The variable hierarchy of the μ-calculus is strict

Dietmar Berwanger[1], Erich Grädel[1], and Giacomo Lenzi[2]

[1] Mathematische Grundlagen der Informatik, RWTH Aachen
52056 Aachen, Germany
{berwanger, graedel}@informatik.rwth-aachen.de

[2] Dipartimento di Matematica, Università di Pisa
via Buonarroti 2, 56127 Pisa, Italy
lenzi@mail.dm.unipi.it

Abstract

Most of the logics commonly used in verification, such as LTL, CTL, CTL*, and PDL can be embedded into the two-variable fragment of the μ-calculus. It is also known that properties occurring at arbitrarily high levels of the alternation hierarchy can be formalised using only two variables. This raises the question whether the number of fixed-point variables in μ-formulae can be bounded in general.

We answer this question negatively, and prove that the variable-hierarchy of the μ-calculus is semantically strict. For any k, we provide examples of formulae with k variables that are not equivalent to any formula with fewer variables. In particular, this implies that Parikh's Game Logic is less expressive than the μ-calculus, thus resolving an open issue raised by Parikh in 1983.

1 Introduction

The μ-calculus L_μ extends basic modal logic by adding monadic variables bound to least and greatest fixed points of definable operators. This provides a notion of recursion which invests the logic with very high expressive power. On the other side, the monadic fixed points import considerable conceptual complexity. One well-studied measure of complexity for L_μ is the alternation depth, that is, the number of genuine alternations between greatest and least fixed-point operators occurring in a formula. It has been shown by Bradfield [6] that the alternation hierarchy of the μ-calculus is strict. Variants of this result have also been proved by Lenzi [14] and Arnold [1].

Interestingly, most of the formalisms commonly used for process description allow translations into low levels of the L_μ alternation hierarchy. On its first level this hierarchy already captures, for instance, PDL as well as CTL, while their expressive extensions ΔPDL and CTL* do not exceed the

second level. Still, the low levels of this hierarchy do not exhaust the significant properties expressible in L_μ. A comprehensive example of formulae distributed over all levels of the alternation hierarchy is provided by parity games. Thus, strictly on level n, there is a formula stating that the first player has a winning strategy in parity games with n priorities.

By reusing fixed-point variables several times it is possible to write many L_μ-formulae, even with highly nested fixed-point definitions, using only very few variables. This is actually the case for Parikh's Game Logic [16] which subsumes the aforementioned formalisms, ΔPDL and CTL*, but also contains formulae describing the winning position in parity games, for any number of priorities [3], thus intersecting non-trivially with all levels of the alternation hierarchy.

In this context, the question arises whether a finite number of variables is sufficient to express all L_μ-definable properties. We answer this question negatively by proving that the variable hierarchy of the μ-calculus is strict.

To witness this, we consider L_μ-formulae that describe a given finite Kripke structure. Specifically, we identify a parameter of directed graphs, called *entanglement*, which measures how many variables are sufficient to describe, up to bisimulation, any Kripke structure over that graph. We prove that every directed graph of entanglement k can be turned into a Kripke structure that cannot be described with less than k variables.

Our proof of the hierarchy theorem consists of two main parts. First, we establish the strictness of the hierarchy for the case of existential formulae, i.e., formulae built without using universal modalities. It is known that these characterise precisely the L_μ-definable properties which are preserved under simulation. We show that no existential formula with less than k variables can define the simulation type of a Kripke structure of entanglement k, under a particular labelling.

In the second part, we prove a preservation theorem stating that every formula defining the simulation type of a strongly connected structure can be transformed into an existential formula without increasing the number of variables. Thus the strictness of the variable hierarchy for the full μ-calculus follows from its strictness in the existential case.

Besides revealing a new aspect of the rich inner structure of the μ-calculus, this result settles an open question formulated by Parikh in [15] regarding the expressive power of Game Logic. When interpreted on Kripke structures, this logic can be translated into the two-variable fragment of L_μ, but it was unknown, up to now, whether the inclusion in L_μ was proper. The strictness of the variable hierarchy implies that already the three-variable fragment of L_μ is more expressive than GL.

Here is an overview of the article. In Section 2 we define the μ-calculus,

explain parity games and introduce the variable hierarchy. Section 3 is dedicated to the notion of entanglement. After defining this measure, we show that, if a Kripke structures has entanglement k, its bisimulation type – and also its simulation type – can be described in L_μ using k variables. Further, we provide a characterisation of entanglement in terms of a game between a thief and several detectives. With the aid of this characterisation, we establish the strictness of the variable hierarchy for the existential fragment of L_μ in Section 4. The technically most involved part of our presentation is the preservation theorem proved in Section 5, which allows us to generalise the strictness of the variable hierarchy for the existential fragment to the full μ-calculus. The hierarchy theorem is stated in Section 6 together with concrete examples of separating formulae over a language of two actions only. We conclude with a discussion on the relation between Parikh's Game Logic with the alternation and variable hierarchy of L_μ.

2 The modal μ-calculus

2.1 Syntax and semantics

For a set ACT of actions, a set PROP of atomic propositions, and a set VAR of monadic variables, the formulae of L_μ are defined by the grammar

$$\varphi ::= \bot \mid \top \mid p \mid \neg p \mid X \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \langle a \rangle \varphi \mid [a]\varphi \mid \mu X.\varphi \mid \nu X.\varphi$$

where $p \in \text{PROP}$, $a \in \text{ACT}$, and $X \in \text{VAR}$.

The *syntax tree* $\mathcal{T}_\psi$ of a formula $\psi \in \mathrm{L}_\mu$ is defined inductively, by associating atomic formulae (propositions and their negations, variables, and the constants $\bot$, $\top$) to isolated nodes, unary constructs $\langle a \rangle \varphi$, $[a]\varphi$, $\mu X.\varphi$, $\nu X.\varphi$ to trees with a single immediate subtree $\mathcal{T}_\varphi$, and binary constructs $\varphi_1 \vee \varphi_2$, $\varphi_1 \wedge \varphi_2$ to trees with two immediate subtrees $\mathcal{T}_{\varphi_1}$ and $\mathcal{T}_{\varphi_2}$. If we introduce, for every leaf corresponding to a variable occurrence X, a link to (the unique node which corresponds) to its binding definition $\mu X.\varphi$ or $\nu X.\varphi$, we obtain an operational representation of ψ as a tree with back edges, which we call its *syntax graph* $\mathcal{S}_\psi$.

The number of distinct fixed-point variables appearing in an L_μ-formula induces the following syntactic hierarchy.

Definition 1. For any $k \in \mathbb{N}$, the *k-variable fragment* $\mathrm{L}_\mu[k]$ of the μ-calculus is the set of formulae $\psi \in \mathrm{L}_\mu$ that contain at most k distinct variables.

Formulae of L_μ are evaluated on Kripke structures at a particular state. A Kripke structure for ACT and PROP is a structure

$$\mathcal{K} = \big(V, (E_a)_{a\in\mathrm{ACT}}, (V_p)_{p\in\mathrm{PROP}}\big)$$

with universe V (whose elements are called *states*), binary *transition* relations $E_a \subseteq V \times V$ for each $a \in \mathrm{ACT}$, and monadic relations $V_p \subseteq V$ for each atomic proposition $p \in \mathrm{PROP}$. In general, we consider *rooted* structures $\mathcal{K}, u$, and assume that all states are reachable from the designated state $u \in V$. On trees, we may omit to mention the root explicitly.

Given a formula ψ and a structure $\mathcal{K}$ with a state u, we write $\mathcal{K}, u \models \psi$ to express that ψ holds in $\mathcal{K}$ at state u. The set of states $u \in V$ such that $\mathcal{K}, u \models \psi$ is denoted by $[\![\psi]\!]^{\mathcal{K}}$. Thus, $[\![\bot]\!]^{\mathcal{K}} := \emptyset$ and $[\![\top]\!]^{\mathcal{K}} := V$. For atomic propositions $p \in \mathrm{PROP}$, we have $[\![p]\!]^{\mathcal{K}} := V_p$, respectively $[\![\neg p]\!]^{\mathcal{K}} := V \setminus V_p$. The propositional operators are interpreted as usual,

$$[\![\varphi_1 \vee \varphi_2]\!]^{\mathcal{K}} := [\![\varphi_1]\!]^{\mathcal{K}} \cup [\![\varphi_2]\!]^{\mathcal{K}} \quad \text{and} \quad [\![\varphi_1 \wedge \varphi_2]\!]^{\mathcal{K}} := [\![\varphi_1]\!]^{\mathcal{K}} \cap [\![\varphi_2]\!]^{\mathcal{K}}.$$

The meaning of the modal operators is given by:

$$[\![\langle a\rangle\varphi]\!]^{\mathcal{K}} := \{\, v \mid \text{there exists } w \text{ such that } (v,w) \in E_a \text{ and } w \in [\![\varphi]\!]^{\mathcal{K}} \,\},$$
$$[\![[a]\varphi]\!]^{\mathcal{K}} := \{\, v \mid \text{for all } w \text{ such that } (v,w) \in E_a, \text{ we have } w \in [\![\varphi]\!]^{\mathcal{K}} \,\}.$$

To understand the semantics of fixed-point operators, note that a formula $\varphi(X)$ with a monadic variable X defines on every Kripke structure $\mathcal{K}$ (providing interpretations for all free variables other than X occurring in φ) an operator $\varphi^{\mathcal{K}} : \mathscr{P}(V) \to \mathscr{P}(V)$ assigning to every set $X \subseteq V$ the set $\varphi^{\mathcal{K}}(X) := [\![\varphi]\!]^{\mathcal{K},X} = \{v \in V : (\mathcal{K}, X), v \models \varphi\}$. As X occurs only positively in φ, the operator $\varphi^{\mathcal{K}}$ is *monotone* for every $\mathcal{K}$, i.e., $X \subseteq X'$ implies that $\varphi^{\mathcal{K}}(X) \subseteq \varphi^{\mathcal{K}}(X')$. Therefore, by a well-known theorem due to Knaster and Tarski, $\varphi^{\mathcal{K}}$ has a least fixed point $\mathrm{lfp}(\varphi^{\mathcal{K}})$ and a greatest fixed point $\mathrm{gfp}(\varphi^{\mathcal{K}})$. Now we put

$$[\![\mu X.\varphi]\!]^{\mathcal{K}} := \mathrm{lfp}(\varphi^{\mathcal{K}}) \text{ and } [\![\nu X.\varphi]\!]^{\mathcal{K}} := \mathrm{gfp}(\varphi^{\mathcal{K}}).$$

Least and greatest fixed points can also be constructed inductively. Given a formula $\nu X.\varphi$, we define for each ordinal α, the stage X^α of the gfp-induction of $\varphi^{\mathcal{K}}$ by $X^0 := V$, $X^{\alpha+1} := [\![\varphi]\!]^{(\mathcal{K},X^\alpha)}$, and $X^\alpha := \bigcap_{\beta<\alpha} X^\beta$ if α is a limit ordinal. Due to monotonicity, the stages of the gfp-induction decrease until a fixed point is reached. By ordinal induction, one easily proves that this inductively constructed fixed point coincides with the greatest fixed point. The *finite approximants* of a formula $\nu X.\varphi$ are defined by $\varphi_0 := \top$ and

$\varphi_{n+1} = \varphi[X/\varphi_n]$ (the formula obtained by replacing every free occurrence of X in φ, by φ_n). For least fixed points the situation is dual. Obviously, $\nu X.\varphi$ implies φ_n for all n, and on finite Kripke structures also the converse holds: If $\mathcal{K}, v \models \varphi_n$ for all n, then also $\mathcal{K}, v \models \nu X.\varphi$.

Simultaneous Fixed Points. There is a variant of L_μ that admits simultaneous fixed points of several formulae. This does not increase the expressive power but allows more transparent formalisations. The mechanism for building simultaneous fixed-point formulae is the following. Given formulae $\varphi_1, \ldots, \varphi_n$ and variables $X_1, \ldots, X_n$, we can write an *equational system* $S := \{X_1 = \varphi_1, \ldots, X_n = \varphi_n\}$ and build formulae $(\mu X_i : S)$ and $(\nu X_i : S)$. On every structure $\mathcal{K}$, the system S defines an operator $S^{\mathcal{K}}$ mapping an n-tuple $\bar{X} = (X_1, \ldots, X_n)$ of sets of states to $S_1^{\mathcal{K}}(\bar{X}), \ldots, S_n^{\mathcal{K}}(\bar{X})$ so that, for each i we have: $S_i^{\mathcal{K}}(\bar{X}) := [\![\varphi_i]\!]^{(\mathcal{K},\bar{X})}$. As $S^{\mathcal{K}}$ is monotone, it has extremal fixed points $\mathrm{lfp}(S) = (X_1^\mu, \ldots, X_n^\mu)$ respectively $\mathrm{gfp}(S) = (X_1^\nu \ldots, X_n^\nu)$, and we set $[\![(\mu X_i : S)]\!]^{\mathcal{K}} := X_i^\mu$ and $[\![(\nu X_i : S)]\!]^{\mathcal{K}} := X_i^\nu$.

It is known that simultaneous least fixed points can be eliminated in favour of nested individual fixed points.

Proposition 2 ([2])**.** *Every formula in* L_μ *with simultaneous fixed points can be translated into an equivalent formula in plain* L_μ *without increasing the number of variables.*

2.2 Simulation and bisimulation

Definition 3. A *simulation* from a structure $\mathcal{K}$ to a structure $\mathcal{K}'$ is a relation $Z \subseteq V \times V'$ respecting the atomic propositions $p \in \textsc{prop}$, in the sense that $\mathcal{K}, v \models p$ iff $\mathcal{K}', v' \models p$, for any $(v, v') \in Z$, which satisfies the following condition:

> For all $(v, v') \in Z$, $a \in \textsc{act}$, and every w such that $(v, w) \in E_a$, there exists a $w' \in V'$ such that $(v', w') \in E'_a$ and $(w, w') \in Z$.

