Succinct Graph Representations of μ-Calculus Formulas

Clemens Kupke ✉
University of Strathclyde, Scotland

Johannes Marti ✉
University of Amsterdam, The Netherlands

Yde Venema ✉
University of Amsterdam, The Netherlands

Abstract

Many algorithmic results on the modal mu-calculus use representations of formulas such as alternating tree automata or hierarchical equation systems. At closer inspection, these results are not always optimal, since the exact relation between the formula and its representation is not clearly understood. In particular, there has been confusion about the definition of the fundamental notion of the size of a mu-calculus formula.

We propose the notion of a parity formula as a natural way of representing a mu-calculus formula, and as a yardstick for measuring its complexity. We discuss the close connection of this concept with alternating tree automata, hierarchical equation systems and parity games. We show that well-known size measures for mu-calculus formulas correspond to a parity formula representation of the formula using its syntax tree, subformula graph or closure graph, respectively. Building on work by Bruse, Friedmann & Lange we argue that for optimal complexity results one needs to work with the closure graph, and thus define the size of a formula in terms of its Fischer-Ladner closure. As a new observation, we show that the common assumption of a formula being clean, that is, with every variable bound in at most one subformula, incurs an exponential blow-up of the size of the closure.

To realise the optimal upper complexity bound of model checking for all formulas, our main result is to provide a construction of a parity formula that (a) is based on the closure graph of a given formula, (b) preserves the alternation-depth but (c) does not assume the input formula to be clean.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Logic and verification

Keywords and phrases modal mu-calculus, model checking, alternating tree automata, hierachical equation systems

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Related Version This paper is largely based on our technical report [15].

Funding *Clemens Kupke*: Supported by Leverhulme Trust Research Project Grant RPG-2020-232. *Johannes Marti*: The research of this author has been made possible by a grant from the Dutch Research Council NWO, project nr. 617.001.857.

1 Introduction

The modal μ-calculus, introduced by Kozen [14] and surveyed in for instance [2, 12, 4, 9], is a logic for describing properties of processes that are modelled by labelled transition systems. It extends the expressive power of propositional modal logic by means of least and greatest fixpoint operators. This addition permits the expression of all bisimulation-invariant monadic second order properties of such processes [13]. As a *logic*, μML has many desirable properties, such as a natural complete axiomatisation [14, 19], uniform interpolation and other interesting model-theoretical properties [8, 11], and a complete cut-free proof system [1]. Here we will be interested in some of its computational properties.

The μ-calculus is generally regarded as a universal specification language for reactive systems, since it embeds most other logics that are used for this purpose, such as LTL, CTL, CTL* and PDL. Given this status, the computational complexity of its model checking and satisfiability problems is of central importance. While the satisfiability problem has been shown to be EXPTIME-complete [10] already thirty years ago, the precise complexity of its model checking problem turned out to be a challenging problem. A breakthrough was obtained by Calude et alii [7] who gave a *quasi-polynomial* algorithm for deciding parity games; since model checking for the modal μ-calculus can be determined by such games, this indicates a quasi-polynomial upper bound of the complexity of the model checking problem.

Generally, to determine the complexity of a proposed algorithm operating on μ-calculus formulas, one needs sensible measures of the complexity of the formula that is (part of) the input to the algorithm; the most important of these concern *size* and *alternation depth*. Different notions of size have been used, depending on how precisely formulas are represented in the input. Standard size measures include: (1) length, corresponding to a representation of the formula as a string or syntax tree; (2) subformula size, corresponding to a representation of the formula as the directed acyclic graph of its subformulas; and (3) closure size, corresponding to a similar representation of a formula via its (Fischer-Ladner) closure.

The choice between these representations is non-trivial because the subformula size of a formula may be exponentially smaller than its length, and, as was shown by Bruse, Friedmann & Lange [6], its closure size may be exponentially smaller than its subformula size. Consequently, complexity results about the μ-calculus may be suboptimal when expressed in terms of subformula size, in the sense that a stronger version of the result holds when formulated in terms of closure size. In other words, it is desirable to design algorithms that operate on a representation of a formula that is based on its closure.

At closer inspection it turns out that generally, the literature on algorithmic aspects of the μ-calculus is crystal clear in terms of the structures on which the algorithms operate, but less so on the precise way in which these structures represent formulas. As a consequence, when formulated in terms of the actual formulas, complexity results as given may be suboptimal or somewhat fuzzy. Our long-term goal is to study the representation of μ-calculus formulas in more detail, and to develop a framework in which various approaches can easily be compared, and in which complexity results can be formulated and proved optimally and unambiguously.

As a starting point, we note that in the literature different frameworks are used to represent μ-calculus formulas. The parity games that feature in model checking algorithms are usually based on an arena which is some kind of Cartesian product of a graph that represents the formula with the model where this formula is evaluated. Other prominent ways to represent formulas are (alternating) tree automata and (hierarchical) equation systems; as we shall see further on, in these cases we can think of the structures that represent formulas in graph-theoretic terms as well. In all cases then, the mathematically fundamental structure representing a formula is a graph, whose nodes are labelled with logical connectives or atomic formulas, and with priorities that are used to determine some winning or acceptance condition. The graph itself can be based on the syntax tree, the subformula dag or the closure graph of the formula that it represents.

We make this fundamental labelled graph structure explicit and call the resulting concept a *parity formula*.[1] Intuitively, parity formulas generalise standard formulas by dropping the

[1] Parity formulas are almost the same structures as the alternating binary tree automata of Emerson &

requirement that the underlying graph structure of the formula is a tree with back edges, and adding an explicit parity acceptance condition. A good way to think about a parity formula is as the formula component of a model checking game. As we shall see below, parity formulas are closely related to alternating tree automata and hierarchical equation systems. Compared to these however, parity formulas have a very simple mathematical structure, which allows for a straightforward and unambiguous definition of its size and its index (alternation depth).

The explicit introduction of this notion is not a goal in itself. We intend to *use* it as a tool to analyse some underexposed sides of the theory of the modal μ-calculus. In this paper we discuss some key constructions turning standard formulas into parity formulas and vice versa. Along the way we make two observations that we consider the key contributions of this paper:

1) A common assumption in the literature on the μ-calculus is that one may assume, without loss of generality, that formulas are clean or well-named, in the sense that bound variables are disjoint from free variables, and each bound variable determines a unique subformula. In Proposition 10 we show that this assumption may lead to an exponential blow-up in terms of closure-size. This means that, if one is interested in optimal complexity results, one should not assume the input formula to be clean.

2) To the best of our knowledge, all representations of μ-calculus formulas known from the literature, are suboptimal in one way or another: they are based on the subformula dag, they presuppose cleanness, or they use a priority function which yields an unnecessarily big index. The main result of our paper, Theorem 12, concerns a construction that provides, for every μ-calculus formula, an equivalent parity formula that is based on its closure graph, and has an index that matches its alternation depth. The fact that we do *not* assume the input formula to be clean makes our proof non-trivial.[2]

Because of Proposition 10, Theorem 12 has an impact on the quasi-polynomial time complexity of the model checking problem for the modal μ-calculus. If one wants to formulate an optimal version of this complexity result, by the observations of Bruse, Friedmann & Lange [6] one needs to measure the formula in terms of closure-size; but then Theorem 12 is needed to ensure that the result applies to all formulas, not just to the ones that are clean.

2 Preliminaries

In this section we briefly review the syntax and semantics of the modal μ-calculus.

Syntax It will be convenient to assume that μ-calculus formulas are in negation normal form. That is, the formulas of the modal μ-calculus $\mu\mathtt{ML}$ are given by the following grammar:

$$\mu\mathtt{ML} \ni \varphi \ ::= \ p \mid \overline{p} \mid \bot \mid \top \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \Diamond\varphi \mid \Box\varphi \mid \mu x\,\varphi \mid \nu x\,\varphi,$$

where p, x are variables, and the formation of the formulas $\mu x\,\varphi$ and $\nu x\,\varphi$ is subject to the constraint that φ is *positive* in x, i.e., there are no occurrences of $\overline{x}$ in φ. Elements of $\mu\mathtt{ML}$ will be called *μ-calculus formulas* or *standard formulas*. Formulas of the form $\mu x.\varphi$ or $\nu x.\varphi$ will be called *fixpoint formulas*. We define $\mathtt{Lit}(\mathsf{Q}) := \{p, \overline{p} \mid p \in \mathsf{Q}\}$ as the set of *literals* over Q, and $\mathtt{At}(\mathsf{Q}) := \{\bot, \top\} \cup \mathtt{Lit}(\mathsf{Q})$ as the set of *atomic formulas* over Q. We will associate μ

Jutla [10] and as the version of Wilke's alternating tree automata where the transition conditions are basic formulas, i.e., contain at most one logical connective [20, 12].

