



THE AUSTRALIAN NATIONAL UNIVERSITY

TR-CS-07-02

**Quantified Mu-Calculus with
Decision Modalities for Concurrent
Game Structures**

Sophie Pinchinat

January 2007

Joint Computer Science Technical Report Series

Department of Computer Science
Faculty of Engineering and Information Technology

Computer Sciences Laboratory
Research School of Information Sciences and Engineering

This technical report series is published jointly by the Department of Computer Science, Faculty of Engineering and Information Technology, and the Computer Sciences Laboratory, Research School of Information Sciences and Engineering, The Australian National University.

Please direct correspondence regarding this series to:

Technical Reports
Department of Computer Science
Faculty of Engineering and Information Technology
The Australian National University
Canberra ACT 0200
Australia

or send email to:

`Technical-DOT-Reports-AT-cs-DOT-anu.edu.au`

A list of technical reports, including some abstracts and copies of some full reports may be found at:

<http://cs.anu.edu.au/techreports/>

Recent reports in this series:

- TR-CS-07-01 Samuel Chang and Peter Strazdins. *A survey of how virtual machine and intelligent runtime environments can support cluster computing*. February 2007.
- TR-CS-06-02 Peter Christen. *A comparison of personal name matching: Techniques and practical issues*. September 2006.
- TR-CS-06-01 Stephen M Blackburn, Robin Garner, Chris Hoffmann, Asjad M Khan, Kathryn S McKinley, Rotem Bentzur, Amer Diwan, Daniel Feinberg, Daniel Frampton, Samuel Z Guyer, Martin Hirzel, Antony Hosking, Maria Jump, Han Lee, J Eliot B Moss, Aashish Phansalkar, Darko Stefanović, Thomas VanDrunen, Daniel von Dincklage, and Ben Wiedermann. *The DaCapo benchmarks: Java benchmarking development and analysis (Extended Version)*. September 2006.
- TR-CS-05-01 Peter Strazdins. *CycleCounter: an efficient and accurate UltraSPARC III CPU simulation module*. May 2005.
- TR-CS-04-04 C. W. Johnson and Ian Barnes. *Redesigning the intermediate course in software design*. November 2004.
- TR-CS-04-03 Alonso Marquez. *Efficient implementation of design patterns in Java programs*. February 2004.

Quantified Mu-Calculus with Decision Modalities for Concurrent Game Structures

Sophie Pinchinat*

Computer Sciences Laboratory

Research School of Information Sciences and Engineering

The Australian National University

Canberra ACT 0200 Australia

Abstract

The emerging technology of interacting systems calls for new verification methods to ensure their reliability. Concurrent Game Structures are expressive abstract models for which several logics have been studied. Yet, these logics are not sufficiently expressive to support certain strategic situation which arise naturally. We propose a second-order mu-calculus enabling a straightforward specification of complex coalition strategies, and also yields a direct synthesis procedure via automata constructions. By translating different alternating-time logics into a natural fragment of our calculus, we recover optimal complexity bounds for these logics.

1 Introduction

Computer-system design currently relies on complex assemblages of *interacting components* which communicate and share resources in order to achieve services.

*Marie Curie Fellow of the European Union, staff member of univ. Rennes 1, France

The combinatorics of such systems is so enormous that the development of adequate formal methods to ensure their reliability has become a major challenge. In this context, games are paradigmatic for providing expressive mathematical models of interactive systems, reflecting their operational semantics and offering adequate reasoning tools. In order to reason formally about interactive models, it is necessary to devise appropriate specification languages in which the desirable behaviors of the system can be stated; once the properties are formulated, methods for automated verification and synthesis can be employed to support the design process.

In the past decade, alternating-time logics such as ATL, ATL*, AMC and GL introduced in [2] have raised considerable interest, in virtue of being natural and of allowing effective decision procedures [19, 10, 6, 18, 14], with reasonable cost for ATL and implementations [8, 1]. Alternating-time logics generalize temporal logics, like CTL, CTL* and the mu-calculus, over models called *Concurrent Game Structures*: these are Kripke structures where the transition to the next state is parametrized by the decisions of *players*. However, these logics show certain limitations in face of the range of strategies required to address realistic situations. We quote two of these.

First, the built-in modalities of ATL, ATL*, and AMC do not distinguish syntactically between quantifications over strategies and over computations. Semantically, this has the consequence that e.g. the *module checking* problem [12] cannot be expressed. To overcome this limitation, the logic GL has been proposed [2]: strategy quantifiers and path quantifiers can now be separated in the formulas. However, the expressive power of this logic is also limited as not every property that can be expressed in the AMC can be expressed in GL.

Second, in each of these logics, the commitment of players within a coalition is binding only until another commitment is specified by the formula. When commitments are conceived as being *bounded* in this sense, it is not possible to express, e.g., strategies which stipulate that a coalition achieves its present objective by merging with another coalition in the future.

In this paper, we propose a formalism with a syntax that distinguishes between strategies and computations and which allows for unbounded commitments. We show that this formalism naturally extends alternating-time logics, and we present a tree-automata construction procedure that can be used to synthesize strategies.

Technically, we consider a subclass of Concurrent Game Structures (CGS) which contains all *turned based* and *Moore* synchronous game structures. In fact, it turns out that any finitary CGS can be translated into an equivalent Moore synchronous game structure, so our results are not restrictive. We encode the CGS

as Kripke structures. Our logic is a monadic second order extension of the propositional mu-calculus [11], augmented with *decision modalities*. The monadic second order extension QD_μ of this logic, allows to quantify over propositions as in [17], but here quantifications and fixed point operators can be arbitrarily interleaved. This extension enables us to directly quantify over strategies.

Turning a CGS into a Kripke Structure consists in moving information about transitions to their origin: we add new propositions, called *player selections*, consisting of pairs composed of a set of directions and a player. Their interpretation is: in a given state, the set of successors along these directions – we simply say the designated successors – form a possible decision for that player; the number of player selections faithfully reflects the size of the CGS transition function. This encoding is valid in a reasonable large class of CGSs which we characterize. Player selections are used to define *decision modalities*: these modalities depend on a fixed player, and express the existence of a player selection whose designated successors satisfy a set of propositions. Indeed, we aim at using a proposition to delimit a sub-tree of the full tree obtained by unfolding the Kripke Structure. Decision modalities are useful to specify that propositions match local decisions of players, ensuring that the sub-tree denotes the outcome of a strategy. Combining such propositions conjunctively enables us to characterize the outcome of a coalition strategy. Quantifications over strategies amounts to quantification over propositions.

It remains to express the desired temporal property of the outcome. Given a set of propositions which denotes a coalition strategy, we compute a new formula by a systematic syntactic transformation of the original formula with respect to those propositions, in such a way that the new formula holds in the full tree unravelling of the given system whenever the original one holds in the outcome (the sub-tree characterized by the propositions). We call this transformation *relativization*. By limiting the depth at which the propositions are injected downwards in the original formula leads to a variant of the relativization called *bounded relativization*, which we use to embed alternating-time logics in our system, thereof exposing their limitations.

Automata constructions for QD_μ are inspired by the ones for the mu-calculus and its second order extension in [17]. However, we need to consider two new issues: the presence of decision modalities, and the ability to nest quantifications under the scope of fixed point operators. Decision modalities are easily handled; the difficulty comes from the latter, as fixed-point operators and quantifiers do not commute in general. To construct automata, we proceed in a bottom-up manner. Existential quantification over a formula amounts to projecting the automaton cor-

responding to this formula [16]. This is a simple operation if the latter automaton is in an adequate form, namely non-deterministic. As classic constructions for the mu-calculus usually yield alternating automata, we use the *Simulation Theorem* [15] to effectively transform alternating automata into non-deterministic ones. However, because quantifiers may occur under the scope of fixed point operators, we also need an automata construction for sub-formulas with possibly free variables. Following [3, Chapter 7], we consider the mu-calculus of automata, where automata may contain variables to which fixed point operators can be applied. It is important to note that by [3, Chapter 9], the Simulation Theorem remains valid for this class of automata. Because each application of the Simulation Theorem causes an exponential blow-up in the size of the automaton, the construction for QD_μ is non-elementary, and this upper bound is tight, by [17]. Although translating AMC and GL into QD_μ may lead to formulas of arbitrary quantification depth, we present a top-down construction where the automata remain small; this method applies whenever the formula is obtained by a bounded relativization. In this way, we can recover the optimal complexity bounds (EXPTIME for AMC and 2-EXPTIME for GL) via embedding into QD_μ .

Whereas quantification over coalition strategies corresponds to projecting automata, keeping track of the projected propositions as in [17] leads to a synthesis procedure: while checking acceptance using a parity game [9], any winning strategy for Player 0 delivers adequate valuations of the propositions, viz. the behavior of the coalition; as positional strategies are sufficient in parity games, there always exist regular solutions.

The paper is organized as follows: we represent the models in Section 2, and the logic QD_μ in Section 3. In Section 4 we revisit the notions of *strategy* and *outcome*, relate them with the *relativization* (Section 4.1); we introduce a QD_μ -definable operator to express the existence of coalition whose commitment persists (Section 4.2). In order to embed alternating-time logics in our system, we use a *bounded* variant of relativization to translate AMC and GL into QD_μ ; this is done in Section 5. Finally, Section 6 is dedicated to automata constructions for QD_μ (Section 6.1). The generic construction is then customized for AMC and GL to recover known complexity bounds (Section 6.2).

2 The models

In the following, for $k \geq 1$, let $[k]$ denote the set $\{1, \dots, k\}$, and assume given an infinite set of atomic propositions $\text{Prop} = \{f, g, h, f^1, f^2, g^1, \dots\}$.

