as hard as* a problem P' if P and P' are polynomial time equivalent, that is, there are polynomial reductions from instances of P to instances of P' and vice versa.

Theorem 6. *The model-checking problem for $\forall MC$ (so, also for $\exists MC$) is as hard as the model-checking problem for the μ-calculus.*

Proof. The idea is similar to the proof of Theorem 5, only that instead of talking about winning a reachability game, we talk about winning a *parity game* [13], to which and from which μ-calculus model checking can be reduced [14]. Without loss of generality, we assume that existential and universal vertices alternate (the game graph is bipartite), and each node has exactly two successors. We also assume that each vertex in V_u has the same color as the incoming existential nodes (otherwise, we can duplicate nodes and get an equivalent game with this property). We assume that each vertex of G is labeled by the color $C(v)$, thus we can refer to G as a Kripke structure with $AP = \{0, \ldots, k-1\}$:

the proposition i holds at vertex v iff $C(v) = i$. From [13], player 1 wins the parity game G at an existential vertex $s \in V_e$ iff

$$G, s \models \lambda_{k-1}x_{k-1} \ldots \mu x_1.\nu x_0. \left(\bigvee_{i\in[0\ldots(k-1)]} (i \wedge EXAXx_i) \right), \tag{1}$$

where $\lambda_n = \nu$ if n is even, and $\lambda_n = \mu$ if n is odd.

The formula above uses both universal and existential next modalities. By transforming G to a Kripke structure $\mathcal{K}$ as in the proof of Theorem 5, we can use left and right labels to vertices in the graph and use only one type of branching mode. Formally, let $\mathcal{K}$ be the Kripke structure induced by G. Then, player 1 wins the parity game in G at a node s iff

$$\mathcal{K}, s \models \lambda_{k-1}x_{k-1} \cdots \mu x_1.\nu x_0. \left(\bigvee_{i\in[0\ldots(k-1)]} (i \wedge ((AX(\neg l \vee AXx_i)) \vee (AX(\neg r \vee AXx_i)))) \right). \tag{2}$$

Since $\mathcal{K}, s \models \varphi$ iff $\mathcal{K}, s \not\models \neg\varphi$, and φ is in $\exists MC$ iff $\neg\varphi$ is in $\forall MC$, the construction also proves the result for $\exists MC$. $\square$

If the syntax of the μ-calculus is equipped with next modalities parameterized by action labels, then the above result follows immediately, because there is no distinction between existential and universal next modalities. Our proof shows that the result follows even if no such labeling is available.

5. Implication

The *implication problem* for a logic asks if one specification logically implies another specification; formally, given formulas φ and ψ of the logic, if the formula $\varphi \to \psi$ is valid. It arises naturally in *modular verification* [27,35], where the antecedent of the implication is the assumption about the behavior of a component's environment, and the consequent is a guarantee about the behavior of the component. For logics closed under negation, the implication problem is equivalent to validity: a formula φ is valid iff *true* $\to \varphi$. Thus, the implication problem for the μ-calculus is EXPTIME-complete. However, for the existential and universal fragments of the μ-calculus, this is not the case: the implication problem combines both universal and existential formulas, and is more general than satisfiability or validity.

Theorem 7. *The implication problem for $\exists MC$ and $\exists AFMC$ (so, also for $\forall MC$ and $\forall AFMC$) is EXPTIME-complete.*

Proof. For formulas φ_1 and φ_2 of $\exists MC$, we have that $\varphi_1 \to \varphi_2$ iff the formula $\varphi_1 \wedge \neg\varphi_2$ is not satisfiable. Membership in EXPTIME follows from the complexity of the satisfiability problem for the μ-calculus. Note that $\neg\varphi_2$ is a formula of $\forall MC$, thus we cannot apply the results of Section 3.

To prove hardness in EXPTIME, we do a reduction from the satisfiability problem of *AFMC*, proved to be EXPTIME-hard in [16]. Given an *AFMC* formula ψ, we construct a formula φ_A of $\forall AFMC$ and a formula φ_E of $\exists AFMC$ such that the conjunction $\varphi = \varphi_E \wedge \varphi_A$ is satisfiable iff ψ is satisfiable. For simplicity, we assume that ψ is satisfied iff it is satisfied in a tree of branching degree two. Note that while our assumption does not hold for all *AFMC* formulas, the EXPTIME-hardness of the satisfiability problem for *AFMC* holds already for such formulas, which is sufficiently good for our goal here.