[2] Proof details, which we could not include here for lack of space, can be found in the technical report [15].

and ν with the odd and even numbers, respectively, and for $\eta \in \{\mu, \nu\}$ define $\overline{\eta}$ by putting $\overline{\mu} := \nu$ and $\overline{\nu} := \mu$. The notion of *subformula* is defined as usual; we write $\varphi \trianglelefteq \psi$ if φ is a subformula of ψ, and define $Sfor(\psi)$ as the set of subformulas of ψ.

We use standard terminology related to the binding of variables. We write $BV(\xi)$ and $FV(\xi)$ for, respectively, the set of *bound* and *free variables* of a formula ξ. A formula ξ is *tidy*[3] if $FV(\xi) \cap BV(\xi) = \varnothing$. We fix a set Q of proposition letters and let $\mu\mathtt{ML}(\mathsf{Q})$ denote the set of formulas ξ with $FV(\xi) \subseteq \mathsf{Q}$. We let $\varphi[\psi/x]$ denote the formula φ, with every free occurrence of x replaced by the formula ψ; we will make sure that we only apply this substitution operation if ψ is free for x in φ (meaning that no free variable of ψ gets bound after substituting). This saves us from involving alphabetical variants when substituting. The *unfolding* of a formula $\eta x.\chi$ is the formula $\chi[\eta x.\chi/x]$; this formula is tidy if χ is so.

Semantics The modal μ-calculus is interpreted over Kripke structures. A *(Kripke) model* is a triple $\mathbb{S} = (S, R, V)$ where S is the set of *states* or *points* of $\mathbb{S}$, $R \subseteq S \times S$ is its *accessibility* relation, and $V : \mathsf{Q} \to \mathcal{P}(S)$ its *valuation*. A *pointed model* is a pair $(\mathbb{S}, s)$ where s is a designated state of $\mathbb{S}$. Inductively we define the *meaning* $[\![\varphi]\!]^{\mathbb{S}} \subseteq S$ of a formula $\varphi \in \mu\mathtt{ML}(\mathsf{Q})$ in a model $\mathbb{S}$ as follows:

$$\begin{array}{lcllcl}
[\![p]\!]^{\mathbb{S}} & := & V(p) & [\![\overline{p}]\!]^{\mathbb{S}} & := & S \setminus V(p) \\
[\![\bot]\!]^{\mathbb{S}} & := & \varnothing & [\![\top]\!]^{\mathbb{S}} & := & S \\
[\![\varphi \vee \psi]\!]^{\mathbb{S}} & := & [\![\varphi]\!]^{\mathbb{S}} \cup [\![\psi]\!]^{\mathbb{S}} & [\![\varphi \wedge \psi]\!]^{\mathbb{S}} & := & [\![\varphi]\!]^{\mathbb{S}} \cap [\![\psi]\!]^{\mathbb{S}} \\
[\![\Diamond\varphi]\!]^{\mathbb{S}} & := & \{s \in S \mid R[s] \cap [\![\varphi]\!]^{\mathbb{S}} \neq \varnothing\} & [\![\Box\varphi]\!]^{\mathbb{S}} & := & \{s \in S \mid R[s] \subseteq [\![\varphi]\!]^{\mathbb{S}}\} \\
[\![\mu x.\varphi]\!]^{\mathbb{S}} & := & \bigcap\{U \subseteq S \mid [\![\varphi]\!]^{\mathbb{S}[x \mapsto U]} \subseteq U\} & [\![\nu x.\varphi]\!]^{\mathbb{S}} & := & \bigcup\{U \subseteq S \mid [\![\varphi]\!]^{\mathbb{S}[x \mapsto U]} \supseteq U\}.
\end{array}$$

Here $\mathbb{S}[x \mapsto U] := (S, R, V[x \mapsto U]$ where $V[x \mapsto U]$ is the $\mathsf{Q} \cup \{x\}$-valuation mapping x to U and any $p \neq x$ to $V(p)$. If a state $s \in S$ belongs to the set $[\![\varphi]\!]^{\mathbb{S}}$, we write $\mathbb{S}, s \Vdash \varphi$, and say that s *satisfies* φ.

Complexity measures The size of a formula $\xi \in \mu\mathtt{ML}$ can be measured in at least three different ways. First, its *length* $|\xi|^{\ell}$ is defined as the number of symbols that occur in ξ. Second, we define its *subformula size* $|\xi|^{s} := |Sfor(\xi)|$ as the number of distinct subformulas of ξ.

Third, we can measure the size of ξ by counting the number of formulas in its (Fischer-Ladner) closure. We need some notation and terminology here, where we assume that ξ is tidy. The set $Clos_0(\xi)$ is defined by the following case distinction:

$$\begin{array}{lcll}
Clos_0(\varphi) & := & \varnothing & \text{if } \varphi \in \mathtt{At}(\mathsf{Q}) \\
Clos_0(\varphi_0 \odot \varphi_1) & := & \{\varphi_0, \varphi_1\} & \text{where } \odot \in \{\wedge, \vee\} \\
Clos_0(\heartsuit\varphi) & := & \{\varphi\} & \text{where } \heartsuit \in \{\Diamond, \Box\} \\
Clos_0(\eta x.\varphi) & := & \{\varphi[\eta x.\varphi/x]\} & \text{where } \eta \in \{\mu, \nu\}.
\end{array}$$

We write $\xi \to_C \varphi$ if $\varphi \in Clos_0(\xi)$ and call $\to_C$ the *trace* relation on $\mu\mathtt{ML}$. We let $\twoheadrightarrow_C$ denote the reflexive and transitive closure of $\to_C$, and define the *closure* of ξ as the set $Clos(\xi) := \{\varphi \mid \xi \twoheadrightarrow_C \varphi\}$. The *closure graph* of ξ is the directed graph $(Clos(\xi), \to_C)$. The *closure size* $|\xi|^{c}$ of ξ is given as $|\xi|^{c} := |Clos(\xi)|$.

[3] In the literature, some authors make a distinction between proposition letters (which can only occur freely in a formula), and propositional variables, which can be bound. Our tidy formulas correspond to *sentences* in this approach, that is, formulas without free variables.

Next to its size, the most important complexity measure of a μ-calculus formula is its *alternation depth*. We shall work with the definition originating with Niwiński [16]. By natural induction we first define classes $\Theta^\mu_n, \Theta^\nu_n \subseteq \mu\mathtt{ML}$ (corresponding to, respectively, the sets Π_{n+1} and Σ_{n+1} in [16]). Intuitively, Θ^η_n consists of those μ-calculus formulas where n bounds the length of any alternating nesting of fixpoint operators of which the most significant formula is an η-formula. For the definition, we set, for $\eta, \lambda \in \{\mu, \nu\}$:

1. all atomic formulas belong to Θ^η_0;
2. if $\varphi_0, \varphi_1 \in \Theta^\eta_n$, then $\varphi_0 \vee \varphi_1, \varphi_0 \wedge \varphi_1, \Diamond\varphi_0, \Box\varphi_0 \in \Theta^\eta_n$;
3. if $\varphi \in \Theta^\eta_n$ then $\overline{\eta}x.\varphi \in \Theta^\eta_n$ (where we recall that $\overline{\mu} = \nu$ and $\overline{\nu} = \mu$);
4. if $\varphi(x), \psi \in \Theta^\eta_n$, then $\varphi[\psi/x] \in \Theta^\eta_n$, provided that ψ is free for x in φ;
5. all formulas in Θ^λ_n belong to Θ^η_{n+1}.

The *alternation depth* $ad(\xi)$ of a formula ξ is the least n such that $\xi \in \Theta^\mu_n \cap \Theta^\nu_n$. It measures the maximal number of alternations between least and greatest fixpoint operators in ξ.

3 Representations of μ-calculus formulas

In this section we discuss two of the most widely used representations for formulas of the modal μ-calculus that one may find in the literature: alternating tree automata (ATAs) and hierarchical equation systems (HESs). Both of these come in many different shapes, and in some of these shapes the two notions are actually very similar to one another. For lack of space we cannot give a proper survey here, and so we focus on a perspective, in which these similarities come out most clearly.[4] In this perspective, both kinds of representation can be defined using the syntactic notion of a *transition condition*. Recall that we have fixed a set Q of proposition letters; in addition to this we need a set A of objects that we shall call *states* in the setting of ATAs and *variables* in that of HESs. Now consider the following definitions of, respectively, the sets of *basic*, *standard* and *extended* transition conditions over Q and A.

$$\begin{array}{rcl}
\mathrm{BTC}(\mathsf{Q},A) \ni \beta & ::= & \bot \mid \top \mid p \mid \overline{p} \mid a \mid \Diamond a \mid \Box a \mid a \wedge a \mid a \vee a, \\
\mathrm{STC}(\mathsf{Q},A) \ni \beta & ::= & \bot \mid \top \mid p \mid \overline{p} \mid a \mid \Diamond a \mid \Box a \mid \beta \wedge \beta \mid \beta \vee \beta, \\
\mathrm{ETC}(\mathsf{Q},A) \ni \beta & ::= & \bot \mid \top \mid p \mid \overline{p} \mid a \mid \Diamond \beta \mid \Box \beta \mid \beta \wedge \beta \mid \beta \vee \beta,
\end{array}$$

where $p \in \mathsf{Q}$ and $a \in A$.

