# Satisfiability and Finite Model Property for the Alternating-Time $\mu$-Calculus*

Sven Schewe and Bernd Finkbeiner

Universität des Saarlandes, 66123 Saarbrücken, Germany
{schewe|finkbeiner}@cs.uni-sb.de

**Abstract.** This paper presents a decision procedure for the alternating-time $\mu$-calculus. The algorithm is based on a representation of alternating-time formulas as automata over concurrent game structures. We show that language emptiness of these automata can be checked in exponential time. The complexity of our construction meets the known lower bounds for deciding the satisfiability of the classic $\mu$-calculus. It follows that the satisfiability problem is EXPTIME-complete for the alternating-time $\mu$-calculus.

## 1 Introduction

In the design of distributed protocols, we are often interested in the strategic abilities of certain agents. For example, in a contract-signing protocol, it is important to ensure that while Alice and Bob can cooperate to sign a contract, Bob never has a strategy to obtain Alice's signature unless, at the same time, Alice has a strategy to obtain Bob's signature as well (cf. [10]). Such properties can be expressed in the alternating-time $\mu$-calculus (AMC) [1], which extends the classic $\mu$-calculus with modalities that quantify over the strategic choices of a group of agents. The models of AMC are a special type of labeled transition systems, called *concurrent game structures*, where each transition results from a set of decisions, one for each agent.

In this paper, we present the first decision procedure for the satisfiability of AMC formulas. The *satisfiability* problem asks for a given AMC formula $\varphi$ whether there exists a concurrent game structure that satisfies $\varphi$. Previous research has focused on the model checking problem [1, 2], which asks whether a *given* concurrent game structure satisfies its specification. By contrast, our procedure checks whether a specification can be implemented at all. For example, we can automatically prove the classic result that it is *impossible* to implement fair contract-signing without a trusted third party [6].

We introduce an automata-theoretic framework for alternating-time logics. *Automata over concurrent game structures* (ACGs) are a variant of alternating

* This work was partly supported by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS).

tree automata, where the atoms in the transition function do not refer to individual successors in the input structure, but instead quantify universally or existentially over all successors that result from the agents' decisions. Specifically, a *universal* atom $(\Box, A')$ refers to *all* successor states for *some* decision of the agents in a set $A'$, and an *existential* atom $(\Diamond, A')$ refers to *some* successor state for *each* decision of the agents *not* in $A'$. In this way, the automaton can run on game structures with arbitrary, even infinite, branching degree. Every AMC formula can be translated into an automaton that accepts exactly the models of the formula. Satisfiability of AMC formulas thus corresponds to language nonemptiness of ACGs.

The core result of the paper is the finite model property for ACGs. We first prove that, given any game structure accepted by an ACG $\mathcal{G}$, we can find a *bounded* game structure that is also accepted by $\mathcal{G}$. In the bounded game structure, the number of possible decisions of each agent is limited by some constant $m$, determined by the size of $\mathcal{G}$.

The emptiness problem of ACGs thus reduces to the emptiness problem of alternating tree automata and, since non-empty automata over finitely-branching structures always accept some finite structure [15, 14], there must exist a finite game structure in the language of $\mathcal{G}$. The drawback of this reduction is that the trees accepted by the alternating tree automaton branch over the decisions of *all* agents: the number of directions is therefore exponential in the number of agents. Since the emptiness problem of alternating tree automata is exponential in the number of directions, this results in a double-exponential decision procedure.

We show that it is possible to decide emptiness in single-exponential time. Instead of constructing an *alternating* automaton that accepts exactly the $m$-bounded game structures in the language of the ACG, we construct a *universal* automaton that only preserves emptiness. Unlike alternating automata, universal automata can be reduced to deterministic automata with just a single exponential increase in the number of states. For deterministic automata, the complexity of the emptiness problem is only linear in the number of directions.

Our approach is constructive and yields a tight complexity bound: the satisfiability problem for AMC is EXPTIME-complete. If the AMC formula is satisfiable, we can synthesize a finite model within the same complexity bound. Since AMC subsumes the alternating-time temporal logic ATL* [4, 1], we obtain a decision procedure for this logic as well.

**Related work.** The automata-theoretic approach to the satisfiability problem was initiated in the classic work by Büchi, McNaughton, and Rabin on monadic second-order logic [3, 13, 15]. For linear-time temporal logic, satisfiability can be decided by a translation to automata over infinite words [18]; for branching-time logics, such as CTL* and the modal $\mu$-calculus, by a translation to automata over infinite trees that branch according to inputs and nondeterministic choices [12, 5, 11, 20]. For alternating-time temporal logics, previous decidability results have been restricted to ATL [17, 19], a sublogic of ATL*.

Automata over concurrent game structures, introduced in this paper, provide an automata-theoretic framework for alternating-time logics. Automata

over concurrent game structures extend symmetric alternating automata [20], which have been proposed as the automata-theoretic framework for the classic $\mu$-calculus. Symmetric automata branch universally into all successors or existentially into some successor.

## 2 Preliminaries

### 2.1 Concurrent Game Structures

Concurrent game structures [1] generalize labeled transition systems to a setting with multiple agents. A *concurrent game structure* (CGS) is a tuple $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$, where

- $P$ is a finite set of atomic propositions,
- $A$ is a finite set of agents,
- $S$ is a set of states, with a designated initial state $s_0 \in S$,
- $l : S \to 2^P$ is a labeling function that decorates each state with a subset of the atomic propositions,
- $\Delta$ is a set of possible decisions for every agent, and
- $\tau : S \times \Delta^A \to S$ is a transition function that maps a state and the decisions of the agents to a new state.

A concurrent game structure is called *bounded* if the set $\Delta$ of decisions is finite, *$m$-bounded* if $\Delta = \mathbb{N}_m = \{1, \ldots, m\}$, and *finite* if $S$ and $\Delta$ are finite.

**Example.** As a running example, we introduce a simple CGS $\mathcal{C}_0$ with an infinite number of states and an infinite number of possible decisions. In every step, two agents each pick a real number and move to the state ${d_2}^2 - {d_1}^2$, where $d_1$ is the decision of agent $a_1$ and $d_2$ is the decision of agent $a_2$. We use two propositions, $p_1$ and $p_2$, where $p_1$ identifies the non-negative numbers and $p_2$ the rational numbers. Let $\mathcal{C}_0 = (P, A, S, s_0, l, \Delta, \tau)$, with $P = \{p_1, p_2\}$, $A = \{a_1, a_2\}$, $S = \mathbb{R}$, $s_0 = 0$, $p_1 \in l(s)$ iff $s \geq 0$, $p_2 \in l(s)$ iff $s \in \mathbb{Q}$, $\Delta = \mathbb{R}$, and $\tau : (s, (d_1, d_2)) \mapsto {d_2}^2 - {d_1}^2$. It is easy to see that in all states of this CGS, agent $a_1$ can enforce that $p_1$ eventually always holds true. Additionally, if agent $a_1$ decides before agent $a_2$, agent $a_2$ can always respond with a decision such that $p_2$ holds in the following state.

