Weak Automata for the Linear Time μ-Calculus

Martin Lange

Institut für Informatik, University of Munich, Germany

Abstract. This paper presents translations forth and back between formulas of the linear time μ-calculus and finite automata with a weak parity acceptance condition. This yields a normal form for these formulas, in fact showing that the linear time alternation hierarchy collapses at level 0 and not just at level 1 as known so far. The translation from formulas to automata can be optimised yielding automata whose size is only exponential in the alternation depth of the formula.

1 Introduction

One of the main reasons for the apparent success of automata theory within computer science is the tight connection that exists between automata and logics. Often, automata are the only tool for deriving (efficient) decision procedures for a logic.

Starting in the 60s, Büchi and Rabin have used automata to show that Monadic Second Order Logic (MSO) over infinite words and trees is decidable [5, 18]. Since then, automata have been found to be particularly useful for temporal logics which usually are fragments of MSO over words or trees. Their emptiness and membership problems are used to decide satisfiability and model checking for various logics, and often certain automata and logics have been shown to be equi-expressive. This characterises logics computationally and automata denotationally.

The type of automaton used – i.e. structures they work upon, rank of determinism, acceptance mode – depends on the type of logic one is interested in: linear time logics need automata over words [21], branching time logics need automata over trees [14]. In any case, alternating automata [16, 6] – i.e. those featuring nondeterministic as well as universal choices – have proved to be most beneficial for two reasons: (1) a temporal logic usually has disjunctions and conjunctions which can uniformly be translated into automata states. (2) Alternating automata are usually more succinct then (non-)deterministic automata, hence, they can lead to more efficient decision procedures.

The acceptance condition needs to match the temporal constructs featured in the logic. LTL for example is happy with a simple Büchi condition since it only has very simple temporal constructs. Logics with more complex temporal operators like extremal fixpoint quantifiers are best matched with more complex acceptance conditions like Rabin, Streett, or Muller conditions for example.

R. Cousot (Ed.): VMCAI 2005, LNCS 3385, pp. 267–281, 2005.

Best suited, however, for fixpoint logics with alternation[1] – the interleaved nesting of least and greatest fixpoint operators – are parity automata. Here, every state is assigned a priority, and acceptance means the parity of the least priority seen infinitely often in a run must be even. The match to fixpoint logics can be explained as follows: both least and greatest fixpoints are recursion mechanisms that get translated into automata states. A least fixpoint quantifier is a recursive program that is bound to terminate eventually. Its dual counterpart, a greatest fixpoint quantifier is a recursive program that is allowed to run ad infinitum. Thus, automata states obtained from least fixpoint quantifiers obtain odd priorities, those obtained from greatest fixpoint quantifiers obtain even priorities.

Fixpoint alternation means a least fixpoint recursion X can call a greatest fixpoint recursion Y which can in return call X, and vice versa. Then, the outermost program, i.e. the one that called the other first, determines whether or not infinite recursion is good or bad. Hence, for example seeing priority 17 infinitely often is alright, as long as priority 8 is also seen infinitely often.

The connection between parity tree automata, parity games – the evaluation of a run of an alternating parity automaton – and the modal μ-calculus is widely known, and much has been written about it [9, 8, 19, 7]. This immediately entails an equal connection between its pendant over infinite words, the linear time μ-calculus μTL [1, 20], and parity word automata. Just as the modal μ-calculus can be seen as a backbone for branching time logics, μTL is the backbone for linear time logics capable of defining at most regular properties.

Fixpoint alternation is what makes formulas hard to evaluate. Equally, the emptiness and word problems for parity automata are harder than those for simple Büchi automata and usually require algorithms that recurse over the number of priorities present in the automaton. It is fair to look for simpler acceptance conditions that still capture the essence of some logic's constructs. One possibility are weak automata. This concept was first introduced by Muller, Saoudi and Schupp [17] as a structural restriction on Büchi automata. Weakness refers to the fact that there are ω-regular languages that cannot be accepted by these automata.

Alternation, however, makes up for weakness [13]: every alternating Büchi automaton can be translated into a weak alternating Büchi automaton. The problem of having established the term "weak" but also knowing that these automata are not any weaker is solved by redefining weakness in terms of acceptance rather than the structure of an automaton's state space. Weak Büchi acceptance is looking for the occurrence of a final state rather than its infinite recurrence. Consequently, a weak parity automaton accepts if the least priority occurring at all is even.

[1] Note that the term *alternation* is overloaded. It describes the type of the transition function in an automaton as well as a structural property of formulas with fixpoint quantifiers. Each time we use the term alternation it should become clear from the context which type is meant. However, we will try to speak of *fixpoint alternation* in the latter case.

The advantage that weak parity automata have over normal parity automata is apparent: it is easy to keep track of the least priority seen in a run so far without worrying whether or not it would occur infinitely often. Consequently, emptiness or word problems for weak automata are easier to solve.

Here we present first of all a direct translation from formulas of the linear time μ-calculus into weak alternating parity automata. The novel part of this is the directness. It is known that this translation is possible via alternating parity automata, alternating Muller automata, and alternating Büchi automata. The complexity of this translation however is exponential in the size of the formula. We also show how to improve the direct translation in order to obtain weak automata that are exponentially large in the alternation depth of the formula only.

Then we present the converse translation from weak alternating parity automata back into μTL. This is based on ideas from [11] and [15]. The latter deals with the connection between weak alternating automata and MSO. The former considered automata models for μTL, obtaining an important result that does not hold true for the modal μ-calculus [4]: every ω-regular language can be defined by a μTL formula of alternation depth at most 1. A simple translation from ω-regular expressions into μTL – just meant to form an intuition about how μTL formulas express ω-regular properties – yields an alternative proof of this result. But the translation back from weak alternating parity automata into formulas of the linear time μ-calculus even improves this: the μTL alternation hierarchy indeed collapses at level 0.

This paper is organised as follows. Section 2 recalls notions about infinite words, alternating automata and the linear time μ-calculus. Section 3 presents the aforementioned translations. Their complexities and possible optimisations are discussed in Section 4. Finally, Section 5 concludes with a discussion about the usefulness of this automaton characterisation for ω-regular word languages.

2 Preliminaries

2.1 Infinite Words and ω-Regular Expressions

Let $\Sigma = \{a, b, \ldots\}$ be a finite set of symbols. As usual, Σ^ω denotes the set of infinite words over Σ. Given a $w \in \Sigma^\omega$ we write w^k for the k-th symbol in w, i.e. $w = w^0 w^1 w^2 \ldots$.