▶ **Definition 1.** *An* alternating tree automaton *or* ATA *is a quadruple* $\mathbb{A} = (A, \Delta, \Omega, a_I)$ *where* A *is a non-empty finite set of* states*, of which* $a_I \in A$ *is the* initial *state,* $\Omega : A \to \omega$ *is the* priority map*, and* $\Delta : A \to \mathrm{STC}(\mathsf{Q}, A)$ *is the* transition map*. An* ATA *will be called* basic *if the range of its transition map consists of basic transition conditions.*

Before we move on to the definition of the semantics of ATAs, we make two comments. First and foremost, the ATAs that were introduced by Wilke [20] are in fact what we call *basic*; as we shall see in the next section, these are the ones that are in close correspondence with our parity formulas. In the subsequent literature however, it seems to have become quite common to allow for the more complex conditions that we here call 'standard', and that may feature nesting of boolean connectives in transition conditions, (possibly restricted to disjunctive normal form).

[4] This means in particular that we only consider *amorphous* tree automata here, i.e., we disregard automata operating on trees where the children of a node are given by a bounded number of functions.

Second, if we think of the powerset $\mathcal{P}(\mathsf{Q})$ as an alphabet, then tree-based Kripke models correspond to $\mathcal{P}(\mathsf{Q})$-labelled trees. In such a setting it is common to consider tree automata with a transition map of the form $\Delta : A \times \mathcal{P}(\mathsf{Q}) \to \mathrm{TC}(\emptyset, A)$ for some set of transition conditions in which the proposition letters in Q may not occur. That is, the proposition letters in Q move from the co-domain of the transition map to its domain. It is in fact quite easy to transform automata of the one kind into devices of the other kind, but for lack of space we cannot go into detail here.

The semantics of alternating tree automata is usually given in terms of run trees, but we may also use parity games [12, ch. 9]. A simple version is the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ for an ATA $\mathbb{A}$ and a model $\mathbb{S} = (S, R, V)$; it takes positions in the set $V_{\mathbb{A}} \times S$, where $V_{\mathbb{A}}$ is given as

$$V_{\mathbb{A}} := \{a_I\} \cup \bigcup_{a \in A} Sfor(\Delta(a)).$$

For each of these positions Table 1 below lists the set of possible moves and the player that is to move. (We need not assign a player to positions that admit a single move only.) As usual in parity games finite matches are lost by the player who gets stuck (i.e., needs to pick an element from the empty set) and infinite matches are won by $\exists$ iff the maximal priority $\Omega(a)$ of all positions $(a, s) \in A \times S$ that occur infinitely often in the match is even. The starting position is (a_I, s), with $(\mathbb{S}, s)$ the pointed model for which we want to check acceptance.

Position	Player	Admissible moves
$(\bot, s)$	$\exists$	$\varnothing$
$(\top, s)$	$\forall$	$\varnothing$
(p, s) for $s \in V(p)$	$\forall$	$\varnothing$
(p, s) for $s \notin V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$ for $s \in V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$ for $s \notin V(p)$	$\forall$	$\varnothing$
(a, s) for $a \in A$	-	$\{(\Delta(a), s)\}$
$(\alpha_0 \vee \alpha_1, s)$	$\exists$	$\{(\alpha_0, s), (\alpha_1, s)\}$
$(\alpha_0 \wedge \alpha_1, s)$	$\forall$	$\{(\alpha_0, s), (\alpha_1, s)\}$
$(\Diamond a, s)$	$\exists$	$\{(a, t) \mid sRt\}$
$(\Box a, s)$	$\forall$	$\{(a, t) \mid sRt\}$

Table 1 The acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$

As a second way of representing μ-calculus formulas we now discuss *hierarchical equation systems* [18, 6]. As with alternating tree automata there are multiple definitions of hierarchical equation systems in the literature. Here we recall the definition from [9] (where they are called *modal* equation systems).

▶ **Definition 2.** *A* hierarchical equation system *or* HES *consists of a finite set of variables* $A = \{X_1, \ldots, X_n\}$*, together with a set*

$$\mathcal{E} = \{X_1 =_{p_1} \beta_1, \ldots, X_n =_{p_n} \beta_n\}.$$

of prioritised modal equations. That is, for each i, the number $p_i \in \omega$ denotes the priority *of the i-th equation, and β_i is an expression in the set* $\mathrm{ETC}(\mathsf{Q}, A)$*.*

By convention the first variable X_1 is the entry point of the equation system, which functions similarly to the initial state of an ATA. In [18, 6] the semantics of hierarchical equation systems is defined on the basis of the Knaster-Tarski fixpoint theorem, as in the compositional semantics of standard formulas defined in Section 2. It is however also possible to give a semantics in terms of parity games, completely analogous to the game semantics for ATAs mentioned above. We leave the details to the reader.

It is clear that there is a close correspondence between hierarchical equation systems and alternating tree automata. In fact one might view an HES as a generalised version of an ATA in which modalities can be nested inside of the transition conditions — such a generalised notion of ATA has been used for example in [5]. With this in mind, in the sequel we will take this generalised perspective on ATAs, so that we include HESs when we refer to ATAs.

It is not entirely obvious what is the right measure for the size of an alternating tree automaton $\mathbb{A} = (A, \Delta, \Omega, a_I)$. One might simply consider the number of states in $\mathbb{A}$, but since any actual representation of the automaton needs to encode the arbitrarily large transition conditions a more adequate measure of the size of $\mathbb{A}$ should take these into account as well. Moreover, since the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ is based on the set $V_{\mathbb{A}} \times S$, it makes sense to define $|\mathbb{A}| := |V_{\mathbb{A}}|$, but also, to consider a representation of $\mathbb{A}$ that is more directly based on this set $V_{\mathbb{A}}$. This is what we will do in the next section.

4 Parity formulas

As the backbone of our framework we introduce the notion of a parity formula. These are like ordinary (modal) formulas, with the difference that (i) the underlying structure of a parity formula is a directed graph, possibly with cycles, rather than a tree; and (ii) one adds a priority labelling to this syntax graph, to ensure a well-defined game-theoretical semantics in terms of parity games.

▶ **Definition 3.** *A* parity formula over Q *is a quintuple* $\mathbb{G} = (V, E, L, \Omega, v_I)$, *where*

- (V, E) *is a finite, directed graph, with* $|E[v]| \leq 2$ *for every vertex* v;
- $L : V \to \mathtt{At}(\mathsf{Q}) \cup \{\wedge, \vee, \Diamond, \Box, \epsilon\}$ *is a labelling function;*
- $\Omega : V \overset{\circ}{\to} \omega$ *is a partial map, the* priority *map of* $\mathbb{G}$; *and*
- v_I *is a vertex in* V, *referred to as the* initial *node of* $\mathbb{G}$;

such that (with $E[v] := \{u \in V \mid Evu\}$*):*

1. $|E[v]| = 0$ *if* $L(v) \in \mathtt{At}(\mathsf{Q})$, *and* $|E[v]| = 1$ *if* $L(v) \in \{\Diamond, \Box\} \cup \{\epsilon\}$;
2. *every cycle of* (V, E) *contains at least one node in* $\mathsf{Dom}(\Omega)$.

A node $v \in V$ *is called* silent *if* $L(v) = \epsilon$, constant *if* $L(v) \in \{\top, \bot\}$, literal *if* $L(v) \in \mathtt{Lit}(\mathsf{Q})$, atomic *if it is either constant or literal,* boolean *if* $L(v) \in \{\wedge, \vee\}$, *and* modal *if* $L(v) \in \{\Diamond, \Box\}$. *The elements of* $\mathsf{Dom}(\Omega)$ *will be called* states.

The semantics of parity formulas is given in terms of a *model checking game*, which is defined as the following parity game between $\exists$ and $\forall$.

▶ **Definition 4.** *Let* $\mathbb{S} = (S, R, V)$ *be a model, and let* $\mathbb{G} = (V, E, L, \Omega, v_I)$ *be a parity formula. We define the* model checking game $\mathcal{E}(\mathbb{G}, \mathbb{S})$ *as the parity game* (G, E, Ω') *of which the board (or arena) consists of the set* $V \times S$, *the priority map* $\Omega' : V \times S \to \omega$ *is given by putting* $\Omega'(v, s) := \Omega(v)$ *if* $v \in \mathsf{Dom}(\Omega)$ *and* $\Omega'(v, s) := 0$ *otherwise. and the game graph is given in Table 2.* $\mathbb{G}$ holds at *or* is satisfied by *the pointed model* $(\mathbb{S}, s)$, *notation:* $\mathbb{S}, s \Vdash \mathbb{G}$, *if the pair* (v_I, s) *is a winning position for* $\exists$ *in* $\mathcal{E}(\mathbb{G}, \mathbb{S})$.