### 2.2 Alternating-Time $\mu$-calculus

The *alternating-time $\mu$-calculus* (AMC) extends the classical $\mu$-calculus with modal operators which express that an agent or a coalition of agents has a strategy to accomplish a goal. AMC formulas are interpreted over concurrent game structures.

**AMC Syntax.** AMC contains the modality $\Box_{A'}\varphi$, expressing that a set $A' \subseteq A$ of agents can enforce that a property $\varphi$ holds in the successor state, and the modality $\Diamond_{A'}\varphi$, expressing that it cannot be enforced against the agents $A'$ that $\varphi$ is violated in the successor state. Let $P$ and $B$ denote disjoint finite sets of atomic propositions and bound variables, respectively. Then

- *true* and *false* are AMC formulas.
- $p$, $\neg p$ and $x$ are AMC formulas for all $p \in P$ and $x \in B$.
- If $\varphi$ and $\psi$ are AMC formulas then $\varphi \wedge \psi$ and $\varphi \vee \psi$ are AMC formulas.
- If $\varphi$ is an AMC formula and $A' \subseteq A$ then $\Box_{A'}\varphi$ and $\Diamond_{A'}\varphi$ are AMC formulas.
- If $x \in B$ and $\varphi$ is an AMC formula where $x$ occurs only free, then $\mu x.\varphi$ and $\nu x.\varphi$ are AMC formulas.

The set of subformulas of a formula $\varphi$ is denoted by $sub(\varphi)$ and its alternation depth by $alt(\varphi)$ (for simplicity we use the syntactic alternation of least and greatest fixed-point operators).

**AMC Semantics.** An AMC formula $\varphi$ with atomic propositions $P$ is interpreted over a CGS $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$. $\|\varphi\|_{\mathcal{C}} \subseteq S$ denotes the set of states where $\varphi$ holds. A CGS $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$ is a *model* of a specification $\varphi$ with atomic propositions $P$ iff $s_0 \in \|\varphi\|_{\mathcal{C}}$.

- Atomic propositions are interpreted as follows: $\|\mathit{false}\|_{\mathcal{C}} = \emptyset$ and $\|\mathit{true}\|_{\mathcal{C}} = S$, $\|p\|_{\mathcal{C}} = \{s \in S \mid p \in l(s)\}$ and $\|\neg p\|_{\mathcal{C}} = \{s \in S \mid p \notin l(s)\}$.
- Conjunction and disjunction are interpreted as intersection and union, respectively: $\|\varphi \wedge \psi\|_{\mathcal{C}} = \|\varphi\|_{\mathcal{C}} \cap \|\psi\|_{\mathcal{C}}$ and $\|\varphi \vee \psi\|_{\mathcal{C}} = \|\varphi\|_{\mathcal{C}} \cup \|\psi\|_{\mathcal{C}}$.
- A state $s \in S$ is in $\|\Box_{A'}\varphi\|_{\mathcal{C}}$ iff the agents $A'$ can make a decision $\upsilon \in \Delta^{A'}$ such that, for all decisions $\upsilon' \in \Delta^{A \setminus A'}$, $\varphi$ holds in the successor state: $\|\Box_{A'}\varphi\|_{\mathcal{C}} = \{s \in S \mid \exists \upsilon \in \Delta^{A'}.\ \forall \upsilon' \in \Delta^{A \setminus A'}.\ \tau(s, (\upsilon, \upsilon')) \in \|\varphi\|_{\mathcal{C}}\}$.
- A state $s \in S$ is in $\|\Diamond_{A'}\varphi\|_{\mathcal{C}}$ iff for all decisions $\upsilon \in \Delta^{A \setminus A'}$ of the agents not in $A'$, the agents in $A'$ have a counter decision $\upsilon' \in \Delta^{A'}$ which ensures that $\varphi$ holds in the successor state: $\|\Diamond_{A'}\varphi\|_{\mathcal{C}} = \{s \in S \mid \forall \upsilon' \in \Delta^{A \setminus A'}.\ \exists \upsilon \in \Delta^{A'}.\ \tau(s, (\upsilon, \upsilon')) \in \|\varphi\|_{\mathcal{C}}\}$.
- The least and greatest fixed points are interpreted as follows: $\|\mu x.\varphi\|_{\mathcal{C}} = \bigcap\{S_x \subseteq S \mid \|\varphi\|_{\mathcal{C}_x^{S_x}} \subseteq S_x\}$, $\|\nu x.\varphi\|_{\mathcal{C}} = \bigcup\{S_x \subseteq S \mid \|\varphi\|_{\mathcal{C}_x^{S_x}} \supseteq S_x\}$, where $\mathcal{C}_x^{S_x} = (P \cup \{x\}, A, S, s_0, l_x^{S_x}, \Delta, \tau)$ denotes the modified CGS with the labeling function $l_x^{S_x} : S \to 2^{P \cup \{x\}}$ with $l_x^{S_x}(s) \cap P = l(s)$ and $x \in l_x^{S_x}(s) \Leftrightarrow s \in S_x \subseteq S$. Since the bound variable $x$ occurs only positive in $\varphi$, $\|\varphi\|_{\mathcal{C}_x^{S_x}}$ is monotone in $S_x$ and the fixed points are well-defined.

AMC contains the classic $\mu$-calculus with the modal operators $\Box$ and $\Diamond$, which abbreviate $\Box_\emptyset$ and $\Diamond_A$, respectively. AMC also subsumes the temporal logic ATL* [1], which is the alternating-time extension of the branching-time temporal logic CTL*. ATL* contains the path quantifier $\langle\langle A' \rangle\rangle$, which ranges over all paths the players in $A'$ can enforce. There is a canonical translation from ATL* to AMC [4].

**Example.** As discussed in Section 2.1, the example CGS $\mathcal{C}_0$ has the property that in all states, agent $a_1$ can enforce that $p_1$ eventually always holds true, and agent $a_2$ can respond to any decision of agent $a_1$ with a counter decision such that $p_2$ holds in the following state. This property is expressed by the AMC formula $\psi = \nu x.(\mu y.\nu z.\, \Box_{\{a_1\}}(p_1 \wedge z \vee y)) \wedge \Diamond_{\{a_2\}} p_2 \wedge \Diamond_{\emptyset} x$.

