PDL is the bisimulation-invariant fragment of Weak Chain Logic

Facundo Carreiro
Institute for Logic, Language and Computation (U. Amsterdam)
Email: fcarreiro@dc.uba.ar

***Abstract*—We introduce a new class of parity automata which, on trees, captures the expressive power of weak chain logic. This logic is a variant of monadic second-order logic which quantifies over finite chains. Using this new tool, we show that the bisimulation-invariant fragment of weak chain logic is equivalent to propositional dynamic logic.**

I. Introduction

A. Expressiveness modulo bisimilarity.

This article concerns the relative expressive power of languages when restricted to properties which are bisimulation-invariant. The interest in such expressiveness questions stems from applications where transition systems model computational processes, and bisimilar structures represent the *same* process. Seen from this perspective, properties of transition structures are only relevant if they are invariant under bisimilarity. This explains the importance of results of the form

$$L' \equiv L/\underline{\leftrightarrow},$$

stating that, one language L' is expressively complete with respect to the relevant (i.e., bisimulation-invariant) properties that can be formulated in another language L. In this setting, generally L is some rich yardstick formalism such as first-order or monadic second-order logic, and L' is some modal-style fragment of L, usually displaying much better computational behavior than the full language L.

A seminal result in the theory of modal logic is van Benthem's Characterization Theorem [1], stating that every bisimulation-invariant first-order formula is actually equivalent to (the standard translation of) a modal formula:

$$\mathrm{ML} \equiv \mathrm{FO}/\underline{\leftrightarrow}.$$

Over the years, a wealth of variants of the Characterization Theorem have been obtained. For a recent, rich source of van Benthem-style characterization results, see Dawar & Otto [2]. In this paper we are mainly interested is the work of Janin & Walukiewicz [3], who extended van Benthem's result to the setting of fixpoint logics, by proving that the modal μ-calculus ($\mu\mathrm{ML}$) is the bisimulation-invariant fragment of monadic second-order logic (MSO):

$$\mu\mathrm{ML} \equiv \mathrm{MSO}/\underline{\leftrightarrow}.$$

Despite the continued study of the connection between modal and classical logics there are still many logics which are not well understood and represent exciting problems. In particular, it is not known whether there is a natural classical logic whose bisimulation-invariant fragment corresponds to PDL (see [4, p. 91]), despite some results in this direction [5], [4], [6].

The language now called Propositional Dynamic Logic was first investigated by Fisher and Ladner [7] as a logic to reason about computer program execution. PDL extends the basic modal logic with an infinite collection of diamonds $\langle\pi\rangle$ where the intended intuitive interpretation of $\langle\pi\rangle\varphi$ is that "some terminating execution of the program π from the current state leads to a state satisfying φ".

One of the most important and characteristic features of PDL is that the program construction π^* (corresponding to iteration) endows PDL with second-order capabilities while still keeping it computationally well-behaved. For an extensive treatment of PDL we refer the reader to [8].

In this article, we take PDL as a starting point, and find a second-order logic whose bisimulation-invariant fragment is PDL. Our main result regarding this question is the following.

Theorem 1. PDL *is effectively equivalent to* $\mathrm{WCL}/\underline{\leftrightarrow}$.

Chain logic (CL) was introduced by Thomas in [9] and further studied in [10], [11], always in the context of *trees*. This logic is a variant of MSO which changes the usual second-order quantifier to a quantifier over chains. A chain on a tree $\mathbb{T}$ is a set such that all of its elements belong to the same branch of $\mathbb{T}$. However, in this article we will only work with a *weak* version of CL, denoted as WCL. This logic restricts quantification to *finite* chains.

B. The role of complete additivity.

The unifying notion across this article is that of *complete additivity*. As we will see, this notion has a strong connection with PDL and WCL, both on the logical and automata side.

A map $F : \wp(M) \to \wp(M)$ is said to be completely additive if for every non-empty family of subsets $\{P_i \subseteq M\}_{i\in I}$ it satisfies:

$$F(\bigcup_i P_i) = \bigcup_i F(P_i).$$

Equivalently, a map F is completely additive if it restricts to singletons, in the following sense:

Fact 1. *A map $F : \wp(M) \to \wp(M)$ is completely additive iff for every $X \subseteq M$ we have $F(X) = \bigcup_{x\in X} F(\{x\}) \cup F(\varnothing)$.*

Complete additivity has been studied (under the name 'continuity') by van Benthem [6], in the context of first-order

operations on relations that are *safe for* (that is, preserve) bisimulations. On the modal side, both Hollenberg [4] and Fontaine and Venema [12] gave syntactic characterizations of complete additivity for the modal μ-calculus.