Position		Player	Admissible moves
(v,s)	with $L(v) = p$ and $s \in V(p)$	$\forall$	$\varnothing$
(v,s)	with $L(v) = p$ and $s \notin V(p)$	$\exists$	$\varnothing$
(v,s)	with $L(v) = \overline{p}$ and $s \in V(p)$	$\exists$	$\varnothing$
(v,s)	with $L(v) = \overline{p}$ and $s \notin V(p)$	$\forall$	$\varnothing$
(v,s)	with $L(v) = \epsilon$	-	$E[v] \times \{s\}$
(v,s)	with $L(v) = \vee$	$\exists$	$E[v] \times \{s\}$
(v,s)	with $L(v) = \wedge$	$\forall$	$E[v] \times \{s\}$
(v,s)	with $L(v) = \Diamond$	$\exists$	$E[v] \times R[s]$
(v,s)	with $L(v) = \Box$	$\forall$	$E[v] \times R[s]$

Table 2 The model checking game $\mathcal{E}(\mathbb{G}, \mathbb{S})$

Equivalence of parity formulas, and between parity formulas and standard formulas (or ATAs or HESs), is defined in the obvious way.

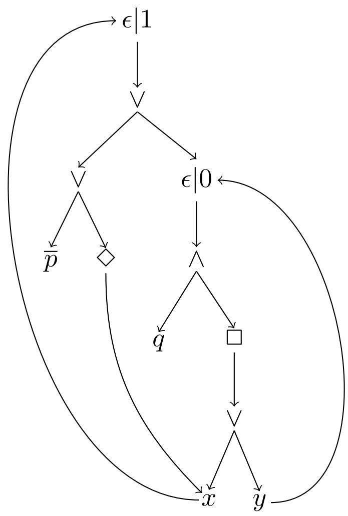


Figure 1 Example of a parity formula

▶ **Example 5.** Figure 1 to the right displays an example of a parity formula that is based on the standard μ-calculus formula $\xi = \mu x.(\overline{p} \vee \Diamond x) \vee \nu y.(q \wedge \Box(x \vee y))$, by adding *back edges* to the subformula dag of ξ. Nodes in the domain of the priority map are indicated by the notation $\cdot|n$, where n is the priority. The initial node is $\epsilon|1$.

▶ **Example 6.** One can also build a parity formula from the closure graph of some standard μ-calculus formula. As an example we consider the formula ξ_2 from our proof of Proposition 10 in Section 5:

$$\xi_2 := \mu x_0.\gamma_2 \wedge (\gamma_1 \wedge x_0),$$

where

$$\gamma_1 := \mu x_1.x_1 \wedge (\mu x_0.\gamma_2 \wedge x_1 \wedge x_0), \text{ and}$$
$$\gamma_2 := \mu x_2.x_2 \wedge \big((\mu x_1.x_1 \wedge (\mu x_0.x_2 \wedge x_1 \wedge x_0)) \wedge (\mu x_0.x_2 \wedge (\mu x_1.x_1 \wedge (\mu x_0.x_2 \wedge x_1 \wedge x_0)) \wedge x_0)\big).$$

A picture of the closure graph $(\mathit{Clos}(\xi_2), \rightarrow_C)$ of ξ_2 is on the left in Figure 2 below (where γ_2 is represented by γ_0). This closure graph gives rise to a parity formula whose vertices are the elements of $\mathit{Clos}(\xi_2)$ and edges are given by the trace relation $\rightarrow_C$. The labelling is obvious and the initial node is the node $\xi_2 = \gamma_0$. The priority map Ω can be defined such that $\Omega(\gamma_0) = \Omega(\gamma_1) = \Omega(\gamma_2) = 1$ and Ω is undefined on all other vertices.

We impose a bound on the outdegree of vertices in parity formulas, so that the size of any reasonable encoding of a parity formula is linear in the number of vertices. This facilitates the following simple definition of size:

▶ **Definition 7.** *The* size *of a parity formula* $\mathbb{G} = (V, E, L, \Omega, v_I)$ *is defined as its number of nodes:* $|\mathbb{G}| := |V|$.

The second fundamental complexity measure for a parity formula is its index, which corresponds to the alternation depth of standard formulas. The most straightforward definition of this notion would be to take the size of the range of the priority map; a slightly more sophisticated approach[5] involves the notions of an *alternating* Ω*-chain* and of a *cluster* (or maximal strongly connected component) of $\mathbb{G}$

▶ **Definition 8.** *Let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be a parity formula.*

A set $C \subseteq V$ is a cluster *in $\mathbb{G}$ if C is a maximal set such that E^*uv and E^*vu for all $u, v \in C$. Clusters are partially ordered by placing one cluster C* higher *than another cluster C' if E^*uu' for all $u \in C$ and $u' \in C'$. A cluster C in $\mathbb{G}$ is* degenerate *if $C = \{v\}$ is a singleton and we do not have Evv; otherwise, C is called* nondegenerate.

An alternating Ω-chain *of* length *k in $\mathbb{G}$ is a finite sequence $v_1 \cdots v_k$ of states that all belong to the same cluster, and satisfy, for all $i < k$, that $\Omega(v_i) < \Omega(v_{i+1})$ while v_i and v_{i+1} have different parity. Such a chain is called an μ-chain (ν-chain) if $\Omega(v_k)$ is odd (even, respectively). Given a cluster C of $\mathbb{G}$ and $\eta \in \{\mu, \nu\}$ we define $ind_\eta(C)$, the η-index of C, as the maximal length of an alternating η-chain in C, and the* index *of C as $ind_{\mathbb{G}}(C) := \max\big(ind_\mu(C), ind_\nu(C)\big)$. Finally, we define*

$$ind(\mathbb{G}) := \max\{ind_{\mathbb{G}}(C) \mid C \in Clus(\mathbb{G})\}.$$

Note that if $\mathbb{G}$ has cycles then $\mathsf{Dom}(\Omega) \neq \varnothing$, so that $\mathbb{G}$ has alternating chains. If $\mathbb{G}$ is cycle-free then we can assume that $\mathsf{Dom}(\Omega)$ is empty, in which case $ind(\mathbb{G}) = 0$.

Parity formulas, alternating tree automata and hierarchical equation systems

It should be clear from the definitions that parity formulas are *very* similar to both alternating tree automata and hierarchical equation systems. To transform a given ATA $\mathbb{A} = (A, \Delta, \Omega, a_I)$ into an equivalent parity formula $\mathbb{G}_{\mathbb{A}} = (V, E, L, \Omega', v_I)$, one just builds a graph on the set $V_{\mathbb{A}}$ in the obvious way, and defines $\Omega' := \Omega$ (with the understanding that Ω' is now a *partial* map on V), and $v_I := a_I$. Finally, one defines $L(a) := \epsilon$ if $a \in A$, whereas $L(\alpha)$ for $\alpha \in \mathrm{STC}(\mathsf{Q}, A) \setminus A$ is given as $L(\alpha) := \alpha$ in case α is atomic, and $L(\alpha)$ is the main connective of α otherwise. It is then straightforward to show that $\mathbb{A} \equiv \mathbb{G}_{\mathbb{A}}$, whereas $\mathbb{G}_{\mathbb{A}}$ obviously has the same size as $\mathbb{A}$. In the opposite direction, it is as straightforward to define, for an arbitrary parity formula $\mathbb{G}$, an equivalent basic ATA $\mathbb{A}$ of the same size and index.

Parity formulas, then, can be seen as a definitional variation of ATAs or HESs. We prefer the graph-based format of parity formulas, since this shows more clearly how to generalise standard formulas, and allows for very perspicuous definitions of complexity measures. What matters most, however, is that the results that we prove in the next two sections apply to ATAs and HESs, in the same way as to parity formulas, see for instance Remark 11 where we make this point explicit.

5 Size issues

It follows from our observations in the previous paragraphs that we may solve the model checking problem for the modal μ-calculus by transforming an arbitrary formula $\xi \in \mu\mathtt{ML}$ into an equivalent parity formula $\mathbb{G}$, and then use the model checking game for parity formulas,

[5] Note that these two definitions almost coincide, since we may shift the priorities of any cluster to either $0, \ldots, d$ or $1, \ldots, d+1$.

together with an algorithm for solving parity games.[6] While the complexity of solving parity games is still not exactly understood, there is no doubt that the key parameters that determine this complexity are the size and the index of the game. Thus, given the definition of the model checking game for parity formulas, it is of crucial importance to find, for an arbitrary μ-calculus formula ξ, an equivalent parity formula $\mathbb{G}$ of minimal size and index. While Kozen [14] already showed that the closure set $Clos(\xi)$ of a clean μ-calculus formula ξ never exceeds the number of subformulas of ξ, Bruse, Friedmann & Lange [6] revealed that $Clos(\xi)$ can in fact be exponentially smaller than $Sfor(\xi)$ of its subformulas. This difference in size indicates that for optimal complexity results, rather than building a parity formula for ξ on the set $Sfor(\xi)$, one should work with the closure graph of ξ.