### 2.3 Automata over Finitely Branching Structures

An *alternating parity automaton* with a finite set $\Upsilon$ of directions is a tuple $\mathcal{A} = (\Sigma, Q, q_0, \delta, \alpha)$, where $\Sigma$ is a finite alphabet, $Q$ is a finite set of states, $q_0 \in Q$ is a designated initial state, $\delta$ is a transition function, and $\alpha : Q \to C \subset \mathbb{N}$ is a coloring function. The transition function $\delta : Q \times \Sigma \to \mathbb{B}^+(Q \times \Upsilon)$ maps a state and an input letter to a positive boolean combination of states and directions.

In the context of this paper, we consider alternating parity automata that run on bounded CGSs with a fixed set $P$ of atomic propositions ($\Sigma$=$2^P$), a fixed set $A$ of agents and a fixed finite set $\Delta$ of decisions ($\Upsilon$=$\Delta^A$). The acceptance mechanism is defined in terms of run trees. As usual, an $\Upsilon$-tree is a prefix-closed subset $Y \subseteq \Upsilon^*$ of the finite words over the set $\Upsilon$ of directions. For given sets $\Sigma$ and $\Upsilon$, a *$\Sigma$-labeled $\Upsilon$-tree* is a pair $\langle Y, l\rangle$, consisting of a tree $Y \subseteq \Upsilon^*$ and a labeling function $l : Y \to \Sigma$ that maps every node of $Y$ to a letter of $\Sigma$. If $\Upsilon$ and $\Sigma$ are not important or clear from the context, $\langle Y, l\rangle$ is called a tree.

A *run tree* $\langle R, r\rangle$ on a given CGS $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$ is a $Q \times S$-labeled tree whose root is decorated with $r(\varepsilon) = (q_0, s_0)$, and for each node $n \in R$ decorated with a label $r(n) = (q, s)$, there is a set $\mathfrak{A}_n \subseteq Q \times \Upsilon$ that satisfies $\delta(q, l(s))$, such that $(q', \upsilon)$ is in $\mathfrak{A}_n$ iff some child of $n$ is decorated with a label $(q', \tau(s, \upsilon))$.

A run tree is *accepting* iff all infinite paths fulfill the *parity condition*. An infinite path fulfills the parity condition iff the highest color of the states appearing infinitely often on the path is even. A CGS is *accepted* by the automaton iff it has an accepting run tree. The set of CGSs accepted by an automaton $\mathcal{A}$ is called its *language* $\mathcal{L}(\mathcal{A})$. An automaton is empty iff its language is empty.

The acceptance of a given CGS $\mathcal{C}$ can also be viewed as the outcome of a *game* played over $Q \times S$, starting in $(q_0, s_0)$. When the game reaches a position $(q, s)$, player *accept* first chooses a set $\mathfrak{A} \subseteq Q \times \Upsilon$ of atoms that satisfies $\delta(q, l(s))$. Player *reject* then chooses one atom $(q', \upsilon)$ from $\mathfrak{A}$ and the game continues in $(q', \tau(s, \upsilon))$. An infinite sequence $(q_0, s_0)(q_1, s_1)(q_2, s_2)\ldots$ of game positions is called a *play*. A play is *winning* for player *accept* iff it satisfies the parity condition. A *strategy* for player *accept* (*reject*) maps each history of decisions of both players to a decision of player *accept* (*reject*). A pair of strategies determines a play. A strategy for player *accept* is *winning* iff, for all strategies of player *reject*, the play determined by the strategies is winning for player *accept*. The CGS $\mathcal{C}$ is accepted iff player *accept* has a winning strategy.

An automaton is *universal* iff the image of $\delta$ consists only of conjunctions, *nondeterministic* iff the image of $\delta$ consists only of formulas that, when rewritten into disjunctive normal form, contain in each disjunct exactly one element of $Q \times \{\upsilon\}$ for each $\upsilon \in \Upsilon$, and *deterministic* iff it is universal and nondeterministic.

For nondeterministic automata, emptiness can be checked with an *emptiness game* over $Q$ where, instead of considering the letter $l(s)$ on some state $s$ of a given CGS, the letter is chosen by player *accept*. The nondeterministic automaton is non-empty iff player *accept* has a winning strategy in the emptiness game.

## 3 Automata over Concurrent Game Structures

In this section, we introduce *automata over concurrent game structures* (ACGs) as an automata-theoretic framework for the alternating-time $\mu$-calculus. The automata over finitely branching structures described in Section 2.3 do not suffice for this purpose, because they are limited to bounded CGSs. Generalizing symmetric automata [20], ACGs contain *universal atoms* $(\Box, A')$, which refer to *all* successor states for *some* decision of the agents in $A'$, and *existential atoms* $(\Diamond, A')$, which refer to *some* successor state for *each* decision of the agents *not* in $A'$. In this way, ACGs can run on CGSs with an arbitrary, even infinite, number of decisions.

An ACG is a tuple $\mathcal{G} = (\Sigma, Q, q_0, \delta, \alpha)$, where $\Sigma$, $Q$, $q_0$, and $\alpha$ are defined as for alternating parity automata in the previous section. The transition function $\delta : Q \times \Sigma \to \mathbb{B}^+(Q \times ((\{\Box, \Diamond\} \times 2^A) \cup \{\varepsilon\}))$ now maps a state and an input letter to a positive boolean combination of three types of atoms: $(\Box, A')$ is a universal atom, $(\Diamond, A')$ is an existential atom, and $\varepsilon$ is an $\varepsilon$-transition, where only the state of the automaton is changed and the state of the CGS remains unchanged. If an ACG has no $\varepsilon$-transitions, it is called *$\varepsilon$-free*.

A run tree $\langle R, r\rangle$ on a given CGS $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$ is a $Q \times S$-labeled tree where the root is labeled with $(q_0, s_0)$ and where, for a node $n$ with a label $(q, s)$ and a set $L = \{r(n \cdot \rho) \mid n \cdot \rho \in R\}$ of labels of its successors, the following property holds: there is a set $\mathfrak{A} \subseteq Q \times (\{\Box, \Diamond\} \times 2^A \cup \{\varepsilon\})$ of atoms satisfying $\delta(q, l(s))$ such that

- for all universal atoms $(q', \Box, A')$ in $\mathfrak{A}$, there exists a decision $\upsilon \in \Delta^{A'}$ of the agents in $A'$ such that, for all counter decisions $\upsilon' \in \Delta^{A \setminus A'}$, $(q', \tau(s, (\upsilon, \upsilon'))) \in L$,
- for all existential atoms $(q', \Diamond, A')$ in $\mathfrak{A}$ and all decisions $\upsilon' \in \Delta^{A \setminus A'}$ of the agents not in $A'$, there exists a counter decision $\upsilon \in \Delta^{A'}$ such that $(q', \tau(s, (\upsilon, \upsilon'))) \in L$, and
- for all $\varepsilon$-transitions $(q', \varepsilon)$ in $\mathfrak{A}$, $(q', s) \in L$.