We say that $\mathcal{K}', u'$ *simulates* $\mathcal{K}, u$ and write $\mathcal{K}, u \preccurlyeq \mathcal{K}', u'$, if there is a simulation from $\mathcal{K}$ to $\mathcal{K}'$ that contains (u, u').

An L_μ-formula in which no universal modality $[a]\varphi$ occurs is called *existential*. The validity of any existential L_μ-formula ψ is preserved under simulation, i.e., $\mathcal{K}, u \models \psi$ and $\mathcal{K}, u \preccurlyeq \mathcal{K}', u'$ implies $\mathcal{K}', u' \models \psi$.

As a modal logic, the μ-calculus distinguishes between Kripke structures only up to behavioural equivalence, captured by the notion of bisimulation.

Definition 4. A *bisimulation* between two Kripke structures $\mathcal{K}$ and $\mathcal{K}'$ is a simulation Z from $\mathcal{K}$ to $\mathcal{K}'$ whose inverse Z^{-1} is a simulation from $\mathcal{K}'$ to $\mathcal{K}$. We say that the structures $\mathcal{K}, u$ and $\mathcal{K}', u'$ are *bisimilar*, and write $\mathcal{K}, u \sim \mathcal{K}', u'$, if between them there is a bisimulation that contains (u, u').

An important model-theoretic feature of modal logics is the *tree model property* meaning that every satisfiable formula is satisfiable in a tree. This is a straightforward consequence of bisimulation invariance, since $\mathcal{K}, u$ is bisimilar to its tree unravelling.

Definition 5. The *unravelling* $\mathcal{T}(\mathcal{K}, u)$ of a Kripke structure $\mathcal{K}$ from a node u is the tree of all paths through $\mathcal{K}$ that start at u. More formally,

- the domain of $\mathcal{T}(\mathcal{K}, u)$ is the set $V^{\mathcal{T}}$ consisting of all sequences

  $$\pi = v_0 a_1 v_1 a_2 \cdots v_{r-1} a_r v_r$$

  where $v_i \in V$ and $a_i \in \textsc{act}$, such that $v_0 = u$ and $(v_{i-1}, v_i) \in E_{a_i}$;
- an atomic proposition $p \in \textsc{prop}$ is true at $v_0 a_1 v_1 a_2 \dots v_{r-1} a_r v_r$ in $\mathcal{T}(\mathcal{K}, u)$ if, and only if, it is true at v_r in $\mathcal{K}$;
- for all actions a, the relation $E_a^{\mathcal{T}}$ contains the pairs $(\pi, \pi a v)$ in $V^{\mathcal{T}} \times V^{\mathcal{T}}$.

Obviously, the natural projection $p\colon \mathcal{T}(\mathcal{K}, u) \to \mathcal{K}, u$ which sends every sequence $\pi = v_0 a_1 v_1 a_2 \dots v_{r-1} a_r v_r \in V^{\mathcal{T}}$ to its last node v_r defines a bisimulation between $\mathcal{T}(\mathcal{K}, u)$ and $\mathcal{K}, u$.

Another significant feature of L_μ is its *finite model property.*

Theorem 6 ([12])**.** *Every satisfiable* L_μ*-formula has a finite model.*

Since the unravelling of a finite model is a finitely branching tree, we obtain the following corollary.

Corollary 7. *Every satisfiable* L_μ*-formula holds in some finitely branching tree.*

For later use, we state a further consequence of the finite model property.

Corollary 8. *For* $\psi \in \mathrm{L}_\mu$*, let* $\psi[\nu^n/\nu]$ *denote the result of replacing every occurrence* $\nu X.\varphi$ *of a* ν*-predicate in* ψ *with its* n*-th approximant* φ_n*. Then, a formula* $\eta \in \mathrm{L}_\mu$ *implies* ψ *if, and only if,* η *implies* $\psi[\nu^n/\nu]$*, for each* n*.*

2.3 Model-checking games

The semantics of L_μ can also be described in terms of *parity games*. Such a game is given by a Kripke structure $\mathcal{G} = (V, V_0, E, \Omega)$, where V is a set of *positions* with a designated subset V_0, $E \subseteq V \times V$ is a transition relation, and $\Omega : V \to \mathbb{N}$ assigns to every position a *priority*. A *play* of $\mathcal{G}$ is a path $v_0, v_1, \ldots$ formed by two players starting from a given position v_0. If the current position v belongs to V_0, Player 0 chooses a move $(v, w) \in E$ and the play proceeds from w. Otherwise, his opponent, Player 1, chooses the move. When no moves are available at the current position, the player who has to choose loses. If this never occurs the play goes on infinitely and the winner is established by looking at the sequence $\Omega(v_0), \Omega(v_1), \ldots$ If the least priority appearing infinitely often in this sequence is even, Player 0 wins the play, otherwise Player 1 wins.

Let $V_1 := V \setminus V_0$ be the set of positions where Player 1 moves. A *memoryless strategy* for Player i in $\mathcal{G}$ is a function $\sigma : V_i \to V$ which indicates a choice $(v, \sigma(v)) \in E$ for every position $v \in V_i$. (It is called memoryless, because it does not depend on the history of the play, but only on the current position.) A strategy σ for Player i is a *winning* strategy if he wins every play in which he moves according to σ. We denote the game in which the moves of Player i are restricted to the strategy σ by $\mathcal{G}_\sigma$.

The Forgetful Determinacy Theorem states that parity games are always determined, and the winner has a memoryless winning strategy.

Theorem 9 (Forgetful Determinacy, [7])**.** *In any parity game, one of the players has a memoryless winning strategy.*

Given a Kripke structure $\mathcal{K}, u$ and a closed formula $\psi \in \mathrm{L}_\mu$, the model-checking game $\mathcal{G}(\mathcal{K}, \psi)$ is a parity game associated with the problem whether $\mathcal{K}, u \models \psi$.

The positions in the game $\mathcal{G}(\mathcal{K}, \psi)$ are pairs (v, φ) of states $v \in V$ and subformulae φ of ψ. To distinguish between different occurrences of the same subformula, we represent φ by the associated node in the syntax graph $\mathcal{S}_\psi$. The first player, here called Verifier, moves at the positions $(v, \varphi_1 \vee \varphi_2)$, $(v, \langle a \rangle \varphi)$, (v, p) with $v \notin p$, and $(v, \neg p)$ with $v \in p$ and his opponent, called Falsifier, moves from every other position. All plays start at position (u, ψ) and proceed as follows:

- no moves are possible from (v, α) where α is atomic or negated atomic;
- from $(v, \varphi_1 \vee \varphi_2)$ or $(v, \varphi_1 \wedge \varphi_2)$ there are two available moves leading to (v, φ_1) and (v, φ_2);

- from $(v, \langle a \rangle \varphi)$ or $(v, [a]\varphi)$ it is possible to move to any position (w, φ) where w is an a-successor of v;
- from each position $(v, \lambda X.\varphi(X))$ there is a move leading to $(v, \varphi(X))$, where λ stands for either μ or ν;
- from any occurrence of a fixed-point variable (v, X), the play moves to its binding definition $(v, \lambda X.\varphi(X))$.

Thus, a play proceeds along the paths in $\mathcal{K}$ and in the syntax tree of ψ, until it hits a fixed-point variable (which is a leaf in the syntax tree). There, the play resumes with the binding definition of the variable. When this occurs, we say that the variable (and its definition) is *regenerated*.

One technically useful property of fixed point formulae is guardedness. In terms of games this guarantees that between the binding definition of a variable and its regeneration we always have at least one modal move.

Definition 10. An L_μ-formula ψ is *guarded* if each path in the syntax tree of ψ from a fixed point definition $\lambda X.\varphi$ to an occurrence of X passes through a modality, $\langle a \rangle \eta$ or $[a]\eta$.

In [13], Kupferman, Vardi, and Wolper give a procedure to transform any L_μ-formula into an equivalent guarded formula. This procedure does not increase the number of variables and preserves existential formulae.

Proposition 11. *Every existential formula in* $\mathrm{L}_\mu[k]$ *is equivalent to a guarded existential formula in* $\mathrm{L}_\mu[k]$.

By repeatedly regenerating fixed-points, it may happen that neither Verifier nor Falsifier, ever gets stuck. To decide the winner of such plays, priorities have to be defined appropriately. The intuition is that, to establish the truth of a μ-formula, Verifier should regenerate it only finitely often whereas ν-formulae can be regenerated infinitely often. Of course the difficulty may be that μ- and ν-formulae are deeply nested and there are several fixed-point formulae that are regenerated infinitely often during a play. But it can be shown that among these, there is always an outermost one, which determines the winner: if it is a ν-formula Verifier wins, if it is a μ-formula, Falsifier wins. Hence, the priority labelling assigns even priorities to positions $(v, \nu X.\varphi)$ and odd priorities to positions $(v, \mu X.\varphi)$. Further, priorities respect dependencies. If $\nu Y.\varphi$ depends on $\mu X.\eta$ then priorities of positions $(v, \nu Y.\varphi)$ are higher than those of positions $(w, \mu X.\eta)$. The remaining positions receive priorities that are higher than those associated with fixed-point formulae. For details (which are not needed in this article), see [4].

Theorem 12 ([19]). *Verifier has a winning strategy in the model-checking game $\mathcal{G}(\mathcal{K}, \psi)$ from position (u, ψ) iff $\mathcal{K}, u \models \psi$.*

The meaning of a subformula within a formula is captured by the notion of closure.

Definition 13. Let $\psi \in \mathrm{L}_\mu$ be a formula without free variables. For each subformula φ in ψ, we define its *closure* $\mathrm{cl}_\psi(\varphi)$ as the formula obtained by replacing recursively every free occurrence of a variable in φ by its binding definition. For instance, if $\psi = \mu X.\nu Y \varphi(X, Y)$, then $\mathrm{cl}_\psi(\varphi) = \varphi(\psi, \nu Y \varphi(\psi, Y))$. For a precise definition, see [11]. By $\mathrm{cl}(\psi)$ we denote the set of closures of all subformulae in ψ.

The following property follows from Theorem 12.

Corollary 14. *Let $\mathcal{K}, u$ be a model of a formula $\psi \in \mathrm{L}_\mu$ and let σ be a winning strategy for Verifier in the associated model-checking game $\mathcal{G}(\mathcal{K}, \psi)$. Then, for every position (v, φ) reachable in $\mathcal{G}_\sigma$ from the initial position (u, ψ), we have $\mathcal{K}, v \models \mathrm{cl}_\psi(\varphi)$.*

A different way to define model-checking games for L_μ refers to the closure of subformulae rather than their occurrences [7, 13]. The games obtained in this way are equivalent to those introduced here – in fact, they are bisimilar. We may therefore use this alternative definition where a more semantic viewpoint is appropriate.

3 Defining bisimulation and simulation types of finite structures

Throughout this article, we are concerned with formulae that describe finite Kripke structures, more precisely, the bisimulation-invariant properties at a given state. In particular, we are interested in existential properties preserved under simulation.

Definition 15. Let $\mathcal{K}$ be a Kripke structure with a designated state u. A formula $\psi \in \mathrm{L}_\mu$ describes the *bisimulation type* of $\mathcal{K}, u$ if, for any structure $\mathcal{K}'$, we have $\mathcal{K}', u' \models \psi$ iff $\mathcal{K}, u \sim \mathcal{K}', u'$. Likewise, we say that ψ describes the *simulation type* of $\mathcal{K}, u$ if, for any Kripke structure $\mathcal{K}'$, we have $\mathcal{K}', u' \models \psi$ iff $\mathcal{K}, u \preccurlyeq \mathcal{K}', u'$.

A straightforward approach to describing a finite structure up to bisimulation consists in forming a system of simultaneous fixed points associated to

the individual states. Given a finite structure $\mathcal{K} = (V, (E_a)_{a\in\mathrm{ACT}}, (V_p)_{p\in\mathrm{PROP}})$, the atomic type of any node $v \in V$ is described by the formula

$$\alpha_v := \bigwedge_{\substack{p\in\mathrm{PROP}\\ v\in V_p}} p \;\wedge \bigwedge_{\substack{p\in\mathrm{PROP}\\ v\notin V_p}} \neg p.$$

Let S be the system defining, for every node $v \in V$, a proposition X_v via the equation

$$X_v = \alpha_v \wedge \bigwedge_{a\in\mathrm{ACT}} \Big(\bigwedge_{(v,w)\in E_a} \langle a\rangle X_w \wedge [a] \big(\bigvee_{(v,w)\in E_a} X_w \big) \Big).$$

It can be easily seen that on any Kripke structure $\mathcal{K}'$, the greatest solution of this system maps each variable X_v to the set $\{\, v' \in V' \mid \mathcal{K}, v \sim \mathcal{K}', v' \,\}$. Hence, the bisimulation type of $\mathcal{K}, u$ is described by $\nu X_u : S$.

If we restrict the definitions of X_v in S to their existential part,

$$X_v = \alpha_v \wedge \bigwedge_{\substack{a\in\mathrm{ACT}\\ (v,w)\in E_a}} \langle a\rangle X_w,$$

the greatest solution of the obtained system maps every variable X_v to the set $\{\, v' \in V' \mid \mathcal{K}, v \preccurlyeq \mathcal{K}', v' \,\}$ and thus $\nu X_u : S$ describes the simulation type of $\mathcal{K}, u$.