In the next section we will give a concrete definition of such a parity formula. Here we point out a complication in this definition that seems to have gone unnoticed until now; it concerns the notion of a formula being *clean* or *well-named*.

▶ **Definition 9.** *A tidy μ-calculus formula ξ is* clean *or* well-named *if we may associate with each $x \in BV(\xi)$ a unique subformula of the form $\eta x.\delta$. This unique subformula will be denoted as $\eta_x x.\delta_x$, and we call x a μ-variable if $\eta_x = \mu$, and a ν-variable if $\eta_x = \nu$.*

It is generally very convenient to work with clean formulas, since the bound variables of a clean formula are in 1-1 correspondence with its fixpoint subformulas.[7] For this reason one often sees in the literature that authors assume that the formulas they work with are clean. It is easy to rewrite a μ-calculus formula into an equivalent clean variant, by a suitable renaming of bound variables. The problem, however, is that such a renaming comes at a high cost, as is stated by the following proposition.

▶ **Proposition 10.** *There exists a family $\xi_1, \xi_2, \dots$ of formulas in $\mu\mathtt{ML}$ such that $|\xi_n|^c \leq 2n+2$, but $|\psi_n|^c \geq 2^n$ for every clean alphabetic variant ψ_n of ξ_n.*

Proof. Fix a number n. The formula ξ_n is defined in terms of simpler families of formulas β_i, γ_i for all $i \in \{0, \dots, n\}$ and $\alpha_{i,j}$ for all $i, j \in \{0, \dots, n\}$ with $j \leq i$. First we define β_i by an induction on $i \leq n$:

$\beta_0 := \mu x_0.x_n \wedge \cdots \wedge x_0$

$\beta_i := \mu x_i.\alpha_{i,i} \wedge \cdots \wedge \alpha_{i,0}$,

where $\alpha_{i,j}$ for $j \leq i$ is defined by an inner downwards induction such that $\alpha_{i,i} := x_i$ and for all j with $0 \leq j < i$ we set

$\alpha_{i,j} := \beta_j[\alpha_{i,i}/x_i] \cdots [\alpha_{i,j+1}/x_{j+1}]$.

Note that $FV(\beta_i) \subseteq \{x_n, \dots, x_{i+1}\}$ and $FV(\alpha_{i,j}) \subseteq \{x_n, \dots, x_i\}$ for all $j \leq i$. In the definition of β_i and the remainder of this section we assume that conjunction associates to the right. We then define γ_i with a downwards induction on i such that

$$\gamma_i := \beta_i[\gamma_n/x_n] \cdots [\gamma_{i+1}/x_{i+1}].$$

Finally, we set $\xi_n := \gamma_0$. Figure 2 depicts the closure graphs for ξ_2 and ξ_3. The formula ξ_2 is given in Example 6. The formula ξ_3 is already too large to be written out.

---

[6] Because the correspondence between parity formulas and ATAs and HESs, this is the standard way of approaching model checking for $\mu\mathtt{ML}$.

[7] In some situations it is even necessary to work with clean formulas. Suppose, for instance, that for a formula $\xi \in \mu\mathtt{ML}$ one wants to base an equivalent ATA $\mathbb{A}_\xi$ on the set of *subformulas* of ξ. If we cannot associate a unique subformula of ξ with some bound variable x of ξ, then there is no sensible way to define the value of the transition map for this x.

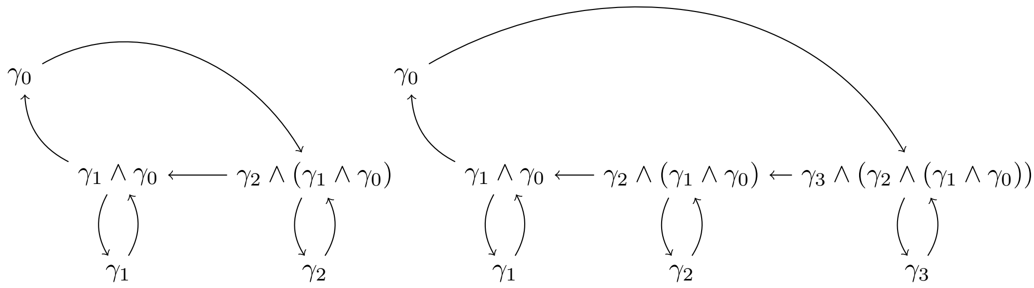


Figure 2 Structure of the closure graphs for ξ_2 (represented by γ_0 in the left graph) and for ξ_3 (represented by γ_0 in the right graph).

To show that $|\xi_n|^c \leq 2n + 2$ one needs to verify that

$$Clos(\xi_n) = \{\gamma_0, \dots, \gamma_n, \gamma_1 \wedge \gamma_0, \gamma_2 \wedge (\gamma_1 \wedge \gamma_0), \dots, \gamma_n \wedge \dots \wedge \gamma_0\}.$$

The crucial observation behind this result is that for all $j \leq i$ it holds that

$$\alpha_{i,j}[\gamma_n/x_n] \cdots [\gamma_{i+1}/x_{i+1}][\gamma_i/x_i] = \gamma_j.$$

This equation can be proved by a downward induction over $j \in \{i, \dots, 0\}$ for every fixed i.

To prove the result on the closure size of clean renamings of ξ_n we use the notion of *fixpoint depth*. Inductively we define $\mathsf{fd}(\varphi) := 0$ if φ is atomic, $\mathsf{fd}(\varphi_0 \odot \varphi_1) := \max(\mathsf{fd}(\varphi_0), \mathsf{fd}(\varphi_1))$, $\mathsf{fd}(\heartsuit\varphi) := \mathsf{fd}(\varphi)$, and $\mathsf{fd}(\eta x.\varphi) := 1 + \mathsf{fd}(\varphi)$. As we sketch below one can then show that

$$\mathsf{fd}(\xi_n) \geq 2^n. \tag{1}$$

To see how the claim about clean alphabetic variants follows from (1) let ψ_n be some clean alphabetical variant of ξ_n; it is not hard to see that we have $\mathsf{fd}(\psi_n) \geq 2^n$ as well. The claim then follows by the observation that

$$\text{every clean } \mu\text{-calculus formula } \chi \text{ satisfies } |\chi|^c \geq \mathsf{fd}(\chi). \tag{2}$$

For a proof of this statement, first observe that for any subformula $\eta x.\varphi \trianglelefteq \chi$, the closure of χ contains a formula of the form $\eta x.\varphi'$. This implies that $|\chi|^c = |Clos(\chi)| \geq |BV(\chi)|$. But if χ is a formula of fixpoint depth k, then there is a chain of subformulas $\eta_1 x_1.\varphi_1 \trianglelefteq \eta_2 x_2.\varphi_2 \trianglelefteq \cdots \trianglelefteq \eta_k x_k.\varphi_k$, and if χ is *clean*, then all these variables x_i must be distinct. This shows that $|BV(\chi)| \geq \mathsf{fd}(\chi)$. Combining these observations, we see that $|\chi|^c \geq \mathsf{fd}(\chi)$ indeed.

To prove (1) we need the auxiliary notion of the fixpoint depth of a variable in a formula. Given a formula φ and variable x, we let $\mathsf{fd}(x, \varphi)$, the *fixpoint depth of x in φ*, denote the maximum number of fixpoint operators that one may meet on a path from the root of the syntax tree of φ to a free occurrence of x in φ, with $\mathsf{fd}(x, \varphi) = -\infty$ if no such occurrence exists. Formally, we set $\mathsf{fd}(x, x) := 0$, $\mathsf{fd}(x, y) := -\infty$ if $x \neq y$, $\mathsf{fd}(x, \varphi_0 \odot \varphi_1) := \max\big(\mathsf{fd}(x, \varphi_0), \mathsf{fd}(x, \varphi_1)\big)$, $\mathsf{fd}(x, \heartsuit\varphi) := \mathsf{fd}(x, \varphi)$, $\mathsf{fd}(x, \eta x.\varphi) := -\infty$, and $\mathsf{fd}(x, \eta y.\varphi) = 1 + \mathsf{fd}(x, \varphi)$ if $x \neq y$. Without proof we mention that, provided $x \neq y$ and ψ is free for y in φ:

$$\mathsf{fd}(x, \varphi[\psi/y]) = \max\big(\mathsf{fd}(x, \varphi), \mathsf{fd}(y, \varphi) + \mathsf{fd}(x, \psi)\big).$$

From this we immediately infer that

$$\mathsf{fd}(x, \varphi[\psi/y]) \geq \mathsf{fd}(y, \varphi) + \mathsf{fd}(x, \psi), \tag{3}$$

which shows that every substitution doubles the fixpoint depth of a variable and leads to the exponential bound in (1). More concretely one can show that for all k and i such that $k > i$ it holds that

$$\mathsf{fd}(x_k, \beta_i) \geq 2^i \tag{4}$$

From this (1) follows because β_n is a subformula of ξ_n. The statement (4) is shown by an induction over i, where in the inductive step one proves with an inner induction over $j \in \{i-1, \ldots, 0\}$ that $\mathsf{fd}(x_k, \alpha_{i,j}) \geq 2^{i-1} + \cdots + 2^j$. We leave the details to the reader. ◀

6 Standard formulas and parity formulas

In this section we show how to move back and forth between standard μ-calculus formulas and parity formulas, in such a way that the closure-size of the standard formula corresponds linearly to the size of the parity formula and the alternation depth is preserved.