Σ^* denotes the set of finite words over Σ. For an $L_1 \subseteq \Sigma^*$ and an $L_2 \subseteq \Sigma^\omega$, their concatenation $L_1 L_2$ consists of all words $w \in \Sigma^\omega$ that can be decomposed into $w = w_1 w_2$ s.t. $w_1 \in L_1$ and $w_2 \in L_2$.

An ω-regular expression is of the form

$$\alpha \;:=\; \epsilon \mid a \mid \alpha \cup \alpha \mid \alpha ; \alpha \mid \alpha^* \mid \alpha^\omega$$

describing, resp. the language containing just the empty word, all words beginning with the letter a, the union and the concatenation of two languages, finite and infinite iteration of a language. We write $[\![\alpha]\!]$ for the language defined by α.

Theorem 1. [5] *For every ω-regular language L there is an ω-regular expression of the form $\delta = \bigcup_{i=1}^{n} \alpha_i ; \beta_i^\omega$ for some $n \in \mathbb{N}$, s.t. $[\![\delta]\!] = L$. Additionally, for all $i = 1, \ldots, n$ we have: neither α_i nor β_i contains a subexpression of the form γ^ω, and $\epsilon \notin [\![\beta_i]\!]$.*

2.2 The Linear Time μ-Calculus μTL

Definition 1. Let $\Sigma = \{a, b, \ldots\}$ be a finite alphabet, and let $\mathcal{V} = \{X, Y, \ldots\}$ be a set of propositional variables. Formulas of the linear time μ-calculus μTL are defined by the following grammar.

$$\varphi \ ::= \ a \mid X \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \mu X.\varphi \mid \nu X.\varphi$$

where $a \in \Sigma$ and $X \in \mathcal{V}$. With $\varphi\{\psi/\chi\}$ we denote the formula that is obtained from φ by replacing every occurrence of χ in it with ψ. We will write σ for either of the fixpoint quantifiers μ or ν.

The set of subformulas of a μTL formula is defined in the usual way, i.e. $Sub(\varphi \vee \psi) = \{\varphi \vee \psi\} \cup Sub(\varphi) \cup Sub(\psi)$ and $Sub(\sigma X.\varphi) = \{\sigma X.\varphi\} \cup Sub(\varphi)$ for example. Equally, $free(\varphi)$ is the usual set of variables occurring in φ which are not in the scope of a binding quantifier. We assume that formulas are well-named, i.e. a variable is not quantified more than once in a formula. Then for every closed φ there is a function $fp_\varphi() : \mathcal{V} \cap Sub(\varphi) \to Sub(\varphi)$ which maps each variable X to its unique defining fixpoint formula $\sigma X.\varphi$. We say that X has fixpoint type μ if $fp_\varphi(X) = \mu X.\psi$ for some ψ, otherwise it is ν.

Assuming that $1 < |\Sigma| < \infty$ it is easy to define the propositional constants true and false as $\mathtt{tt} := \bigvee_{a \in \Sigma} a$ and $\mathtt{ff} := a \wedge b$ for some $a, b \in \Sigma$ with $a \neq b$. If the assumption does not hold then one can also include $\mathtt{tt}$ and $\mathtt{ff}$ as primitives in the logic.

Formulas of μTL are interpreted over ω-words $w \in \Sigma^\omega$. Since the semantics is defined inductively, one needs to explain the meaning of open formulas. This is done using an *environment* which is a mapping $\rho : \mathcal{V} \to 2^{\mathbb{N}}$. With $\rho[X \mapsto M]$ we denote the function that maps X to M and behaves like ρ on all other arguments.

$$\begin{array}{lcl}
[\![a]\!]^w_\rho & := & \{i \in \mathbb{N} \mid w^i = a\} \\
[\![X]\!]^w_\rho & := & \rho(X) \\
[\![\varphi \vee \psi]\!]^w_\rho & := & [\![\varphi]\!]^w_\rho \cup [\![\psi]\!]^w_\rho \\
[\![\varphi \wedge \psi]\!]^w_\rho & := & [\![\varphi]\!]^w_\rho \cap [\![\psi]\!]^w_\rho \\
[\![\bigcirc\varphi]\!]^w_\rho & := & \{i \in \mathbb{N} \mid i+1 \in [\![\varphi]\!]^w_\rho\} \\
[\![\mu X.\varphi]\!]^w_\rho & := & \bigcap\{M \subseteq \mathbb{N} \mid [\![\varphi]\!]^w_{\rho[X \mapsto M]} \subseteq M\} \\
[\![\nu X.\varphi]\!]^w_\rho & := & \bigcup\{M \subseteq \mathbb{N} \mid M \subseteq [\![\varphi]\!]^w_{\rho[X \mapsto M]}\}
\end{array}$$

We write $w \models_\rho \varphi$ iff $0 \in [\![\varphi]\!]^w_\rho$. If φ does not contain free variables we also drop ρ since in this case the positions in w satisfying φ do not depend on it. The set of all models of φ is denoted $L(\varphi) := \{w \in \Sigma^\omega \mid w \models \varphi\}$. Two formulas φ and ψ are equivalent, $\varphi \equiv \psi$, if $L(\varphi) = L(\psi)$.

Approximants of a formula $\mu X.\varphi$ are defined for all $k \in \mathbb{N}$ as

$$\mu^0 X.\varphi \;:=\; \mathtt{ff} \qquad\qquad \mu^{k+1} X.\varphi \;:=\; \varphi\{\mu^k X.\varphi / X\}$$

Dually, approximants of a $\nu X.\varphi$ are defined as

$$\nu^0 X.\varphi \;:=\; \mathtt{tt} \qquad\qquad \nu^{k+1} X.\varphi \;:=\; \varphi\{\nu^k X.\varphi / X\}$$

The next result is a standard result about approximants.

Lemma 1.
a) $w \models \mu X.\varphi$ *iff there is a* $k \in \mathbb{N}$ *s.t.* $w \models \mu^k X.\varphi$.
b) $w \models \nu X.\varphi$ *iff for all* $k \in \mathbb{N}$*:* $w \models \nu^k X.\varphi$.