Definition 1. A Concurrent Game Structure (CGS) is a tuple $\mathcal{S} = \langle \mathbf{P}, S, \Lambda, \lambda, d, \delta \rangle$, with:

- A finite set of players $\mathbf{P} = \{p, c, c', \dots\}$ of the form $[n]$. Subsets $C \subseteq \mathbf{P}$ are coalitions.
- A set of states $S = \{s, s', \dots\}$ and a finite set $\Lambda \subseteq \text{Prop}$ of propositions with a function $\lambda : \Lambda \rightarrow 2^S$; a state s is labelled by g whenever $s \in \lambda(g)$. For $\Gamma \subseteq \Lambda$, we set $\lambda(\Gamma) = \bigcap_{g \in \Gamma} \lambda(g)$.
- For each player $p \in \mathbf{P}$, and each state $s \in S$, a natural number $d_p(s) \geq 1$ of decisions available at state s to player p ; we identify the decisions of player p with the numbers $1, \dots, d_p(s)$. A decision vector at s is a tuple $\langle j_1, j_2, \dots, j_n \rangle$, such that $1 \leq j_p \leq d_p(s)$, for all players p .
- For each state $s \in S$ and each decision vector $\langle j_1, j_2, \dots, j_n \rangle$ at s , a state $\delta(s, j_1, j_2, \dots, j_n)$ that results from the state s if each player $p \in \mathbf{P}$ takes decision j_p . The function δ is the transition function.

For two states s and s' , we say that s' is a *successor* of s if $\delta(s, j_1, j_2, \dots, j_n) = s'$ for some decision vector $\langle j_1, j_2, \dots, j_n \rangle$ at s ; we denote by $R \subseteq S \times S$ the binary successor relation, and by sR is the set of successors of s . Let $\text{bd}(s)$ be the value $|sR|$, and let the *branching degree* of \mathcal{S} be $m = \max\{\text{bd}(s) \mid s \in S\}$.

Given $s \in S$, $p \in \mathbf{P}$, and $1 \leq j \leq d_p(s)$, we uniquely associate the set $\text{dec}(s, (j, p)) \subseteq sR$ composed of states $s' = \delta(s, j_1, j_2, \dots, j_n)$ for some decision vector $\langle j_1, j_2, \dots, j_n \rangle$ at s with $j_p = j$; we also set $\text{dec}(s, p) = \{\text{dec}(s, (j, p)) \mid 1 \leq j \leq d_p(s)\}$ as the set of possible such decisions.

Given a coalition C and $s \in S$, a C -move from s is a subset of sR which elements result from fixing a particular decision for each player $c \in C$. Notice that because the decisions of the players might be dependent, a C -move is not in general some element of $\bigcap_{c \in C} \text{dec}(s, c)$, as in e.g. the matching pennies game. Nevertheless, one easily proves that this is the case for the sub-classes of *turned based synchronous* and *Moore synchronous* game structures (see [2]), as the decisions of players are decomposable.

In the rest of the paper, we restrict ourselves to the class of Concurrent Game Structures such that for any $C \subseteq \mathbf{P}$ and any $s \in S$, the set of C -moves from s is $\bigcap_{c \in C} \text{dec}(s, c)$.

We propose a representation of CGS's by Kripke structures. Assume given a CGS \mathcal{S} with branching degree m , and pick up an arbitrary total order \prec over the set of states; for any state s , we canonically order its set sR accordingly to obtain

a list $\{s_1, \dots, s_{\text{bd}(s)}\}$; the indices $1, \dots, \text{bd}(s)$ are the *directions* in s , and $(s)_x$ for the successor of s along the direction x . Subsets of sR and subsets of $[\text{bd}(s)]$ are in a one-to-one correspondence. For each $p \in \mathbf{P}$ and each $1 \leq j \leq d_p(s)$, let \mathcal{D}_j be the set of directions corresponding to $\text{dec}(s, (j, p))$; we label s by the *player selection* (\mathcal{D}_j, p) .

Definition 2. The Kripke Structure representation of \mathcal{S} is defined by $K_{\mathcal{S}} = \langle S, \Lambda \cup (\mathcal{P}([m]) \times \mathbf{P}), \lambda \cup \theta, R \rangle$, where for each $(\mathcal{D}, p) \in \mathcal{P}([m]) \times \mathbf{P}$, $s \in \theta((\mathcal{D}, p))$ if and only if there exists $1 \leq j \leq d_p(s)$ such that $\{(s)_x\}_{x \in \mathcal{D}} = \text{dec}(s, (j, p))$.

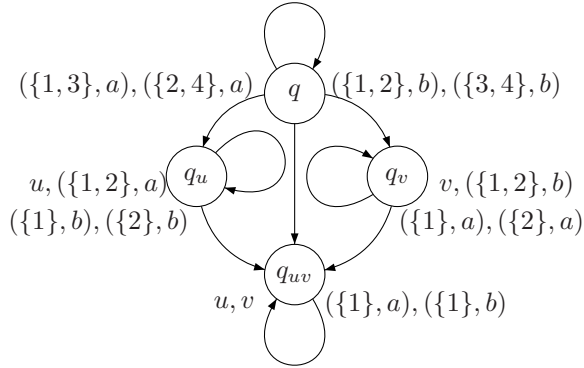


Figure 1: A system with two processes

We borrow from [2] an example of a Moore synchronous game structure to illustrate Definition 2: two processes a and b assign Boolean values to variables u and v , respectively, in an independent manner, starting with the value false for both variables. By setting $q \prec q_u \prec q_v \prec q_{uv}$, we have e.g. $\text{bd}(q_v) = 2$, $(q_v)_1 = q_v$, and $(q_v)_2 = q_{uv}$. Player selections are added accordingly, as depicted in Figure 1.

Lemma 1. A Kripke Structure $\langle S, \Lambda \cup (\mathcal{P}([m]) \times \mathbf{P}), \lambda \cup \theta, R \rangle$, with $\lambda : \Lambda \rightarrow 2^S$ and $\theta : \mathcal{P}([m]) \times \mathbf{P} \rightarrow 2^S$, denotes a CGS over Λ whenever for each state $s \in S$, and for each collection $\{(\mathcal{D}^p, p)\}_{p \in \mathbf{P}}$ with $s \in \theta((\mathcal{D}^p, p))$, the set $\cap_p \mathcal{D}^p$ has a single element.

From now on, we tacitly refer to CGS's for Kripke structures which fulfill Lemma 1, and by abuse of notations we write \mathcal{S} instead of $K_{\mathcal{S}}$, when it is clear from the context. We also let $\bar{\Lambda}$ denote the 'extended' set of propositions $\Lambda \cup (\mathcal{P}([m]) \times \mathbf{P})$ and $\bar{\lambda}$ denote $\lambda \cup \theta$.

3 The logical framework

We propose a generalization of [17] which is twofold: we enrich the propositional mu-calculus [11] by allowing *decision modalities*, and we consider its monadic second order extension by allowing quantifications over atomic propositions, even under the scope of fixed points operators.

The logic D_μ is the mu-calculus with *decision modalities*. Given a set Prop of atomic propositions and a set of variables $\text{Var} = \{Z, Y, \dots\}$, the syntax of D_μ is:

$$g \mid \textcircled{p}\Gamma \mid \top \mid \neg\beta \mid \beta_1 \vee \beta_2 \mid \mathbf{EX} \beta \mid Z \mid \mu Z.\beta(Z)$$

where $g \in \text{Prop}$, $p \in \mathbf{P}$, and $\Gamma \subseteq \text{Prop}$. Additionally, we require that in each formula $\mu Z.\beta(Z)$, the variable Z occurs under an even number of negation symbols \neg in $\beta(Z)$. A variable $Z \in \text{Var}$ is *free* in β if it does not occur under the scope of a μZ . operator. A *sentence* is a formula with no free variable. The subset of formulas which do not contain any decision modality is the standard propositional mu-calculus, whence the use of notations \perp , $\mathbf{AX} \beta$, $\beta_1 \wedge \beta_2$, $\beta_1 \Rightarrow \beta_2$, and $\nu Z.\beta(Z)$ to denote $\neg\top$, $\neg\mathbf{EX} \neg\alpha$, $\neg(\neg\beta_1 \vee \neg\beta_2)$, $\neg\beta_1 \vee \beta_2$, and $\neg\mu Z.\neg\beta(\neg Z)$, respectively. Also, by abuse of notation when convenient we might expand $\textcircled{p}\Gamma$ as $\textcircled{p}\bigwedge_{f \in \Gamma} f$. A formula $\beta \in D_\mu$ is interpreted in a CGS $\mathcal{S} = \langle S, \bar{\Lambda}, \bar{\lambda}, R \rangle$ supplied with a valuation $\text{val} : \text{Var} \rightarrow 2^S$. Its semantics $\llbracket \beta \rrbracket_{\mathcal{S}}^{\text{val}}$ is a subset of S , defined by induction over the structure of β as follows:

$$\begin{aligned} \llbracket g \rrbracket_{\mathcal{S}}^{\text{val}} &= \{s \in S \mid s \in \lambda(g)\} \\ \llbracket \textcircled{p}\Gamma \rrbracket_{\mathcal{S}}^{\text{val}} &= \{s \in S \mid sR \cap \lambda(\Gamma) \in \text{dec}(s, p)\} \\ \llbracket \top \rrbracket_{\mathcal{S}}^{\text{val}} &= S \\ \llbracket \neg\beta \rrbracket_{\mathcal{S}}^{\text{val}} &= S \setminus \llbracket \beta \rrbracket_{\mathcal{S}}^{\text{val}} \\ \llbracket \beta_1 \wedge \beta_2 \rrbracket_{\mathcal{S}}^{\text{val}} &= \llbracket \beta_1 \rrbracket_{\mathcal{S}}^{\text{val}} \cap \llbracket \beta_2 \rrbracket_{\mathcal{S}}^{\text{val}} \\ \llbracket \mathbf{EX} \beta \rrbracket_{\mathcal{S}}^{\text{val}} &= \{s \in S \mid \exists s' \in sR \wedge s' \in \llbracket \beta \rrbracket_{\mathcal{S}}^{\text{val}}\} \\ \llbracket Z \rrbracket_{\mathcal{S}}^{\text{val}} &= \text{val}(Z) \\ \llbracket \mu Z.\beta(Z) \rrbracket_{\mathcal{S}}^{\text{val}} &= \bigcap \{S' \subseteq S \mid \llbracket \beta(Z) \rrbracket_{\mathcal{S}}^{\text{val}(S'/Z)} \subseteq S'\} \end{aligned}$$