Given a formula φ with a free variable p, the set of elements $s \in M$ of some model $\mathbb{M}$ which satisfy φ clearly depends on the set of elements in which p holds. This dependency can be formalized as a map:

$$F_p^{\varphi}(Y) := \{s \in M \mid \mathbb{M}[p \mapsto Y], s \Vdash \varphi\}.$$

A formula is then called completely additive in p if the induced map is completely additive.

Carreiro and Venema [13] showed that PDL is equivalent to the fragment μ_aML of the μ-calculus where the application of the least fixpoint operator is restricted to completely additive formulas.

Fact 2 ([13]). PDL *is effectively equivalent to* μ_aML.

Another important occurrence of complete additivity is in the context of automata for PDL and the automata that we will introduce for WCL.

C. Automata and expressiveness on trees.

It is difficult to over-stress the importance of automata-theoretic techniques in logical questions. The literature is vast in this topic, and we only name a few results which are relevant for the current article: Walukiewicz introduced *parity automata* for MSO on arbitrary trees [14]. These automata were crucial in proving that the modal μ-calculus is the bisimulation-invariant fragment of MSO [3]. On the modal side, other classes of parity automata were used to prove the small model property [15] and uniform interpolation [16] for the μ-calculus.

In this article we introduce a new class of parity automata characterizing WCL on trees. Before we turn to a description of these automata, we first have a look at the parity automata introduced by Walukiewicz [14], corresponding to MSO (over tree models).

We fix a set of proposition letters P and think of $\wp(\mathsf{P})$ as an *alphabet* or set of *colors*. An MSO-automaton is then a tuple $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$, where A is a finite set of states, a_I an initial state, and $\Omega : A \to \mathbb{N}$ is a parity function. The transition function Δ maps a pair $(a, c) \in A \times \wp(\mathsf{P})$ to a sentence in the monadic first-order language with equality $\mathrm{FOE}_1(A)$, of which the state space A provides the set of predicates. We shall refer to FOE_1 as the *one-step language* of MSO-automata, and denote this class of automata with $Aut(\mathrm{FOE}_1)$.

The automata that we consider in this article run on labelled transition systems and decide whether to accept or reject them. To take such decision we associate an acceptance game for an automaton $\mathbb{A}$ and a transition system $\mathbb{S}$. A match of this game consists of two players, $\exists$ and $\forall$, moving a token from one position to another. When such a match arrives at a so-called *basic* position, i.e., a position of the form $(a, t) \in A \times S$, the players consider the sentence $\Delta(a, c_t) \in \mathrm{FOE}_1^+(A)$, where $c_t \in \wp(\mathsf{P})$ is the color of t (that is, the set of proposition letters true at t). At this position $\exists$ has to turn the set $R[t]$ of successors of s into a *model* for the formula $\Delta(a, c_t)$ by coming up with an interpretation I of the monadic predicates $a \in A$ as subsets of $R[s]$, so that the resulting first-order structure $(R[s], I)$ makes the formula $\Delta(a, c_t)$ true. If a player cannot move, it loses the game, and infinite games are decided using the parity map.

Walukiewicz's result for MSO-automata is the following:

Fact 3 ([14]). MSO *and* $Aut(\mathrm{FOE}_1)$ *are effectively equivalent over tree models.*

These automata are shown to be closed under the operations of MSO, i.e., Boolean operators and quantification over sets.

The additive-weak parity automata that we introduce in Section IV are a restriction of MSO-automata. In order to state the constraints, observe that given a parity automaton $\mathbb{A}$ we can induce a graph on A by setting a transition from a to b if b occurs in $\Delta(a, c)$ for some $c \in \wp(\mathsf{P})$. A parity automaton $\mathbb{A}$ is called an additive-weak automaton if it satisfies the following constraints for every *maximal* strongly connected component $C \subseteq A$, states $a, b \in C$ and color $c \in \wp(\mathsf{P})$:

(weakness) $\Omega(a) = \Omega(b)$.

(additivity) for every color $c \in \wp(\mathsf{P})$:

If $\Omega(a)$ is odd, $\Delta(a, c)$ is completely additive in C.