The fixpoint depth $fpd(\varphi)$ of φ measures the maximal number of fixpoint quantifiers seen on any path in $\varphi's$ syntax tree. It is defined as

$$\begin{aligned} fpd(a) = fpd(X) &:= 0 \\ fpd(\varphi \vee \psi) = fpd(\varphi \wedge \psi) &:= \max\{fpd(\varphi), fpd(\psi)\} \\ fpd(\bigcirc\varphi) &:= fpd(\varphi) \\ fpd(\sigma X.\varphi) &:= 1 + fpd(\varphi) \end{aligned}$$

We say that X depends on Y in φ, written $Y \prec_\varphi X$, if $Y \in free(fp_\varphi(X))$. We write $<_\varphi$ for the transitive closure of $\prec_\varphi$. The nesting depth $nd(\varphi)$ of φ is the length n of a maximal chain $X_0 <_\varphi \ldots <_\varphi X_n$. The alternation depth $ad(\varphi)$ is the length of such a maximal chain in which adjacent variables have different fixpoint types. Note that for any φ we have $ad(\varphi) \leq nd(\varphi) \leq fpd(\varphi)$. Let $\mu\mathrm{TL}^k := \{\varphi \in \mu\mathrm{TL} \mid free(\varphi) = \emptyset \text{ and } ad(\varphi) \leq k+1\}$.

A formula φ is guarded if every occurrence of a variable $X \in Sub(\varphi)$ is in the scope of a $\bigcirc$ operator inside of $fp_\varphi(X)$. It is strictly guarded if every occurrence of a variable $X \in Sub(\varphi)$ is immediately preceeded by a $\bigcirc$ operator.

Lemma 2. *Every* $\varphi \in \mu\mathrm{TL}^k$ *for any* $k \in \mathbb{N}$ *is equivalent to a strictly guarded* $\varphi' \in \mu\mathrm{TL}^k$.

Proof. It is known from [12] or [22] for example that every formula of the modal μ-calculus can equivalently be translated into a guarded formula. There, guardedness means occurrence in the scope of either a $\langle a \rangle$ or a $[a]$. The alternation depth is not effected by this process. The construction for μTL formulas proceeds in just the same way.

Finally, strict guardedness can easily be achieved by pushing the next operator inwards using the equivalences $\bigcirc(\varphi \vee \psi) \equiv \bigcirc\varphi \vee \bigcirc\psi$ and $\bigcirc(\varphi \wedge \psi) \equiv \bigcirc\varphi \wedge \bigcirc\psi$. What remains to be seen is that the next operator also commutes with the fixpoint quantifiers. Take a formula of the form $\bigcirc\mu X.\varphi(X, Y)$. By induction hypothesis, X is already strictly guarded in it, but Y may not. Let $\varphi^{(\bigcirc X)}$ be the formula that results from $\bigcirc\varphi$ by pushing the $\bigcirc$ operators in as far as possible

and removing it right in front of every occurrence of X. By hypothesis it exists. Now

$$\begin{aligned}
\bigcirc\mu X.\varphi(X,Y) &\equiv \bigcirc \bigvee_{i\in\mathbb{N}} \mu^i X.\varphi(X,Y) \\
&\equiv \bigcirc \mathtt{ff} \vee \bigvee_{i\geq 1} \bigcirc \mu^i X.\varphi(X,Y) \\
&\equiv \bigvee_{i\geq 1} \bigcirc \varphi(\mu^{i-1} X.\varphi(X,Y), Y) \\
&\equiv \bigvee_{i\geq 1} \varphi^{(\bigcirc X)}(\bigcirc \mu^{i-1} X.\varphi(X,Y), Y) \\
&\equiv \bigvee_{i\geq 1} \mu^i X.\varphi^{(\bigcirc X)}(X,Y) \\
&\equiv \mu X.\varphi^{(\bigcirc X)}(X,Y)
\end{aligned}$$

The penultimate step requires a straight-forward induction on i. The temporary introduction of infinitary formulas is justified by Lemma 1. The case of a greatest fixpoint formula is analogous. □

Just like the modal μ-calculus can define all (bisimulation-invariant) regular languages of infinite trees [10], μTL can define all ω-regular word languages. We will show this by giving a translation from ω-regular expressions into μTL.

Definition 2. For any ω-regular expression α we define $tr_X(\alpha) \in \mu\mathrm{TL}$ that describes the same ω-language over Σ. The inductive and uniform translation uses a free variable X that will eventually be bound by an $^\omega$-operator. This is essentially continuation-passing if the ω-regular expression is regarded as a computation.

$$\begin{aligned}
tr_X(\epsilon) &:= X \\
tr_X(a) &:= a \wedge \bigcirc X \\
tr_X(\alpha_0 \cup \alpha_1) &:= tr_X(\alpha_0) \vee tr_X(\alpha_1) \\
tr_X(\alpha_0;\alpha_1) &:= tr_X(\alpha_0)\{tr_X(\alpha_1)/X\} \\
tr_X(\alpha^*) &:= \mu Y.X \vee tr_Y(\alpha) \\
tr_X(\alpha^\omega) &:= \nu X.tr_X(\alpha)
\end{aligned}$$

This translation does not only give an automata-free proof of the fact that μTL can describe all ω-regular languages. It also yields an alternative way of showing that the alternation hierarchy in μTL collapses at level 1.

Theorem 2. *For every* μTL *formula* φ *there is a* $\varphi' \in \mu\mathrm{TL}^1$ *s.t.* $\varphi \equiv \varphi'$.

Proof. It is well-known that a μTL formula can be regarded as an alternating parity automaton. These can be translated into alternating Muller automata,

then into alternating Büchi automata, then into nondeterministic Büchi automata, and finally into ω-regular expressions – whilst preserving equivalence in each step. According to Theorem 1, the resulting expressions do not contain nested $^\omega$-operators. Using Definition 2 they can be translated into μTL formulas whose subformulas of the form $\mu X.\psi$ contain at most one free variable that is bound by a ν-operator. Hence, the alternation depth of the resulting formula is at most 1. □

Not only can this result be improved in terms of the level at which the hierarchy collapses – see below. Such a translation is also of no practical relevance since it is at least double exponential in the size of the formula.