In general, however, this approach uses many more variables than needed. Any acyclic finite structure can be described already in basic modal logic. Typically, this is achieved by a formula whose syntax follows the finite tree obtained by unravelling the structure. We may proceed similarly to describe structures with cycles in the μ-calculus. Syntactically, L_μ-formulae are trees with back edges; each reference to a fixed-point variable semantically instantiates its binding definition, which occurred previously in the syntax tree. This allows us to describe any Kripke structure over a tree with back edges by associating greatest fixed-point variables to each node with incoming back edges. We obtain a defining formula following the tree edges, as in the acyclic case, additionally referencing for every back edge the fixed-point variable associated to its target. For instance, in a language without propositional constants, the simulation type of the structure from Figure 1 at state 0 is described by $\nu X.\big(\langle a\rangle X \wedge \langle b\rangle\langle b\rangle\langle a\rangle X\big)$.

Likewise, it is possible to characterise any finite structure $\mathcal{K}$ by describing a tree with back edges bisimilar to $\mathcal{K}$. Such a tree can be obtained, for example, by partially performing an unravelling of $\mathcal{K}$ as in Definition 5, but with the difference that, whenever a node that occurred previously on the

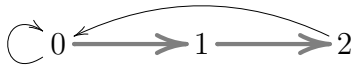


Figure 1: A simple structure with cycles, (a-transitions plain, b-transitions thicker)

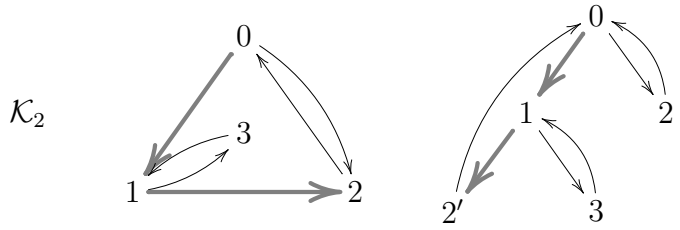


Figure 2: Viewing a structure as a tree with back edges

current path is reached, a back edge to this occurrence is added instead of creating a new copy. (Later on, we will formally introduce the notion of unravelling by generalising this procedure.) For the simulation type of the structure from Figure 2, we thus obtain the formula:

$$\nu X.\Big(\langle b\rangle \nu Y.\Big(\langle b\rangle\langle a\rangle X \wedge \langle a\rangle\langle a\rangle Y\Big) \wedge \langle a\rangle\langle a\rangle X\Big).$$

Notice, however, that a given structure may have several structurally different trees with back edges as bisimilar companions, leading to syntactically different descriptions. In particular, since we introduce variables for every node entered by a back edge, the number of variables involved in those descriptions may differ, as illustrated by the formulae obtained for the two bisimilar structures in Figure 3:

$$\begin{aligned}&\nu X.\langle b\rangle \nu Y.\langle b\rangle(\ \langle a\rangle X \wedge \langle a\rangle Y \wedge \nu Z.\langle a\rangle Z\ )\\ &\quad \equiv \langle b\rangle\langle b\rangle \nu X.\big(\langle a\rangle\langle b\rangle\langle b\rangle X \wedge \langle a\rangle\langle b\rangle X \wedge \langle a\rangle X\big).\end{aligned}$$

To control this phenomenon, we introduce a structural parameter for the complexity of finite directed graphs which measures to what extent the cycles of the graph are intertwined. The definition of this measure, called entanglement, is characterised in terms of a game similar in spirit to the cops and robbers games used to describe tree width, directed tree width, and hypertree width [18, 10, 8]. Nevertheless, there are significant differences between entanglement and the various incarnations of tree width.

As we will show, the entanglement of a finite Kripke structure provides an upper bound for the number of fixed-point variables needed to describe it

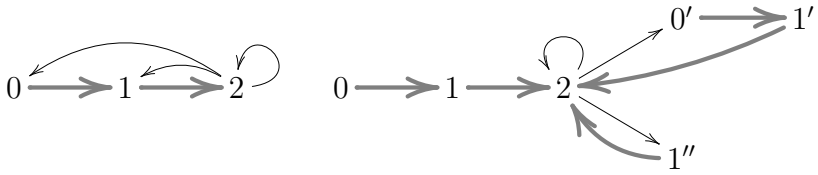


Figure 3: Bisimilar companions with different cyclic structure

up to bisimulation. In Section 4 and 6 we will further prove that this bound is tight, in a quite general sense.

3.1 Trees with back edges

Let $\mathcal{T} = (V, E)$ be a directed tree. We write $\preceq_E$ for the associated partial order on $\mathcal{T}$, i.e, the reflexive, transitive closure of E.

Definition 16. A directed graph $\mathcal{T} = (V, F)$ is a *tree with back edges* if there is a partition $F = E \cup B$ of the edges into tree edges and back edges such that (V, E) is a directed tree, and whenever $(u, v) \in B$, then $v \preceq_E u$.

The following observation shows that up to the choice of the root, the decomposition into tree edges and back edges is unique.

Lemma 17. *Let $\mathcal{T} = (V, F)$ be a tree with back edges and $v \in V$. Then there exists at most one decomposition $F = E \cup B$ into tree edges and back edges such that (V, E) is a tree with root v.*

Definition 18. Let $\mathcal{T} = (V, E, B)$ be a tree with back edges. The *feedback* of a node v of $\mathcal{T}$ is the number of ancestors of v that are the target of a back-edge from a descendant of v. The feedback of $\mathcal{T}$, denoted $\mathrm{fb}(\mathcal{T})$ is the maximal feedback of nodes in $\mathcal{T}$. More formally,

$$\mathrm{fb}(T) = \max_{v \in V} |\{u \in V : \exists w(u \preceq_E v \preceq_E w \wedge (w, u) \in B)\}|.$$

We call a node u *active* at the node v in $\mathcal{T}$, if u is the target of some back edge (w, u) such that $u \preceq_E v \preceq_E w$.

Notice, that the feedback of the syntax graph of a formula φ in L_μ equals the maximum number of variables occurring simultaneously in a subformula of φ. Thus, after appropriate renaming, the number of variables in a formula equals the feedback of its syntax graph.

We now show that the feedback of a tree with back edges provides an upper bound on the number of variables needed to describe any Kripke structure over this graph. Subsequently, we derive therefrom a corresponding bound for arbitrary Kripke structures via finite unravellings.

Lemma 19. *Let $\mathcal{T} = (V, E, B)$ be a tree with back edges of feedback k. Then there exists a partial labelling $i : V \to \{0, \dots, k-1\}$ assigning to every target u of a back edge an index $i(u)$ in such a way that no two nodes u, u' that are active at the same node v have the same index.*

Proof. The labelling i is constructed by a breadth-first traversal of $\mathcal{T}$. At each target u of a back-edge, we choose the least value in $\{0, \dots, k-1\}$ that is not the label of a node active at u. Such a value must exist, because the number of nodes active at u (including u) is at most k. Except for u, the index of these nodes is already defined since they are ancestors of u. In this way, we ensure that no conflict between simultaneosuly active nodes arises. □

Lemma 20. *Let $\mathcal{T}$ be a Kripke structure over a finite tree with back edges of feedback k. Then, the bisimulation and the simulation type of $\mathcal{T}$ at its root can be described by a formula in $\mathrm{L}_\mu[k]$.*

Proof. Let $i : \mathcal{T} \to \{0, \dots, k-1\}$ be the partial labelling of $\mathcal{T}$ defined in Lemma 19. On the basis of this labelling, we construct a sequence of formulae $(\psi_v)_{v \in T}$ over fixed-point variables $X_0, \dots, X_{k-1}$ while traversing the nodes of $\mathcal{T}$ in reverse breadth-first order. For every action $a \in \textsc{act}$, the transitions in $\mathcal{T}$ are partitioned into tree edges and back edges $E_a \uplus B_a$.

To describe a state $v \in V$ and the relationship with its successors, let

$$\varphi_v := \alpha_v \wedge \bigwedge_{a \in \textsc{act}} \Bigg(\bigwedge_{(v,w) \in E_a} \langle a \rangle \psi_w \wedge \bigwedge_{(v,w) \in B_a} \langle a \rangle X_{i(w)} \\ \wedge\ [a] \Bigg(\bigvee_{(v,w) \in E_a} \psi_w \vee \bigvee_{(v,w) \in B_a} X_{i(w)} \Bigg) \Bigg),$$

where α_v expresses the atomic type of v, as in the beginning of the section. If v has an incoming back-edge, we set $\psi_v := \nu X_{i(v)} \,.\, \varphi_v$; otherwise, we let $\psi_v := \varphi_v$.

Note that since we proceed from the leaves of $\mathcal{T}$ to the root, this process is well-defined, and that in ψ_v the variables $X_{i(u)}$ occur free, for any node $u \neq v$ that is active at v. In particular, all variables in the formula ψ_u, corresponding to the root of $\mathcal{T}$, are bound.

We claim that $\mathcal{K}, v \models \psi_u$ iff $\mathcal{K}, v \sim \mathcal{T}, u$. First, we show that $\mathcal{T}, u \models \psi_u$, and hence $\mathcal{K}, v \models \psi_u$ for any $\mathcal{K}, v \sim \mathcal{T}, u$. To see this, we prove that Verifier has a winning strategy for the associated model-checking game.

Note that, since ψ_u has only greatest fixed points, any infinite play of the model-checking game is won by Verifier. It thus suffices to show that from any position of form (v, φ_v), Verifier has a strategy to make sure that the play proceeds to a next position of form (w, φ_w), unless Falsifier moves to position (v, α_v) and then loses in the next move. But by the construction of the formula, it is obvious that Verifier can play so that any position at which he moves is of one of the following three types.

(i) $(v, \langle a \rangle \psi_w)$, where $(v, w) \in E_a$: then, Verifier moves to position (w, ψ_w).

(ii) $(v, \langle a \rangle X_{i(w)})$, where $(v, w) \in B_a$: in this case, he moves to $(w, X_{i(w)})$.

(iii) $(w, \bigvee_{(v,z)\in E_a} \psi_z \vee \bigvee_{(v,z)\in B_a} X_{i(z)})$ for some edge $(v, w) \in E_a \cup B_a$: in this case, Verifier selects the appropriate disjunct with $z = w$ and moves accordingly either to (w, ψ_w) or to $(w, X_{i(w)})$.

In all cases the play will proceed to (w, φ_w). Hence, Falsifier can force a play to be finite only by moving to a position (v, α_v), where he loses. Otherwise the resulting play is infinite and thus always won by Verifier.

For the converse, suppose that $\mathcal{K}, v \not\sim \mathcal{T}, u$. Since $\mathcal{T}$ is finite, the non-bisimilarity is witnessed at a finite stage. That is, there is a basic modal formula separating $\mathcal{K}, v$ from $\mathcal{T}, u$, and Falsifier can force the model-checking game for ψ_u on $\mathcal{K}, v$ in finitely many moves to a position of form (w, α_w) such that w and w' have distinct atomic types. This proves that $\mathcal{K}, v \not\models \psi_u$.

By the same argument, we obtain a description of the simulation type of $\mathcal{K}, u$ using formulae φ_v restricted to their existential part:

$$\varphi_v := \alpha_v \wedge \bigwedge_{a \in \mathrm{ACT}} \Big(\bigwedge_{(v,w)\in E_a} \langle a \rangle \psi_w \wedge \bigwedge_{(v,w)\in B_a} \langle a \rangle X_{i(w)} \Big).$$

□

3.2 Finite unravellings and entanglement

According to Definition 5, every graph $\mathcal{G}$ can be unravelled from any node v to a tree $\mathcal{T}_{G,v}$ whose nodes are the paths in $\mathcal{G}$ from v. Clearly $\mathcal{T}_{G,v}$ is infinite unless $\mathcal{G}$ is finite and no cycle in $\mathcal{G}$ is reachable from v. A *finite unravelling* of a (finite) graph $\mathcal{G}$ is defined in a similar way, but rather than an infinite tree, it produces a finite tree with back edges. To construct a finite

unravelling we proceed as in the usual unravelling process with the following modification: whenever we have a path $v_0 v_1 \dots v_n$ in $\mathcal{G}$ with corresponding node $\pi = v_0 v_1 \dots v_n$ in the unravelling, and a successor w of v_n that coincides with v_i, for some $i \leq n$, then we may either create a new node πw together with a tree edge from π to πw, or put a back edge from π to its ancestor $v_0 \dots v_i$. Clearly this process is nondeterministic. In this way, any finite graph can be unravelled, in many different ways, to a finite tree with back edges.

Note that different finite unravellings of a graph may have different feedback.

Definition 21. The *entanglement* of a rooted graph $\mathcal{G}, u$ is the minimal feedback of its finite unravellings from v:

$$\mathrm{ent}(\mathcal{G}, u) = \min\{\, \mathrm{fb}(\mathcal{T}) \mid \mathcal{T} \text{ is a finite unravelling of } \mathcal{G}, u \,\}.$$

Likewise, for Kripke structures $\mathcal{K} = \big(V, (E_a)_{a \in \mathrm{ACT}}, (V_p)_{p \in \mathrm{PROP}}\big)$, we define $\mathrm{ent}(\mathcal{K}, u)$ as the entanglement of the underlying graph (V, E) with edges $E = \bigcup_{a \in \mathrm{ACT}} E_a$ from u.