As before, a run tree is accepting iff all paths satisfy the parity condition, and a CGS is accepted iff there exists an accepting run tree.

The acceptance of a CGS can again equivalently be defined as the outcome of a game over $Q \times S$, starting in $(q_0, s_0)$. Each round of the game now consists of two stages. In the first stage, player *accept* chooses a set $\mathfrak{A}$ of atoms satisfying $\delta(q, l(s))$, and player *reject* picks one atom from $\mathfrak{A}$. If the result of the first stage is an $\varepsilon$-transition $(q', \varepsilon)$, then the round is finished and the game continues in $(q', s)$ with the new state of the automaton. If the result of the first stage is

a universal atom $(q', (\Box, A'))$, the second stage begins by player *accept* making the decisions $\upsilon \in \Delta^{A'}$ for the agents in $A'$, followed by player *reject* making the decisions $\upsilon' \in \Delta^{A \setminus A'}$ for the remaining agents. Finally, if the result of the first stage is an existential atom $(q, (\Diamond, A'))$, the order of the two choices is reversed: first, player *reject* makes the decisions $\upsilon' \in \Delta^{A \setminus A'}$ for the agents in $A \setminus A'$; then, player *accept* makes the decisions $\upsilon \in \Delta^{A'}$ for the players in $A'$. After the decisions are made, the game continues in $(q', \tau(s, (\upsilon, \upsilon')))$.

A winning strategy for player *accept* uniquely defines an accepting run tree, and the existence of an accepting run tree implies the existence of a winning strategy. The game-theoretic characterization of acceptance is often more convenient than the characterization through run trees, because parity games are memoryless determined [5]. A CGS is therefore accepted by an ACG iff player *accept* has a memoryless winning strategy in the acceptance game, i.e., iff she has a strategy where her choices only depend on the state of the game and the previous decisions in the current round.

As an additional complexity measure for an ACG $\mathcal{G}$, we use the set $atom(\mathcal{G}) \subseteq Q \times \{\Box, \Diamond, \varepsilon\} \times 2^A$ of atoms that actually occur in some boolean function $\delta(q, \sigma)$. The elements of $atom(\mathcal{G})$ are called the *atoms of* $\mathcal{G}$.

**Example.** The CGSs that satisfy the AMC formula $\psi = \nu x.(\mu y.\nu z.\, \Box_{\{a_1\}}(p_1 \wedge z \vee y)) \wedge \Diamond_{\{a_2\}} p_2 \wedge \Diamond_{\emptyset} x$ from Section 2.2 are recognized by the ACG $\mathcal{G}_\psi = (\Sigma, Q, q_0, \delta, \alpha)$, where $\Sigma = 2^{\{p_1, p_2\}}$ and $Q = \{q_0, q_\mu, q_\nu, q_{p_2}\}$. The transition function $\delta$ maps

- $(q_{p_2}, \sigma)$ to *true* if $p_2 \in \sigma$, and to *false* otherwise,
- $(q_\mu, \sigma)$ and $(q_\nu, \sigma)$ to $(q_\nu, \Box, \{a_1\})$ if $p_1 \in \sigma$, and to $(q_\mu, \Box, \{a_1\})$ otherwise, and
- $(q_0, \sigma)$ to $\delta(q_\mu, \sigma) \wedge (q_{p_2}, \Diamond, \{a_2\}) \wedge (q_0, \Diamond, \emptyset)$.

The coloring function $\alpha$ maps $q_\mu$ to 1 and the remaining states to 0.

Consider again the example CGS $\mathcal{C}_0$ from Section 2.1, which satisfies $\psi$. In the acceptance game of $\mathcal{G}_\psi$ for $\mathcal{C}_0$, player *accept* has no choice during the first stage of each move, and can win the game by making the following decisions during the second stage:

- If one of the atoms $(q_\mu, \Box, \{a_1\})$ or $(q_\nu, \Box, \{a_1\})$ is the outcome of the first stage, agent $a_1$ makes the decision 0.
- If the atom $(q_{p_2}, \Diamond, \{a_2\})$ is the outcome of the first stage and agent $a_1$ has made the decision $d_1$, agent $a_2$ chooses $d_2 = d_1$.
- For all other atoms $(q, \circ, A')$, the decision for all agents in $A'$ is 0.

### 3.1 From AMC Formulas to Automata over Concurrent Game Structures

The following theorem provides a translation of AMC formulas to equivalent ACGs. It generalizes the construction for the modal $\mu$-calculus suggested in [20] and can be proved analogously.

**Theorem 1.** *Given an AMC formula $\varphi$, we can construct an ACG $\mathcal{G}^{\varepsilon}_{\varphi} = (2^V, sub(\varphi), \varphi, \delta, \alpha)$ with $|sub(\varphi)|$ states and atoms and $O(|alt(\varphi)|)$ colors that accepts exactly the models of $\varphi$.*

**Construction:** W.l.o.g., we assume that the bound variables have been consistently renamed to ensure that for each pair of different subformulas $\lambda x.\psi$ and $\lambda' x'.\psi'$ $(\lambda, \lambda' \in \{\mu, \nu\})$ of $\varphi$, the bound variables are different $(x \neq x')$.

- The transition function $\delta$ is defined, for all free variables $p$ and all bound variables $x$, by
  - $\delta(p, \sigma) = \mathit{true}$, $\delta(\neg p, \sigma) = \mathit{false} \quad \forall p \in \sigma$;
  - $\delta(\neg p, \sigma) = \mathit{true}$, $\delta(p, \sigma) = \mathit{false} \quad \forall p \in P \setminus \sigma$;
  - $\delta(\varphi \wedge \psi, \sigma) = (\varphi, \varepsilon) \wedge (\psi, \varepsilon)$ and $\delta(\varphi \vee \psi, \sigma) = (\varphi, \varepsilon) \vee (\psi, \varepsilon)$;
  - $\delta(\Box_{A'} \varphi, \sigma) = (\varphi, (\Box, A'))$ and $\delta(\Diamond_{A'} \varphi, \sigma) = (\varphi, (\Diamond, A'))$;
  - $\delta(x, \sigma) = (\lambda x.\varphi, \varepsilon)$ and $\delta(\lambda x.\varphi, \sigma) = (\varphi, \varepsilon) \quad \lambda \in \{\mu, \nu\}$.
- The coloring function $\alpha$ maps every subformula that is not a fixed point formula to 0. The colors of the fixed point formulas are defined inductively:
  - Every least fixed point formula $\mu p.\psi$ is colored by the smallest odd color that is greater or equal to the highest color of each subformula of $\psi$.
  - Every greatest fixed point formula $\nu p.\psi$ is colored by the smallest even color that is greater or equal to the highest color of each subformula of $\psi$. □

### 3.2 Eliminating $\varepsilon$-Transitions

Given an ACG $\mathcal{G}^{\varepsilon} = (\Sigma, Q, q_0, \delta, \alpha)$ with $\varepsilon$-transitions, we can find an $\varepsilon$-free ACG that accepts the same language. The idea of our construction is to consider the sequences of transitions from some position of the acceptance game that *exclusively* consist of $\varepsilon$-transitions: if the sequence is infinite, we can declare the winner of the game without considering the rest of the game; if the sequence is finite, we skip forward to the next non-$\varepsilon$-atom.