2.3 Positive Boolean Formulas

For a given set Q let $\mathbb{B}^+(Q)$ be the set of positive boolean formulas over Q. I.e. $\mathbb{B}^+(Q)$ is the least set that contains Q and fulfils: if $f, g \in \mathbb{B}^+(Q)$ then $f \vee g, f \wedge g \in \mathbb{B}^+(Q)$. We say that $P \subset Q$ is a model of f if f evaluates to $\mathtt{tt}$ when every $q \in P$ in it is replaced by $\mathtt{tt}$ and every $q \notin P$ in it is replaced by $\mathtt{ff}$.

We write $f[q'/q]$ for the positive boolean formula that results from f by replacing every occurrence of q in it with q'.

Later we will use a simple operation $()^l$ that tags the elements of Q as q^l. This is extended to $\mathbb{B}^+(Q)$ in the obvious way: $(f \vee g)^l := (f)^l \vee (g)^l$, and $(f \wedge g)^l := (f)^l \wedge (g)^l$.

2.4 Alternating Automata

An alternating parity automaton (APA) is a tuple $\mathcal{A} = (Q, \Sigma, q_0, \delta, \Omega)$ where Q is a finite set of states, Σ a finite alphabet, and $q_0 \in Q$ the designated starting state, $\delta : Q \times \Sigma \to \mathbb{B}^+(Q)$ is the transition function. $\Omega : Q \to \mathbb{N}$ assigns to each state a priority.

A run r of $\mathcal{A}$ on a word $w \in \Sigma^\omega$ is an infinite tree rooted with q_0, s.t. for every node q_j on level i the set $\{p_1, \ldots, p_n\}$ of its children is a model of $\delta(q_j, w^i)$.

Let $\pi = q_0, q_1, \ldots$ be a path of a run r of $\mathcal{A}$ on w. Let

$$Occ(\pi) := \{q \mid \exists i \in \mathbb{N} \text{ s.t. } q_i = q\}$$
$$Inf(\pi) := \{q \mid \forall j \in \mathbb{N} : \exists i \geq j \text{ s.t. } q_i = q\}$$

A run r of an APA $\mathcal{A}$ on w is accepting iff for every path π of r: $\min\{\Omega(q) \mid q \in Inf(\pi)\}$ is even. $\mathcal{A}$ accepts w if there is an accepting run of $\mathcal{A}$ on w. The language $L(\mathcal{A})$ is the set of all words accepted by $\mathcal{A}$.

A weak alternating parity automaton (WAPA) is an APA $\mathcal{A}$ with a less demanding acceptance condition. A run r of a WAPA $\mathcal{A}$ on w is accepting if for all paths π of r: $\min\{\Omega(q) \mid q \in Occ(\pi)\}$ is even. $L(\mathcal{A})$ is defined in the same way.

An alternating Büchi automaton (ABA) is an APA $\mathcal{A} = (Q, \Sigma, q_0, \delta, \Omega)$ where $\Omega : Q \to \{0, 1\}$. We usually write an ABA as $(Q, \Sigma, q_0, \delta, F)$ with $F := \{q \in Q \mid$

$\Omega(q) = 0\}$. Acceptance then boils down to a state in F being visited infinitely often.

A weak alternating Büchi automaton (WABA) could just be defined as a WAPA with two priorities only. However, for technical reasons we prefer the equivalent and original definition from [17]. A WABA is an ABA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ where Q can be partitioned into components $C_0, \ldots, C_n$ s.t.

- for all $q \in Q$, $i, j \in \{0, \ldots, n\}$, $a \in \Sigma$: if $q \in C_i$ and $q' \in C_j$ and $\delta(q, a) = f(\ldots, q', \ldots)$ for some f then $j \leq i$,
- for all $0 \leq i \leq n$: $C_i \subseteq F$ or $C_i \cap F = \emptyset$.

3 From μTL to WAPA and Back

Definition 3. A WAPA with a hole q is a $\mathcal{A} = (Q, \Sigma, q_0, \delta, \Omega)$ with $q \in Q$ s.t. $\delta(q, a) = \bot$ (undefined) for any $a \in \Sigma$, and $\Omega(q) = \bot$. Intuitively, a WAPA with a hole is a WAPA whose construction is not finished yet.

Let $\mathcal{A}$ be a WAPA with a hole q, $\mathcal{B}$ be another WAPA and $L \subseteq \Sigma^\omega$. We write $\mathcal{A}[q : \mathcal{B}]$ for the WAPA that results from $\mathcal{A}$ by replacing q in $\mathcal{A}$ with the starting state of $\mathcal{B}$. We also write $\mathcal{A}[q : L]$ instead of $\mathcal{A}[q : \mathcal{B}]$ for some $\mathcal{B}$ with $L(\mathcal{B}) = L$.

Let $\mathcal{A}_{\mathtt{ff}} = (\{\mathtt{ff}\}, \Sigma, \mathtt{ff}, \{(\mathtt{ff}, a) \mapsto \mathtt{ff} \mid a \in \Sigma\}, \{\mathtt{ff} \mapsto 1\})$ be a WAPA that accepts the empty language.

Theorem 3. *For every closed $\varphi \in \mu$TL there is a WAPA $\mathcal{A}_\varphi$ s.t. $L(\mathcal{A}_\varphi) = L(\varphi)$.*

Proof. The proof proceeds by induction on the structure of φ. According to Lemma 2, φ can be assumed to be strictly guarded.

Despite closeness of φ, we need to handle open subformulas. This will be done using WAPAs with holes. Furthermore, we need to strengthen the inductive hypothesis: for every φ we will construct a WAPA $\mathcal{A} = (Q, \Sigma, q_0, \delta, \Omega)$ with $\Omega(q_0) > \Omega(q)$ for all $q \neq q_0$.

Figure 1 shows the intuition behind some of the cases below.

Case $\varphi = a$. Let $\mathcal{A}_a = (\{a, \mathtt{tt}, \mathtt{ff}\}, \Sigma, a, \delta, \Omega)$ with $\delta(a, a) = \mathtt{tt}$, $\delta(a, b) = \mathtt{ff}$ for any $b \neq a$, $\delta(q, a) = q$ for any $a \in \Sigma, q \in \{\mathtt{tt}, \mathtt{ff}\}$ and $\Omega(a) = 2$, $\Omega(\mathtt{ff}) = 1$, $\Omega(\mathtt{tt}) = 0$. Then $L(\mathcal{A}_a) = \{a\}\Sigma^\omega = L(a)$.

Case $\varphi = X$. Let $\mathcal{A}_X$ be the WAPA that consists of the hole X only.