Notice that if, in a structure $\mathcal{K}$, we consider two states u, u' from which all other states are reachable, then $\mathrm{ent}(\mathcal{K}, u) = \mathrm{ent}(\mathcal{K}, u')$. Since we assume that all states of a structure are reachable from some root, we may simply write $\mathrm{ent}(\mathcal{K})$ instead of $\mathrm{ent}(\mathcal{K}, u)$, for any root u.

As a direct consequence of Lemma 20, every Kripke structure of entanglement k can be described, up to bisimulation, in the μ-calculus using only k fixed-point variables.

Proposition 22. *Let $\mathcal{K}$ be a finite Kripke structure with $\mathrm{ent}(\mathcal{K}) = k$. Then, for any node v of $\mathcal{K}$, the bisimulation type of $\mathcal{K}, v$ is described by a formula of $\mathrm{L}_\mu[k]$ and its simulation type by an existential formula of $\mathrm{L}_\mu[k]$.*

Proof. For bisimulation types, we show that there exists a formula ψ_v such that,

$$\mathcal{K}', v' \models \psi_v \quad \text{iff} \quad \mathcal{K}', v' \sim \mathcal{K}, v.$$

By definition of entanglement, the structure $\mathcal{K}$ can be unravelled from v to a finite tree with back edges $\mathcal{T}$ of feedback at most k. Clearly, we have $\mathcal{T} \sim \mathcal{K}, v$. Hence, we obtain ψ_v by describing the bisimulation type of $\mathcal{T}$ at its root, as in Lemma 20. The case of simulation types is analogous. □

3.3 Game characterisation of entanglement

We characterise the entanglement of a graph $\mathcal{G} = (V, E)$ by way of a game, played by a thief against k detectives on $\mathcal{G}$ according to the following rules. At the beginning, the thief is at the given initial position u of $\mathcal{G}$ and the detectives are outside the graph. In any round, the detectives may either stay where they are, or place one of themselves on the current position v of the thief. The thief, in turn, has to move to a successor w of v that is not occupied by any detective. If no such position exists, the thief is caught and the detectives have won. Note that the thief sees the move of the detectives before he decides on his own move, and that he is forced to leave his current position, regardless of whether the detectives move or not.

Lemma 23. *Let $\mathcal{T}$ be a tree with back edges of feedback k. Then k detectives have a strategy to capture the thief on $\mathcal{T}$.*

Proof. Suppose that $\mathrm{fb}(\mathcal{T}) = k$. By Lemma 19 there is a labelling i of the targets of the back edges in $\mathcal{T}$ by numbers $0, \ldots, k-1$ assigning different values to any two nodes u, u' that are active at the same node v. This labelling induces the following strategy for the k detectives: at every node v reached by the thief, send detective number $i(v)$ to that position or, if the value is undefined, do nothing. By induction over the stages of the play, we can now show that this strategy maintains the following invariant: at every node v occurring in a play on $\mathcal{T}$, all active nodes $u \neq v$ are occupied and, if the current node is itself active, a detective is on the way. To see this, let us trace the evolution of the set $Z \subseteq T$ of nodes occupied by a detective. In the beginning of the play, Z is empty. A node v can be included into Z if it is visited by the thief and active with regard to itself. At this point, our strategy appoints detective $i(v)$ to move to v. Since, by construction of the labelling, the designated detective $i(v)$ must come from a currently inactive position and, hence, all currently active positions except v remain in Z. But if every node which becomes active is added to Z and no active node is ever given up, the thief can never move along a back-edge, so that after a finite number of steps he reaches a leaf of the tree and loses. But this means that we have a winning strategy for k detectives. □

Proposition 24. *The minimal number $k \in \mathbb{N}$ such that k detectives have a strategy to catch the thief on a graph $\mathcal{G}$ starting from a node u is the entanglement of $\mathcal{G}, u$.*

Proof. Notice that any winning strategy for the k detectives on a finite unravelling of $\mathcal{G}, u$ immediately translates to a winning strategy on $\mathcal{G}, u$. Accordingly, for any finite unravelling $\mathcal{T}$ of a graph $\mathcal{G}, u$, we have $k \leq \mathrm{fb}(\mathcal{T})$.

It remains to show that for any graph $\mathcal{G}, u$ there exists some finite unravelling $\mathcal{T}$ with $\mathrm{fb}(\mathcal{T}) \leq k$. To prove this, we associate winning strategies for k detectives to finite unravellings of $\mathcal{G}$ with feedback k.

A strategy for k detectives is a k-tuple $(g_0, \ldots, g_{k-1})$ of partial functions mapping every initial segment π of a possible play in $\mathcal{G}, u$ to a path $g_i(\pi)$ that is a prefix of π. The intended meaning is the following: if the thief covers the path $g_i(\pi)$, then detective i is placed at the current node and there he remains as long as the thief proceeds along π.

An *unravelling function* for a rooted graph $\mathcal{G}, u$ is a partial function ρ between finite paths from u through $\mathcal{G}$ that maps every path $v_0, \ldots, v_{r-1}, v_r$ in its domain to a strict prefix $v_0, v_1, \cdots, v_{j-1}$ such that $v_{j-1} = v_r$. The *unravelling* of $\mathcal{G}, u$ *driven* by ρ is the tree with back edges $\mathcal{T}$ defined as follows:

- the domain of $\mathcal{T}$ is the smallest set T which contains $v_0 := u$ and for each path $\pi \in T$, it also contains all prolongations πv in $\mathcal{G}$ at which ρ is undefined;
- the tree-edge partition is

$$E^{\mathcal{T}} := \{\, (v_0, \ldots, v_{r-1},\ v_0, \ldots, v_{r-1}, v_r) \in T \times T \mid (v_{r-1}, v_r) \in E^{\mathcal{G}} \,\};$$

- for all paths $\pi := v_0, \ldots, v_{r-1} \in T$ where $\rho(\pi v)$ is defined, the back-relation $B^{\mathcal{T}}$ contains the pair $(\pi,\ \rho(\pi v))$ if $(v_{r-1}, v) \in E^{\mathcal{G}}$.

Informally, a function ρ describes an unravelling that starts at the root and follows finite paths through $\mathcal{G}$; whenever the current path π can be prolonged by a position v and the value of ρ at πv is undefined, a fresh copy of v corresponding to πw is created as a successor of π. (This happens in particular whenever v has not yet been visited.) Otherwise, if $\rho(\pi\, v)$ is defined, then the current path π is prolonged by inserting a back-edge to its prefix $\rho(\pi)$ which also corresponds to a copy of v.

To any strategy $(g_0, \ldots, g_{k-1})$, we now associate an unravelling function ρ as follows: for every path π and any possible prolongation by v, if for some i, the end node of $g_i(\pi)$ is v, we set $\rho(\pi\, v) := g_i(\pi)$; otherwise we leave the value of $\rho(\pi\, v)$ undefined. It is easy to verify that, if g is a winning strategy for the detectives, the associated unravelling is finite and has feedback k. □

4 The existential hierarchy

As we have seen in the previous section, the entanglement of a graph provides an upper bound for the number of variables required to describe any Kripke

structure over this graph. However, the descriptive complexity depends not only on the underlying graph, but also on the labelling of transitions and states with actions and atomic propositions. For instance, the simulation type of any strongly connected Kripke structure over a language with only one action and no propositional symbols is described by the formula $\nu X.\langle a\rangle X$, regardless of the entanglement of the underlying graph.

In the sequel of this article we show that, with a particular labelling of edges, the structural complexity of a graph, in terms of entanglement, is reflected in the descriptive complexity of its simulation type measured by the number of variables needed to describe it in the μ-calculus.

Definition 25. A Kripke structure $\mathcal{K}$ is *deterministic* if every state $v \in V$ has at most one a-successor, for all actions $a \in \textsc{act}$; it is *co-deterministic* if every state has at most one a-predecessor, for all actions a. Further, we say that a structure is *singular* with respect to simulation, if there are no two states $v \neq w$ such that $\mathcal{K}, v \preccurlyeq \mathcal{K}, w$. A finite structure is *rigid*, if it is deterministic, co-deterministic, and singular with respect to simulation.

Lemma 26. *Every connected finite graph can be labelled in such a way that the resulting Kripke structure is rigid.*

Proof. Given a finite graph $\mathcal{G} = (V, E)$, the Kripke structure which assigns to every edge $(v, w) \in E$ a distinct action label is obviously rigid. Formally, this yields a structure over a set of actions $\textsc{act} := E$, with state domain V and singleton transition relations $E_{vw} := \{(v, w)\}$, for all $(v, w) \in E$. □

According to Proposition 22, the simulation type of any structure with entanglement k can be described by an existential formula in $\mathrm{L}_\mu[k]$. In this section we prove that, if the structure is rigid, no existential formula from $\mathrm{L}_\mu[k-1]$ can describe its simulation type. This establishes that the variable hierarchy is strict for the existential fragment of L_μ.

For a simple example of a rigid Kripke structure with entanglement k, consider the complete graph over k vertices labelled as in Lemma 26.

Our argument pivots around the model-checking game $\mathcal{G}(\mathcal{K}, \psi)$ associated to a Kripke structure $\mathcal{K}, u$ and a formula ψ defining its simulation type. Obviously, Verifier has a winning strategy in this game. In general, we may understand the subgame $\mathcal{G}_\sigma$ induced by a (memoryless) winning strategy σ of Verifier in $\mathcal{G}(\mathcal{K}, \psi)$ as a proof for $\mathcal{K}, u \models \psi$. We will argue that, on the one hand, if $\mathcal{K}$ is rigid, the entanglement of such a proof cannot be lower than the entanglement of $\mathcal{K}$ itself. On the other hand, we will show that this proof is already contained in the syntax graph of ψ, and hence its entanglement is not higher than the number of variables used in ψ.

4.1 Definite formulae

The rigidity of a structure ensures that the simulation types of its states do not overlap. This allows us to narrow the gap between the semantics and the syntax of formulae ψ describing the simulation type of a rigid structure $\mathcal{K}, u$. Concretely, we show that the proof of $\mathcal{K}, u \models \psi$, i.e., the subgame induced by a winning strategy in the associated model-checking game, can be embedded into the syntax graph of ψ.

Definition 27. We call a formula ψ *definite* on a Kripke structure $\mathcal{K}$, if for every subformula $\eta \in \mathrm{cl}(\psi)$, there exists precisely one state v such that $\mathcal{K}, v \models \eta$.

The notion is meaningful only over structures without propositional symbols. Notice that, if we consider rigid structures under this proviso, the formulae constructed in Lemma 20 as a description of their simulation type are indeed definite.

Lemma 28. *Let $\mathcal{K}$ be a rigid Kripke structure with a designated state u. Then, every existential formula $\psi \in \mathrm{L}_\mu$ defining the simulation type of $\mathcal{K}, u$ can be transformed, without increasing the number of variables, into an equivalent existential formula that is definite on $\mathcal{K}$.*

Proof. First, we dispose of the subformulae of ψ that do not hold at any node of $\mathcal{K}$. Let ψ' be the formula obtained from ψ by replacing every such subformula with $\bot$. Then, ψ' is still true on $\mathcal{K}$ and, being existential, on all models of ψ. On the other hand, ψ' obviously implies ψ so that we have $\psi' \equiv \psi$.

Further, we successively eliminate all subformulae true at more than one node. Assume that for some $\eta \in \mathrm{cl}(\psi)$ we have $\mathcal{K}, v_1 \models \eta$ and $\mathcal{K}, v_2 \models \eta$ with $v_1 \neq v_2$ and let ψ' be the formula obtained from ψ by replacing η with $\top$.

Clearly, ψ implies ψ'. To prove the converse, we will construct for every tree an extension that satisfies η at all nodes while preserving the validity of ψ. Notice that every extension of a tree $\mathcal{T}$ is similar to $\mathcal{T}$. Consequently, existential formulae are preserved under extensions.

Let $\mathcal{T}$ be a tree with edges labelled by ACT. We establish a matching correspondence between the nodes of $\mathcal{T}$ and $\mathcal{K}$ in the following way. For any node $x \in T$, consider the sequence of actions on the path from the root to x. In $\mathcal{K}$, there is at most one node w reachable from the designated root u via this sequence of actions, since the structure is deterministic. We set $\mathrm{match}_T(x) := w$, if the sequence is indeed executable in $\mathcal{K}$, otherwise we leave the value undefined. Now let $\mathcal{T}'$ be the extension of $\mathcal{T}$ obtained by attaching at every node x the unravelling $\mathcal{T}^{\mathcal{K}}_w$ of $\mathcal{K}$ from the node $w := v_1$ if $\mathrm{match}_T(x) \neq v_1$ and $w := v_2$ otherwise.

Claim. *For any tree $\mathcal{T}$, the constructed extension $\mathcal{T}'$ has the following properties:*

(i) *If $\mathcal{T} \models \psi'$ then $\mathcal{T}' \models \psi$.*

(ii) *If $\mathcal{T}' \models \psi$ then $\mathcal{T} \models \psi$.*

(i) Every subtree of $\mathcal{T}'$ rooted at a node $x \in T$ extends an unravelling $\mathcal{T}_w^{\mathcal{K}}$ of $\mathcal{K}, w$, where η holds. Since η is existential and thus preserved under extensions, it follows that also $\mathcal{T}'_x$, the subtree of $\mathcal{T}'$ rooted at x, is a model of η. Moreover, if σ_w is a winning strategy for Verifier in $\mathcal{G}(\mathcal{T}_w^{\mathcal{K}}, \eta)$, it will also be a winning strategy in $\mathcal{G}(\mathcal{T}'_x, \eta)$.