From standard formulas to parity formulas

Our main theorem states that for an arbitrary tidy formula, we can find an equivalent parity formula that is based on the formula's closure graph, and has an index which is bounded by the alternation depth of the formula.

▶ Remark 11. To stress our point that our results apply to ATAs and HESs too, suppose that we want to base an ATA $\mathbb{A}_\xi$ on the closure set of a formula ξ, or, for the sake of a perspicuous definition, on the set $A := \{\widehat{\varphi} \mid \varphi \in \mathit{Clos}(\xi)\}$. It is clear how to define the transition map Δ: we simply put $\Delta(\widehat{\varphi}) := \varphi$ if φ is atomic, $\Delta(\widehat{\varphi \odot \psi}) := \widehat{\varphi} \odot \widehat{\psi}$ (for $\odot \in \{\wedge, \vee\}$), $\Delta(\widehat{\heartsuit\varphi}) := \heartsuit\widehat{\varphi}$ (for $\heartsuit \in \{\Diamond, \Box\}$), and $\Delta(\widehat{\eta x.\varphi}) := \varphi[\widehat{\eta x.\varphi}/x]$ (for $\eta \in \{\mu, \nu\}$). What is *not* obvious, however, is how to define the priority map on the set A (unless ξ is clean); this is exactly the issue we address here.

▶ **Theorem 12.** *There is a construction transforming an arbitrary tidy formula $\xi \in \mu\mathtt{ML}$ into an equivalent parity formula $\mathbb{G}_\xi$, which is based on the closure graph of ξ, so that $|\mathbb{G}_\xi| = |\xi|^c$ and $ind(\mathbb{G}_\xi) \leq ad(\xi)$.*

The formula $\mathbb{G}_\xi = (V, E, L, \Omega, v_I)$ is defined such that (V, E) is the closure graph of ξ, $v_I = \xi$ and L is the labelling that maps a literal to itself, a boolean or modal formula to its main connective and a fixpoint formula to ϵ. Clearly this guarantees $|\mathbb{G}_\xi| = |\xi|^c$. The main difficulty is in defining the priority map Ω on $\mathit{Clos}(\xi)$ such that $\mathbb{G}_\xi$ is equivalent to ξ and $ind(\mathbb{G}_\xi) \leq ad(\xi)$, *without assuming that ξ is clean.*

The definition of Ω is such that it assigns priorities to the fixpoint formulas in the closure of ξ. Because every cycle in the trace relation needs to pass over at least one fixpoint formula this makes sure that condition 2) of Definition 3 is satisfied by $\mathbb{G}_\xi$. In fact we can take Ω to be the restriction of a global priority map Ω_g, which uniformly assigns a priority to every tidy fixpoint formula in $\mu\mathtt{ML}$. The function Ω_g itself is defined cluster-wise from a strict partial ordering $\sqsubset_C$ over the set of all tidy fixpoint formulas. To define $\sqsubset_C$ we make use of the following notion of a *free* subformula.

▶ **Definition 13.** *Let φ and ψ be μ-calculus formulas. We say that φ is a* free *subformula of ψ, notation: $\varphi \trianglelefteq_f \psi$, if $\psi = \psi'[\varphi/x]$ for some formula ψ' such that $x \in FV(\psi')$ and φ is free for x in ψ'.*

The following is a useful characterisation of the free subformula relation (see [15] for a proof):

$$\varphi \trianglelefteq_f \psi \text{ iff } \varphi \in \mathit{Sfor}(\psi) \cap \mathit{Clos}(\psi).$$

▶ **Definition 14.** *We let $\equiv_C$ denote the equivalence relation generated by the relation $\to_C$, in the sense that: $\varphi \equiv_C \psi$ if $\varphi \twoheadrightarrow_C \psi$ and $\psi \twoheadrightarrow_C \varphi$. We will refer to the equivalence classes of $\equiv_C$ as* (closure) clusters*, and denote the cluster of a formula φ as $C(\varphi)$.*

We define the closure priority relation $\sqsubseteq_C$ on fixpoint formulas by putting $\varphi \sqsubseteq_C \psi$ precisely if $\psi \twoheadrightarrow^{\psi}_C \varphi$, where $\twoheadrightarrow^{\psi}_C$ is the relation given by $\rho \twoheadrightarrow^{\psi}_C \sigma$ if there is a trace $\rho = \chi_0 \to_C \chi_1 \to_C \cdots \to_C \chi_n = \sigma$ such that $\psi \trianglelefteq_f \chi_i$, for every $i \in [0,..,n]$. We write $\varphi \sqsubset_C \psi$ if $\varphi \sqsubseteq_C \psi$ and $\psi \not\sqsubseteq_C \varphi$.

Using $\sqsubset_C$ we can define the priority of a fixpoint formula as follows:

▶ **Definition 15.** *An* alternating $\sqsubset_C$-chain *of length n is a sequence $(\eta_i x_i.\chi_i)_{i\in[1,..,n]}$ of tidy fixpoint formulas such that $\eta_i x_i.\chi_i \sqsubset_C \eta_{i+1} x_{i+1}.\chi_{i+1}$ and $\eta_{i+1} = \overline{\eta_i}$ for all $i \in [0,..,n-1]$. We say that such a chain* starts *at $\eta_1 x_1.\chi_1$ and* leads up to *$\eta_n x_n.\chi_n$.*

Given a tidy fixpoint formula ξ, we let $h^{\uparrow}(\xi)$ and $h^{\downarrow}(\xi)$ denote the maximal length of any alternating $\sqsubset_C$-chain starting at, respectively leading up to, ξ. Given a closure cluster D, we let $cd(D)$ denote the maximal length of an alternating $\sqsubset_C$-chain in D.

The global priority function *$\Omega_g : \mu\mathtt{ML}^t \to \omega$ is defined cluster-wise, as follows. Take an arbitrary tidy fixpoint formula $\eta y.\varphi$, and define*

$$\Omega_g(\eta y.\varphi) := \begin{cases} cd(C(\psi)) - h^{\uparrow}(\psi)) & \text{if } cd(C(\psi) - h^{\uparrow}(\psi)) \text{ has parity } \eta \\ \big(cd(C(\psi)) - h^{\uparrow}(\psi)\big) + 1 & \text{if } cd(C(\psi)) - h^{\uparrow}(\psi)) \text{ has parity } \overline{\eta}, \end{cases}$$

where we recall that we associate μ and ν with odd and even parity, respectively. If ψ is not of the form $\eta y.\varphi$, we leave $\Omega_g(\psi)$ undefined.

Finally we define the priority function Ω of the parity formula $\mathbb{G}_\xi$ to be $\Omega := \Omega_g \restriction_{Clos(\xi)}$.

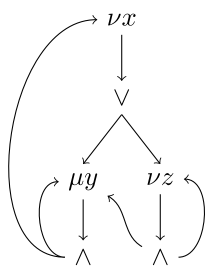
▶ Remark 16. The definition of the priority map Ω_g and of the priority order $\sqsubset_C$ on which it is based, may look overly complicated. In fact, simpler definitions would suffice if we are only after the equivalence of ξ with $\mathbb{G}_\xi$ and we do not need an exact match of index and alternation depth.

In particular, we could have introduced an alternative priority order $\sqsubset'_C$ by putting $\varphi \sqsubset'_C \psi$ if $\varphi \equiv_C \psi$ and $\psi \triangleleft_f \varphi$. This definition of $\sqsubset'_C$ is similar to the definition of a valid thread in [3]. If we would base a priority map Ω'_g on $\sqsubset'_C$ instead of on $\sqsubset_C$, then we could prove the equivalence of any tidy formula ξ with the associated parity formula $\mathbb{G}'_\xi$ that is just like $\mathbb{G}$ but uses Ω'_g as its priority map. However, we would not be able to prove that the index of $\mathbb{G}'_\xi$ is bounded by the alternation depth of ξ.