As a valuation val does not influence the semantics of a sentence $\beta \in D_\mu$, we then simply write $\llbracket \beta \rrbracket_{\mathcal{S}}$. By definition, $s \in \llbracket \textcircled{p}\Gamma \rrbracket_{\mathcal{S}}$ means that the successors of s labeled by all elements of Γ form a decision available at s to player p . Formulas like $\textcircled{p}\Gamma$ are called *decision modalities*. For example in Fig.1, we have

$q_v \in \llbracket \textcircled{u} \rrbracket_S$. In the following, given $\beta \in D_\mu$, we use the concise CTL-like notation $\mathbf{AG}(\beta)$ for $\nu Z.(\mathbf{AX} Z \wedge \beta)$, which expresses that β is globally true in the future, and $\mathbf{EF}(\beta)$ for $\neg \mathbf{AG}(\neg \beta)$.

We define $\mathbf{Q}D_\mu$, for “quantified D_μ ”; its syntax is as for D_μ but with quantifications over sets of atomic propositions $\Gamma \subseteq \text{Prop}$:

$$g \mid \textcircled{p}\Gamma \mid \top \mid \neg\alpha \mid \alpha_1 \vee \alpha_2 \mid \mathbf{EX} \alpha \mid Z \mid \mu Z.\alpha(Z) \mid \exists \Gamma.\alpha$$

The semantics of $\mathbf{Q}D_\mu$ generalizes the one of D_μ : the cases of g , $\textcircled{p}\Gamma$, \top , $\neg\alpha$, $\alpha_1 \vee \alpha_2$, $\mathbf{EX} \alpha$, Z , and $\mu Z.\alpha(Z)$ are dealt inductively. The semantics of quantified formulas relies on the notion of *labeling*, in the spirit of [17]: basically, labelings are one player ’complete’ CGS’s which witness the existential statement about Γ .

Definition 3. Let $\Gamma \subseteq \text{Prop}$ and let $m \geq 1$. A Γ -labeling of branching degree m is a pair (\mathcal{E}, r) where $\mathcal{E} = \langle E, \Gamma, \gamma, R' \rangle$ is a Kripke Structure over Γ , $r \in E$ is the root, and \mathcal{E} is complete, viz. $\text{bd}(e) = m$ for every $e \in E$.

We compose labelings and CGS’s. Given a rooted CGS (\mathcal{S}, s) over $\bar{\Lambda}$, and a Γ -labeling (\mathcal{E}, r) , with the same branching degree, the *labeling of (\mathcal{S}, s) by \mathcal{E}* is the CGS over $\bar{\Lambda} \cup \Gamma$, written $(\mathcal{S}, s) \times (\mathcal{E}, r) = \langle S \times E, \bar{\Lambda} \cup \Gamma, \lambda', R'' \rangle$, rooted at (s, r) , and defined by: $\lambda'(g) = \bar{\lambda}(g) \cup \gamma(g)$, for each $g \in \bar{\Lambda} \cup \Gamma$ with the convention that if $g \notin \bar{\Lambda}$ (or $\notin \Gamma$) then $\bar{\lambda}(g)$ (respectively $\gamma(g)$) is the empty set. And $(s_1, e_1)R''(s_2, e_2)$ if and only if $s_1 R s_2$, $e_1 R' e_2$ and the directions in each components match. Since the labeling function γ does not add any player selection, by Lemma 1, $(\mathcal{S}, s) \times (\mathcal{E}, r)$ is a CGS. Moreover, as propositions of compound states accumulate, and because \mathcal{E} is complete, $(\mathcal{S}, s) \times (\mathcal{E}, r)$ is bisimilar to (\mathcal{S}, s) in the usual sense, if we restrict to propositions that are not in Γ . In particular if $\Gamma = \emptyset$, $(\mathcal{S}, s) \times (\mathcal{E}, r)$ is bisimilar to (\mathcal{S}, s) . We have now the material to define the meaning of our new quantifiers: $s \in \llbracket \exists \Gamma.\alpha \rrbracket_S^{\text{val}}$ if, and only if, there exists a Γ -labeling (\mathcal{E}, r) such that $(s, r) \in \llbracket \alpha \rrbracket_{(\mathcal{S}, s) \times (\mathcal{E}, r)}^{\text{val}'}$ with $\text{val}'(Z) = \text{val}(Z) \times E$.

4 Strategies and Outcomes

In CGS’s, *strategy* and *outcome* are central concepts underlying the semantics of their logics. We revisit these notions: strategies are labelings, and outcomes are CGS’s resulting from the pruning of the original structure w.r.t. labelings. [2] originally defined *outcomes* in a way we informally explain: a strategy of a player p from some state s determines for each finite computation starting in s

and ending in some current state s' a decision in $\text{dec}(s', p)$. Given a coalition of players $C \subseteq \mathbf{P}$, a state s and a set F_C of strategies from s for each player of C , the set of outcomes $\text{outcome}(F_C, \mathcal{S}, s)$ resulting from these strategies is composed of all the plays (paths) consistent with F_C , heretofore unstructured. We aim at structuring it as a CGS. To do so, we consider particular labelings:

Definition 4. Given $C \subseteq \mathbf{P}$, and a set $f_C = \{f^c\}_{c \in C}$ of atomic propositions, a (f_C, C) -strategy from s is a $\{f^c\}_{c \in C}$ -labeling of (\mathcal{S}, s) , (\mathcal{E}, r) such that

$$(s, r) \in \llbracket \mathbf{AG} \left(\bigwedge_{c \in C} \odot f^c \right) \rrbracket_{(\mathcal{S}, s) \times (\mathcal{E}, r)} \quad (1)$$

We simply say f_C -strategy when clear from the context. Assume fixed a coalition $C \subseteq \mathbf{P}$, and a f_C -strategy (\mathcal{E}, r) . We define the structure $\text{CGS}(f_C, \mathcal{S}, s)$, called a C -outcome of (\mathcal{S}, s) , as follows: if $C = \emptyset$ then $\text{CGS}(f_\emptyset, \mathcal{S}, s)$ is (\mathcal{S}, s) itself. Otherwise, $\text{CGS}(f_C, \mathcal{S}, s)$ results from retaining of $(\mathcal{S}, s) \times (\mathcal{E}, r)$ only those states that would be reachable if the strategies of f_C were applied. Formally, we prune $(\mathcal{S}, s) \times (\mathcal{E}, r)$ according to the following procedure:

Step 1: Remove any state of $(\mathcal{S}, s) \times (\mathcal{E}, r)$ except the root (s, r) where some proposition f^c does not hold, and remove the propositions f^c as they are now trivial.

Step 2: Update the player selections (\mathcal{D}, p) in each remaining state as some successors might have been deleted in Step 1: in each state s' which ordered set of successors is $\{s_x\}_{x \in J_0}$, assume the set $\{s_y\}_{y \in J_1}$ of successors have been deleted in Step 1. Write $m_0 = |J_0|$, $m_1 = |J_1|$, and $m_2 = m_0 - m_1$. Let $\pi : [m_0] \setminus J_1 \rightarrow [m_2]$ be the canonical monotonic mapping which renumbers the remaining successors. Each player selection (\mathcal{D}, p) is updated accordingly as $(\pi(\mathcal{D}), p)$. In particular, the player selections (\mathcal{D}, c) where $c \in C$ all become $([m_2], c)$ and can be removed since constant.

Essentially, by Step 2 and Lemma 1, $\text{CGS}(f_C, \mathcal{S}, s)$ is a CGS (rooted at s). It is not difficult to see that the set of maximal paths in $\text{CGS}(f_C, \mathcal{S}, s)$ coincides with the definition of 'outcome' in the sense of [2].

Lemma 2. Given two distinct sets $C_1, C_2 \subseteq \mathbf{P}$, and any two f_{C_i} -strategies (\mathcal{E}_i, r_i) ($i \in \{1, 2\}$), $\text{CGS}(f_{C_1 \cup C_2}, \mathcal{S}, s)$ and $\text{CGS}(f_{C_1}, \text{CGS}(f_{C_2}, \mathcal{S}, s), (s, r_2))$ are isomorphic.