By means of this, we can extend any winning strategy σ of Verifier in $\mathcal{G}(\mathcal{T}, \psi')$ to a strategy in $\mathcal{G}(\mathcal{T}', \psi)$ as follows. At every position (x, φ) where $x \in T$ and $\varphi \neq \eta$ choose according to σ. As Falsifier cannot move in the tree, the play will stay on nodes of $\mathcal{T}$ unless a position (x, η) is reached. When this occurs, Verifier drops σ and proceeds with the strategy σ_w which is winning in $\mathcal{G}(\mathcal{T}_w^{\mathcal{K}}, \eta)$ and thus in $\mathcal{G}(\mathcal{T}'_x, \eta)$. In that way, every play of $\mathcal{G}(\mathcal{T}', \psi)$ is won by Verifier which means that $\mathcal{T}' \models \psi$.

(ii) Assuming that $\mathcal{T}' \models \psi$, let Z be a simulation relation witnessing that $\mathcal{K}, u \preccurlyeq \mathcal{T}'$. Then, the relation

$$Z' := \{\, (v, x) \in Z \mid x \in T \text{ and } v = \mathrm{match}_{T'}(x) \,\}$$

is a simulation from $\mathcal{K}, u$ to $\mathcal{T}$.

Obviously, Z' relates u with the root of $\mathcal{T}$. Since Z is a simulation, for any $(v, x) \in Z'$, $a \in \textsc{act}$, and every a-successor u of v there exists an a-successor y of x such that $(u, y) \in Z$. Clearly, $\mathrm{match}_{T'}(y) = u$, so we just need to show that $y \in T$. Let us assume, towards a contradiction, that y is a new node, $y \in T' \setminus T$. Then, in $\mathcal{K}$ there is a node $w \neq \mathrm{match}_T(x)$ with a-successor u', so that $y \sim u'$. On the other hand, $(u, y) \in Z$, hence $u \preccurlyeq y$. As $\mathcal{K}$ is singular with respect to simulation, this means that u and u' are actually the same node. But then this node would have two different a-predecessors, w and v, in contradiction to the fact that $\mathcal{K}$ is co-deterministic. Hence, Z' witnesses the simulation $\mathcal{K} \preccurlyeq \mathcal{T}$. This proves the second part of our claim and we can conclude that $\psi' \equiv \psi$.

Notice that whenever the subformulae $\langle a \rangle \top$ and $\eta \vee \top$ occur, they are also removed as they hold at more than one node; if the atom $\top$ appears as a conjunct we can safely drop it.

With this rewriting, ψ will eventually consist only of subformulae satisfied at precisely one node of $\mathcal{K}$. □

In particular, definiteness implies that every fixed-point definition holds at precisely one state. Accordingly, for any winning strategy σ in this game, the projection $(v, \varphi) \mapsto \varphi$ induces an embedding of $\mathcal{G}_\sigma(\mathcal{K}, \psi)$ into the syntax graph $\mathcal{S}_\psi$.

Corollary 29. *Let ψ be an existential formula that is definite on a rigid structure $\mathcal{K}$ and assume $\mathcal{K}, u \models \psi$. Then, for any winning strategy σ for Verifier in the game $\mathcal{G}(\mathcal{K}, \psi)$ the induced subgame $\mathcal{G}_\sigma(\mathcal{K}, \psi)$, is embeddable into the syntax graph of ψ.*

4.2 Cast structure

Up to now, we have seen that, in our specific setting, any formula describing a structure contains a proof of its validity on that structure. In the next step we argue that, moreover, this proof essentially contains (a bisimilar copy of) the structure in question.

Observe that a model-checking game does not necessarily explore the entire structure on which it is played. For example, if we are interested in the property $\mu X.(\langle a \rangle X \vee \langle b \rangle \top)$ expressing that a b-transition is reachable in the model, a winning strategy for Verifier would just display an a-path ending with a b-transition. To capture the part of a model explored by a winning strategy, we introduce the notion of structure cast by a strategy.

Definition 30. Given a Kripke structure $\mathcal{K}$, and an existential L_μ-formula ψ such that $\mathcal{K}, u \models \psi$, let σ be a winning strategy for Verifier in the model-checking game $\mathcal{G}(\mathcal{K}, \psi)$, inducing the subgame $\mathcal{G}_\sigma$. The *cast* of σ, is the Kripke structure $\widehat{\mathcal{G}}_\sigma = (\widehat{V}, (\widehat{E}_a)_{a \in \mathrm{ACT}})$ over a subset $\widehat{V}$ of vertices from $\mathcal{G}_\sigma$ consisting of the root (u, ψ) and the target (w, η) of every possible move $(v, \langle a \rangle \eta) \to (w, \eta)$ in $\mathcal{G}_\sigma$. Between two of these vertices $(v, \varphi), (w, \eta)$ we allow an $\widehat{E}_a$-transition if in $\mathcal{G}_\sigma$ there is a path from (v, φ) to a predecessor $(v, \langle a \rangle \eta)$ of (w, η) which avoids $\widehat{V}$.

Model-checking games are constructed out of a Kripke structure and a formula. By casting a winning strategy we perform a reverse operation, where we set out from a specific game, or a proof, and extract the relevant structural component. In line with this intuition, the following lemma points out that every model-checking game for an existential formula contains a model of the formula.

Lemma 31. *Let ψ be an existential L_μ-formula and let $\mathcal{K}, u$ be a model of ψ. Then, for any winning strategy σ of Verifier in the model-checking game $\mathcal{G} := \mathcal{G}(\mathcal{K}, \psi)$, the cast of $\mathcal{G}_\sigma$ at state (u, ψ) is also a model of ψ.*

Proof. We show that Verifier wins the model-checking game $\mathcal{G}' := \mathcal{G}(\widehat{\mathcal{G}}_\sigma, \psi)$ for ψ on the cast structure starting from position $((u, \psi), \psi)$. Towards this, we perform a generic play of $\mathcal{G}'$ while replicating, on the side, a play of $\mathcal{G}$ according to the Verifier strategy σ. Thereby we transfer every move of Falsifier from $\mathcal{G}'$ to $\mathcal{G}$ and, conversely, every move of Verifier back to $\mathcal{G}'$ so that the parallel plays maintain the following invariant in each turn: whenever the current position in $\mathcal{G}'$ is $((v, \alpha), \beta)$, the current position in $\mathcal{G}$ is (v, β).

This is done in the following way. At the starting position, the proposed invariant obviously holds. If Falsifier moves in the main game $\mathcal{G}'$ from some position $((v, \alpha), \eta_1 \wedge \eta_2)$ to $((v, \alpha), \eta_i)$, we move in the secondary game $\mathcal{G}$ from the current position $(v, \eta_1 \wedge \eta_2)$ to (v, η_i). If Verifier is in turn to move in the main game $\mathcal{G}'$ at a disjunction, e.g., $((v, \alpha), \eta_1 \vee \eta_2)$, the current position in the secondary game $\mathcal{G}$ is $(v, \eta_1 \vee \eta_2)$. In this case we first execute the move in $\mathcal{G}$ to (v, η_i) according to σ and then choose $((v, \alpha), \eta_i)$ in $\mathcal{G}'$. Similarly, when the current position in the main game is $((v, \alpha), \langle a \rangle \beta)$, and, hence, $(v, \langle a \rangle \beta)$ in the secondary game, we first perform in $\mathcal{G}$ the move (w, β) indicated by σ and then choose $((w, \beta), \beta)$ in $\mathcal{G}'$.

It can be easily checked that the choices transferred between the games are always available. Particularly, in the case of modal moves, this follows from the definition of the cast structure. Hence, Verifier can always move in $\mathcal{G}'$ if he can move in $\mathcal{G}$. Since σ is a winning strategy for the latter game, a play can end only at positions $(v, \top)$ in which case the play of $\mathcal{G}'$ also reaches a terminal position $((w, \beta), \top)$, where Verifier wins. Otherwise, both plays are infinite and the sequences of formulae they visit are the same; accordingly, thanks to his winning strategy in $\mathcal{G}$, Verifier simultaneously wins $\mathcal{G}'$. □

For the case of formulae describing the simulation type of a Kripke structure, the cast of a winning strategy must, hence, be similar to the structure itself. Moreover, for structures that are singular with respect to simulation, this relation has natural witnesses.

Lemma 32. *Given a structure $\mathcal{K}, u$ that is singular with respect to simulation, let $\psi \in \mathrm{L}_\mu$ be an existential formula describing its simulation type, and let σ be a winning strategy for Verifier in $\mathcal{G}(\mathcal{K}, \psi)$. Then, there exists a simulation from $\mathcal{K}, u$ to $\widehat{\mathcal{G}}_\sigma, (u, \psi)$ such that, $Z \subseteq \{ (v, (v, \varphi)) \mid \mathcal{K}, v \models \varphi \}$.*

Proof. According to Lemma 31, we have $\widehat{\mathcal{G}}_\sigma, (u, \psi) \models \psi$. As ψ describes the simulation type of $\mathcal{K}, u$, this implies that $\mathcal{K}, u \preccurlyeq \widehat{\mathcal{G}}_\sigma, (u, \psi)$. Among the simulations witnessing this, let Z be minimal (with respect to set inclusion). Then, for any pair $(v, (w, \varphi)) \in Z$, we have $\mathcal{K}, v \preccurlyeq \widehat{\mathcal{G}}_\sigma, (w, \varphi)$. On the other hand, we also have $\widehat{\mathcal{G}}_\sigma, (w, \varphi) \preccurlyeq \mathcal{K}, w$, since $\mathcal{G}_\sigma$ is a model checking game and

the modal moves follow the transitions of $\mathcal{K}$. Thus, we obtain $\mathcal{K}, v \preccurlyeq \mathcal{K}, w$ and, by our assumption that $\mathcal{K}$ is singular, it follows that $v = w$. □

It is not hard to show that a relation Z of the above kind is, in fact, a bisimulation between $\mathcal{K}'$ and the structure induced by its range in $\widehat{\mathcal{G}}_\sigma$.

4.3 The separation theorem

As a last step towards proving that simulation-type descriptions are hard formulae for the existential variable hierarchy, we show that the entanglement of a structure is bounded by the number of variables of any existential formula describing its simulation type.

Lemma 33. *Let $\mathcal{K}$ be a singular structure with respect to simulation and consider a distinguished node u. If there exists an existential formula $\psi \in \mathrm{L}_\mu[k]$ that is definite on $\mathcal{K}$ and describes the simulation type of $\mathcal{K}, u$, then* $\mathrm{ent}(\mathcal{K}) \leq k$.

Proof. According to Lemma 10, we may assume that ψ is guarded. Let $\psi_1, \ldots \psi_n$ be the fixed-point subformulae in ψ, i.e., $\psi_i = \lambda Y.\varphi_i$ with $\lambda \in \{\mu, \nu\}$ and $Y \in \{X_0, \ldots, X_{k-1}\}$. Throughout this proof we write $\mathrm{cl}(\psi_i)$ rather than $\mathrm{cl}_\psi(\psi_i)$ for the closure of ψ_i in ψ. Recall that by the definiteness of ψ, there is a unique node v_i with $\mathcal{K}, v_i \models \mathrm{cl}(\psi_i)$. Recall further that ψ_j *depends* on ψ_i, if in the syntax graph S_ψ (which is a tree with back edges), the node ψ_i is active at ψ_j, i.e., there is a descendent Y of ψ_j with a back-edge to ψ_i.

Consider a winning strategy σ for Verifier in the model checking game associated to $\mathcal{K}, u \models \psi$. By Corollary 29, the induced subgame $\mathcal{G}_\sigma$ is embedded in the syntax graph $\mathcal{S}_\psi$ via the projection $(v, \varphi) \mapsto \varphi$. On the other hand, its cast simulates $\mathcal{K}, u$. We fix a simulation Z from $\mathcal{K}, u$ to $\widehat{\mathcal{G}}_\sigma, (u, \psi)$ as in Lemma 32.

On the basis of this, we define a strategy for k detectives in the entanglement game on $\mathcal{K}$ starting at u. To each state v of $\mathcal{K}$ reached by the thief in a play against this strategy, we will associate a position (v, φ) in $\widehat{\mathcal{G}}_\sigma$ such that $(v, (v, \varphi)) \in Z$.

The initial state u is associated to position (u, ψ). Suppose that, in a round of the play, the thief sits at some position v in $\mathcal{K}$ which is associated to (v, φ) in $\widehat{\mathcal{G}}_\sigma$. Each free variable X_j in φ is defined at a fixed-point subformula $\psi_{j,\varphi} \in \{\psi_1, \ldots, \psi_n\}$ and, by definiteness, there exists precisely one state $v_{\varphi,j}$ in $\mathcal{K}$ where the closure $\mathrm{cl}(\psi_{j,\varphi})$ holds. The strategy of the detectives is to move those detectives $j < k$ to v for which $v_{j,\varphi} = v$. If now the thief, in turn, moves from v to some successor w not occupied by any detective, we associate with w a successor (w, ϑ) of (v, φ) in $\widehat{\mathcal{G}}_\sigma$ such that $(w, (w, \vartheta)) \in Z$, and proceed to

the next round. Lemma 32 guarantees that a suitable successor (w, ϑ) always exists in $\widehat{\mathcal{G}_\sigma}$. Accordingly, in $\mathcal{G}_\sigma$ there is a path from (v, φ) leading to (w, ϑ) via positions of the form (v, φ'). This establishes a correspondence between plays of the entanglement game on $\mathcal{K}, u$ and paths in $\mathcal{G}_\sigma$ and furthermore, their projections to paths in $\mathcal{S}_\psi$.