Intuitively, the formula φ_E would require the states of models of φ to be attributed by directions so that at least one successor is labeled by l and at least one successor is labeled by r. In addition, φ_A would contain a conjunct that requires each state to be labeled by at most one direction. Thus, states that are labeled by l cannot be labeled by r, and vice versa. Then, the other conjunct of φ_A is obtained from ψ by replacing an existential next modality by a disjunction over the successors of a state.

Formally, the formula

$$\varphi_E = \nu y.EX(l \wedge y) \wedge EX(r \wedge y)$$

requires each state (except for the initial state) to have at least two successors, labeled by different directions, and the formula

$$\varphi_A^1 = \nu y.((\neg l) \vee (\neg r)) \wedge AXy$$

requires each state to be labeled by at most one direction.

Then, the formula φ_A^2 is obtained from ψ by replacing a subformula of the form $EX\theta$ by the formula $AX(r \vee \theta) \vee AX(l \vee \theta)$. We show that for every ψ such that ψ is satisfiable iff it is satisfiable in a model of branching degree two, we have that ψ is satisfiable iff $\varphi_E \wedge \varphi_A^1 \wedge \varphi_A^2$ is satisfiable. First, if ψ is satisfiable, then there is a tree of branching degree two that satisfies it. This tree can be attributed with l and r so that it satisfies the formula $\varphi_E \wedge \varphi_A^1 \wedge \varphi_A^2$, by labeling the left successor of each node with $\{l, \neg r\}$ and the right successor of each node with $\{\neg l, r\}$. On the other hand, assume that the formula $\varphi_E \wedge \varphi_A^1 \wedge \varphi_A^2$ is satisfiable in a model M. The subformula $\varphi_E \wedge \varphi_A^1$ guarantees that each state of M has at least one successor that is not labeled l and at least one successor that is not labeled r. Accordingly, each subformula of the form $AX(r \vee \theta) \vee AX(l \vee \theta)$ is satisfied in a state w of M iff w has a successor that satisfies θ, thus w satisfies $EX\theta$. Hence, the model M also satisfies ψ. $\square$

The above proof constructs, given a formula φ of the μ-calculus, two formulas φ_E and φ_A such that φ_E is an existential formula, φ_A is a universal formula, and φ is satisfiable iff $\varphi_E \wedge \varphi_A$ is satisfiable. However, one cannot in general construct formulas φ_E and φ_A such that $\varphi_E \wedge \varphi_A$ is equivalent to φ. This can be proved considering two states of a Kripke structure that are similar, but not bisimilar, and the formula of μ-calculus that distinguishes them.

The proofs of Theorems 6 and 7 make use of the fact that the successors of a state in a Kripke structure can be attributed by new atomic propositions, corresponding to the successor's direction. Alternatively, one could have used the fact that the existential and universal fragments of μ-calculus embody the μ-calculus over binary trees (see [1]). We find our direct proof more illuminating, especially for readers not familiar with μ-calculus over binary trees.

Note that the implication problem for $\forall$CTL* and $\exists$CTL* is EXPSPACE-complete [27], and hence easier than the satisfiability problem for CTL*, which is 2EXPTIME-complete. The above construction does not work for $\exists$CTL*, as the formula φ_E used to label the states of a model by directions specifies an unbounded number of unfoldings of the structure. On the other hand, the number of unfoldings expressible by an $\exists$CTL* formula is bounded by the size of the formula; thus, the formula φ_E does not have an equivalent formula in $\exists$CTL*.

6. The alternation hierarchy

The existential and universal μ-calculi still contain the second source of complexity of the μ-calculus: alternation of fixpoint operators.