To see this, consider the following formula:

$$\alpha_x := \nu x.\big((\mu y.x \wedge y) \vee \nu z.(z \wedge \mu y.x \wedge y)\big).$$

We leave it for the reader to verify that this formula has alternation depth two, and that its closure graph looks as in the picture to the right (where we only indicate the main connective of the formulas):



Let α_y and α_z be the other two fixpoint formulas in the cluster of α_x, that is, let $\alpha_y := \mu y.\alpha_x \wedge y$ and $\alpha_z := \nu z.z \wedge \alpha_y$. These formulas correspond to the nodes in the graph that are labelled μy and νz, respectively. Now observe that we have $\alpha_x \triangleleft_f \alpha_y \triangleleft_f \alpha_z$, so that this cluster has an alternating $\sqsubset'_C$-chain of length *three*: $\alpha_z \sqsubset'_C \alpha_y \sqsubset'_C \alpha_x$. Note however, that any trace from α_y to α_z must pass through α_x, the $\sqsubset_C$-maximal element of the cluster. In particular, we do *not* have $\alpha_z \sqsubset_C \alpha_y$, so that there is *no* $\sqsubset_C$-chain of length three in the cluster.

A different kind of simplification of the global priority map would be to define

$$\Omega''_g(\psi) := \begin{cases} h^{\downarrow}(\psi) & \text{if } h^{\downarrow}(\psi) \text{ has parity } \eta \\ h^{\downarrow}(\psi) - 1 & \text{if } h^{\downarrow}(\psi) \text{ has parity } \overline{\eta}. \end{cases} \tag{5}$$

Using this definition for a priority map Ω''_g, we would again obtain the equivalence of ξ and the resulting parity formula $\mathbb{G}''_\xi := (\mathbb{C}_\xi, \Omega''_g{\upharpoonright}_{Clos(\xi)})$. In addition, we would achieve that the index of the parity formula $\mathbb{G}''_\xi$ satisfies $ind(\mathbb{G}''_\xi) \leq ad(\xi) + 1$. However, the above formula α_x would be an example of a formula ξ where $ind(\mathbb{G}''_\xi)$ exceeds $ad(\xi)$: We leave it for the reader to verify that we would get $\Omega''_g(\alpha_z) = 0$, $\Omega''_g(\alpha_y) = 1$ and $\Omega''_g(\alpha_x) = 2$, implying that $ind(\mathbb{G}''_\xi) = 3$.

With our definition of the priority map Ω_g, we find the same values for α_y and α_x as with Ω''_g, but we obtain $\Omega_g(\alpha_z) = 2$, implying that $ind(\mathbb{G}_x) = 2 = ad(\xi)$ as required.

In our technical report [15] we prove in detail that $\mathbb{G}_\xi$ is in fact equivalent to ξ and that $ind(\mathbb{G}_\xi) \leq ad(\xi)$. The proof of the equivalence proceeds by induction on the length of ξ, where we use the strengthened inductive hypothesis that each formula $\varphi \in Clos(\xi)$ is equivalent to $\mathbb{G}_\xi\langle\varphi\rangle$ (that is, the version of $\mathbb{G}$ where we take φ as the initial state). In the crucial case of the inductive step we have $\xi = \eta x.\chi$ and because of our strengthened inductive hypothesis we can assume that $\xi \notin Clos(\chi)$. We then apply the inductive hypothesis to the tidy variant $\chi[x'/x]$ of χ. The claim follows from a comparison of the evaluation games for $\mathbb{G}_\xi$ with the evaluation games for $\mathbb{G}_{\chi[x'/x]}$. For this we need the following proposition:

▶ **Proposition 17.** *Let $\xi = \eta x.\chi$ be a tidy fixpoint formula such that $x \in FV(\chi)$ and $\xi \notin Clos(\chi)$. Let $\chi' := \chi[x'/x]$ for some fresh variable x'. Then χ' is tidy and we have:*

1. *the substitution ξ/x' is a bijection between $Clos(\chi')$ and $Clos(\xi)$.*

Let $\varphi, \psi \in Clos(\chi')$. Then we have

2. *if $\varphi \neq x'$, then $\varphi \to_C \psi$ iff $\varphi[\xi/x'] \to_C \psi[\xi/x']$ and $L_C(\varphi) = L_C(\varphi[\xi/x'])$;*
3. *if $x' \in FV(\varphi)$ then $\varphi \trianglelefteq_f \psi$ iff $\varphi[\xi/x'] \trianglelefteq_f \psi[\xi/x']$;*
4. *if φ and ψ are fixpoint formulas then $\psi \sqsubseteq_C \varphi$ iff $\psi[\xi/x'] \sqsubseteq_C \varphi[\xi/x']$;*
5. *if $(\varphi_n)_{n\in\omega}$ is an infinite trace through $Clos(\chi')$, then $(\varphi_n)_{n\in\omega}$ has the same winner as $(\varphi_n[\xi/x'])_{n\in\omega}$.*

The crucial step in proving that $ind(\mathbb{G}_\xi) \leq ad(\xi)$ is to establish a link between the alternation depth of ξ and the length of alternating $\sqsubset_C$-chains in the closure graph of ξ. This is done by the following proposition, which can be seen as giving an alternative characterisation of the alternation depth of a formula. With $\eta \in \{\mu, \nu\}$, we let $cd_\eta(\xi)$ denote the maximal length of an alternating $\sqsubset_C$-chain in $Clos(\xi)$ that leads up to an η-formula.

▶ **Proposition 18.** *For any tidy formula ξ and $\eta \in \{\mu, \nu\}$, we have*

$$cd_\eta(\xi) \leq n \text{ iff } \xi \in \Theta^\eta_n. \tag{6}$$

Hence the alternation depth of ξ is equal to the length of its longest alternating $\sqsubset_C$-chain.

The main challenge in proving Proposition 18 is the direction from right to left, and more specifically the case of the definition of alternation depth that concerns the closure of Θ^{η}_{n} under substitutions. Here we carefully analyse how the alternating $\sqsubset_C$-chains in $C(\psi[\xi/x])$ relate to the ones in $C(\psi)$. For the details, which are fairly complex, we refer to our technical report [15]. Here we just state the crucial proposition that establishes this relation.

▶ **Proposition 19.** *Let ξ and χ be formulas such that ξ is free for x in χ, $\xi \not\trianglelefteq_f \chi$, and $x \notin FV(\xi)$. Furthermore, let $\psi \in Clos(\chi)$ be such that $\psi[\xi/x] \notin Clos(\chi) \cup Clos(\xi)$. Then*

1. *the substitution $\xi/x : C(\psi) \to C(\psi[\xi/x])$ is a bijection between $C(\psi)$ and $C(\psi[\xi/x])$.*

Let $\varphi_0, \varphi_1 \in C(\psi)$. Then we have

2. *$\varphi_0 \to_C \varphi_1$ iff $\varphi_0[\xi/x] \to_C \varphi_1[\xi/x]$ and $L_C(\varphi_0) = L_C(\varphi_0[\xi/x])$;*
3. *$\varphi_0 \trianglelefteq_f \varphi_1$ iff $\varphi_0[\xi/x] \trianglelefteq_f \varphi_1[\xi/x]$;*
4. *$h^{\downarrow}(\varphi_0) = h^{\downarrow}(\varphi_0[\xi/x])$, if φ_0 is a fixpoint formula.*

From parity formulas to standard formulas

The construction of an equivalent μ-calculus formula from a parity formula is well known, see for instance [17, 20]. The following theorem provides an analysis on how it behaves in terms of closure size and alternation depth. Given a parity formula $\mathbb{G}$, we let $\mathbb{G}\langle v\rangle$ denote its variant that takes v as its initial state.