We shall prove that the strategy defined in this way is winning for the detectives. Towards a contradiction, assume that the thief can form an infinite path π from u through $\mathcal{K}$ when playing against this strategy. We look at the associated path π' through $\mathcal{G}_\sigma$ and at its projection π'' to a path through the syntax graph S_ψ. Since π, and hence π'' is infinite, some fixed-point definition ψ_j must be regenerated infinitely often on π''. We want to show that this cannot happen.

Indeed, suppose that at node (v, φ) the fixed-point formula ψ_i is regenerated. This means that there is a variable X_j such that $\psi_{j,\varphi} = \psi_i$ and $v_{j,\varphi} = v$. Since ψ is guarded, X_j must be free in φ. By definiteness, any next regeneration of ψ_i must also take place at v. But, at the moment when the thief moves from v to w, detective j is at v and stays there until, on the corresponding path π'' a new fixed point formula ψ_ℓ with the same variable X_j is opened, and a node $v' \neq v$ is reached where $\mathrm{cl}(\psi_\ell)$ holds. Before this has happened, the thief cannot move back to v.

Thus, in order to have a further regeneration of ψ_i the path π'' must go through the following steps:

1. From ψ_i the path proceeds to a fixed point definition $\psi_m = \lambda Y.\varphi_m$ with a different variable $Y \neq X_j$ so that ψ_m depends on ψ_i (i.e., ψ_i is active at ψ_m);

2. from there the path must reach a definition $\psi_\ell = \lambda X_j \varphi_\ell$, so that in the corresponding path on $\mathcal{K}$, the detective j is lured away from v;

3. then the path must regenerate Y to ψ_m, and

4. proceed from ψ_m to X_j where it can finally regenerate ψ_i.

Hence, we have seen that between any two regenerations of ψ_i on π'' we must have a regeneration of a formula ψ_m that depends on ψ_i. As a consequence, all fixed point formulae are regenerated only finitely often on π''. □

At this point we are ready to state our separation theorem.

Theorem 34. *Let $\mathcal{G}$ be a finite directed graph of entanglement k such that every node of $\mathcal{G}$ is reachable from u. Then, there exists a Kripke structure $\mathcal{K}$*

over $\mathcal{G}$ so that the simulation type of $\mathcal{K}, u$ can be described by an existential formula in $\mathrm{L}_\mu[k]$, but not by any existential formula in $\mathrm{L}_\mu[k-1]$.

Proof. According to Lemma 26, it is possible to assign transition labels to the edges of $\mathcal{G}$ so that the resulting Kripke structure $\mathcal{K}$ is rigid; no atomic atomic predicates are set.

Since $\mathrm{ent}(\mathcal{K}) = k$, an existential formula $\chi \in \mathrm{L}_\mu[k]$ describing the simulation type of $\mathcal{K}, u$ can be constructed, according to Proposition 22.

Towards a contradiction, assume that there is an existential formula $\psi \in \mathrm{L}_\mu[k-1]$ defining the simulation type of $\mathcal{K}, u$. According to Lemma 28, we can assume without loss of generality, that ψ is definite. But then, by Lemma 33 it would follow that $\mathrm{ent}(\mathcal{K}) \leq k-1$. □

As a conclusion, this shows that every existential formula describing the simulation type of a k-entangled rigid structure requires at least k variables. However, this does not yet exclude the existence of equivalent L_μ-formula over fewer variables but with universal modalities. In the next section we argue that this can not be the case.

Also notice that at this stage, we consider vocabularies with arbitrarily many action symbols. However, in Section 6 we will construct formulae that separate the levels of the variable hierarchy using only two variables.

5 An existential preservation theorem

The key argument in our proof of the hierarchy theorem consists in the following preservation property, which implies that the formulae we used to separate the hierarchic levels of the existential fragment also witness the strictness of the variable hierarchy in the case of the full μ-calculus.

Theorem 35. *Let $\mathcal{K}$ be a finite Kripke structure over a strongly connected graph. Then every formula $\psi \in \mathrm{L}_\mu[k]$ that defines the simulation type of a state $\mathcal{K}, u$ is equivalent to an existential formula $\psi' \in \mathrm{L}_\mu[k]$.*

To show that universal modalities can be safely eliminated from any formula ψ of the considered kind, we take a detour and first show that they can be eliminated from the formula expressing that some node at which ψ holds is reachable. To refer to this formula, we use a shorthand borrowed from temporal logics:

$$\mathsf{F}\psi := \mu X.\psi \vee \bigvee_{a \in \mathrm{ACT}} \langle a \rangle X.$$

Lemma 39 in the second part of this section then states that from any formula equivalent to $\mathsf{F}\psi$, an existential formula equivalent to ψ can be recovered without increasing the number of variables.

Lemma 36. *Let $\mathcal{K}$ be a finite strongly connected structure with a distinguished state u and let $\psi^{\mathcal{K}}$ be a formula defining the simulation type of $\mathcal{K}, u$. Then, every formula $\chi \equiv \mathsf{F}\psi^{\mathcal{K}}$ can be transformed, without increasing the number of variables, into an equivalent formula χ' with the following properties:*

(i) *no universal modalities occur in χ';*

(ii) *χ' is of shape $\mathsf{F}\psi$, where ψ contains no μ-operators;*

(iii) *every formula $\varphi \in \mathrm{cl}(\chi')$ holds at some state of $\mathcal{K}$.*

Proof. (i) Given an L_μ-formula χ, we say that a subformula $\langle a \rangle \varphi$ starting with a diamond is *vital*, if $\mathrm{cl}_\chi(\varphi)$ implies $\mathsf{F}\psi^{\mathcal{K}}$. Dually, a subformula $[a]\varphi$ starting with a box is vital, if the negation $\neg \mathrm{cl}_\chi(\varphi)$ implies $\mathsf{F}\psi^{\mathcal{K}}$.

We will perform several transformation steps and use χ' to also denote intermediary formulae equivalent to χ.

Eliminating vital boxes. For $\chi \equiv \mathsf{F}\psi^{\mathcal{K}}$, let χ' be the formula obtained by replacing any occurrence of a vital box-subformula $[a]\varphi$ with $\top$. Then, χ obviously implies χ'. For the converse, let us consider a tree model $\mathcal{T}$ of χ'. If, at all its nodes, $\mathcal{T}, v \models [a]\mathrm{cl}_\chi(\varphi)$ holds, then $\mathcal{T} \models \chi$. Else, there exists a node $v \in T$ with $\mathcal{T}, v \models \langle a \rangle \neg \mathrm{cl}_\chi(\varphi)$. But, since $[a]\varphi$ is vital, this means that $\mathcal{T}, v$ and hence $\mathcal{T}$ verifies $\mathsf{F}\psi^{\mathcal{K}}$. Either way, we obtain $\mathcal{T} \models \chi$ and hence $\chi \equiv \chi'$.

Eliminating non-vital modalities. By iterating the above elimination step a finite number of times, we obtain a formula $\chi \equiv \mathsf{F}\psi^{\mathcal{K}}$ without vital box-subformulae. Let now χ' be the formula obtained from χ by substituting simultaneously all remaining (i.e., non-vital) box-subformulae with $\bot$ and all non-vital diamond-subformulae with $\top$.

We will first show that the resulting formula χ' implies χ. Let $\mathcal{T}$ be a tree model of χ' and, for every non-vital subformula $\langle a \rangle \varphi$ of χ, let $\mathcal{T}_\varphi$ be a tree model of $\mathrm{cl}_\chi(\varphi) \wedge \neg \mathsf{F}\psi^{\mathcal{K}}$. Using the latter models, we construct an extension $\mathcal{T}'$ of $\mathcal{T}$ by introducing for every node $v \in T$ and every non-vital subformula $\langle a \rangle \varphi$ of χ, a fresh copy of $\mathcal{T}_\varphi$ which we connect to v via an a-edge.

Since χ' contains no box-subformulae, it is preserved under extensions. Consequently $\mathcal{T}' \models \chi'$ and Verifier has a winning strategy σ in the model-checking game $\mathcal{G}(\mathcal{T}', \chi')$. Also, for every tree $\mathcal{T}_\varphi$, Verifier has a winning

strategy σ_φ in the game $\mathcal{G}(\mathcal{T}_\varphi, \mathrm{cl}_\chi(\varphi))$. We can combine these strategies, to obtain a winning strategy for Verifier in the game $\mathcal{G}(\mathcal{T}', \chi)$ as follows. Move according to σ unless a position with a non-vital subformula of χ is met; up to that point, the play cannot leave T, otherwise, since $\mathsf{F}\psi^{\mathcal{K}}$ is falsified at any node $w \in T' \setminus T$, any vital subformula $\langle a \rangle \varphi$ would fail at w. Moreover, no subformula $[a]\varphi$ can occur, as it would correspond to a $\bot$ position in $\mathcal{G}(\mathcal{T}', \chi')$. Consequently, σ leads the play to a position $(v, \langle a \rangle \varphi)$ where $v \in T$ and $\langle a \rangle \varphi$ is non-vital. At that event, let the Verifier choose the a-successor to the root of $\mathcal{T}_\varphi$ and proceed with his memoryless winning strategy σ_φ for the remaining game. In this way, Verifier finally wins any play of $\mathcal{G}(\mathcal{T}', \chi)$. Notice that, for all nodes $w \in T' \setminus T$, we have $\mathcal{T}', w \not\models \mathsf{F}\psi^{\mathcal{K}}$, and hence $\mathcal{T}'$ verifies $\mathsf{F}\psi^{\mathcal{K}}$ (or, equivalently, χ) if, and only if, $\mathcal{T}$ does. Hence, we have the following chain of implications, showing that χ' implies χ:

$$\mathcal{T} \models \chi' \implies \mathcal{T}' \models \chi' \implies \mathcal{T}' \models \chi \implies \mathcal{T} \models \chi.$$

For the converse, consider a tree model $\mathcal{T} \models \chi$ and, for every (non-vital) subformula $[a]\varphi$ of χ, a tree model $\mathcal{T}_\neg\varphi \models \neg\mathrm{cl}_\chi(\varphi) \wedge \neg\mathsf{F}\psi^{\mathcal{K}}$. As in the previous step, we construct an extension $\mathcal{T}'$ of $\mathcal{T}$ by connecting every node $v \in T$ via an a-edge to a fresh copy of $\mathcal{T}_\neg\varphi$, for every subformula $[a]\varphi$ of χ. Since $\chi \equiv \mathsf{F}\psi^{\mathcal{K}}$ is preserved under extensions, $\mathcal{T}'$ is still a model of χ. Let σ be a winning strategy for Verifier in the model-checking game $\mathcal{G}(\mathcal{T}', \chi)$. We will show that σ is also a winning strategy for Verifier in $\mathcal{G}(\mathcal{T}, \chi')$.

Notice that, in $\mathcal{G}(\mathcal{T}', \chi)$ Falsifier has a winning strategy from every position $(v, [a]\varphi)$ with $v \in T$, by moving to the a-successor of v at the root of $\mathcal{T}_\neg\varphi$. Consequently, any play according to Verifier's strategy σ will avoid such positions. Besides this, at every position $(v, \langle a \rangle \varphi)$ where $v \in T$ and $\langle a \rangle \varphi$ is a vital subformula of χ, the strategy σ will appoint a successor position (w, φ) with $w \in T$, otherwise, since any a-successor $w' \in T' \setminus T$ falsifies $\mathsf{F}\psi^{\mathcal{K}}$, φ would fail too. Summarising, every play of $\mathcal{G}(\mathcal{T}', \chi)$ according to σ, will avoid universal modalities and meet only nodes $v \in T$, unless a position with a non-vital subformula $\langle a \rangle \varphi$ occurs. But under these conditions, we can replicate every play of $\mathcal{G}(\mathcal{T}', \chi)$ according to σ as a play of $\mathcal{G}(\mathcal{T}, \chi')$: in case a non-vital subformula $\langle a \rangle \varphi$ of χ is met in the former game, Verifier immediately wins $\mathcal{G}(\mathcal{T}, \chi')$, since the non-vital diamond-subformulae have been replaced by $\top$. Otherwise, the outcome of the play is the same for both games and Verifier wins as well.

This concludes the proof that $\chi \equiv \chi'$.

(ii) By the above result, we can assume without loss that $\chi \equiv \mathsf{F}\psi^{\mathcal{K}}$ contains no box-modalities. For n the number of states in $\mathcal{K}$, let ψ be the formula obtained by replacing every occurrence of a least fixed-point subformula $\mu X.\varphi$

in χ by it's n-th approximant φ_n. Then, by definition of the μ-operator, ψ implies χ and thus $\mathsf{F}\psi$ implies $\mathsf{F}\chi$, which is equivalent to χ. Conversely, since $\mathcal{K}, u \models \chi$ and $\mathcal{K}$ has n states, we have $\mathcal{K}, u \models \psi$. As ψ is preserved under simulation, this means that $\psi^{\mathcal{K}}$ implies ψ. Accordingly $\mathsf{F}\psi^{\mathcal{K}}$, which is equivalent to χ, implies $\mathsf{F}\psi$. Hence, $\chi \equiv \mathsf{F}\psi$.

Note that if χ contains any fixed-point variables, the transformation into $\mathsf{F}\psi$ does not increase the number of variables, as we can pick any of the variables already occurring in χ to expand the F-notation.