4.1 Relativization of formulas

We present a simple mechanism called the *relativization* which transforms a formula by propagating downward in the formula a set of atomic propositions. For $\Gamma \subseteq \text{Prop}$, the Γ -relativization is a mapping written $(\cdot|\Gamma) : \mathbb{Q}D_\mu \rightarrow \mathbb{Q}D_\mu$ defined by induction over the formulas. We concisely write Γ for $\bigwedge_{g \in \Gamma} g$ and we take the convention that $\bigwedge_{g \in \emptyset} g$ is equivalent to \top . Formulas of the form g , \top , and Z are left unchanged by the mapping $(\cdot|\Gamma)$, and

$$\begin{aligned} (\neg\alpha|\Gamma) &= \neg(\alpha|\Gamma) \\ (\mathcal{D}f|\Gamma) &= \mathcal{D}(\Gamma \wedge f) \\ (\alpha_1 \vee \alpha_2|\Gamma) &= (\alpha_1|\Gamma) \vee (\alpha_2|\Gamma) \\ (\mathbf{EX} \alpha|\Gamma) &= \mathbf{EX} [\bigwedge_{g \in \Gamma} g \wedge (\alpha|\Gamma)] \\ (\mu Z.\alpha(Z)|\Gamma) &= \mu Z.(\alpha(Z)|\Gamma) \\ (\exists \Gamma'.\alpha|\Gamma) &= \exists \Gamma'.(\alpha|\Gamma) \end{aligned}$$

From this definition, we immediately obtain the equivalences:

$$(\alpha|\emptyset) \equiv \alpha \quad \text{and} \quad (\alpha|\Gamma \cup \{g\}) \equiv ((\alpha|\Gamma)|g). \quad (2)$$

Theorem 3 below enlightens the semantics of the *relativization*: under the assumption that the coalition C follows some f_C -strategy, a property will hold in the resulting C -outcome if and only if the f_C -relativization of this property holds in \mathcal{S} labeled by f_C .

Theorem 3. *Given a rooted CGS (\mathcal{S}, s) , $C \subseteq \mathbf{P}$, and a f_C -strategy (\mathcal{E}, r) from s , we have: for any $\alpha \in \mathbb{Q}D_\mu$, and any valuation $\text{val} : \text{Var} \rightarrow 2^S$, let $\text{val}'(Z) = \text{val}(Z) \times E$:*

$$\llbracket (\alpha|f_C) \rrbracket_{(\mathcal{S}, s) \times (\mathcal{E}, r)}^{\text{val}'} = \llbracket \alpha \rrbracket_{\text{CGS}(f_C, \mathcal{S}, s)}^{\text{val}'}$$

Proof. The proof is conducted by a double induction on the set C and on the structure of α . The case $C = \emptyset$ is trivial and independent of α , since $(\alpha|f_C)$ is α by (2), and both $(\mathcal{S}, s) \times (\mathcal{E}, r)$ and $\text{CGS}(f_C, \mathcal{S}, s)$ are isomorphic to (\mathcal{S}, s) . Let $C = C' \cup \{c\}$, with $c \notin C'$. The f_C -strategy (\mathcal{E}, r) can be decomposed into $(\mathcal{E}', r') \times (\mathcal{E}_c, r_c)$, where (\mathcal{E}', r') is a $(f_{C'}, C')$ -strategy and (\mathcal{E}_c, r_c) is a $(\{f^c\}, \{c\})$ -strategy; write (\mathcal{S}', r') for $(\mathcal{S}, s) \times (\mathcal{E}', r')$. By (2):

$$\llbracket (\alpha|f_C) \rrbracket_{(\mathcal{S}, s) \times (\mathcal{E}, r)}^{\text{val}'} = \llbracket ((\alpha|f_{C'})|f^c) \rrbracket_{(\mathcal{S}', s') \times (\mathcal{E}_c, r_c)}^{\text{val}'} \quad (3)$$

Lemma 4. For any rooted CGS (S', s') , any $(\{f^c\}, \{c\})$ -strategy (\mathcal{E}, r) , any $\alpha \in \text{QD}_\mu$, and any valuation $\text{val} : \text{Var} \rightarrow 2^S$, $\llbracket (\alpha | f) \rrbracket_{(S', s') \times (\mathcal{E}, r)}^{\text{val}'} = \llbracket \alpha \rrbracket_{\text{CGS}(f, S, s)}^{\text{val}'}$ where $\text{val}'(Z) = \text{val}(Z) \times E$.

The proof of Lemma 4 is done by a simple induction over the formulas, as in [17]. Now, the right hand side of (3) is equal to $\llbracket (\alpha | f_{C'}) \rrbracket_{\text{CGS}(f, S', s')}^{\text{val}'}$. Since $\text{CGS}(f, S', s')$ is isomorphic to $\text{CGS}(f, S, s) \times (\mathcal{E}', r')$, it is also equal to $\llbracket (\alpha | f_{C'}) \rrbracket_{\text{CGS}(f, S, s) \times (\mathcal{E}', r')}^{\text{val}'}$ which by induction hypothesis coincides with $\llbracket \alpha \rrbracket_{\text{CGS}(f_{C'}, \text{CGS}(f, S, s), (s, r_c))}^{\text{val}'}$. We conclude by applying Lemma 2. \square

4.2 Expressing properties of strategies

We specialize the logic QD_μ to specify strategies of players. We equip the language with syntactic macros: given a coalition $C \subseteq \mathbf{P}$, and a set $f_C = \{f^c\}_{c \in C}$ of propositions, let us define

$$\hat{\exists}f_C.\alpha \stackrel{\text{def}}{=} \exists f_C. [\mathbf{AG} (\bigwedge_{c \in C} \odot f^c) \wedge \alpha]$$

Theorem 5. Given a CGS S , a coalition $C \subseteq \mathbf{P}$, and a sentence $\alpha \in \text{QD}_\mu$, the formula $\hat{\exists}f_C.(\alpha | f_C)$ (where f_C is a fresh set of atomic propositions indexed over C) characterizes the set of states from which there exists a C -outcome of (S, s) satisfying α .

Proof. Write $f_C = \{f^c\}_{c \in C}$, and let $s \in \llbracket \hat{\exists}f_C.(\alpha | f_C) \rrbracket_s$. By definition, there exists a f_C -labeling from s , (\mathcal{E}, r) such that

$$(s, r) \in \llbracket \mathbf{AG} (\bigwedge_{c \in C} \odot f^c) \rrbracket_{(s, s) \times (\mathcal{E}, r)}, \text{ and} \quad (4)$$

$$(s, r) \in \llbracket (\alpha | f_C) \rrbracket_{(s, s) \times (\mathcal{E}, r)}. \quad (5)$$

By (4), (\mathcal{E}, r) is a f_C -strategy. By Theorem 3, (5) is equivalent to $s \in \llbracket \alpha \rrbracket_{\text{CGS}(f_C, S, s)}$, which concludes. For the reciprocal, unroll the reasoning backward. \square

Let us discuss the modality $\hat{\exists}f_C.(\cdot | f_C)$. By the above theorem, $\hat{\exists}f_C.(\alpha | f_C)$ states the existence of a C -outcome fixed once for all in which α shall be interpreted. Accordingly, any sub-formula of α of the form $\hat{\exists}f_{C'}.(\alpha' | f_{C'})$ states

the existence of a C' -outcome 'inside' the former C -outcome, where α' holds. Notice that C and C' need not be disjoint: consider the formula

$$\hat{\exists}\{f^1, f^2\}.(\mathbf{EF} [\hat{\exists}\{g^2, g^3\}.(\mathbf{AG} h|\{g^2, g^3\})]|\{f^1, f^2\})$$

By relativization (Section 4.1), it specifies the existence of a $\{1, 2\}$ -outcome S' which contains a path eventually hitting the property $(\hat{\exists}\{g^2, g^3\}.(\mathbf{AG} h|\{g^2, g^3\})|\{f^1, f^2\})$. Again by the relativization, this property means the existence of two propositions g^2 and g^3 such that $\mathbf{AG} (\mathcal{Q}(f^1 \wedge f^2 \wedge g^2))$, $\mathbf{AG} (\mathcal{Q}(f^1 \wedge f^2 \wedge g^3))$, and $(\mathbf{AG} h|\{f^1, f^2, g^2, g^3\})$ hold. By Theorem 3, $\mathbf{AG} h$ is interpreted inside S' , rather than in S . Moreover, the innermost strategy of player 2 denoted by g^2 (if it exists) is such that $\mathbf{AG} (\mathcal{Q}(f^1 \wedge f^2 \wedge g^2))$ and is therefore consistent with f^2 . The modality $\hat{\exists}f_C.(|f_C)$ enables to 'superpose' strategies, e.g. the $\{g^2, g^3\}$ -strategy on top of the $\{f^1, f^2\}$ -strategy. Nevertheless, superposition is also avoidable by considering a variant of the relativization, as explained below.

5 Alternating Time Logics

We consider the logics AMC and GL of [2]. Results for weaker logics such as ATL, Fair ATL, and ATL* follow from their natural embedding either into AMC or GL. We show that AMC and GL are natural fragments of QD_μ .

5.1 Bounded relativization

For $\Gamma \subseteq \text{Prop}$, the *bounded Γ -relativization* is a mapping written $(\cdot] \Gamma) : QD_\mu \rightarrow QD_\mu$ defined by induction over the formulas. It is like the relativization of Section 4.1, except that the downward propagation of propositions terminates when encountering a quantified sub-formula:

$$(\exists \Gamma'. \alpha'] \Gamma) = \exists \Gamma'. \alpha' \tag{6}$$

Using bounded relativization, we define the modality $\hat{\exists}f_C(|f_C)$ with the following semantics: $\hat{\exists}f_C(\alpha |f_C)$ states the existence a C -outcome where α holds, but where any sub-formula $\hat{\exists}f_{C'}.(\alpha' |f_{C'})$ of α , or even $\hat{\exists}f_{C'}.(\alpha' |f_{C'})$, is interpreted in the entire structure, and not only in the designated C -outcome. We retrieve the spirit of the coalition modalities in ATL.