The construction is related to the elimination of $\varepsilon$-transitions in ordinary alternating automata [20] and will be included in the full version.

**Lemma 1.** *Given an ACG $\mathcal{G}^{\varepsilon}$ with $n$ states, $c$ colors and $a$ atoms, we can construct an equivalent $\varepsilon$-free ACG with at most $c \cdot n$ states, $c$ colors and $c \cdot a$ atoms.* □

## 4 Bounded Models

We now show that for every ACG $\mathcal{G}$ there exists a bound $m$ such that $\mathcal{G}$ is empty if and only if $\mathcal{G}$ does not accept any $m$-bounded CGSs. Consider an $\varepsilon$-free ACG $\mathcal{G}$ and a CGS $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$ accepted by $\mathcal{G}$. In the following, we define a finite set $\Gamma$ of decisions and a transition function $\tau' : S \times \Gamma^A \to S$, such that the resulting bounded CGS $\mathcal{C}' = (P, A, S, s_0, l, \Gamma, \tau')$ is also accepted by $\mathcal{G}$. Before

we formally define the construction in the proof of Theorem 2 below, we first give an informal outline.

Let us begin with the special case where all atoms of $\mathcal{G}$ are of the form $(q, \Box, \{a\})$, i.e., a *universal* atom with a *single* agent. We use the set of atoms as the new set of decisions of each agent. The new transition function is obtained by first mapping the decision of each agent in $\mathcal{C}'$ to a decision in $\mathcal{C}$, and then applying the old transition function.

To map the decisions, we fix a memoryless winning strategy for player *accept* in the acceptance game for $\mathcal{C}$. After an atom $(q, \Box, \{a\})$ has been chosen in the first stage of the acceptance game, player *accept* begins the second stage by selecting a decision $d_a$ for agent $a$. We map each decision $(q, \Box, \{a\})$ in $\mathcal{C}'$ to this decision $d_a$ in $\mathcal{C}$.

Player *accept* wins the acceptance game for $\mathcal{C}'$ with the following strategy: In the first stage of each move, we apply the winning strategy of player *accept* in the acceptance game for $\mathcal{C}$. In the second stage, we simply select the atom $(q', \Box, \{a'\})$ that was chosen in the first stage as the decision for agent $a'$. Since the strategy for $\mathcal{C}$ wins for all possible decisions of the agents in $A \setminus \{a'\}$, it wins in particular for the decisions selected in the transition function.

Suppose next that we still have only *universal* atoms $(q, \Box, A')$, but that the set $A'$ of agents is not required to be singleton. There is no guarantee that the decisions of the agents in $A'$ are consistent: an agent $a$ may choose an atom $(q, \Box, A')$ where $A'$ does not contain $a$ or contains some other agent $a'$ who made a different decision. For the purpose of computing the transition function, we therefore *harmonize* the decisions by replacing, in such cases, the decision of agent $a$ with a fixed decision $(q_0, \Box, \{a\})$.

To win the acceptance game for $\mathcal{C}'$, player *accept* selects, after an atom $(q, \Box, A')$ has been chosen in the first stage, this atom $(q, \Box, A')$ for all agents in $A'$. The selection is therefore consistent for all agents in $A'$. Since the strategy wins for all decisions of the agents in $A \setminus A'$, it does not matter if some of their decisions have been replaced. Note that, this way, only decisions of player *reject* are changed in the harmonization.

Finally, suppose that $\mathcal{G}$ contains *existential* atoms. If an existential atom $(q, \Diamond, A')$ is the outcome of the first stage of the acceptance game, player *accept* only decides *after* the decisions of the agents in $A \setminus A'$ have been made by player *reject*. To implement this order of the choices in the computation of the transition function, we allow the player who chooses the last existential atom to override all decisions for existential atoms of his opponent. We add the natural numbers $\leq |A|$ as an additional component to the decisions of the agents. For a given combined decision of the agents, the sum over the numbers in the decisions of the agents, modulo $|A|$, then identifies one *favored* agent $a_0 \in A$. In this way, whichever player chooses *last* can determine the favored agent. Given the decision of agent $a_0$ for some atom $(q'', \Diamond, A'')$ or $(q'', \Box, A'')$, we replace each decision for an existential atom by an agent in $A \setminus A''$ by the fixed decision $(q_0, \Box, \{a\})$.

To win the acceptance game for $\mathcal{C}'$, the strategy for player *accept* makes the following choice after an atom $(q', \Diamond, A')$ has been chosen in the first stage and

player *reject* has made the decisions for all agents in $A \setminus A'$: for all agents in $A'$, she selects the atom $(q', \Diamond, A')$, combined with some number that ensures that the favored agent $a_0$ is in $A'$.

**Example.** Consider again the CGS $\mathcal{C}_0$, which is accepted by the ACG $\mathcal{G}_\psi$ with the winning strategy for player *accept* described in Section 3.1. The new transition function consists of two steps:

In the first step, we harmonize the given combined decision of the agents by replacing the inconsistent decisions. In the acceptance game, this may change the decisions of the agents controlled by player *reject*. If, for example, the atom $(q_{p_2}, \Diamond, \{a_2\})$ is the outcome of the first stage of the acceptance game and player *reject* makes the decision $(q_0, \Diamond, \emptyset, 1)$ for agent $a_1$, player *accept* responds by making the decision $(q_{p_2}, \Diamond, \{a_2\}, 1)$ for agent $a_2$. The sum of the natural numbers $(1+1)$ identifies agent $a_2$, and all existential choices for groups of agents *not* containing $a_2$ are overridden. The resulting choices are $(q_0, \Box, \{a_1\})$ for agent $a_1$ and $(q_{p_2}, \Diamond, \{a_2\})$ for agent $a_2$.