(iii) By the previous argument, we can assume that χ is of shape $\mathsf{F}\psi$ where ψ contains no boxes, i.e., $\chi = \mu X.\psi \vee \bigvee_a \langle a \rangle X$. Clearly, χ itself holds at every node of $\mathcal{K}$ and therefore, for every transition a occurring in $\mathcal{K}$, there is a node $v \in V$ where $\langle a \rangle \chi$, and thus $\mathrm{cl}_\chi\big(\langle a \rangle X\big)$, holds. Hence, any subformula φ of χ, with $\mathcal{K}, v \not\models \mathrm{cl}_\chi(\varphi)$ for all v, must actually be a subformula of ψ. Let ψ' be the formula obtained by replacing every such occurrence φ in ψ with $\bot$. On the one hand, ψ' then obviously implies ψ. On the other hand, as $\mathcal{K}, u \models \mathsf{F}\psi$, there must exists a node v of $\mathcal{K}$ where ψ holds. At that node we also have $\mathcal{K}, v \models \psi'$ and, because ψ' is preserved under simulation, this means that $\psi'^{\mathcal{K}}_v$ implies ψ'. But then $\mathsf{F}\psi^{\mathcal{K}}$ implies $\mathsf{F}\psi'$ and, by $\mathsf{F}\psi \equiv \mathsf{F}\psi^{\mathcal{K}}$, it follows that $\mathsf{F}\psi$ implies $\mathsf{F}\psi'$. □

Radical formulae and crisp models. Before we proceed towards proving the Theorem 35, we will introduce some notions which will be useful in the proof of Lemma 39

Given a formula $\psi \in \mathrm{L}_\mu$, we call a subformula φ *radical*, if it appears directly under a modal quantifier in ψ. We refer to the closure of radicals in ψ by

$$\mathrm{cl}_0(\psi) := \{\psi\} \cup \{\, \varphi \in \mathrm{cl}(\psi) \mid \langle a \rangle \varphi \in \mathrm{cl}(\psi) \text{ or } [a]\varphi \in \mathrm{cl}(\psi) \text{ for some } a \in \textsc{act} \,\}.$$

Radical formulae are the first to be met when a play of the model-checking game reaches a new node of the Kripke structure. For this reason, we need to take care of game positions carrying radical formulae when merging strategies of different games.

Let $\mathcal{K}, u$ be a model of $\psi \in \mathrm{L}_\mu$ and σ a winning strategy for Verifier in $\mathcal{G}(\mathcal{K}, \psi)$. For any node $v \in V$, we define the *strategic type* of v in $\mathcal{K}, u$ under σ as follows:

$$\mathrm{tp}^{\mathcal{K}}_\sigma(v) := \{\, \varphi \in \mathrm{cl}_0(\psi) \mid \text{position } (v, \varphi) \text{ is reachable in } \mathcal{G}_\sigma(\mathcal{K}, \psi) \,\}.$$

In arbitrary games, the type of a node can be rather complex. However, for existential formulae, Verifier has full control over the moves in the Kripke structure. In the ideal case, he can foresee for every node, a single radical formula to be proved there.

Given a Kripke structure $\mathcal{K}, u$ and a formula ψ, we say that a Verifier strategy σ in the model-checking game $\mathcal{G}(\mathcal{K}, \psi)$ is *crisp*, if the strategic type $\mathrm{tp}^{\mathcal{K}}_{\sigma}(v)$ of any $v \in V$ consists of not more than one radical. Accordingly, we call a model $\mathcal{K}, u$ of ψ crisp (under σ), if Verifier has a crisp winning strategy σ in the associated model-checking game.

The subsequent lemmas, that can be easily proved, provide us with sharp tools for manipulating models of existential formulae.

Lemma 37. *Given a structure $\mathcal{K}, u$ every existential formula $\psi \in \mathrm{L}_{\mu}$ with $\mathcal{K}, u \models \psi$ also has a tree model $\mathcal{T}$ bisimilar to $\mathcal{K}, u$ which is crisp. Moreover, if $\mathcal{K}$ is finitely branching, then $\mathcal{T}$ can be chosen so as well.*

Lemma 38. *Let $\mathcal{T}$ be a crisp tree model of a formula $\psi \in \mathrm{L}_{\mu}$ under a strategy σ and let $x \in T$ be a node with strategic type $\mathrm{tp}^{\mathcal{T}}_{\sigma}(x) = \{\varphi\}$. Then, for every crisp tree model $\mathcal{S}$ of φ, the tree $\mathcal{T}[x/\mathcal{S}] \models \psi$, obtained by replacing the subtree of $\mathcal{T}$ rooted at x with $\mathcal{S}$, is still a crisp model of ψ.*

We are now ready for the final step, the elimination of the F-operator.

Lemma 39. *Let $\mathcal{K}$ be a finite strongly connected structure with a state u and let $\psi^{\mathcal{K}}$ describe the simulation type of $\mathcal{K}, u$. If $\psi \in \mathrm{L}_{\mu}$ is a formula such that $\mathsf{F}\psi \equiv \mathsf{F}\psi^{\mathcal{K}}$, then it can be transformed, without increasing the number of variables, into a formula ψ' without universal modalities, such that $\psi' \equiv \psi^{\mathcal{K}}$.*

Proof. According to Lemma 36, we can assume that ψ contains no universal modalities or least fixed point operators and that (the closure of) every subformula is true at some node in $\mathcal{K}$.

We will first show that for any node v in $\mathcal{K}$, there is a subformula φ of ψ whose closure $\mathrm{cl}_{\psi}(\varphi)$ implies $\psi^{\mathcal{K}}_{v}$. Actually, we always find a radical formula with this property.

Towards a contradiction, let us assume that $\psi^{\mathcal{K}}_{v}$ is not implied by any radical subformula of ψ. This means that every $\varphi \in \mathrm{cl}_0(\psi)$ has a tree model $\mathcal{T}_{\varphi}$ which falsifies $\psi^{\mathcal{K}}_{v}$. According to Corollary 8, we can choose $\mathcal{T}_{\varphi}$ to be a finitely branching tree that falsifies already an approximant of $\psi^{\mathcal{K}}_{v}$ to some finite stage m_{φ}. Observe that this approximant $(\psi^{\mathcal{K}}_{v})[\nu^{m_{\varphi}}/\nu]$ is a modal formula. Let us denote its modal depth by n_{φ}. Further, let us fix a number n which is greater than any n_{φ} for $\varphi \in \mathrm{cl}_0(\psi)$ and co-prime to every number up to the size of the domain V.

By Lemma 37, we can assume without loss of generality that each $\mathcal{T}_\varphi$ is a crisp model of φ, this being witnessed by a crisp winning strategy for Verifier in the game $\mathcal{G}(\mathcal{T}_\varphi, \varphi)$. In particular, $\mathcal{T}_\psi$ is a crisp model of ψ. Let σ_ψ be a crisp winning strategy for Verifier in the model-checking game $\mathcal{G}(\mathcal{T}_\psi, \psi)$.

With the aid of these, we construct a sequence of trees $(\mathcal{T}_i)_{0 \leq i < \omega}$, together with crisp Verifier strategies σ_i witnessing that $\mathcal{T}_i \models \psi$. To start, we set $\mathcal{T}_0 := \mathcal{T}_\psi$ and $\sigma_0 := \sigma_\psi$. In every step $i > 0$, the tree $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by performing the following manipulations at depth $n(i+1)$. For each subtree of $\mathcal{T}_i$ rooted at a node x of this depth, we check whether $\mathcal{T}_i, x \models \psi_v^{\mathcal{K}}$. If this is not the case, the subtree remains unchanged. Else, we look at the strategic type of x under σ_i. If the type is empty, we simply cut all successors of x. Otherwise, $\mathrm{tp}_{\sigma_i}^{\mathcal{T}_i}(x)$ consists of a single radical formula φ, and we replace the subtree $\mathcal{T}_i, x$ with $\mathcal{T}_\varphi$. According to Lemma 38, the resulting tree $\mathcal{T}_{i+1}$ is a model of ψ, and the composition of the strategy σ_i with the crisp strategies σ_φ on the newly appended subtrees $\mathcal{T}_\varphi$ yields a crisp Verifier strategy σ_{i+1} for the model-checking game $\mathcal{G}(\mathcal{T}_{i+1}, \psi)$.

By construction, each of the trees $\mathcal{T}_i$ is finitely branching and the sequence $(\mathcal{T}_i)_{0 \leq i < \omega}$ converges in the prefix topology of finitely branching trees (see [9]). Let $\mathcal{T}_\omega$ be the limit of this sequence. Since no μ-operators occur in ψ, its model class is topologically closed on finitely branching trees, according to [9]. Consequently, $\mathcal{T}_\omega$ is still a model of ψ. By our hypothesis, ψ implies $\mathsf{F}\psi^{\mathcal{K}}$. Thus, at some depth d in $\mathcal{T}_\omega$ a node x with $\mathcal{T}_\omega, x \models \psi_v^{\mathcal{K}}$ appears. Since $\mathcal{K}$ is strongly connected, v must lie on a cycle in $\mathcal{K}$. Hence, for $k \leq |V|$ being the length of such a cycle, there exist nodes y with $\mathcal{T}_\omega, y \models \psi_v^{\mathcal{K}}$ at every depth $d + jk$. However, our construction eliminated all subtrees carrying the similarity type of v at depths multiple of n. Since n was chosen to be co-prime to any integer up to $|V|$, it follows that $\mathcal{T}_\omega$ cannot satisfy ψ. This is a contradiction which invalidates our assumption that $\psi_v^{\mathcal{K}}$ is not implied by any $\varphi \in \mathrm{cl}_0(\psi)$.

Hence, for every node $v \in V$, there exists a formula $\varphi_v \in \mathrm{cl}_0(\psi)$ implying $\psi_v^{\mathcal{K}}$. We can show that the converse also holds, if v is maximal with respect to the preorder $\preccurlyeq$, in the sense that for every w with $v \preccurlyeq w$ we have $w \preccurlyeq v$. Recall that, by Lemma 36 (iii), the formula φ_v must be verified at some node w in $\mathcal{K}$. Since φ_v is existential and thus preserved under extension, it follows that $\psi_w^{\mathcal{K}}$ implies φ_v, which further implies $\psi_v^{\mathcal{K}}$. But this means that $v \preccurlyeq w$ and, by maximality of v, that w and v are bisimilar. Hence, $\mathcal{K}, v \models \varphi_v$ and consequently $\psi_v^{\mathcal{K}} \equiv \varphi_v$.

This concludes the proof for the case when u is maximal in $\mathcal{K}$ with respect to $\preccurlyeq$. Otherwise, we could not guarantee, of course, that $\varphi_u \equiv \psi_u^{\mathcal{K}}$. But in that case, a formula equivalent to $\psi_u^{\mathcal{K}}$ can be recovered from $\mathrm{cl}_0(\psi)$ without great difficulty. □

6 The hierarchy theorem

Up to now we have shown how to construct, for every level k of the variable hierarchy, existential formulae which are not equivalent to any existential formula from a lower hierarchical level. However, this left open the question whether there exist equivalent formulae in $\mathrm{L}_\mu[k-1]$ which use universal quantification. Due to Theorem 35, we are now able to assert that this cannot be the case.

Theorem 40. *For every k, there exist formulae $\psi \in \mathrm{L}_\mu[k]$ that are not equivalent to any formula in $\mathrm{L}_\mu[k-1]$.*

Proof. Consider a rigid strongly connected Kripke structure $\mathcal{K}$ of entanglement k and let $\psi^{\mathcal{K}} \in \mathrm{L}_\mu$ be a formula describing the simulation type of $\mathcal{K}, u$ for some state u.

Towards a contradiction, assume that there exists a formula $\psi \in \mathrm{L}_\mu[k-1]$ equivalent to $\psi^{\mathcal{K}}$. Since ψ defines the simulation type of $\mathcal{K}$, a finite strongly connected structure, we can apply Theorem 35 to conclude that there also exists a formula $\psi' \in \mathrm{L}_\mu[k-1]$ using only existential modalities which is equivalent to $\psi^{\mathcal{K}}$. But this contradicts the separation theorem 34 for the existential fragment. □

6.1 Separating formulae with two modalities

The results of the previous sections provide us with a generic technique to construct witnesses for the L_μ-variable hierarchy. The first examples for the strictness of the existential hierarchy, which turn out to be valid witnesses for the unrestricted case too, have been presented in [5]. They rely on rigid k-cliques where every action is labelled differently, leading to formulae over a vocabulary with k^2 modalities.

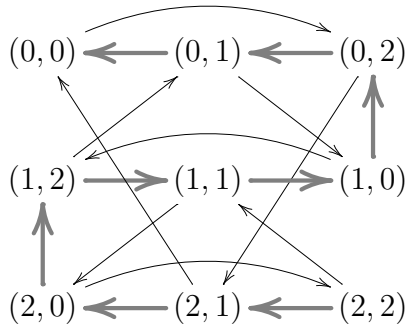
To show that already over a fixed vocabulary, the variable hierarchy remains strict, we construct rigid Kripke structures over only two modalities leading to formulae that are strict at each level of the variable hierarchy.

Definition 41. For every $k > 0$, let $\mathcal{C}^k := (V, E_a, E_b)$ be the Kripke structure with state set $V = \{0, \dots, k-1\} \times \{0, \dots, k-1\}$ and transition relations

$$\begin{aligned} E_a &:= \{ \big((i,j), (i, j-1) \big) \mid i \geq 0, j > 0 \} \\ &\quad \cup \{ \big((i,0), (i-1, k-1) \big) \mid i > 0 \} \text{ and} \\ E_b &:= \{ \big((i,j), ((i+j) \bmod k, (j-1) \bmod k) \big) \mid 0 \leq i, j < k \}. \end{aligned}$$

Let us first verify that these structures $\mathcal{C}^k$ indeed fulfil the premises formulated in the proof of the separation theorem 34.