5.2 The Alternating-time μ -calculus

We recall the syntax of AMC: formulas are of the form

$$g \mid \top \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid Z \mid \mu Z.\varphi(Z) \mid \langle\langle C \rangle\rangle \circ \varphi$$

with $g \in \text{Prop}$, $C \subseteq \mathbf{P}$, and where each $Z \in \text{Var}$ occurs under an even number of negation symbols \neg in $\varphi(Z)$. These formulas are interpreted over CGS's (in the sense of Def 1), supplied with a valuation $\text{val} : \text{Var} \rightarrow 2^S$. Given $\varphi \in \text{AMC}$, its interpretation $\varphi^S(\text{val}) \subseteq S$ is inductively defined by:

$$\begin{aligned} g^S(\text{val}) &= \lambda(g) & (\neg\varphi)^S(\text{val}) &= S \setminus \varphi^S(\text{val}) \\ \top^S(\text{val}) &= S & Z^S(\text{val}) &= \text{val}(Z) \\ (\varphi_1 \vee \varphi_2)^S(\text{val}) &= \varphi_1^S(\text{val}) \cup \varphi_2^S(\text{val}) \\ (\mu Z.\varphi(Z))^S(\text{val}) &= \bigcap \{S' \subseteq S \mid \varphi(Z)^S(\text{val}[S'/Z]) \subseteq S'\} \\ (\langle\langle C \rangle\rangle \circ \varphi)^S(\text{val}) & \text{ is the set of states } s \in S \text{ such that there exists a } C\text{-move from } s \\ & \text{ contained in } \varphi^S(\text{val}). \end{aligned}$$

We define the mapping $\hat{\cdot} : \text{AMC} \rightarrow \text{QD}_\mu$ inductively by: formulas like g , \top and Z are left unchanged, formulas like $\neg\varphi$, $\varphi_1 \vee \varphi_2$, and $\mu Z.\varphi(Z)$ are dealt inductively, and finally we set

$$\widehat{\langle\langle C \rangle\rangle \circ \varphi} \stackrel{\text{def}}{=} \exists f_C. (\mathbf{AX} \hat{\varphi} \mid f_C) \quad (7)$$

where $f_C = \{f^c\}_{c \in C}$ is a set of fresh atomic propositions. Notice that the size of $\hat{\varphi}$ is linear in the size of φ .

Theorem 6. *Given a CGS \mathcal{S} , $\varphi \in \text{AMC}$, and a valuation $\text{val} : \text{Var} \rightarrow 2^S$, we have*

$$\varphi^S(\text{val}) = \llbracket \hat{\varphi} \rrbracket_{K_S}^{\text{val}}$$

where K_S is the Kripke Structure representation of \mathcal{S} .

Proof. We conduct the proof by induction over φ . The only non trivial case is $\langle\langle C \rangle\rangle \circ \varphi$. Notice that by the definition of $\hat{\cdot}$ and the bounded relativization, $(\hat{\varphi} \mid f_C) = \hat{\varphi}$.

(\Rightarrow) Let $s \in (\langle\langle C \rangle\rangle \circ \varphi)^S(\text{val})$. We first prove

$$s \in \llbracket \exists f_C. \bigwedge_{c \in C} \odot f^c \wedge \mathbf{AX} [f_C \Rightarrow \hat{\varphi}] \rrbracket_{K_S}^{\text{val}} \quad (8)$$

If $s \in (\langle\langle C \rangle\rangle \circ \varphi)^S(\text{val})$, then there exists C -move $M_{C,s} \subseteq \varphi^S(\text{val})$. Let \mathcal{D} be the set of directions associated with $M_{C,s}$. Let m be the branching degree of \mathcal{S} ,

and let $\mathcal{E} = \langle \{r\} \cup E, f_C, \gamma, R' \rangle$ be the f_C -labeling where r is the root, E is the disjoint union of $E^0 = \{e_x^0\}_{x \in [m]}$ and $E^1 = \{e_x^1\}_{x \in [m]}$; denote by $E_D^0 \subseteq E^0$ the set $\{e_x^0\}_{x \in \mathcal{D}}$. The relation R' is $\{(r, e^0) \mid e^0 \in E^0\} \cup (E^0 \times E^1) \cup (E^1 \times E^1)$, and the function $\gamma : f_C \rightarrow \{r\} \cup E$ is constant equal to $E_D^0 \cup E^1$, therefore

$$(s, r) \in \llbracket \bigwedge_{c \in C} \mathcal{O}f^c \rrbracket_{(K_S, s) \times (\mathcal{E}, r)} \quad (9)$$

Moreover, since for any $x \in \mathcal{D}$, the structures $(K_S, (s)_x)$ and $(K_S, (s)_x) \times (\mathcal{E}, e_x)$ are bisimilar up to propositions f^c , and because no proposition f^c occurs in $\widehat{\varphi}$, we apply the induction hypothesis $(s)_x \in \llbracket \widehat{\varphi} \rrbracket_{K_S}^{\text{val}}$. Consequently, for $\text{val}'(Z) = \text{val}(Z) \times (\{r\} \cup E)$, we have

$$(s, r) \in \llbracket \mathbf{AX} [f_C \Rightarrow \widehat{\varphi}] \rrbracket_{(K_S, s) \times (\mathcal{E}, r)}^{\text{val}'} \quad (10)$$

as the only successors of s labeled by the f^c 's are the $(s)_x$'s with $x \in \mathcal{D}$. By (9) and (10) we obtain (8). It is actually possible to modify the definition of E^1 and of the mapping γ accordingly so that for any compound state $(s', e) \in S \times E$ $(s', e) \in \llbracket \bigwedge_{c \in C} \mathcal{O}f^c \rrbracket_{(K_S, s) \times (\mathcal{E}, r)}$. The construction is tedious and we omit it. By the above and by (9), the labeling (\mathcal{E}, r) becomes a f_C -strategy:

$$(s, r) \llbracket \mathbf{AG} \left(\bigwedge_{c \in C} \mathcal{O}f^c \right) \rrbracket_{(K_S, s) \times (\mathcal{E}, r)}^{\text{val}'} \quad (11)$$

We conclude by (8), (9), and (11).

(\Leftarrow) Assume $s \in \llbracket \hat{\exists} f_C. (\mathbf{AX} \widehat{\varphi}) f_C \rrbracket_{K_S}^{\text{val}}$. By the definitions of $\hat{\exists}$ and the bounded relativization, there exists (\mathcal{E}, r) a f_C -strategy from s such that $(s, r) \in \llbracket \mathbf{AG} (\bigwedge_{c \in C} \mathcal{O}f^c) \wedge \mathbf{AX} [f_C \Rightarrow \widehat{\varphi}] \rrbracket_{(K_S, s) \times (\mathcal{E}, r)}^{\text{val}'}$. There is no difficulty in exhibiting a C -move $M_{C, s}$ contained in $\llbracket \widehat{\varphi} \rrbracket_{(K_S, s) \times (\mathcal{E}, r)}^{\text{val}'}$, and to use the induction hypothesis to conclude $M_{C, s} \subseteq \varphi^{\mathcal{S}}(\text{val})$. \square

5.3 The logic GL

The set of formulas of GL splits into three types: the state formulas, the tree formulas, and the path formulas; the last two ones are inherited from CTL*.

- *State formulas* are of the form g , \top , $\neg\varphi$, or $\varphi_1 \vee \varphi_2$ – where φ , φ_1 , and φ_2 are state formulas – , and $\exists C.\theta$ – where θ is a tree formula – .

- *Tree formulas* are of the form φ – where φ is a state formula \neg , $\neg\theta$, or $\theta_1 \vee \theta_2$ – where θ , θ_1 , and θ_2 are path formulas \neg , and $\mathbf{E} \psi$ – where ψ is a path formula \neg .
- *Path Formulas* are of the form θ – where θ is a tree formula \neg , $\neg\psi$, $\psi_1 \vee \psi_2$, $\bigcirc\psi$, or $\psi_1 \mathbf{U} \psi_2$ – where ψ , ψ_1 , and ψ_2 are path formulas \neg .

We simply sketch the semantics of GL, and we assume the reader is familiar with CTL* (see [2] for details). Let φ be a state formula, and let (\mathcal{S}, s) be a rooted CGS. $\mathcal{S}, s \models \varphi$ indicates that s satisfies φ in \mathcal{S} ; it is defined by induction over φ . Let us focus on formulas of the form $\exists C.\theta$, as the others are dealt inductively or follows the semantics of CTL*: $\mathcal{S}, s \models \exists C.\theta$ whenever there exists a C -outcome CGS (f_C, \mathcal{S}, s) which satisfies θ . Now, θ is a tree formula and a CTL* interpretation up to sub-formulas $\exists C'.\varphi'$ interpreted back inside \mathcal{S} .

Regarding the translation of GL into QD_μ , first we find more pedagogical to first establish a translation of GL into the monadic second order extension of CTL* with decision modalities, written QDCTL*; it generalizes the proposal of [5] since quantifications may occur in sub-formulas. Second, statements of QDCTL* can be expressed in QD_μ by adapting the translation procedure of [4].