In the second step, the decisions of the agents are mapped to decisions in the CGS $\mathcal{C}_0$. First, the universal choices are evaluated: The winning strategy maps $(q_0, \Box, \{a_1\})$ to the decision $d_1 = 0$ for agent $a_1$. Then, the existential choice is evaluated: The winning strategy maps $(q_{p_2}, \Diamond, \{a_2\})$ and the decision $d_1 = 0$ for agent $a_1$ to the decision $d_2 = d_1 = 0$ for agent $a_2$.

The resulting bounded CGS is very simple: the new transition function maps all decisions to state 0.

**Theorem 2.** *An $\varepsilon$-free ACG $\mathcal{G} = (\Sigma, Q, q_0, \delta, \alpha)$ is non-empty iff it accepts a $(|atom(\mathcal{G})| \cdot |A|)$-bounded CGS.*

*Proof.* If $\mathcal{C} = (P, A, S, s_0, l, \Delta, \tau)$ is accepted by the $\varepsilon$-free ACG $\mathcal{G} = (2^P, Q, q_0, \delta, \alpha)$, then player *accept* has a memoryless winning strategy in the acceptance game for $\mathcal{C}$. We fix such a memoryless strategy and use it to construct the bounded CGS $\mathcal{C}' = (P, A, S, s_0, l, \Gamma, \tau')$.

*Decisions* For convenience, we assume that the set $A$ of agents is an initial sequence of the natural numbers. The new set of decisions $\Gamma = atom(\mathcal{G}) \times A$ consists of pairs of atoms and numbers. If the first component is an existential atom $(q, \Diamond, A')$, then the sum of the second components of the decisions of all agents is used to validate the choice.

We say that two decisions $d_1, d_2 \in \Gamma$ are *equivalent* if they agree on their first component: $(\mathbf{a}_1, a_1') \sim (\mathbf{a}_2, a_2') :\Leftrightarrow \mathbf{a}_1 = \mathbf{a}_2$.

We say that a combined decision $\upsilon \in \Gamma^A$ *favors* an agent $a \in A$, $\upsilon \rightarrowtail a$, if the sum of the second arguments, modulo $|A|$, of this combined decision is equal to $a$.

We say that the decision $d_a \in \Gamma$ of agent $a$ *prevails* in the combined decision $\upsilon \in \Gamma^A$ if the following conditions hold for $d_a = ((q, \circ, A'), a'')$, $\circ \in \{\Box, \Diamond\}$:

- $a \in A'$,

- all agents $a' \in A'$ have made a decision $d_{a'} \sim d_a$ equivalent to the decision of $a$, and
- if $\circ = \Diamond$, then $a$ cooperates with the agent favored by the combined decision $\upsilon$ ($\upsilon \rightarrowtail a' \in A'$).

*Harmonization* Let $\mathbf{A} = Q \times \{\Box, \Diamond\} \times 2^A$. The harmonization $h : \Gamma^A \to \mathbf{A}^A$ maps the decision of the agents to a harmonic decision. Harmonic decisions are elements of $\mathbf{A}^A$ such that

- each agent $a \in A$ chooses an atom $(q, \circ, A')$ with $q \in Q$, $\circ \in \{\Box, \Diamond\}$, and $a \in A' \subseteq A$,
- if an agent $a \in A$ chooses an atom $(q, \circ, A') \in \mathbf{A}$, then all agents $a' \in A'$ choose the same atom, and
- if an agent $a \in A$ chooses an existential atom $(q, \Diamond, A') \in \mathbf{A}$, then all agents $a' \notin A'$ choose universal atoms.

For prevailing decisions, the harmonization $h$ only deletes the second component. Non-prevailing decisions of an agent $a$ are replaced by the fixed decision $(q_0, \Box, \{a\})$ (which is not necessarily in $atom(\mathcal{G})$).

*Direction* We define the function $f_s : \mathbf{A}^A \to \Delta^A$ that maps a harmonic decision to a direction $\upsilon \in \Delta^A$ in $\mathcal{C}$. $f_s$ depends on the state $s \in S$ of $\mathcal{C}$ and is determined by the second stage of the fixed memoryless strategy.

First, the universal decisions are evaluated: if an agent makes the harmonic decision $(q, \Box, A')$, then $\upsilon' \in \Delta^{A'}$ is determined by the choice of player *accept* in the second stage of the winning strategy in state $s$, when confronted with the atom $(q, \Box, A')$.

Then, the existential decisions are evaluated: If an agent makes the harmonic decision $(q, \Diamond, A')$ then $\upsilon' \in \Delta^{A'}$ is determined by the choice of player *accept* in the second stage of the winning strategy in state $s$, when confronted with the atom $(q, \Diamond, A')$ and the decision $\upsilon'' \in \Delta^{A \setminus A'}$ fixed by the evaluation of the universal harmonic decisions.

The new transition function $\tau' : S \times \Gamma^A \to S$ is defined as $\tau' : (s, \upsilon) \mapsto \tau(s, f_s(h(\upsilon)))$.

*Acceptance* In the acceptance game for $\mathcal{C}'$, player *accept* has the following strategy: in the first stage of each round, she applies the winning strategy of the acceptance game for $\mathcal{C}$. The strategy for the second stage depends on the outcome of the first stage:

- If an atom $(q, \Box, A')$ is chosen in the first stage, player *accept* fixes the prevailing decision $((q, \Box, A'), 1)$ for all agents $a \in A'$.
- If an atom $(q, \Diamond, A')$ with $A' \neq \emptyset$ is chosen in the first stage and player *reject* has made the decisions $d_a$ for all agents $a \notin A'$, player *accept* fixes the prevailing decisions $((q, \Diamond, A'), n_a)$ for the agents $a \in A'$ such that an agent $a' \in A'$ is favored.

– If an atom $(q, \Diamond, \emptyset)$ is chosen in the first stage, then player *accept* does not participate in the second stage.

We now show that the run tree $\langle R', r' \rangle$ defined by this strategy is accepting. Let $\langle R, r \rangle$ be the run tree defined by the winning strategy in the acceptance game for $\mathcal{C}$. In the following, we argue that for each branch labeled $(q_0, s_0)(q_1, s_1)(q_2, s_2) \ldots$ in $\langle R', r' \rangle$, there is an identically labeled branch in $\langle R, r \rangle$. Since all branches of $\langle R, r \rangle$ satisfy the parity condition, $\langle R', r' \rangle$ must be accepting as well.