Figure 4: The Kripke structure $\mathcal{C}^3$ (a-transitions thicker, b-transitions plain)



Lemma 42. *For every k, the structure $\mathcal{C}^k = (V, E_a, E_b)$ satisfies the following conditions:*

(i) $\mathrm{ent}(\mathcal{C}^k) = k$;

(ii) $\mathcal{C}^k$ *is deterministic and co-deterministic;*

(iii) $\mathcal{C}^k$ *is singular with respect to simulation.*

Proof. To prove (i), we use our characterisation of entanglement in terms of games, and show that the thief has a winning strategy in any game on $\mathcal{C}^k$ with less than k detectives, but he loses when they come in k or more.

We will refer to the rows $C_i := \{\, i, j \mid 0 \leq j < k \,\}$ of the state set as *islands*. Each island induces a cycle, and there is an edge between any pair of islands. Intuitively, if there are less than k cops, at every moment at least one of the islands must be unguarded and the thief can always navigate from his current position to that island without meeting a detective by pursuing the following strategy: Whenever the current island i is unguarded and, moreover, no detective is on his way to the current position, move to a successor in C_i. In the event that a detective is sent to the current position, at least one island j must be left unguarded. Since the current island was previously unguarded, the path from the current position to the safe island j is free. Hence, set out on this path and follow it until reaching the island j. Upon arrival, j will still be an unguarded island so that the strategy can be reiterated.

In case k or more detectives are available, they can distribute to the different islands, e.g., by following the thief to any position $(i, 0)$ he reaches during the play. Then the thief must move to a fresh island after at most

$k-1$ steps. But after k iteration, there are no unguarded islands left, so the thief loses.

It is easily seen that $\mathcal{K}$ is deterministic and co-deterministic. To verify that it is also singular with respect to simulation, observe that the subgraph of $\mathcal{C}^k$ induced by a-transitions is acyclic and each node is uniquely determined by the length of the maximal a-path available from it. □

According to this, the structures $\mathcal{C}^k$ can be used as witnesses in the proof of the separation theorem 34 yielding strict formulae for each level k of the existential variable hierarchy. Since $\mathcal{C}^k$ is strongly connected, Theorem 35, establishes that these formulae actually witness the strictness of the variable hierarchy of the μ-calculus, already over a language with two modalities only.

Corollary 43. *For every integer k, there are bimodal existential formulae $\psi^k \in \mathrm{L}_\mu[k]$ that are not equivalent to any formula in $\mathrm{L}_\mu[k-1]$.*

We explicitly construct witnessing formulae describing the simulation type of $\mathcal{C}^k, (0,0)$, as in the proof of Proposition 22. Towards this, we build a sequence of formulae $(\varphi_{i,j})_{0 \leq i,j < k}$ over the fixed-point variables $X_0, \ldots, X_{k-1}$ by induction on j, setting for all i simultaneously $\varphi_{i,0} := X_i$ and for every $j > 0$:

$$\varphi_{i,j} := \langle a \rangle \varphi_{i,j-1} \wedge \langle b \rangle \varphi_{i+j,j-1}.$$

Then, we define the system S of rules

$$X_0 := \langle b \rangle \varphi_{0,n-1} \text{ and } X_i := \langle a \rangle \varphi_{i-1,k-1} \wedge \langle b \rangle \varphi_{i,n-1} \text{ for } 0 < i < k.$$

The formula $\nu X_0.S$ obtained as a description for the simulation type of $\mathcal{C}^k$ at state (0.0) is strict for the level k of the variable hierarchy.

7 Parikh's Game Logic and hierarchies of the μ-calculus

Parikh's Game Logic GL, introduced in [16], is a formalism for reasoning about the dynamics of games. It extends propositional logic by the addition of modal operators whose meanings are assigned by games. Specifically, path-forming games for two players, Angel and Demon, are described. The syntax of GL is based on the language of PDL augmented with a game dualisation operator.

Definition 44. Starting from a set PROP of atomic propositions and a set ACT of atomic game actions, the expressions of GL are of two sorts, formulae and games, generated respectively by the following grammar:

$$\varphi := \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\gamma\rangle\varphi$$
$$\gamma := a \mid \varphi? \mid \gamma;\gamma \mid \gamma \cup \gamma \mid \gamma^* \mid \gamma^{\mathrm{d}}$$

where $p \in$ PROP and $a \in$ ACT.

Typically, $\langle\gamma\rangle\varphi$ expresses that Angel has a strategy to play the game γ starting at v in such a way that either φ is true when the game ends, or the game breaks and Demon fails. The game expressions specify a schedule for the two players, according to the following outline. The sequential composition of two games $\gamma_1;\gamma_2$ means: play γ_1 first, then γ_2. The nondeterministic choice operator $\gamma_1 \cup \gamma_2$ lets Angel decide which of the two games γ_1 or γ_2 shall be played. The iteration operator γ^* allows to play the game γ repeatedly, for a finite number of times, where Angel can decide before each round whether a new round is to be played. Finally, the test operator ($\varphi?$) invokes an independent observer to check whether φ holds; if so, the play just ends, otherwise it breaks and Angel loses. As a new construct, beyond PDL, the formalism introduces an explicit alternation operator γ^{d}, which directs Angel and Demon to play γ with interchanged roles. This notion of dualisation, corresponds to a form of game-theoretic negation.

Originally, the semantics of Game Logic was defined over neighbourhood structures, which are more general than Kripke structures, with transition relations between *sets* of states. But GL is also a very interesting logic when interpreted on Kripke structures. In this variant, atomic games are associated to transitions. For precise definitions, we refer to Pauly's thesis [17] which offers an extensive survey of Game Logic.

If the dualisation operator is not used, Game Logic over Kripke structures is just PDL. However, with dualisation, the expressive power of the language increases significantly. For instance ΔPDL, the extension of PDL with a looping operator can easily be captured. This already implies that GL is more expressive than CTL*, which is subsumed by ΔPDL [20].

Furthermore, it turns out that the expressive power of GL goes far beyond that of commonly used process logics.

Proposition 45 ([3])**.** *For every $n \in \mathbb{N}$, there exists a* GL*-formula* W^n *so that for any parity game $\mathcal{G}$ with n priorities we have $\mathcal{G}, u \models \mathrm{W}^n$ if, and only if, Player 0 has a winning strategy for $\mathcal{G}$ starting from position u.*

Since, for every n, the property described by the formula W_n is hard for the n-th level of the μ-calculus alternation hierarchy [6], and the size of any

W_n is linear in n, this has significant consequences regarding the conceptual and computational complexity of GL.

Corollary 46. (i) *No finite level of the μ-calculus alternation hierarchy captures the expressive power of* GL*.*

(ii) *Model-checking for the μ-calculus can be performed efficiently iff this is the case for* GL*.*

On the other hand, when compared to the μ-calculus, the syntax of GL seems considerably simpler, as it does not explicitly introduce variables. Indeed, it turns out that, even though GL-formulae may involve deeply nested iterations, they can be written in L_μ using, and re-using, only two fixed-point variables.

Lemma 47. *Every* GL*-formula over Kripke structures, can be translated into an equivalent formula in* $\mathrm{L}_\mu[2]$*.*

Proof. For a formula $\psi \in \mathrm{GL}$, we construct three mappings $\cdot^X$, $\cdot^Y$, and $\cdot^\sharp$ inductively over its subexpressions. The translations $\cdot^X$ and $\cdot^Y$ associate to every game expression γ an L_μ-formula $\gamma^X(X)$ and $\gamma^Y(Y)$, respectively, with one free fixed-point variable:

$$\begin{array}{rlrl}
a^X &:= \Diamond X & a^Y &:= \Diamond Y \\
(\gamma_1 \cup \gamma_2)^X &:= \gamma_1^X \vee \gamma_2^X & (\gamma_1 \cup \gamma_2)^Y &:= \gamma_1^Y \vee \gamma_2^Y \\
(\gamma_1;\gamma_2)^X &:= \gamma_1^X[X := \gamma_2^X] & (\gamma_1;\gamma_2)^Y &:= \gamma_1^Y[Y := \gamma_2^Y] \\
(\varphi?)^X &:= \varphi^\sharp \wedge X & (\varphi?)^Y &:= \varphi^\sharp \wedge Y \\
(\gamma^d)^X &:= \neg\gamma^X[X := \neg X] & (\gamma^d)^Y &:= \neg\gamma^Y[Y := \neg Y] \\
(\gamma^*)^X &:= \mu Y.X \vee \gamma^Y & (\gamma^*)^Y &:= \mu X.Y \vee \gamma^X
\end{array}$$

The mapping $\cdot^\sharp$ associates to every GL-formula an L_μ-formula without free variables, setting $p^\sharp := p$, and

$$(\neg\varphi)^\sharp := \neg\varphi^\sharp, \quad (\varphi_1 \vee \varphi_2)^\sharp := \varphi_1^\sharp \vee \varphi_2^\sharp, \quad (\langle\gamma\rangle\varphi)^\sharp := \gamma^X[X := \varphi^\sharp].$$

In this way, we obtain for any $\psi \in \mathrm{GL}$ a formula $\psi^\sharp \in \mathrm{L}_\mu[2]$ with the same Kripke models. □

In his first paper on this topic [15], Parikh announced Game Logic as

> [a logic] which lies in expressive power between the PDL of Fischer and Ladner and the μ-calculus of Kozen. It is stronger than the first, but might possibly be equal in expressive power to the second.

As a direct consequence of our hierarchy theorem, we can now separate the expressive power of GL and the μ-calculus. Since Game Logic can be translated into the two variable fragment of L_μ, its expressive power is strictly subsumed already by $L_\mu[3]$. It is an open problem whether Game Logic is equivalent to $L_\mu[2]$.

Corollary 48. *The modal μ-calculus is strictly more expressive than Game Logic interpreted over Kripke structures.*

Moreover, since our examples of strict formulae for the variable hierarchy involve only greatest fixed points, it follows that already the alternation-free level of L_μ contains formulae inexpressible in Game Logic. This shows that the variable hierarchy and the alternation hierarchy of L_μ are orthogonal.

Acknowledgements

This research has been partially supported by the European Community Research Training Network "Games and Automata for Synthesis and Validation". The authors wish to thank Thomas Colcombet for his valuable feedback on this article. The third author expresses his gratitute to André Arnold for his continuing support and advice.

References

[1] A. Arnold, *The mu-calculus alternation-depth is strict on binary trees*, RAIRO Informatique Thorique et Applications, 33 (1999), pp. 329–339.

[2] A. Arnold and D. Niwiński, *Rudiments of μ-calculus*, North Holland, 2001.

[3] D. Berwanger, *Game logic is strong enough for parity games*, Studia Logica, 75 (2003), pp. 205–219. Special issue on Game Logic and Game Algebra edited by M. Pauly and R. Parikh.

[4] D. Berwanger and E. Grädel, *Games and model checking for guarded logics*, in Proceedings of LPAR 2001, Lecture Notes in Computer Science Nr. 2250, Springer, 2001, pp. 70–84.

[5] D. Berwanger, E. Grädel, and G. Lenzi, *On the variable hierarchy of the modal mu-calculus*, in Computer Science Logic, CSL 2002, J. Bradfield, ed., vol. 2471 of LNCS, Springer-Verlag, 2002, pp. 352–366.

[6] J. Bradfield, *The modal μ-calculus alternation hierarchy is strict*, Theoretical Computer Science, 195 (1998), pp. 133–153.

[7] A. Emerson and C. Jutla, *Tree automata, mu-calculus and determinacy*, in Proc. 32nd IEEE Symp. on Foundations of Computer Science, 1991, pp. 368–377.

[8] G. Gottlob, N. Leone, and F. Scarcello, *Robbers, marshals, and guards: Game theoretic and logical characterizations of hypertree width*, in Proc. 20th ACM Symp. on Principles of Database Systems, 2001, pp. 195–201.

[9] D. Janin and G. Lenzi, *On the logical definability of topologically closed recognizable languages of infinite trees*, Computing and Informatics, 21 (2002), pp. 185–203.

[10] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas, *Directed tree-width*, J. Comb. Theory Ser. B, 82 (2001), pp. 138–154.

[11] D. Kozen, *Results on the propositional μ-calculus*, Theoretical Computer Science, 27 (1983), pp. 333–354.

[12] D. Kozen, *A finite model theorem for the propositional μ-calculus*, Studia Logica, 47 (1988), pp. 233–241.

[13] O. Kupferman, M. Vardi, and P. Wolper, *An automata-theoretic approach to branching-time model checking*, Journal of the ACM, 47 (2000), pp. 312–360.

[14] G. Lenzi, *A hierarchy theorem for the mu-calculus*, in Proceedings of the 23rd International Colloquium on Automata, Languages and Programming, ICALP '96, F. Meyer auf der Heide and B. Monien, eds., vol. 1099 of Lecture Notes in Computer Science, Springer-Verlag, July 1996, pp. 87–97.

[15] R. Parikh, *Propositional game logic*, in IEEE Symposium on Foundations of Computer Science, IEEE, 1983, pp. 195–200.

[16] R. Parikh, *The logic of games and its applications*, Annals of discrete mathematics, 24 (1985), pp. 111–140.

[17] M. Pauly, *Logic for Social Software*, PhD thesis, University of Amsterdam, 2001.

[18] P. D. Seymour and R. Thomas, *Graph searching and a min-max theorem for tree-width*, J. Comb. Theory Ser. B, 58 (1993), pp. 22–33.

[19] C. Stirling, *Bisimulation, modal logic and model checking games*, Logic Journal of the IGPL, 7 (1999), pp. 103–124.

[20] P. Wolper, *Temporal logic can be more expressive*, Information and Control, 56 (1983), pp. 72–99.