Case $\varphi = \bigcirc\psi$. By hypothesis there is a WAPA $\mathcal{A}_\psi = (Q, \Sigma, q_0, \delta, \Omega)$ with $L(\mathcal{A}_\psi) = L(\psi)$. Assume $\varphi \notin Q$, and let $p := 1 + \max\{\Omega(q) \mid q \in Q\}$. Define $\mathcal{A}_\varphi = (Q \cup \{\varphi\}, \Sigma, \varphi, \delta', \Omega')$ with $\delta' = \delta \cup \{(q, a) \mapsto q_0 \mid a \in \Sigma\}$, $\Omega' = \Omega \cup \{q \mapsto p\}$. Then $L(\mathcal{A}_\varphi) = \Sigma L(\mathcal{A}_\psi) = L(\bigcirc\psi)$. Note that every run of $\mathcal{A}_\varphi$ starts with φ but then contains only states with strictly smaller priorities. Thus, a word aw is accepted by $\mathcal{A}_\varphi$ iff w is accepted by $\mathcal{A}_\psi$.

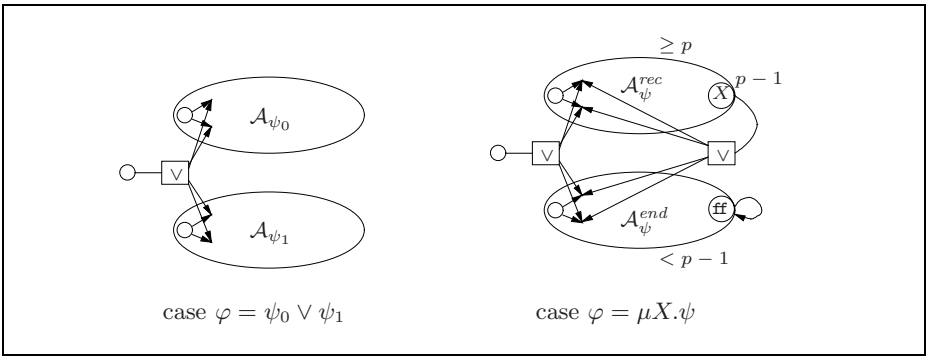

Fig. 1. Illustrations of the translation from μTL to WAPA.

Case $\varphi = \psi_0 \vee \psi_2$. By hypothesis there are WAPAs $\mathcal{A}_{\psi_i} = (Q_i, \Sigma, q_{0,i}, \delta_i, \Omega_i)$ for $i \in \{0,1\}$ s.t. $L(\mathcal{A}_{\psi_i}) = L(\psi_i)$. We can assume Q_0 and Q_1 to be disjoint. Let $Q := Q_0 \cup Q_1 \cup \{\varphi\}$. This is where strict guardedness is needed. It ensures that inside each $\mathcal{A}_{\psi_i}$ there is a proper transition between the starting state and any hole. Define $\mathcal{A}_\varphi = (Q, \Sigma, \varphi, \delta, \Omega)$ where for any $a \in \Sigma$, $q \in Q$:

$$\delta(q,a) := \begin{cases} \delta_0(q_{0,0}, a) \vee \delta(q_{0,1}, a) & \text{if } q = \varphi \\ \delta_i(q,a) & \text{if } q \in Q_i, i \in \{0,1\} \end{cases}$$

$$\Omega(q) := \begin{cases} 1 + \max\{\Omega_i(q) \mid q \in Q_i, i \in \{0,1\}\} & \text{if } q = \varphi \\ \Omega_i(q) & \text{if } q \in Q_i, i \in \{0,1\} \end{cases}$$

A run of $\mathcal{A}_\varphi$ on any w is also a run of either $\mathcal{A}_{\psi_0}$ or $\mathcal{A}_{\psi_1}$ on w with the exception that the root of the run in $\mathcal{A}_\varphi$ has a higher priority. Hence, if $w \in L(\mathcal{A}_\varphi)$ then $w \in L(\mathcal{A}_{\psi_i})$ for some $i \in \{0,1\}$. For the converse direction we need the stronger hypothesis. Suppose $w \in L(\mathcal{A}_{\psi_i})$ for some $i \in \{0,1\}$. Thus, on any path the minimal priority that occurs is even. However, this cannot be the priority of the root since it is the greatest occurring at all. But then the corresponding run of $\mathcal{A}_\varphi$ accepts w, too.

Case $\varphi = \psi_0 \wedge \psi_1$. Analogous to the previous case. The automaton for φ is obtained as the disjoint union of the automata for the conjuncts with a new starting state which does the conjunction of the two components' starting states.

Case $\varphi = \mu X.\psi$. By hypothesis there is a WAPA $\mathcal{A}_\psi = (Q, \Sigma, q_0, \delta, \Omega)$. Let $p' := \max\{\Omega(q) \mid q \in Q\}$ and $p := p' + 2 + (p' \mod 2)$ a strict even upper bound on all the priorities occurring in $\mathcal{A}_\psi$.

Note that $\mu X.\psi \equiv \psi$ if $X \notin \mathit{free}(\psi)$. Thus, $\mathcal{A}_\psi$ can be assumed to contain a hole X. Note that one hole suffices since holes are states that clearly behave in the same way – namely not at all – and hence, can be collapsed.

$\mathcal{A}_\varphi$ will consist of two disjoint copies of $\mathcal{A}_\psi$: one that unfolds the fixpoint a finite number of times and one that puts a halt to the recursion. Technically, let

$\mathcal{A}_\varphi = (Q', \Sigma, \varphi, \delta', \Omega')$ where

$$\begin{aligned} Q^{rec} &:= \{q^{rec} \mid q \in Q \setminus \{X\}\} \cup \{X\} \\ Q^{end} &:= \{q^{end} \mid q \in Q \setminus \{X\}\} \cup \{\mathtt{ff}\} \\ Q' &:= \{\varphi\} \cup Q^{rec} \cup Q^{end} \end{aligned}$$

$$\delta'(q,a) := \begin{cases} (\delta(q_0,a))^{rec} \vee (\delta(q_0,a))^{end} & \text{if } q = \varphi \\ (\delta(q_0,a))^{rec} \vee (\delta(q_0,a))^{end} & \text{if } q = X \\ \mathtt{ff} & \text{if } q = \mathtt{ff} \\ (\delta(q,a))^{rec} & \text{if } q \in Q^{rec} \setminus \{X\} \\ (\delta(q,a))^{end}[\mathtt{ff}/X] & \text{if } q \in Q^{end} \setminus \{\mathtt{ff}\} \end{cases}$$

$$\Omega'(q) := \begin{cases} 2 \cdot p - 1 & \text{if } q = \varphi \\ p - 1 & \text{if } q = X \\ 1 & \text{if } q = \mathtt{ff} \\ \Omega(q) + p & \text{if } q \in Q^{rec} \setminus \{X\} \\ \Omega(q) & \text{if } q \in Q^{end} \setminus \{\mathtt{ff}\} \end{cases}$$