If $\Omega(a)$ is even, $\Delta(a, c)$ is completely multiplicative in C.

The class of these automata is denoted by $Aut_{wa}(\mathrm{FOE}_1)$.

As opposed to $Aut(\mathrm{FOE}_1)$, this class is not closed under the existential quantification of MSO. We prove that, instead, it is closed under the operations of WCL. The second main contribution of this article is the following.

Theorem 2. *On trees, the following formalisms are expressively equivalent:*

(i) $Aut_{wa}(\mathrm{FOE}_1)$*: additive-weak automata on* FOE_1*,*

(ii) WCL*: weak chain logic.*

Moreover, the equivalence is given by effective *translations.*

The definition of the automata for WCL is similar to the definition of automata for PDL introduced by Carreiro and Venema [13]. PDL is characterized by additive-weak parity automata based on the one step language of first-order logic *without* equality:

Fact 4 ([13]). PDL *is effectively equivalent to* $Aut_{wa}(\mathrm{FO}_1)$.

In order to prove Theorem 2, we need a detailed analysis of certain fragments of FOE_1. The third contribution of this article is to provide, in Section III, a syntactical characterization of the monotone, completely additive and completely multiplicative fragments of *multi-sorted* monadic first-order logic with and without equality.

II. Preliminaries

A. General notation, transition systems and trees

Throughout this article we fix a set P of elements that will be called *proposition letters* and denoted with small Latin letters p, q, etc. We also fix a set D of *atomic actions*.

We use overlined boldface letters to represent sequences, for example a list of variables $\overline{\mathbf{x}} := x_1, \ldots, x_n$ or a sequence of sets $\overline{\mathbf{T}} \in \wp(S)^n$. We blur the distinction between sets and sequences: a sequence may be used as a set comprised of the elements of the list; in a similar way, we may assume a fixed order on a set and see it as a list. To simplify the notation, given a map $f : A^{n+m} \to B$ and $\overline{\mathbf{a}} \in A^n$, $\overline{\mathbf{a}}' \in A^m$ we write $f(\overline{\mathbf{a}}, \overline{\mathbf{a}}')$ to denote $f(a_1, \ldots, a_n, a'_1, \ldots, a'_m)$.

Given a binary relation $R \subseteq X \times Y$, for any element $x \in X$, we indicate with $R[x]$ the set $\{y \in Y \mid (x, y) \in R\}$ while R^+ and R^* are defined respectively as the transitive closure of R and the reflexive and transitive closure of R.

Transition systems. A *labeled transition system* (LTS) on propositions P and actions D is a tuple $\mathbb{S} = \langle S, R_{\ell \in \mathsf{D}}, \kappa, s_I \rangle$ where S is the universe or domain of $\mathbb{S}$; the map $\kappa : S \to \wp(\mathsf{P})$ is a marking (or coloring) of the elements in S; $R_\ell \subseteq S^2$ is the accessibility relation for the atomic action $\ell \in \mathsf{D}$; and $s_I \in S$ is a distinguished node. We use R without a subscript to denote the binary relation defined as $R := \bigcup_{\ell \in \mathsf{D}} R_\ell$.

Observe that a marking $\kappa : S \to \wp(\mathsf{P})$ can be seen as a valuation $\kappa^\natural : \mathsf{P} \to \wp(S)$ given by $\kappa^\natural(p) = \{s \in S \mid p \in \kappa(s)\}$. We say that $\mathbb{S}$ is *p-free* if $p \notin \mathsf{P}$ or $p \notin \kappa(s)$ for all $s \in S$. Given a set of propositions P' and $p \notin \mathsf{P}'$, a *p-extension* of a LTS $\mathbb{S} = \langle S, R_{\ell \in \mathsf{D}}, \kappa, s_I \rangle$ over P' is a transition system $\langle S, R_{\ell \in \mathsf{D}}, \kappa', s_I \rangle$ over $\mathsf{P}' \cup \{p\}$ such that $\kappa'(s) \setminus \{p\} = \kappa(s)$ for all $s \in S$. Given a set $X_p \subseteq S$, we use $\mathbb{S}[p \mapsto X_p]$ to denote the p-extension where $p \in \kappa'(s)$ iff $s \in X_p$.

Trees. A P*-tree* $\mathbb{T}$ is a LTS over P in which every node can be reached from s_I, and every node except s_I has a unique R-predecessor; the distinguished node s_I is called the *root* of $\mathbb{T}$. A tree is called *strict* when the range of R_ℓ and $R_{\ell'}$ is disjoint for every pair of actions $\ell \neq \ell'$.