The root of both run trees is labeled by $(q_0, s_0)$. If a node labeled $(q_i, s_i)$ in $\langle R', r' \rangle$ has a child labeled $(q_{i+1}, s_{i+1})$, then there must be an atom $(q_{i+1}, \circ, A') \in Q \times \{\Box, \Diamond\} \times 2^A$ in the set of atoms chosen by player *accept*, such that following holds: for the decision $\upsilon' \in \Gamma^{A'}$ defined by the strategy of player *accept*, there is a decision $\upsilon'' \in \Gamma^{A \setminus A'}$ such that $s_{i+1} = \tau'(s_i, (\upsilon', \upsilon'')) = \tau(s_i, f_{s_i}(h(\upsilon', \upsilon'')))$.

Now consider a node labeled $(q_i, s_i)$ in $\langle R, r \rangle$. Since the strategy of player *accept* in the first stage of each round is identical for the two acceptance games, the atom $(q_{i+1}, \circ, A')$ is also included in the set of atoms chosen by player *accept* in the acceptance game for $\mathcal{C}$. Player *reject* can enforce the decision $\upsilon = f_{s_i}(h(\upsilon', \upsilon''))$ as follows:

– If $\circ = \Box$, player *accept* chooses the $\Delta^{A'}$ part of $\upsilon$ under the fixed memoryless strategy for the acceptance game of $\mathcal{C}$, and player *reject* can respond by choosing the $\Delta^{A \setminus A'}$ part of $\upsilon$.
– If $\circ = \Diamond$, player *reject* can choose the $\Delta^{A \setminus A'}$ part of $\upsilon$, and player *accept* will react by choosing the $\Delta^{A'}$ part of $\upsilon$ under the fixed memoryless strategy for the acceptance game of $\mathcal{C}$, guaranteeing that $\upsilon = f_{s_i}(h(\upsilon', \upsilon''))$.

In both cases, the new state $s_{i+1} = \tau(s_i, \upsilon)$ is chosen. The node labeled $(q_i, s_i)$ in $\langle R, r \rangle$ must therefore have a child labeled $(q_{i+1}, s_{i+1})$. □

## 5 Satisfiability and Complexity

An AMC formula is satisfiable if and only if the language of its ACG is nonempty. A simple procedure for deciding emptiness of ACGs is immediately suggested by Theorem 2: since we can restrict our attention to $m$-bounded CGSs with fixed $m = |atom(\mathcal{G})| \cdot |A|$, we can replace $(q, (\Box, A'))$ and $(q, (\Diamond, A'))$ by the corresponding positive boolean combinations: the resulting automaton accepts exactly the $m$-bounded concurrent game structures in the language of $\mathcal{G}$. To decide emptiness, we nondeterminize the automaton [7, 14] and then solve the emptiness game. The complexity of this construction is double-exponential: solving the emptiness game of the nondeterministic automaton is exponential in the number of directions, which is already exponential in the number of agents ($m^{|A|}$).

We now describe an alternative algorithm with only single-exponential complexity. Instead of going through an alternating automaton to a nondeterministic automaton, we go through a universal automaton to a *deterministic* automaton. The advantage of solving the emptiness game for a deterministic automaton instead of for a nondeterministic automaton is that the set of atoms chosen by

player *accept* is uniquely determined by the input letter; this reduces the number of choices from exponential in the number of directions to linear in the size of the input alphabet.

The construction of the universal automaton is based on the observation that the winning strategy of player *accept* that we defined in the previous section can be represented by assigning a function $f_s : Q \to 2^{atom(\mathcal{G})}$ to each state $s$ of $\mathcal{C}$. The set $f_s(q)$ contains the atoms that player *accept* chooses in the first stage of the game at position $(q, s)$. Since the strategy for the second stage depends only on the chosen atom and not on the states of $\mathcal{C}$ and $\mathcal{G}$, $f_s$ determines the entire strategy of player *accept*.

The universal automaton runs on bounded CGSs that are annotated by this function $f_s$; i.e., we extend the alphabet from $\Sigma$ to $\Sigma \times (Q \to 2^{atom(\mathcal{G})})$ and a CGS is accepted iff $f_s$ identifies a winning strategy for player *accept* in the acceptance game of the ACG. The construction does not change the set of states and increases the input alphabet by an exponential factor in the number of states and atoms of the ACG.

**Lemma 2.** *Given an $\varepsilon$-free ACG $\mathcal{G} = (\Sigma, Q, q_0, \delta, \alpha)$ and a set $A$ of agents, we can construct a universal parity automaton $\mathcal{U} = (\Sigma \times (Q \to 2^{atom(\mathcal{G})}, Q, q_0, \delta', \alpha)$ on $\Sigma \times (Q \to 2^{atom(\mathcal{G})})$-labeled CGSs with the set $atom(\mathcal{G}) \times A$ of decisions such that $\mathcal{U}$ has the following properties:*

- *If $\mathcal{U}$ accepts a CGS $\mathcal{C} = (P, A, S, s_0, l \times strat_1, atom(\mathcal{G}) \times A, \tau)$ then $\mathcal{G}$ accepts its $\Sigma$ projection $\mathcal{C}' = (P, A, S, s_0, l, atom(\mathcal{G}) \times A, \tau)$.*
- *If $\mathcal{U}$ is empty, then $\mathcal{G}$ is empty.*

*Proof.* We denote with $strat_2$ the function that maps each atom $\mathbf{a}$ of $\mathcal{G}$ to the set $D \subseteq (atom(\mathcal{G}) \times A)^A$ of decisions that are the outcome of the second stage of the acceptance game for some strategy of player reject, when the outcome of the first stage is $\mathbf{a}$ and player *accept* follows the simple strategy for the second stage described in the proof of Theorem 2. Generalizing $strat_2$ to sets of atoms, we define the transition function $\delta'$ of $\mathcal{U}$ by setting $\delta'(q; \sigma, s)$ to *false* if $s(q)$ does not satisfy $\delta(q, \sigma)$, and to a conjunction over $strat_2(s(q))$ otherwise.

If $\mathcal{U}$ accepts a CGS $\mathcal{C} = (P, A, S, s_0, l \times strat_1, atom(\mathcal{G}) \times A, \tau)$, then player *accept* has a winning strategy for $\mathcal{C}' = (P, A, S, s_0, l, atom(\mathcal{G}) \times A, \tau)$ in the acceptance game of $\mathcal{G}$, where the strategy in the first stage is defined by $strat_1$ and the strategy in the second stage is as defined in the proof of Theorem 2.