For QDCTL*, we take the convention to note α a tree formula (it may contain quantifications) and π a path formula, and to write $\mathbf{A} \pi$ for $\neg\mathbf{E} \neg\pi$, and $\mathbf{G} \alpha$ for $\neg(\top \mathbf{U} \neg\alpha)$. We adapt the definition of the bounded relativization (Section 5.1): for each set $\Gamma \subseteq \text{Prop}$, we consider two syntactic transformations of QDCTL* formulas, $(\cdot]_{\forall}\Gamma)$ and $(\cdot]_{\exists}\Gamma)$, defined by: for all tree formulas α , α_1 , and α_2 , we set $(\neg\alpha]_{\exists}\Gamma) = \neg(\alpha]_{\exists}\Gamma)$, $(\neg\alpha]_{\forall}\Gamma) = \neg(\alpha]_{\forall}\Gamma)$, $(\alpha_1 \vee \alpha_2]_{\exists}\Gamma) = (\alpha_1]_{\exists}\Gamma) \vee (\alpha_2]_{\exists}\Gamma)$, and $(\alpha_1 \vee \alpha_2]_{\forall}\Gamma) = (\alpha_1]_{\forall}\Gamma) \vee (\alpha_2]_{\forall}\Gamma)$. We set similar definitions for path formulas. For the remaining cases:

$$\begin{aligned}
(g]_{\forall}\Gamma) &= (g]_{\exists}\Gamma) &&= g \\
(\top]_{\forall}\Gamma) &= (\top]_{\exists}\Gamma) &&= \top \\
(\exists\Gamma'.\alpha]_{\forall}\Gamma) &= (\exists\Gamma'.\alpha]_{\exists}\Gamma) &&= \exists\Gamma'.\alpha \\
(\mathbf{A} \pi]_{\forall}\Gamma) &= (\mathbf{A} \pi]_{\exists}\Gamma) &&= \mathbf{A} (\pi]_{\forall}\Gamma) \\
(\mathbf{E} \pi]_{\forall}\Gamma) &= (\mathbf{E} \pi]_{\exists}\Gamma) &&= \mathbf{E} (\pi]_{\exists}\Gamma) \\
(\pi_1 \mathbf{U} \pi_2]_{\forall}\Gamma) &= [\Gamma \Rightarrow (\pi_1]_{\forall}\Gamma)] \mathbf{U} [\Gamma \Rightarrow (\pi_2]_{\forall}\Gamma)] \\
(\pi_1 \mathbf{U} \pi_2]_{\exists}\Gamma) &= [\Gamma \wedge (\pi_1]_{\exists}\Gamma)] \mathbf{U} [\Gamma \wedge (\pi_2]_{\exists}\Gamma)]
\end{aligned}$$

The bounded Γ -relativization of α is $(\alpha]_{\exists}\Gamma) \stackrel{\text{def}}{=} (\alpha]_{\forall}\Gamma)$. The proof is analogous to the one from Section 5.1. As an example: consider the CTL* formula $\mathbf{E} \mathbf{F} g_1 \wedge \mathbf{E} \mathbf{F} g_2$ and its equivalent mu-calculus formula $(\mu Z. \mathbf{E} \mathbf{X} Z \vee g_1) \wedge$

$(\mu Z.\mathbf{EX} Z \vee g_2)$. Their bounded $\{g\}$ -relativization are $\mathbf{EF}(g \wedge g_1) \wedge \mathbf{EF}(g \wedge g_2)$ and $(\mu Z.\mathbf{EX}(g \wedge Z) \vee g_1) \wedge (\mu Z.\mathbf{EX}(g \wedge Z) \vee g_2)$ respectively. We define $\widehat{\cdot} : \text{GL} \rightarrow \text{QDCTL}^*$ by induction: atomic propositions and \top are left unchanged; formulas like $\neg\varphi$, $\varphi_1 \vee \varphi_2$ are dealt inductively, and

$$\widehat{\exists C.\theta} \stackrel{\text{def}}{=} \hat{\exists} f_C.(\widehat{\theta}) f_C$$

Clearly, the size of $\widehat{\varphi}$ is linear in the size of φ , for any $\varphi \in \text{GL}$. Also, since $\hat{\exists} f_C.\alpha \in \text{QD}_\mu$ is definable in QDCTL^* provided α is, the co-domain of $\widehat{\cdot}$ is indeed QDCTL^* . Let φ^S denote the set $\{s \in S \mid \mathcal{S}, s \models \varphi\}$.

Theorem 7. For any state formula $\varphi \in \text{GL}$, $\varphi^S = \llbracket \widehat{\varphi} \rrbracket_S$.

Proof. The proof is made by induction over φ . The non trivial case is $\exists C.\theta$. We reason by recurrence on $N \geq 1$ the number of occurrences of symbols \exists in φ .

(1) Assume $N = 1$. We use Proposition 8 below, and the fact that $(\widehat{\theta}) f_C = (\widehat{\theta}) f_C$ since $\widehat{\theta}$ is quantifier-free.

Proposition 8. Given a rooted CGS (\mathcal{S}, s) , $C \subseteq \mathbf{P}$, and (\mathcal{E}, r) a f_C -strategy from s , we have: for any $\alpha \in \text{QDCTL}^*$, $s' \in S$, $e \in E$, $(\mathcal{S}, s) \times (\mathcal{E}, r), (s', e) \models (\alpha|f_C)$ if and only if $\text{CGS}(f_C, \mathcal{S}, s), (s', e) \models \alpha$.

The proof of Proposition 8 is omitted. It consists in a simple induction over α , just as for Theorem 3.

(2) Assume $N > 1$. Remark first that Proposition 8 generalizes to non-closed formulas provided QDCTL^* is generalized to formulas with variables of Var . We rewrite α as $\alpha_0(Y_1, \dots, Y_N)$, where $\alpha_0(Y_1, \dots, Y_N)$ is a CTL^* -like formula and where the (fresh) variables $Y_i \in \text{Var}$ are interpreted according to $\text{val}(Y_i) = (\exists C_i.\alpha_i)^S$. Notice that the number N_i of \exists symbols in $\exists C_i.\alpha_i$ is strictly less than N . It is routine to terminate the proof by using the generalization of Proposition 8 and the induction hypothesis. \square

6 Automata constructions

We assume the reader is familiar with alternating parity tree automata, and their relationship with the mu-calculus and parity games (we refer to [3], [13], and [20]). In the following, we use 'automata' for 'alternating parity tree automata'.

6.1 Automata for QD_μ

The difficulty in building automata for QD_μ essentially comes from the interplay between fixed-point operators and quantifiers, which can be subtle, as they do not commute in general: consider the formulas $\alpha_\perp = \exists g.\nu Z.(\mathbf{A}\mathbf{X} Z \wedge g \wedge \mathbf{E}\mathbf{X} \neg g)$ and $\alpha_\top = \nu Z.(\mathbf{A}\mathbf{X} Z \wedge \exists g.g \wedge \mathbf{E}\mathbf{X} \neg g)$, interpreted on a single infinite path CGS. Whereas the interpretation of α_\perp is the empty set, the one of α_\top is the entire set of states. Theorem 9 below relies on a powerful construction which generalizes [17]. Existential quantification corresponds to the projection operation over non-deterministic automata [16]; by the *Simulation Theorem* [15], every alternating automaton is equivalent to a non-deterministic automaton, and the procedure is effective with one exponential blow-up in the size of the automaton. Fixed point operators of the propositional mu-calculus also have their counterpart in the automata-theoretic approach: as proposed by [3] [Chapter 7, 7.2], automata are equipped with variables; their semantics considers inputs of the form $((\mathcal{S}, s), \text{val})$, where (\mathcal{S}, s) is as usual a model, and $\text{val} : \text{Var} \rightarrow 2^S$ is a valuation to interpret the variables, in the same line we interpret non-closed formulas. This class of automata can be turned into a mu-calculus, where in particular fixed point operators apply. Given an automaton \mathcal{A} , one can effectively define the automaton $\mu Z.\mathcal{A}$, which semantics is as expected: for example, if \mathcal{A} denotes a non-closed formula $\exists \Gamma.\alpha(Z)$, where $Z \in \text{Var}$ occurs free in $\alpha(Z)$, the automaton $\mu Z.\mathcal{A}$ denotes the sentence $\mu Z.(\exists \Gamma.\alpha(Z))$. Informally, our construction consists in three main steps. First, we build the automaton for $\alpha(Z)$. Second, using the projection operation, we compute the automaton for $\exists \Gamma.\alpha(Z)$. Third, and finally, we build the automaton for $\mu Z.(\exists \Gamma.\alpha(Z))$. Notice that the automaton for $\alpha(Z)$ may not be non-deterministic in general, either because e.g. $\alpha(Z)$ is of the form $\neg\alpha'(Z)$, or of the form $\alpha_1(Z) \wedge \alpha_2(Z)$. Projecting to forget propositions of Γ may therefore require the preliminary application of the Simulation Theorem, entailing one exponential blow-up. We now turn to the formal argument.

Theorem 9. *Let $m \geq 1$. For any $\alpha \in QD_\mu$, write $\kappa \in \mathbb{N}$ for the maximal number of nested quantifiers in α . Then, there exists an alternating parity tree automaton \mathcal{A}_α^m with $\max(\kappa, 0)$ -EXPTIME($|\alpha|$) states and $\max(\kappa - 1, 0)$ -EXPTIME($|\alpha|$) priorities, which accepts exactly the models of α of branching degree m .*

Proof. Without loss of generality, we will assume α in positive guarded normal form, in the sense that (1) negation symbols are pushed innermost, possibly changing least fixed points into greatest fixed points, and existential quantifiers into universal ones, and (2) each occurrence of a variable Z is under the scope of some

EX or **AX** operator. Formulas of the form $\neg(\mathcal{P}\Gamma)$ are in normal form. The proof is conducted by induction on κ , and the construction proceeds in a bottom-up manner in the formula.