Let $\mathcal{A}_\psi^{end}$ and $\mathcal{A}_\psi^{rec}$ be the respective restrictions of $\mathcal{A}_\varphi$ to Q^{end} and Q^{rec}. Note the following facts:

- All the priorities in $\mathcal{A}_\psi^{end}$ are strictly smaller than those in $\mathcal{A}_\psi^{rec}$.
- In $\mathcal{A}_\psi^{rec}$, state X has the smallest priority which is odd.
- $\mathcal{A}_\psi^{end}$ is isomorphic to $\mathcal{A}_\psi[X : \emptyset]$.
- $\mathcal{A}_\psi^{rec}$ has the same structure as $\mathcal{A}_\psi$ except for state X which accepts either $L(\mathcal{A}_\psi^{end})$ or $L(\mathcal{A}_\psi^{rec})$.
- Every path in an accepting run of $\mathcal{A}_\varphi$ must not visit state X infinitely often, for otherwise the least priority seen on this path at all would be odd.

Thus, $L(\mathcal{A}_\varphi)$ is the least solution to the equation

$$L = L(\mathcal{A}_\psi[X : \emptyset]) \cup L(\mathcal{A}_\psi[X : L])$$

Using the hypothesis twice as well as the approximant characterisation of least fixpoints and Lemma 1 we get

$$L(\mathcal{A}_\varphi) = \bigcup_{k \geq 1} L(\mu^k X.\psi) = \bigcup_{k \geq 0} L(\mu^k X.\psi) = L(\mu X.\psi) = L(\varphi)$$

Case $\varphi = \nu X.\psi$. This is dual to the previous case. In order to adhere to Lemma 1, the starting state as well as the recursion state X conjunctively combine the transitions of q_0^{rec} and q_0^{end}. State X obtains an even priority. The whole X in the *end*-component is filled by a state $\mathtt{tt}$ which has priority 0 and loops back to itself with any alphabet symbol. □

The next theorem is featured as an observation in [15] already. However, we include its proof here in order to have a complete and effective translation from WAPAs to μTL formulas. The result is also very similar to the theorem in [13] stating that ABAs can be translated into WABAs. Consequently, its proof proceeds along the same lines.

Theorem 4. [15, 13] *For every WAPA $\mathcal{A}$ there is a WABA $\mathcal{A}'$ s.t. $L(\mathcal{A}') = L(\mathcal{A})$.*

Proof. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, \Omega)$ with $\Omega : Q \to \{0, \ldots, p\}$. Define $\mathcal{A}' = (Q \times \{0, \ldots, p\}, \Sigma, (q_0, \Omega(q_0)), \delta', F)$ where for all $q \in Q$, $a \in \Sigma$:

$$\delta'((q,k),a) \;:=\; \langle \delta(q,a) \rangle^{\min\{k, \Omega(q)\}}$$

with $\langle f \vee g \rangle^k := \langle f \rangle^k \vee \langle f \rangle^k$, $\langle f \wedge g \rangle^k := \langle f \rangle^k \wedge \langle f \rangle^k$, and $\langle q \rangle^k := (q,k)$. Finally, let $F := \{\ (q,k) \mid k \text{ is even}\ \}$.

First observe that $\mathcal{A}'$ is indeed a WABA. Let $C_i := \{(q,i) \mid q \in Q\}$. Then $C_0, \ldots, C_p$ is a partition on $Q \times \{0, \ldots, p\}$, transitions either stay inside a C_i or lead to a C_j with $j < i$. At last, for every C_i we either have $C_i \subseteq F$ or $C_i \cap F = \emptyset$.

Now suppose that $w \in L(\mathcal{A})$. A run r of $\mathcal{A}$ on w naturally induces a run r' of $\mathcal{A}'$ on w. Every node in this run carries an extra component which remembers the minimal priority seen on this path so far. Hence, if r is accepting, so is r' since every path will eventually be trapped in a component that remembers even priorities. The converse direction is proved in the same way. If every path of a run in $\mathcal{A}'$ visits infinitely many final states, then the corresponding run of $\mathcal{A}$ must have seen even priorities as the least on every path. □

Theorem 5. *For every WABA $\mathcal{A}$ there is a $\mu\mathrm{TL}^0$ formula $\varphi_{\mathcal{A}}$ s.t. $L(\varphi_{\mathcal{A}}) = L(\mathcal{A})$.*

Proof. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ with Q being disjointly partitioned into components $C_0, \ldots, C_p$. W.l.o.g. we can assume for any $i \in \{0, \ldots, p\}$ that $C_i \subseteq F$ if i is even, and $C_i \cap F = \emptyset$ if i is odd. Since transitions can at most lead into components with smaller indices we can also assume $q_0 \in C_p$.

Take such a component C_i and a state $q \in C_i$. For each component we use the same method as proposed in [11] in order to come up with a μTL formula. First, C_i is unfolded into a tree-like structure with root q that admits loops but no merging of paths. A state at the beginning of a loop is called a (q,i)-loop state. With C_i^q we denote this unfolding of C_i.

We will translate every unfolded C_i^q into a formula φ_i^q using auxiliary formulas that describe the local behaviour of state q in component i with root q':

$$\psi^q_{q',i} \;:=\; \begin{cases} \sigma_i X_{q',i}.\ \bigwedge\limits_{a \in \Sigma} a \to \bigcirc \| \delta(q',a) \|_i^q & \text{if } q' \text{ is a } (q,i) - \text{loop state} \\ \bigwedge\limits_{a \in \Sigma} a \to \bigcirc \| \delta(q',a) \|_i^q & \text{o.w.} \end{cases}$$

where $\sigma_i = \mu$ if i is odd, $\sigma_i = \nu$ if it is even, and

$$\begin{aligned} \|f \vee g\|_i^q &:= \|f\|_i^q \vee \|g\|_i^q \\ \|f \wedge g\|_i^q &:= \|f\|_i^q \wedge \|g\|_i^q \\ \|q'\|_i^q &:= \begin{cases} X_{q',i} & \text{if } q' \text{ is a } (q,i)-\text{loop state} \\ \psi_{q',i} & \text{o.w.} \end{cases} \end{aligned}$$

Finally, let $\varphi_i^q := \psi_{q,i}$.