A formula ϕ is in the classes Σ_0^μ and Π_0^μ iff it contains no fixpoint operators. The class Σ_{n+1}^μ is constructed by taking $\Sigma_n^\mu \cup \Pi_n^\mu$ and closing under (i) boolean operations, (ii) modal operations EX and AX, and (iii) $\mu x.\phi$, for $\phi \in \Sigma_{n+1}^\mu$, and (iv) substitution of $\phi' \in \Sigma_{n+1}^\mu$ for a free variable of $\phi \in \Sigma_{n+1}^\mu$ provided that no free variable of ϕ' is captured by ϕ. The class Π_{n+1}^μ is $\{\phi \mid \neg\phi \in \Sigma_{n+1}^\mu\}$. The *alternation depth* of a formula φ is the least n such that $\varphi \in \Sigma_{n+1}^\mu \cap \Pi_{n+1}^\mu$ [34,3,15]. A formula φ is $(n+1)$-*strict* if it is equivalent to some formula in Σ_{n+1}^μ and there is no formula φ' in Σ_n^μ that is equivalent to φ. The alternation hierarchy is strict if for each $n > 0$, there is an n-strict formula.

It is known [3,4] that the μ-calculus alternation hierarchy is strict, that is, there is a class of transition systems such that for each $n \geqslant 0$, there is a formula ϕ_n in the class Σ_n^μ, such that the semantics of ϕ_n is not equivalent to the semantics of any Σ_{n-1}^μ formula. We show a similar theorem for the restricted logics.

We define the classes $\Sigma_n^{\exists MC}$, $\Pi_n^{\exists MC}$ for $\exists MC$ (respectively, $\Sigma_n^{\forall MC}$, $\Pi_n^{\forall MC}$ for $\forall MC$) as for the classes Σ_n^μ and Π_n^μ, but replacing condition (ii) by (ii′) modal operation EX (respectively, (ii′) modal operation AX). We show that the alternation hierarchy of $\exists MC$ (and so also $\forall MC$) is strict. This shows that restricting the expressive power in terms of the next modalities does not restrict the expressiveness of alternating fixpoint operators.

Theorem 8. *The existential μ-calculus $\exists MC$ alternation hierarchy is strict. The universal μ-calculus $\forall MC$ alternation hierarchy is strict.*

The proof of the theorem takes the proof in [4] and adds to it the same construction as in Theorem 6.

Before we proceed, we have to convert the results of [4] that talk about *transition systems* to corresponding results on Kripke structures. In particular, the models considered in [4] are labeled transition systems $(S, AP, \to)$ with a set S of states, AP of labels, and $\to \subseteq S \times AP \times S$ of labeled transitions. Moreover, the μ-calculus is defined using labels on the next-modalities rather than on the states, that is, the formulas are inductively defined as

$$\varphi ::= true \mid false \mid x \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \langle p\rangle\varphi_1 \mid [p]\varphi_1 \mid \mu x.\varphi_1 \mid \nu x.\varphi_1,$$

for variables x and labels $p \in AP$. Let us call this variant the *labeled μ-calculus*. In order to apply the results, we must be able to translate labeled μ-calculus formulas over transition systems to μ-calculus formulas over Kripke structures. For our purposes, we restrict attention to *binary* transition systems, where each state has two successor states, one reachable with an edge labeled l and the second with an edge labeled r. Each binary transition system $\mathcal{T}$ can be translated to a Kripke structure $\mathcal{K}(\mathcal{T})$ over the atomic propositions $\{l, r\}$, by moving the labels on the edges to labels on the successor nodes. We claim the following results. For any labeled μ-calculus formula φ over a binary transition system, there exists an $\exists MC$ formula $T(\varphi)$ over Kripke structures with the same alternation depth such that $s \in \varphi^{\mathcal{T}}$ iff $s \in T(\varphi)^{\mathcal{K}(\mathcal{T})}$. The translation simply pushes the labels on the next-modalities to the successor states, and is obtained by inductively replacing each occurrence of $\langle l\rangle\varphi_1$ to $EX(l \wedge T(\varphi_1))$, and $\langle r\rangle\varphi_1$ to $EX(r \wedge T(\varphi_1))$. For example, the formula $\nu x.\langle l\rangle x \wedge \langle r\rangle x$ is translated to $\nu x.EX(l \wedge x) \wedge EX(r \wedge x)$. Conversely, for any μ-calculus formula ψ on Kripke structures over the atomic propositions $\{l, r\}$ (not just existential), we can show there exists a labeled μ-calculus formula φ over binary transition systems with the same alternation depth such that $T(\varphi)$ is equivalent to ψ on all Kripke structures. As expected, we first translate ψ to $T(\psi)$ by recursively replacing $EX\varphi_1$ to $\langle l\rangle(l \wedge T(\varphi_1)) \vee \langle r\rangle(r \wedge T(\varphi_1))$ and $AX\varphi_1$ to $\langle l\rangle(l \wedge T(\varphi_1)) \wedge \langle r\rangle(r \wedge T(\varphi_1))$. Then, this transformed formula is written in disjunctive normal form [21], and each conjunction $l \wedge r$ is replaced by *false*. Finally, the propositions l and r are replaced by *true* to get the formula φ. Notice that in both translations, the alternation depth of the formula is not affected.