(1) Assume $\kappa = 0$, henceforth $\alpha = \beta \in D_\mu$, with possibly with free variables.

As β is in positive normal form, each variable $Z \in \text{Var}$ occurs positively. We define the set $\text{cl}(\beta)$ of sub-formulas of β inductively:

$$\begin{aligned} \text{cl}(Z) &= \{Z\}, & \text{cl}(g) &= \{g\}, \\ \text{cl}(\mathcal{P}\Gamma) &= \{\mathcal{P}\Gamma\} \cup \bigcup_{f \in \Gamma} \{f, \neg f\}, \\ \text{cl}(\neg \mathcal{P}\Gamma) &= \{\neg \mathcal{P}\Gamma, \mathcal{P}\Gamma\} \cup \bigcup_{f \in \Gamma} \{f, \neg f\}, \\ \text{cl}(\beta_1 \vee \beta_2) &= \{\beta_1 \vee \beta_2\} \cup \text{cl}(\beta_1) \cup \text{cl}(\beta_2), \\ \text{cl}(\beta_1 \wedge \beta_2) &= \{\beta_1 \wedge \beta_2\} \cup \text{cl}(\beta_1) \cup \text{cl}(\beta_2), \\ \text{cl}(\mathbf{EX} \beta) &= \{\mathbf{EX} \beta\} \cup \text{cl}(\beta), \\ \text{cl}(\mu Z. \beta(Z)) &= \{\mu Z. \beta(Z)\} \cup \text{cl}(\beta(Z)[\mu Z. \beta(Z)/Z]), \\ \text{cl}(\nu Z. \beta(Z)) &= \{\nu Z. \beta(Z)\} \cup \text{cl}(\beta(Z)[\nu Z. \beta(Z)/Z]). \end{aligned}$$

The set of states of \mathcal{A}_β^m is $\text{cl}(\beta)$ and we distinguish between the *concrete-states* as elements of $\text{cl}(\beta) \setminus \text{Var}$, and the *variable-states* as elements of $\text{cl}(\beta) \cap \text{Var}$.

Regarding the transition function, we follow [13]. However, we need to take variable-states into account. Let us write $\mathcal{B}^+([m] \times \text{cl}(\beta) \cup \text{Var})$ for the set of positive Boolean formulas over $([m] \times \text{cl}(\beta)) \cup \text{Var}$; typical elements of $\mathcal{B}^+([m] \times \text{cl}(\beta) \cup \text{Var})$ are θ_1, θ_2 . Preliminarily to the definition of the transition function, we introduce a function $\text{split} : \mathcal{B}^+([m] \times \text{cl}(\beta) \cup \text{Var}) \rightarrow \mathcal{B}^+([m] \times \text{cl}(\beta) \cup \text{Var})$ which we will use to avoid Boolean combinations of sub-formulas as states: for any $x \in [m]$ and any $\beta' \in \text{cl}(\beta)$ of the form $g, \neg g, \mathbf{EX} \beta_1, \mathbf{AX} \beta_1, \mu Z. \beta_1, \nu Z. \beta_1$, we set $\text{split}((x, \beta')) = (x, \beta')$; also $\text{split}(x, \beta_1 \wedge \beta_2) = \text{split}(x, \beta_1) \wedge \text{split}(x, \beta_2)$ and $\text{split}(x, \beta_1 \vee \beta_2) = \text{split}(x, \beta_1) \vee \text{split}(x, \beta_2)$; finally, for all $Z \in \text{Var}$, $\text{split}(Z) = Z$, and for all $\theta_1, \theta_2 \in \mathcal{B}^+([m] \times \text{cl}(\beta))$, $\text{split}(\theta_1 \wedge \theta_2) = \text{split}(\theta_1) \wedge \text{split}(\theta_2)$, $\text{split}(\theta_1 \vee \theta_2) = \text{split}(\theta_1) \vee \text{split}(\theta_2)$.

The transition function of \mathcal{A}_β^m is a mapping $t : \text{cl}(\beta) \times \mathcal{P}(\bar{\Lambda}) \rightarrow \mathcal{B}^+([m] \times \text{cl}(\beta) \cup \text{Var})$. It follows the same lines as for the classic mu-calculus, where additionally free variables and decision modalities are taken into account. Let $l \in 2^{\bar{\Lambda}}$, and let $k \leq m$; when $\mathcal{D} \subseteq [k]$, write $\bar{\mathcal{D}}$ for $[k] \setminus \mathcal{D}$. Recall that in the context of formulas, Γ denotes $\bigwedge_{f \in \Gamma} f$. We define:

- $t(g, l, k) = \text{true}$ if $g \in l$, false otherwise.
- $t(\neg g, l, k) = \text{false}$ if $g \in l$, true otherwise.
- $t(Z, l, k) = Z$.
- $t(\bigoplus \Gamma, l, k) = \bigvee_{\{(\mathcal{D}, p) \in l\}} B(\mathcal{D}, \Gamma)$, where

$$B(\mathcal{D}, \Gamma) = \bigwedge_{x \in \mathcal{D}} \text{split}((x, \Gamma)) \wedge \bigwedge_{y \in \bar{\mathcal{D}}} \text{split}(((y, \neg \Gamma))).$$

- $t(\neg \bigoplus \Gamma, l, k) = \bigwedge_{\{(\mathcal{D}, p) \in l\}} \bar{B}(\mathcal{D}, \Gamma)$ where

$$\bar{B}(\mathcal{D}, \Gamma) = \bigvee_{x \in \mathcal{D}} \text{split}((x, \neg \Gamma)) \vee \bigvee_{y \in \bar{\mathcal{D}}} \text{split}((y, \Gamma))$$

- $t(\beta_1 \wedge \beta_2, l, k) = \text{split}(t(\beta_1, l, k) \wedge t(\beta_2, l, k))$.
- $t(\beta_1 \vee \beta_2, l, k) = \text{split}(t(\beta_1, l, k) \vee t(\beta_2, l, k))$.
- $t(\mathbf{EX} \beta, l, k) = \text{split}(\bigvee_{x \in [k]} (x, \beta))$.
- $t(\mathbf{AX} \beta, l, k) = \text{split}(\bigwedge_{x \in [k]} (x, \beta))$.
- $t(\mu Z. \beta(Z), l, k) = \text{split}(t(\beta(\mu Z. \beta(Z))), l, k)$.
- $t(\nu Z. \beta(Z), l, k) = \text{split}(t(\beta(\nu Z. \beta(Z))), l, k)$.

For all $\beta' \in \text{cl}(\beta)$, and $l \in 2^{\bar{\Lambda}}$, we define $t(\beta', l)$ as $\bigvee_{k \in [m]} t(\beta', l, k)$.

The initial state of \mathcal{A}_β^m is β , and concrete-states are assigned priorities in a standard manner as for the mu-calculus, see for example [7, Chapter 10]; it is known that the number of priorities corresponds to the number of fixed-point alternations in β , hence it is linear in the size of the formula. On the contrary, variable-states do not have priority as they are meant to be interpreted by valuations.

We recall the principles for the acceptance by the automaton \mathcal{A}_β^m of a given input (\mathcal{S}, s_0) with the valuation $\text{val} : \text{Var} \rightarrow 2^S$: it is based on a two-players parity game played over $\text{cl}(\beta) \times S$, and starting in (β, s_0) ; call Player 0 and Player 1 the players. Given $s \in S$, let us write $l(s)$ for $\{\bar{g} \in \bar{\Lambda} \mid s \in \bar{\lambda}(g)\}$. In position (β', s) , Player 0 chooses a subset χ of $([m] \times \text{cl}(\beta)) \cup \text{Var}$ that satisfies $t(\beta', l(s))$, where for each $Z \in \chi$ it must be the case that $s \in \text{val}(Z)$. If Player 0 cannot make such a choice, then Player 1 wins. If $\chi \subseteq \text{Var}$, then Player 0 wins. Otherwise, Player 1

chooses an element of $(x, \beta^m) \in \chi \cap [m] \times \text{cl}(\beta)$ and the game continues in position $(\beta^m, (s)_x)$ (where $(s)_x$ is the successor of s along the direction x). An infinite play is winning for Player 0 whenever it satisfies the parity condition inferred by the priorities of the concrete-state in the first component of the positions.

By the above, \mathcal{A}_β^m has a number of states and a number of priorities which are linear in the size of β . Notice that it is exponential in m , which we assume fixed. Correctness of the construction relies on a standard arguments, see for example [7, Chapter 10].

(2) Assume $\kappa > 0$. We proceed by inducting over the structure of α . Constructions for $\exists\Gamma.\alpha$ consist in building a non-deterministic automaton \mathcal{A}' equivalent to \mathcal{A}_α^m and to project it in order to abstract from propositions in Γ ; \mathcal{A}' has exponentially more states than \mathcal{A}_α^m , and its number of priorities is linear in the size of \mathcal{A}_α^m . It is important to note that by [3, Chapter 9], the Simulation Theorem remains valid for the class of automata with variables, and that \mathcal{A}' has the same variable-states as \mathcal{A}_α^m . Projecting \mathcal{A}' is $O(1)$. The case of $\forall\Gamma.\alpha$ is very similar: since $\forall\Gamma.\alpha$ can be rewritten as $\neg\exists\Gamma.\neg\alpha$, we can use the same techniques but by invoking twice the complementation operation for (alternating) automata which is linear time.