Note that every connected component of $\mathcal{A}$ is translated into a formula without free variables that only uses closed formulas from components with lower indices. Furthermore, the formulas created from one component have a single fixpoint type only. Hence, the resulting formula is alternation-free.

The correctness of this construction can be shown using tableaux or games for μTL as it is done in [11]. □

Corollary 1. *Every $\varphi \in \mu\mathrm{TL}$ is equivalent to a $\varphi' \in \mu\mathrm{TL}^0$.*

Proof. By composition of Theorems 3, 4 and 5. □

4 Optimising the Translations

Proposition 1. *For every $\varphi \in \mu\mathrm{TL}$ there is a WAPA $\mathcal{A}_\varphi$ with $L(\mathcal{A}_\varphi) = L(\varphi)$ s.t. $|\mathcal{A}| = O(|\varphi| \cdot 2^{fpd(\varphi)})$.*

Proof. Immediate from the proof of Theorem 3. All inductive constructions are linear, except those for fixpoint quantifiers. They double the size of the automata. □

Proposition 2. *For every WAPA $\mathcal{A}$ there is a $\varphi_\mathcal{A} \in \mu\mathrm{TL}^0$ with $L(\varphi_\mathcal{A}) = L(\mathcal{A})$ s.t. $|\varphi| = O(|\mathcal{A}|^4)$.*

Proof. Using Theorem 4, $\mathcal{A}$ can be transformed into a WABA of size $O(|\mathcal{A}|^2)$. Note that it is fair to assume that there are not more priorities than there are states. Furthermore, Theorem 5 constructs for every state, every component and every state in that component – i.e. every pair of states – a μTL formula of linear size. Composing these two constructions yields a formula of size $O(|\mathcal{A}|^4)$. □

In the following we discuss how to improve the translation from μTL formulas into WAPAs. The main focus is on optimising the costly translation of fixpoint quantifiers. The first attempt reduces the size of the automaton by not duplicating automata for nested but closed fixpoint formulas.

Take a formula of the form $\varphi = \sigma_1 X_1.\psi_1(\sigma_2 X_2.\psi_2)$ s.t. $X_1 \notin \mathit{free}(\psi_2)$. Thus, X_2 does not depend on X_1 and $nd(\varphi) < \mathit{fpd}(\varphi)$. A clever algorithm for calculating the semantics of φ would calculate the semantics of $\sigma_2 X_2.\psi_2$ only once and reuse it in every iteration needed to calculate the semantics of φ. Note that the automaton constructed in the proof of Theorem 3 would include the automaton $\mathcal{A}_2$ for $\sigma_2 X_2.\psi_2$ twice. Since there is no path out of $\mathcal{A}_2$ it suffices to include a

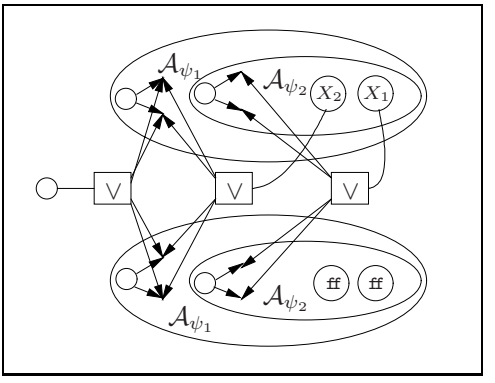

Fig. 2. Simultaneously translating fixpoint formulas into WAPAs.

single copy of $\mathcal{A}$. However, in the worst case $\mathcal{A}_2$ is a lot smaller than the automaton for ψ_1 and no asymptotic improvement compared to $O(|\varphi| \cdot 2^{fpd(\varphi)})$ is achieved.

The second attempt is based on Békič's Theorem. Let $\mu x.f$ denote the least fixpoint of an arbitrary function f that takes an argument x.

Theorem 6. [2] *Let $A \times B$ be a complete lattice and $F : A \times B \to A \times B$ a monotone function defined by $F(x, y) = (f_1(x, y), f_2(x, y))$. Then $\mu(x, y).F = (\mu x.f_1(x, \mu y.f_2(x, y)), \mu y.f_2(\mu x.f_1(x, \mu y.f_2(x, y)), y)$.*

The same holds for greatest fixpoints. Regarding formulas of μTL, Theorem 6 says that fixpoints of the same type can be computed simultaneously. We will show how to build automata that do so for two fixpoint formulas. It can easily be extended to formulas of arbitrary nesting depth as long as all variables are of the same type.

Take a formula $\varphi = \mu X_1.\psi_1(X_1, \mu X_2.\psi_2(X_1, X_2))$. Instead of building an automaton for $\mu X_2.\psi_2(X_1, X_2)$ and then one for φ we will construct an automaton for φ directly. By hypothesis we can assume that we already have automata $\mathcal{A}_{\psi_2}$ with two holes X_1 and X_2, as well as an automaton $\mathcal{A}_{\psi_1}$ with a hole for X_1 and another hole Z.

First let $\mathcal{A}' := \mathcal{A}_{\psi_1}[Z : \mathcal{A}_{\psi_2}]$. We can collapse holes and assume that $\mathcal{A}'$ only contains two holes X_1 and X_2. Then a WAPA $\mathcal{A}_\varphi$ for $L(\varphi)$ can be built by duplicating $\mathcal{A}'$ in the same way as it is done in the proof of Theorem 3. The two states X_1 and X_2 get odd priorities that lie between those in the *rec*-part and those in the *end*-part. Additionally, X_1 has transitions back to either of the beginnings of $\mathcal{A}_{\psi_1}$, X_2 has transitions back to either of the beginnings of $\mathcal{A}_{\psi_2}$. An illustration of this construction is given in Figure 2.