As a result, if φ is a strict formula on binary transition systems, then its translation $T(\varphi)$ is a strict formula on Kripke structures. Thus, we can use the results of [4] via this translation, and we omit this explicit intermediate translation step in what follows for clarity.

We now recollect the proof of [3,4], where the alternation hierarchy of the μ-calculus is shown by reducing formulas of *μ-arithmetic* to it, and using the strictness for μ-arithmetic formulas. The μ-arithmetic is a logic obtained by adding fixpoint operators to formulas in first-order arithmetic. For a full definition see [32]. We define syntactic alternation classes for the μ-arithmetic in a way similar to the syntactic alternation classes of the μ-calculus. Set variables and first-order formulas are Σ_0 and Π_0 formulas. Then, Σ_{n+1} formulas and set terms are constructed from $\Sigma_n \cup \Pi_n$ formulas and set terms by closing under the first order connectives $\wedge$, $\vee$, $\forall$, $\exists$, and $\in$, and forming the formula $\mu(x, X)\phi$ for $\phi \in \Sigma_{n+1}$. Also, if ϕ is in Σ_n, then $\neg\phi$ is in Π_n. The definition of *n-strictness* for μ-arithmetic formulas is similar to the definition in the case of μ-calculus. We say that a set $X \subseteq \mathbb{N}$ is arithmetic Σ_n-hard if it is definable by a Σ_n formula but not definable by a $\Sigma_{n-1} \cup \Pi_{n-1}$ formula. It is shown in [32] that the hierarchy of sets of integers definable by Σ_n formulas of the μ-arithmetic is a strict hierarchy.

A Kripke structure $\mathcal{K} = \langle AP, W, R, w_0, L\rangle$ is *recursively presented* if AP and W are (recursively codeable as) recursive sets of numbers, and R and L are recursive. For a μ-calculus formula $\varphi \in \Sigma_n^\mu$, [4] shows that the set of states $\varphi^{\mathcal{K}}$ on any recursively presented Kripke structure $\mathcal{K}$ of branching degree two is an arithmetic Σ_{n-1} set of integers, that is, for the μ-calculus formula φ in Σ_n^μ and the recursively presented Kripke structure $\mathcal{K}$, we can write a Σ_n formula $\psi(z)$ of μ-arithmetic with a free individual variable z such that $\mathcal{K}, s \models \varphi$ iff $\psi(s)$ holds.

Conversely, for any arithmetic Σ_n formula $\psi(z)$ in μ-arithmetic with a free individual variable z, there is a recursively presented Kripke structure $\mathcal{K}$ of branching degree two and a Σ_{n+1}^μ formula φ of the μ-calculus such that $\psi(s)$ iff $\mathcal{K}, s \models \varphi$. Thus, if $\psi(z)$ is not a Σ_{n-1} definable set, then $\varphi^{\mathcal{K}}$ is not Σ_n^μ definable, that is, if $\psi(z)$ is Σ_n-hard, then so is $\varphi^{\mathcal{K}}$. Now, take an n-strict Σ_n formula $\psi(z)$ of μ-arithmetic, and construct the recursively presented Kripke structure T and μ-calculus formula φ in Σ_{n+1} as above. From this, we can recursively construct a parity game G of rank $n+1$, consisting of states (s, Ψ) of pairs of states s of T and subformulas Ψ of φ, such that $G, (s, \Psi) \models \Phi_{n+1}$ iff $T, s \models \varphi$. Here, Φ_n is the winning condition from Eq. (1) in a parity game with n indices [13]. Thus, Φ_{n+1}^G is also an arithmetic Σ_n-hard set. This is used in [3,4] to show that the μ-calculus alternation hierarchy is strict.