Constructions for $\mu Z.\alpha$ does not induce any particular cost: the variable-state Z is made a concrete-state with priority $2\lfloor \frac{M}{2} \rfloor + 1$, where M is the maximum priority in \mathcal{A}_α^m . Constructions for $\nu Z.\alpha$ is similar but using priority $2\lceil \frac{M}{2} \rceil$. Also, constructions for formulas of the form $\neg\alpha$, $\alpha_1 \vee \alpha_2$, $\alpha_1 \wedge \alpha_2$, **EX** α , **AX** α are not difficult, and the automata constructions corresponding to these logical operators may increase in the size of the automata only with polynomial bounds; we refer to [3] for the formal definitions. \square

Theorem 9 gives an upper bound for the complexity of the model-checking problem of QD_μ formulas: in the light of the complexity resulting from solving two-players parity games [9], it is non-elementary, but polynomial in the size of the structure \mathcal{S} (that is $|S| + |\bar{A}|$). If we fix the maximal number κ of $\hat{\exists}$ symbols in the formulas, the model-checking is κ -EXPTIME in the size of the formula. It is κ -EXPTIME-complete by [17] whose logic is contained in QD_μ .

6.2 Automata for alternating time logics

Although our translation $\hat{\cdot}$ of AMC or GL into QD_μ may generate an arbitrary large number of nested symbols $\hat{\exists}$, the corresponding automata nevertheless remain

small, if their construction is carefully conducted; applying Theorem 9 is actually avoidable: because formulas $\widehat{\varphi}$ are obtained by bounded relativizations of QD_μ formulas, a quantified proposition never occurs in strict quantified sub-formulas. This observation enables us to construct automata in a top-down manner, as opposed to the bottom-up procedure of Theorem 9. Consequently, the Simulation Theorem applies independently on sub-formulas, leading to a single exponential blow-up in the size of $\widehat{\varphi}$. Notice however that for GL, $\widehat{\varphi}$ is in QDCTL* which causes an additional exponential blow-up in the construction.

Let us consider formulas $\widehat{\varphi} \in QD_\mu$ when $\varphi \in \text{AMC}$. We construct an automaton $\mathcal{B}_{\widehat{\varphi}}^m$ equivalent to $\mathcal{A}_{\widehat{\varphi}}^m$ (Theorem 9), whose size is only exponential in the size of $\widehat{\varphi}$, hence in the size of φ , as opposed to $\mathcal{A}_{\widehat{\varphi}}^m$. The construction is done by induction on the structure of φ .

(1) If $\varphi = \langle\langle C \rangle\rangle \circ \varphi'$, then $\widehat{\varphi} = \exists f_C. (\mathbf{AX} \widehat{\varphi}]. f_C)$ can be rewritten $\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})[\sigma]$, where $\beta(\mathbf{Z}, \mathbf{Y}) \in D_\mu$ with free variables in $\mathbf{Z} \cup \mathbf{Y} \subseteq \text{Var}$; \mathbf{Z} contains the variables that are meant to be bound by fixed point operators, whereas \mathbf{Y} contains the variables that are mapped onto sub-formulas $\widehat{\varphi}_1$ by the substitution σ , where φ_1 has the form $\langle\langle C' \rangle\rangle \circ \varphi''$ and is a maximal state sub-formula of φ . By Theorem 9, build $\mathcal{A}_{\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})}^m$ the automaton for $\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})$; its number of states is in $O(2^{|\varphi|})$ and its number of priorities is in $O(|\varphi|)$, since $|\beta(\mathbf{Z}, \mathbf{Y})| < |\varphi|$, $\mathbf{AG}(\odot f^c)$ is independent of φ , and we have applied the Simulation Theorem once. Let ρ be the following valuation: each $Y \in \mathbf{Y}$ is mapped onto the automaton $\mathcal{B}_{[\sigma(Y)]}^m$, which exists by induction hypothesis and whose size is in $\text{EXPTIME}(|\sigma(Y)|)$; $Z \in \mathbf{Z}$ is left unchanged. We apply the composition operation of [3, Chapter 7] to $\mathcal{A}_{\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})}^m$, by setting $\mathcal{B}_{\langle\langle C \rangle\rangle \circ \varphi}^m$ equals $\mathcal{A}_{\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})}^m[\rho]$; notice that the size of $\mathcal{A}_{\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})}^m[\rho]$ is polynomial in $(\sum_{Y \in \mathbf{Y}} |\rho(Y)|) \cdot |\mathcal{A}_{\exists f_C. \beta(\mathbf{Z}, \mathbf{Y})}^m|$.

Lemma 10. *The models of $\mathcal{A}_{\langle\langle C \rangle\rangle \circ \varphi}^m$ (from Theorem 9) and $\mathcal{B}_{\langle\langle C \rangle\rangle \circ \varphi}^m$ coincide.*

Due to lack of space, we omit this proof which results from a direct application of [3], and from the crucial fact that the alphabets of the sub-automata $\rho(Y) = \mathcal{B}_{[\sigma(Y)]}^m$ ($Y \in \mathbf{Y}$) do not contain any propositions of f_C . By construction, the size of $\mathcal{B}_{\langle\langle C \rangle\rangle \circ \varphi}^m$ is in $\text{EXPTIME}(|\langle\langle C \rangle\rangle \circ \varphi|)$.

(2) For the formulas of the form $g, \top, \neg\varphi, \varphi_1 \vee \varphi_2, Z, \mu Z. \varphi(Z)$, it is routine to run an inductive reasoning, where in the later case, the construction for $\mu Z. \mathcal{B}^m$, where \mathcal{B}^m is the automaton for $\widehat{\varphi}(Z) \in D_\mu$ given by Theorem 9, remains linear in the size of \mathcal{B}^m .

Regarding the logic GL, let us only explain the construction for a formula $\exists C.\varphi$. A formula $\widehat{\exists C}.\varphi = \hat{\exists}f_C.(\hat{\varphi})f_C$ can be rewritten as $\hat{\exists}f_C.\theta(Z, Y)[\sigma]$, where θ is a CTL* formula with decision modalities, and with possibly free variables. Following the same lines as for AMC, the blow-up in the construction arises when building the non-deterministic automaton for $\theta(Z, Y)$, before applying the projection: first, by [13], it is possible to build an alternating parity automaton of size $\text{EXPTIME}(|\theta(Z, Y)|)$ (and 3 priorities, as a special case of hesitant alternating automata) equivalent to $\theta(Z, Y)$; second, this automaton is conjunctively combined with the one for $\text{AG}(\odot f^c)$; third, the Simulation Theorem is applied obtaining the 2-EXPTIME blow-up in the size of $\theta(Z, Y)$.

References

- [1] B. Thomas Adler, Luca de Alfaro, Leandro Dias da Silva, Marco Faella, Axel Legay, Vishwanath Raman, and Pritam Roy. Ticc: A tool for interface compatibility and composition. In Thomas Ball and Robert B. Jones, editors, *CAV*, volume 4144 of *Lecture Notes in Computer Science*, pages 59–62. Springer, 2006.
- [2] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.
- [3] A. Arnold and D. Niwinski. *Rudiments of mu-calculus*. North-Holland, 2001.
- [4] M. Dam. CTL* and ECTL* as fragments of the modal μ -calculus. *Theoretical Computer Science*, 126(1):77–96, 1994.
- [5] E. A. Emerson and A. P. Sistla. Deciding full branching time logic. *Information and Control*, 61:175–201, 1984.
- [6] V. Goranko and G. van Drimmelen. Complete axiomatization and decidability of the alternating-time temporal logic. *Theoretical Computer Science*, 353(1):93–117, 2006.
- [7] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

- [8] Thomas A. Henzinger, Orna Kupferman, and Shaz Qadeer. From *re*-historic to *ost*-modern symbolic model checking. In Alan J. Hu and Moshe Y. Vardi, editors, *CAV*, volume 1427 of *Lecture Notes in Computer Science*, pages 195–206. Springer, 1998.
- [9] M. Jurdziński. Deciding the winner in parity games is in $UP \cap co-UP$. *Information Processing Letters*, 68(3):119–124, November 1998.
- [10] M. Kacprzak and W. Penczek. Unbounded model checking for alternating-time temporal logic. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 646–653, Washington, DC, USA, 2004. IEEE Computer Society.
- [11] D. Kozen. Results on the propositional μ -calculus. *Theoretical Computer Science*, 27(3):333–354, 1983.
- [12] O. Kupferman and M. Y. Vardi. Module checking revisited. In *Computer Aided Verification, Proc. 9th Int. Conference*, volume 1254 of *Lecture Notes in Computer Science*, pages 36–47. Springer-Verlag, 1997.
- [13] 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.
- [14] François Laroussinie, Nicolas Markey, and Ghassan Oreiby. On the expressiveness and complexity of ATL. In Helmut Seidl, editor, *Proceedings of the 10th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'07)*, Lecture Notes in Computer Science, Braga, Portugal, March 2007. Springer. To appear.
- [15] David E. Muller and Paul E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1–2):69–107, 17 April 1995.
- [16] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.
- [17] Stéphane Riedweg and Sophie Pinchinat. Quantified mu-calculus for control synthesis. In Branislav Rován and Peter Vojtás, editors, *MFCS*, volume 2747 of *Lecture Notes in Computer Science*, pages 642–651. Springer, 2003.

- [18] Sven Schewe and Bernd Finkbeiner. Satisfiability and finite model property for the alternating-time μ -calculus. In *15th Annual Conference on Computer Science Logic (CSL 2006)*. Springer Verlag, 2006.
- [19] Govert van Drimmelen. Satisfiability in alternating-time temporal logic. In *LICS*, pages 208–217. IEEE Computer Society, 2003.
- [20] Thomas Wilke. Alternating tree automata, parity games, and modal μ -calculus. *Bull. Soc. Math. Belg.*, 8(2), May 2001.