▶ **Theorem 20.** *For any parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ there is a map $\mathtt{tr}_{\mathbb{G}} : V \to \mu\mathtt{ML}$ such that, for every $v \in V$:*

1. *$\mathbb{G}\langle v\rangle \equiv \mathtt{tr}_{\mathbb{G}}(v)$;*
2. *$|\mathtt{tr}_{\mathbb{G}}(v)|^c \leq 2 \cdot |\mathbb{G}|$;*
3. *$ad(\mathtt{tr}_{\mathbb{G}}(v)) \leq ind(\mathbb{G})$.*

The details of the definition of $\mathtt{tr}_{\mathbb{G}}$ and the proofs of items 1–3 can be found in our technical report [15]. Here, we illustrate the basic idea behind the construction by considering the simplified case where the priority map Ω is injective.[8] The definition of $\mathtt{tr}_{\mathbb{G}}$ proceeds by an induction on the lexicographic order over the pairs of numbers $(|\mathsf{Dom}(\Omega)|, |\mathbb{G}|)$, and we allow ourselves to be sloppy in considering structures consisting of parity formulas without initial vertex. Let T be a top cluster of $\mathbb{G}$, that is, the states in T are not reachable from any state outside T. We make the following case distinction:

Case 1: T is degenerate. In this case we have $T = \{v\}$ for some $v \notin \mathsf{Ran}(E)$. Let $\mathbb{G}'$ be the structure we obtain from $\mathbb{G}$ by removing v from V. We may apply the induction hypothesis to $\mathbb{G}'$ because it is strictly smaller than $\mathbb{G}$, while having no more elements in the domain of the priority map. We define $\mathtt{tr}_{\mathbb{G}}(u) := \mathtt{tr}_{\mathbb{G}\langle u\rangle}(u)$ for $u \neq v$, while for v we set define $\mathtt{tr}_{\mathbb{G}}(v)$ by connecting the formulas $\mathtt{tr}_{\mathbb{G}\langle u\rangle}(u)$ for $u \in E(v)$ with $L(v)$ in the obvious way.
Case 2: T is non-degenerate. In this case we have $T \cap \mathsf{Dom}(\Omega) \neq \varnothing$; let $m \in T$ be the state in T of maximal priority, which is unique because of our assumption that Ω is injective.

For the induction we then consider a fresh propositional variable p_m and define $\mathbb{G}^- = (V^-, E^-, L^-, \Omega^-, v_I)$ as the parity formula over $\mathsf{Q} \cup \{p_m\}$, given by

$$\begin{array}{lcl} V^- & := & V \cup \{m^*\} \\ E^- & := & \{(v,x) \mid (v,x) \in E, x \neq m\} \cup \{(v,m^*) \mid (v,m) \in E\} \\ \Omega^- & := & \Omega\restriction_{V\setminus\{m\}}, \end{array}$$

[8] In fact, it is not hard to see that by shifting priorities we can reduce the general case to this.

while its labelling L^- is defined by putting

$$L^-(v) := \begin{cases} L(v) & \text{if } v \in V \\ p_m & \text{if } v = m^*. \end{cases}$$

Since $|\mathsf{Dom}(\Omega^-)| < |\mathsf{Dom}(\Omega)|$, inductively we have a map $\mathtt{tr}_{\mathbb{G}^-} : V^- \to \mu\mathtt{ML}(\mathsf{Q} \cup \{p_m\})$. Let η be the parity of m and define $\mathtt{tr}_{\mathbb{G}}$ as

$$\begin{array}{lll} \mathtt{tr}_{\mathbb{G}}(m) & := & \eta p_m.\mathtt{tr}_{\mathbb{G}^-}(m) \\ \mathtt{tr}_{\mathbb{G}}(v) & := & \mathtt{tr}_{\mathbb{G}^-}(v)[\mathtt{tr}_{\mathbb{G}}(m)/p_m] \quad \text{for } v \in V. \end{array}$$

The key claim that entails item 2 of Theorem 20 is that

$$|\mathit{Clos}(\mathbb{G})| \leq |\mathbb{G}| + |\mathsf{Dom}(\Omega)|,$$

where $\mathit{Clos}(\mathbb{G}) := \bigcup \{ \mathit{Clos}(\mathtt{tr}_{\mathbb{G}}(v)) \mid v \in V \}$. This claim can be proved by the same induction as is used in the definition of $\mathtt{tr}_{\mathbb{G}}$: The point is to treat the closures of all the translations for vertices in $\mathbb{G}$ *in parallel*. The inductive case for non-degenerate clusters then follows with the observation that $\mathit{Clos}(\mathbb{G}) \subseteq \{\varphi[\mathtt{tr}_{\mathbb{G}}(m)/p_m] \mid \varphi \in \mathit{Clos}(\mathbb{G}^-)\}$.

7 Conclusion

This paper contributes to the theory of the modal μ-calculus by studying in detail some representations that are commonly used in order to prove complexity-theoretic results on problems such as model checking or satisfiability. We introduced the notion of a parity formula as a natural graph-based structure for representing formulas, and, building on work by Bruse, Friedmann & Lange [6] we focused on defining succinct parity formula representation on the closure graph of a standard formula. We showed in Proposition 10 that the renaming of bound variables can cause an exponential blow-up if the target formula is required to be clean. To realise the optimal upper complexity bound of model checking for all μ-calculus formulas, as our main contribution, Theorem 12 provides a construction of a parity formula that is based on the closure graph of a given formula, preserves its alternation-depth but does *not* assume the input formula to be clean.

There is a lot more to say about parity formulas as graph-based representations of μ-calculus formulas, but here we confine ourselves to the following.

Our example in Section 5 shows that closure size is not invariant under alphabetical equivalence. This matter could be investigated more thoroughly — here are some pertinent questions. Can we compute alphabetical variants of *minimal* closure size? If we make the reasonable assumption that alphabetical variants should be identified, then we should define the size of a formula as the size of its closure, up to alpha-equivalence; but can we base a parity formula on the quotient of the closure set under α-equivalence? Some answers to these questions can be found in our technical report [15].

Second, we used parity formulas here as a means to understand complexity-theoretic results pertaining to the modal μ-calculus, but it could be interesting to study these structures in their own right. A natural first question is to find a good notion of a morphism or an equivalence between parity formulas. One might then for instance investigate whether Kozen's expansion map [14] is a morphism from the parity formula based on the subformula dag to the parity formula on the closure. Furthermore, because parity formulas are representations of μ-calculus formulas one might also take a more logical perspective, and develop, for instance, their model theory or proof theory.

References

1 B. Afshari and G. Leigh. Cut-free completeness for modal mu-calculus. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic In Computer Science (LICS'17)*, pages 1–12. IEEE Computer Society, 2017.

2 A. Arnold and D. Niwiński. *Rudiments of μ-calculus*, volume 146 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2001.

3 D. Baelde, A. Doumane, and A. Saurin. Infinitary proof theory: the multiplicative additive case. In J.-M. Talbot and L. Regnier, editors, *Proceedings of the 25th EACSL Annual Conference on Computer Science Logic, CSL 2016*, volume 62 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

4 J. Bradfield and C. Stirling. Modal μ-calculi. In J. van Benthem, P. Blackburn, and F. Wolter, editors, *Handbook of Modal Logic*, pages 721–756. Elsevier, 2006.

5 J.C. Bradfield and I. Walukiewicz. The mu-calculus and model checking. In E. M. Clarke, Th.A. Henzinger, H. Veith, and R. Bloem, editors, *Handbook of Model Checking*, pages 871–919. Springer, 2018. `doi:10.1007/978-3-319-10575-8\_26`.

6 F. Bruse, O. Friedmann, and M. Lange. On guarded transformation in the modal μ-calculus. *Logic Journal of the IGPL*, 23(2):194–216, 2015.

7 C.S. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. Deciding parity games in quasipolynomial time. In H. Hatami, P. McKenzie, and V. King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, (STOC 2017)*, pages 252–263, 2017.

8 G. D'Agostino and M. Hollenberg. Logical questions concerning the μ-calculus. *Journal of Symbolic Logic*, 65:310–332, 2000.

9 S. Demri, V. Goranko, and M. Lange. *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016.

10 E.A. Emerson and C.S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of the 32nd Symposium on the Foundations of Computer Science*, pages 368–377. IEEE Computer Society Press, 1991.

11 G. Fontaine and Y. Venema. Some model theory for the modal mu-calculus: syntactic characterizations of semantic properties. *Logical Methods in Computer Science*, 14(1), 2018.

12 E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logic, and Infinite Games*, volume 2500 of *LNCS*. Springer, 2002.

13 D. Janin and I. Walukiewicz. On the expressive completeness of the propositional μ-calculus w.r.t. monadic second-order logic. In *Proceedings of the Seventh International Conference on Concurrency Theory, CONCUR '96*, volume 1119 of *LNCS*, pages 263–277, 1996.

14 D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

15 C. Kupke, J. Marti, and Y. Venema. Size matters in the modal μ-calculus, 2020. `arXiv:2010.14430`.

16 D. Niwiński. On fixed point clones. In L. Kott, editor, *Proceedings of the 13th International Colloquium on Automata, Languages and Programming (ICALP 13)*, volume 226 of *LNCS*, pages 464–473, 1986.

17 D. Niwiński. Fixed point characterization of infinite behavior of finite-state systems. *Theoretical Computer Science*, 189:1–69, 1997.

18 H. Seidl and A. Neumann. On guarding nested fixpoints. In *Computer Science Logic, 13th International Workshop, CSL '99, 8th Annual Conference of the EACSL, Madrid, Spain, September 20-25, 1999, Proceedings*, pages 484–498, 1999.

19 I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. *Information and Computation*, 157:142–182, 2000.

20 T. Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bulletin of the Belgian Mathematical Society*, 8:359–391, 2001.