Now, using the same construction as in Theorem 6, we label the edges in G with directions to get the game G' and a formula Ψ_{n+1} of $\forall MC$ (Eq. (2)) such that $G, s \models \Phi_{n+1}$ iff $G', s \models \Psi_{n+1}$. This shows that $\Psi_{n+1}^{G'}$ encodes an arithmetic

Σ_n set. Since the alternation hierarchy is strict for μ-arithmetic formulas, this also shows that the alternation hierarchy is strict for $\forall MC$ and hence $\exists MC$ formulas (since formulas of arbitrary arithmetic complexity can be encoded).

7. Discussion

We studied the complexity of the satisfiability, validity, model-checking, and implication problems for the universal and existential fragments of the μ-calculus and its alternation-free fragment. We proved that the linear-size model property, which is known for ∃CTL, holds also for $\exists MC$. Interestingly, the property does not hold for ∃CTL*, which is less expressive than $\exists MC$. Thus, the picture we obtain for $\exists MC$ and $\exists AFMC$ is different than the one known for ∃CTL* and ∃CTL. For the universal fragments $\forall MC$ and $\forall AFMC$, the picture does agree with the one known for ∀CTL* and ∀CTL, and the complexity of the satisfiability problem coincides with the complexity of the linear-time versions of the logics (obtained by omitting all universal path quantifiers).

We showed how labeling of states with directions makes the model-checking and implication problems for the universal and existential fragments as hard as for the full logics. While such a labeling is straightforward in the case of model checking, it is not always possible for implication. Indeed, in the case of CTL* and CTL, formulas cannot specify a legal labeling, making the implication problem for ∀CTL* and ∃CTL* strictly easier than the implication problem for CTL*, and similarly for CTL. In contrast, we were able to label the directions legally using a $\forall AFMC$ formula, making the implication problems for $\forall MC$ and $\exists MC$ as hard as the one for MC, and similarly for the alternation-free fragments. Another way to see the importance of the fixed-point quantifiers is to observe that the implication problems for Modal Logic (μ-calculus without fixed-point quantifiers) and its universal and existential fragments are co-NP-complete.

Finally, the *equivalence problem* for a logic asks, given formulas φ and ψ, if the formula $\varphi \leftrightarrow \psi$ is valid. The equivalence problem for the μ-calculus is EXPTIME-complete, by easy reductions to and from satisfiability. This gives an EXPTIME upper bound for the equivalence problem for $\exists MC$ and $\forall MC$. By a reduction from satisfiability or validity (whichever is harder), we also get a PSPACE lower bound. However, the exact complexity for the equivalence problem for the universal and existential fragments of the μ-calculus remains open (also for the alternation-free fragments).

The gap above highlights the difficulty in studying the universal and existential fragments of the μ-calculus. It is easy to see that in all formalisms that are closed under complementation (in particular, full MC), equivalence is as hard as satisfiability. Indeed, φ and ψ are equivalent iff $(\varphi \wedge \neg\psi) \vee (\psi \wedge \neg\varphi)$ is not satisfiable. When a formalism is not closed under complementation, equivalence is not harder than implication, and is not easier than satisfiability or validity, whichever is harder. In the case of CTL, for example, it is easy to see that the equivalence problems for ∀CTL and ∃CTL are PSPACE-complete, as implication has the same complexity as satisfiability or validity (whichever is harder). The same holds for word automata: if we identify the existential fragment with nondeterministic automata, and the universal fragment with universal automata, then in both cases the language-containment problem (the automata-theoretic counterpart of implication) has the same complexity as the harder one of the nonemptiness and universality problems (the automata-theoretic counterparts of satisfiability and validity). Once we do not allow fixed-point quantifiers, the same holds for the μ-calculus: the equivalence problem for Modal Logic and its universal and existential fragments is co-NP-complete, as the co-NP-hardness of the implication problem applies already for the validity problem. So, in all the cases we know, except for the universal and existential fragments of CTL* and the μ-calculus and its alternation-free fragment, the above immediate upper and lower bounds do not induce a gap, and the exact complexity of the equivalence problem is known.