If $\mathcal{G}$ accepts a CGS $\mathcal{C}$, then there exists, as described in the proof of Theorem 2, a CGS $\mathcal{C}' = (P, A, S, s_0, l, atom(\mathcal{G}) \times A, \tau)$, such that player *accept* wins the acceptance game using some memoryless strategy $strat_1$ in the first stage and the canonical strategy in the second stage. The CGS $\mathcal{C}'' = (P, A, S, s_0, l \times strat_1, atom(\mathcal{G}) \times A, \tau)$ is accepted by $\mathcal{U}$. □

We transform the universal parity automaton $\mathcal{U}$ into a deterministic parity automaton by first transforming $\mathcal{U}$ into a universal co-Büchi automaton with $O(c \cdot n)$ states and then using Safra's construction [16, 7].

**Lemma 3.** *Given a universal automaton $\mathcal{U}$ with $n$ states and $c$ colors, we can construct an equivalent deterministic parity automaton $\mathcal{D}$ with $n^{O(c \cdot n)}$ states and $O(c \cdot n)$ colors.* □

Our transformation of the ACG to the deterministic automaton $\mathcal{D}$ thus increases both the number of states and the size of the input alphabet to at most exponential in the number of states of the ACG. The emptiness game of $\mathcal{D}$ is solved in polynomial time both in the number of states and in the size of the input alphabet, providing an exponential-time procedure for deciding emptiness of an ACG.

**Lemma 4.** *Given a deterministic parity automaton $\mathcal{D} = (\Sigma, Q, q_0, \delta, \alpha)$ with $n$ states and $c$ colors, we can, in time $(n \cdot |\Sigma|)^{O(c)}$, decide emptiness and, if $\mathcal{L}(\mathcal{D}) \neq \emptyset$, construct a finite CGS $\mathcal{C} \in \mathcal{L}(\mathcal{D})$.*

*Proof.* The emptiness problem can be reduced to a bipartite parity game with $n \cdot (1 + |\Sigma|)$ positions and $c$ colors: Player *accept* owns the positions $Q$ and chooses a label $\sigma \in \Sigma$. Player *reject* owns the resulting pairs $Q \times \Sigma$ and can move from a position $(q, \sigma)$ with $\delta(q, \sigma) = \bigwedge_{\upsilon \in \Upsilon}(q_\upsilon, \upsilon)$ to a position $q_\upsilon$ (intuitively by choosing a direction $\upsilon \in \Upsilon$). The colors of the positions owned by player *accept* are defined by the coloring function $\alpha$, while all states owned by player *reject* are colored by the minimum color in the mapping of $\alpha$. This parity game can be solved in time $(n \cdot |\Sigma|)^{O(c)}$ [8].

$\mathcal{D}$ is empty iff player *reject* has a winning strategy, and the $\Sigma$-projection of a memoryless winning strategy for player *accept* defines a CGS in the language of $\mathcal{D}$. □

Combining Lemma 1, Theorem 2, and Lemmata 2, 3 and 4, we obtain the finite model property of automata over concurrent game structures.

**Theorem 3.** *Every non-empty ACG with $n$ states, $c$ colors, $a$ atoms and $a'$ agents accepts some finite CGS with $a \cdot a'$ directions and at most $n^{O(c^3 \cdot n^2 \cdot a^2 \cdot a')}$ states, which can be constructed in time $n^{O(c^3 \cdot n^2 \cdot a^2 \cdot a')}$.* □

Combining Theorem 3 with Theorem 1, we furthermore obtain the finite model property for the alternating-time $\mu$-calculus:

**Theorem 4.** *Given an AMC formula $\varphi$ with alternation depth $d$, $n$ subformulas, and $a$ agents, we can decide satisfiability of $\varphi$ and, if $\varphi$ is satisfiable, construct a model of $\varphi$ in time $n^{O(d^3 \cdot n^4 \cdot a)}$.* □

Matching lower bounds for the AMC satisfiability and synthesis problems are given by the lower bounds for the classic $\mu$-calculus [9,11].

**Corollary 1.** *The satisfiability and synthesis problems for the alternating-time $\mu$-calculus are EXPTIME-complete.* □

## References


1. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.
2. R. Alur, T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Tasiran. Mocha: Modularity in model checking. In *Proc. CAV*, pages 521–525. Springer-Verlag, June 1998.
3. J. R. Büchi. On a decision method in restricted second order arithmetic. *Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.
4. L. de Alfaro, T. A. Henzinger, and R. Majumdar. From verification to control: Dynamic programs for omega-regular objectives. In *Proc. LICS*, pages 279–290. IEEE Computer Society Press, June 2001.
5. E. A. Emerson and C. S. Jutla. Tree automata, $\mu$-calculus and determinacy. In *Proc. FOCS*, pages 368–377. IEEE Computer Society Press, October 1991.
6. S. Even and Y. Yacobi. Relations among public key signature systems. Technical Report 175, Technion, Haifa, Israel, March 1980.
7. B. Finkbeiner and S. Schewe. Uniform distributed synthesis. In *Proc. LICS*, pages 321–330. IEEE Computer Society Press, June 2005.
8. M. Jurdziński. Small progress measures for solving parity games. In *Proc. STACS*, pages 290–301. Springer-Verlag, 2000.
9. D. Kozen and R. J. Parikh. A decision procedure for the propositional $\mu$-calculus. In *Proc. Logic of Programs*, pages 313–325. Springer-Verlag, 1983.
10. S. Kremer and J.-F. Raskin. A game-based verification of non-repudiation and fair exchange protocols. *Journal of Computer Security*, 11(3):399–430, 2003.
11. O. Kupferman and M. Y. Vardi. $\mu$-calculus synthesis. In *Proc. MFCS*, pages 497–507. Springer-Verlag, 2000.
12. 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.
13. R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9(5):521–530, October 1966.
14. D. E. Muller and P. E. Schupp. Simulating alternating tree automata by nondeterministic automata: new results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theor. Comput. Sci.*, 141(1-2):69–107, 1995.
15. M. O. Rabin. *Automata on Infinite Objects and Church's Problem*, volume 13 of *Regional Conference Series in Mathematics*. Amer. Math. Soc., 1972.
16. S. Safra. On the complexity of the $\omega$-automata. In *Proc. FoCS*, pages 319–327. IEEE Computer Society Press, 1988.
17. G. van Drimmelen. Satisfiability in alternating-time temporal logic. In *Proc. LICS*, pages 208–217. IEEE Computer Society Press, June 2003.
18. M. Y. Vardi and P. Wolper. Reasoning about infinite computations. *Journal of Information and Computation*, 115(1):1–37, May 1994.
19. D. Walther, C. Lutz, F. Wolter, and M. Wooldridge. Atl satisfiability is indeed exptime-complete. *Journal of Logic and Computation*, 2006. To appear.
20. T. Wilke. Alternating tree automata, parity games, and modal $\mu$-calculus. *Bull. Soc. Math. Belg.*, 8(2), May 2001.