Proposition 3. *Every μTL formula φ can be translated into an equivalent WAPA of size $O(|\varphi| \cdot 2^{ad(\varphi)})$.*

5 Conclusion

We reinforced the importance of weak alternating automata in the algorithmics of ω-regular languages by giving direct translations from formulas of the linear

time μ-calculus into these automata and back. Definition 2 has shown that every ω-regular expression can easily be translated into μTL.

Remember that – just as the modal μ-calculus can be used as a specification language for the verification of branching time properties – μTL is a straightforward temporal logic for the verification of linear time properties. In fact, the work presented here is part of the development of a verification tool under the term "bounded model checking for all ω-regular properties". Bounded model checking [3] only considers paths of finite length through a transition system, and uses SAT-solvers for finding counterexamples to unsatisfied properties. It is incomplete in the sense that it cannot show the absence of errors. However, it is very successful as a symbolic verification method because of two reasons: (1) often, errors occur "early", i.e. small boundedness parameters suffice for finding them. (2) In recent years, much effort has been put into the development of SAT-solvers that behave efficiently despite SAT's NP-hardness.

So far, bounded model checking is – to the best of our knowledge – only done for LTL, hence, is only suitable for the verification of star-free, resp. first-order definable properties. The work presented here yields a computational model for all regular, i.e. monadic second-order definable properties that is easy to handle algorithmically. It will be used in a bounded model checker that verifies regular properties. Weak alternating parity automata will serve as the link between a denotational specification language like μTL or ω-regular expressions on one hand, and the actual model checker that generates formulas of propositional logic on the other hand. These formulas then only need to describe the run of a WAPA on a finite word or a word of the form wv^ω. Weakness, i.e. checking for occurrence of priorities rather than infinite recurrence simplifies this process vastly.

We believe that the direct translation from μTL, resp. ω-regular properties to weak alternating automata can prove to be useful for other verification purposes as well.

References

1. H. Barringer, R. Kuiper, and A. Pnueli. A really abstract concurrent model and its temporal logic. In *Proc. 13th Annual ACM Symp. on Principles of Programming Languages*, pages 173–183. ACM, 1986.
2. H. Békič. *Programming Languages and Their Definition, Selected Papers*, volume 177 of *LNCS*. Springer, 1984.
3. A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu. Symbolic model checking without BDDs. In R. Cleaveland, editor, *Proc. 5th Int. Conf. on Tools and Algorithms for the Analysis and Construction of Systems, TACAS'99*, volume 1579 of *LNCS*, Amsterdam, NL, March 1999.
4. J. C. Bradfield. The modal μ-calculus alternation hierarchy is strict. In U. Montanari and V. Sassone, editors, *Proc. 7th Conf. on Concurrency Theory, CONCUR'96*, volume 1119 of *LNCS*, pages 233–246, Pisa, Italy, August 1996. Springer.
5. J. R. Büchi. On a decision method in restricted second order arithmetic. In *Proc. Congress on Logic, Method, and Philosophy of Science*, pages 1–12, Stanford, CA, USA, 1962. Stanford University Press.

6. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, January 1981.
7. S. Dziembowski, M. Jurdziński, and I. Walukiewicz. How much memory is needed to win infinite games? In *Proc. 12th Symp. on Logic in Computer Science, LICS'97*, pages 99–110, Warsaw, Poland, June 1997. IEEE.
8. E. A. Emerson. Model checking and the μ-calculus. In N. Immerman and P. G. Kolaitis, editors, *Descriptive Complexity and Finite Models*, volume 31 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*, chapter 6. AMS, 1997.
9. E. A. Emerson and C. S. Jutla. Tree automata, μ-calculus and determinacy. In *Proc. 32nd Symp. on Foundations of Computer Science*, pages 368–377, San Juan, Puerto Rico, October 1991. IEEE.
10. D. Janin and I. Walukiewicz. On the expressive completeness of the propositional μ-calculus with respect to monadic second order logic. In U. Montanari and V. Sassone, editors, *Proc. 7th Conf. on Concurrency Theory, CONCUR'96*, volume 1119 of *LNCS*, pages 263–277, Pisa, Italy, August 1996. Springer.
11. R. Kaivola. *Using Automata to Characterise Fixed Point Temporal Logics*. PhD thesis, LFCS, Division of Informatics, The University of Edinburgh, 1997. Tech. Rep. ECS-LFCS-97-356.
12. D. Kozen. Results on the propositional μ-calculus. *TCS*, 27:333–354, December 1983.
13. O. Kupferman and M. Y. Vardi. Weak alternating automata are not that weak. *ACM Transactions on Computational Logic*, 2(3):408–429, 2001.
14. O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, March 2000.
15. C. Löding and W. Thomas. Alternating automata and logics over infinite words. In *Proc. IFIP Int. Conf. on Theoretical Computer Science, IFIP TCS2000*, volume 1872 of *LNCS*, pages 521–535. Springer, August 2000.
16. D. Muller and P. Schupp. Alternating automata on infinite objects: determinacy and rabin's theorem. In M. Nivat and D. Perrin, editors, *Proc. Ecole de Printemps d'Informatique Théoretique on Automata on Infinite Words*, volume 192 of *LNCS*, pages 100–107, Le Mont Dore, France, May 1984. Springer.
17. D. E. Muller, A. Saoudi, and P. E. Schupp. Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In *Proc. 3rd Symp. on Logic in Computer Science, LICS'88*, pages 422–427, Edinburgh, Scotland, July 1988. IEEE.
18. M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. of Amer. Math. Soc.*, 141:1–35, 1969.
19. C. Stirling. Local model checking games. In I. Lee and S. A. Smolka, editors, *Proc. 6th Conf. on Concurrency Theory, CONCUR'95*, volume 962 of *LNCS*, pages 1–11, Berlin, Germany, August 1995. Springer.
20. M. Y. Vardi. A temporal fixpoint calculus. In ACM, editor, *Proc. Conf. on Principles of Programming Languages, POPL'88*, pages 250–259, NY, USA, 1988. ACM Press.
21. M. Y. Vardi. *An Automata-Theoretic Approach to Linear Temporal Logic*, volume 1043 of *LNCS*, pages 238–266. Springer, New York, NY, USA, 1996.
22. I. Walukiewicz. Completeness of Kozen's axiomatization of the propositional μ-calculus. In *Proc. 10th Symp. on Logic in Computer Science, LICS'95*, pages 14–24, Los Alamitos, CA, 1995. IEEE.