Acknowledgements

We thank the anonymous reviewers for their detailed comments on the organization and on the proofs.

References

[1] A. Arnold, D. Niwinski, Rudiments of the μ-Calculus, Elsevier Science, Amsterdam, 2001.
[2] B. Banieqbal, H. Barringer, Temporal logic with fixed points, in: B. Banieqbal, H. Barringer, A. Pnueli (Eds.), Temporal Logic in Specification, Lecture Notes in Computer Science, vol. 398, Springer, Berlin, 1987, pp. 62–74.
[3] J.C. Bradfield, The modal μ-calculus alternation hierarchy is strict, Theoret. Comput. Sci. 195 (2) (1998) 133–153.

[4] J.C. Bradfield, Fixpoint alternation: arithmetic, transition systems, and the binary tree, Theoret. Inform. Appl. 33 (4/5) (1999) 341–356.
[5] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, L.J. Hwang, Symbolic model checking: 10^{20} states and beyond, Inform. and Comput. 98 (2) (1992) 142–170.
[6] E.M. Clarke, E.A. Emerson, Design and synthesis of synchronization skeletons using branching time temporal logic, in: D. Kozen (Ed.), Proc. Workshop on Logic of Programs, Lecture Notes in Computer Science, vol. 131, Springer, Berlin, 1981, pp. 52–71.
[7] E.M. Clarke, E.A. Emerson, A.P. Sistla, Automatic verification of finite-state concurrent systems using temporal-logic specifications, ACM Trans. Programming Languages Systems 8 (2) (1986) 244–263.
[8] E.M. Clarke, O. Grumberg, D. Peled, Model Checking, MIT Press, Cambridge, MA, 1999.
[9] R.J. Cleaveland, B. Steffen, A linear-time model-checking algorithm for the alternation-free modal μ-calculus, Formal Methods System Design 2 (1993) 121–147.
[10] M. Dam, CTL* and ECTL* as fragments of the modal μ-calculus, Theoret. Comput. Sci. 126 (1994) 77–96.
[11] E.A. Emerson, J.Y. Halpern, Sometimes and not never revisited: on branching versus linear time, J. ACM 33 (1) (1986) 151–178.
[12] E.A. Emerson, C. Jutla, The complexity of tree automata and logics of programs, in: Proc. 29th IEEE Symp. on Foundations of Computer Science, IEEE Computer Society Press, Silver Spring, MD, 1988, pp. 328–337.
[13] E.A. Emerson, C. Jutla, Tree automata, mu-calculus and determinancy, in: Proc. 32nd Annu. Symp. on Foundations of Computer Science, IEEE Computer Society Press, Silver Spring, MD, 1991, pp. 368–377.
[14] E.A. Emerson, C. Jutla, A.P. Sistla, On model-checking for fragments of μ-calculus, in: C. Courcoubetis (Ed.), Computer Aided Verification, Lecture Notes in Computer Science, vol. 697, Springer, Berlin, 1993, pp. 385–396.
[15] E.A. Emerson, C.-L. Lei, Efficient model checking in fragments of the propositional μ-calculus, in: Proc. First Symp. Logic in Computer Science, IEEE Computer Society Press, Silver Spring, MD, 1986, pp. 267–278.
[16] M.J. Fischer, R.E. Ladner, Propositional dynamic logic of regular programs, J. Comput. Systems Sci. 18 (1979) 194–211.
[17] R. Greenlaw, H.J. Hoover, W.L. Ruzzo, Limits of Parallel Computation, Oxford University Press, Oxford, 1995.
[18] O. Grumberg, D.E. Long, Model checking and modular verification, ACM Trans. Programming Languages Systems 16 (3) (1994) 843–871.
[19] M. Hennessy, R. Milner, Algebraic laws for nondeterminism and concurrency, J. ACM 32 (1985) 137–161.
[20] N. Immerman, Number of quantifiers is better than number of tape cells, J. Comput. System Sci. 22 (3) (1981) 384–406.
[21] D. Janin, I. Walukiewicz, Automata for the modal μ-calculus and related results. in: J. Wiedermann, P. Hájek (Eds.), Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, vol. 969, Springer, Berlin, 1995, pp. 552–562.
[22] D. Janin, I. Walukiewicz, On the expressive completeness of the propositional μ-calculus with respect to the monadic second order logic, in: U. Montanari, V. Sassone (Eds.), Conference on Concurrency Theory, Lecture Notes in Computer Science, vol. 1119, Springer, Berlin, 1996, pp. 263–277.
[23] M. Jurdzinski, Deciding the winner in parity games is in UP $\cap$ co-UP, Inform. Process. Lett. 68 (3) (1998) 119–124.
[24] M. Jurdzinski, Small progress measures for solving parity games, in: H. Reichel, S. Tison (Eds.), Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science, vol. 1770, Springer, Berlin, 2000, pp. 290–301.
[25] D. Kozen, Results on the propositional μ-calculus, Theoret. Comput. Sci. 27 (1983) 333–354.
[26] D. Kozen, A finite model theorem for the propositional μ-calculus, Studia Logica 47 (3) (1988) 333–354.
[27] O. Kupferman, M.Y. Vardi, An automata-theoretic approach to modular model checking, ACM Trans. Programming Languages Systems 22 (2000) 87–128.
[28] O. Kupferman, M.Y. Vardi, P. Wolper, An automata-theoretic approach to branching-time model checking, J. ACM 47 (2) (2000) 312–360.
[29] R.P. Kurshan, FormalCheck User's Manual, Cadence Design, Inc., 1998.
[30] C. Loiseaux, S. Graf, J. Sifakis, A. Bouajjani, S. Bensalem, Property-preserving abstractions for the verification of concurrent systems, Formal Methods in System Design 6 (1995) 11–44.
[31] D. Long, A. Brown, E. Clarke, S. Jha, W. Marrero, An improved algorithm for the evaluation of fixpoint expressions, in: D.L. Dill (Ed.), Computer Aided Verification, Lecture Notes in Computer Science, vol. 818, Springer, Berlin, 1994, pp. 338–350.
[32] R.S. Lubarsky, μ-Definable sets of integers, J. Symbolic Logic 58 (1993) 291–313.
[33] K.L. McMillan, Symbolic Model Checking, Kluwer Academic Publishers, Dordrecht, 1993.
[34] D. Niwiński, On fixed point clones, in: L. Kott (Ed.), International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, vol. 226, Springer, Berlin, 1986, pp. 464–473.
[35] A. Pnueli, In transition from global to modular temporal reasoning about programs, in: K. Apt (Ed.), Logics and Models of Concurrent Systems, NATO Advanced Summer Institutes, vol. F-13, Springer, Berlin, 1985, pp. 123–144.
[36] A. Pnueli, R. Rosner, On the synthesis of a reactive module, in: Proc. 16th ACM Symp. Principles of Programming Languages, ACM Press, New York, 1989, pp. 179–190.
[37] P.J.G. Ramadge, W.M. Wonham, The control of discrete event systems, IEEE Trans. Control Theory 77 (1989) 81–98.
[38] H. Seidl, Fast and simple nested fixpoints, Inform. Process. Lett. 59 (6) (1996) 303–308.
[39] M.Y. Vardi, A temporal fixpoint calculus, in: Proc. 15th Annu. Symp. Principles of Programming Languages, ACM Press, New York, 1988, pp. 250–259.
[40] M.Y. Vardi, L. Stockmeyer, Improved upper and lower bounds for modal logics of programs, in: Proc. 17th ACM Symp. Theory of Computing, 1985, pp. 240–251.