The *tree unravelling* of an LTS $\mathbb{S}$ is given by the tree $\hat{\mathbb{S}} := \langle \hat{S}, \hat{R}_{\ell \in \mathsf{D}}, \hat{\kappa}, s_I \rangle$ where $\hat{S}$ is the set of (D-decorated) finite paths $s_I \to_{\ell_1} e_1 \to_{\ell_2} \cdots \to_{\ell_n} e_n$ in $\mathbb{S}$ stemming from s_I; $\hat{R}_\ell(t, t')$ holds iff t' is an extension of t through the relation ℓ; and the color of a path $t \in \hat{S}$ is given by the color of its last node in S. The ω-unravelling $\mathbb{S}^\omega$ of $\mathbb{S}$ is an unravelling which has ω-many copies of each node different from the root.

Remark 1. *The (ω-)unravelling of a LTS is a* strict *tree.*

Also observe that if there is only one action in D, the notion of tree and strict tree coincide.

Chains and generalized chains. Let $\mathbb{S}$ be an arbitrary model. A *chain* on $\mathbb{S}$ is a set $X \subseteq S$ such that (X, R^*) is a totally ordered set; i.e., it satisfies the following, for all $x, y \in X$:

(antisymmetry) if xR^*y and yR^*x then $x = y$,
(transitivity) if xR^*y and yR^*z then xR^*z,
(totality) xR^*y or yR^*x.

A *finite chain* (f.c.) is a chain on a finite set. A *generalized chain* (g.c.) is a set $X \subseteq S$ such that $X \subseteq P$, for some path P of $\mathbb{S}$. A *generalized finite chain* (g.f.c.) is a finite subset $X \subseteq S$ such that $X \subseteq P$, for some finite path P of $\mathbb{S}$.

Proposition 1. *Every chain on $\mathbb{S}$ is also a g.c. on $\mathbb{S}$.*

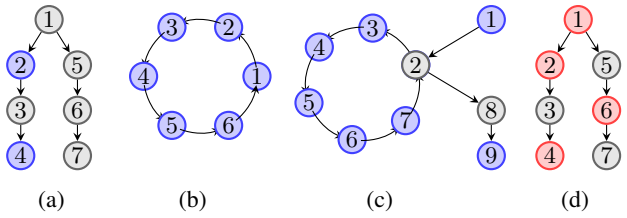

Figure 1. Examples and counter-examples of chains.

In Fig. 1 we show some examples of (generalized) chains and non-chains: in (a) the set $X_a = \{2, 4\}$ is a finite chain. In (b) the generalized finite chain $X_b = \{1, 2, 3, 4, 5, 6\}$ is witnessed, among others, by the path $3, 4, 5, 6, 1, 2$. Observe, however, that X_b is *not* a chain, since there is no possible total ordering of X_b by R^* (antisymmetry fails). In (c) the generalized finite chain $X_c = \{1, 3, 4, 5, 6, 7, 9\}$ is witnessed by the path $1, 2, \ldots, 7, 2, 8, 9$; observe that the element 2 is repeated in the path. Again, X_c is also not a finite chain. In the last example (d), the set $X_d = \{1, 2, 4, 6\}$ is not a generalized chain (and hence not a chain).

Fact 5. *On trees, chains and generalized chains coincide.*

B. Games

A match of a parity game $\mathcal{G}$ consists of two players, $\exists$ and $\forall$, moving a token from one position to another over a partitioned board $G = G_\exists \cup G_\forall$. For every position, players have a set of available moves. If during a match a player reaches a position with no admissible moves, he loses the match. If the match goes on forever then a parity map $\Omega : G \to \mathbb{N}$ is used to choose a winner. The winner is $\exists$ if the *minimum* parity which occurs infinitely often in the match is even, otherwise $\forall$ wins.

A *strategy* for player $\Pi \in \{\exists, \forall\}$ is, intuitively, a specification of choices to be made in the positions belonging to Π. Strategies for parity games can be taken to be *positional* or memory-free (see e.g. [17]) and therefore can be represented as a function $f_\Pi : G_\Pi \to G$. A match is f_Π*-guided* if for each position $u \in G_\Pi$ player Π chooses $f_\Pi(u)$ as next position.

We say that f is a *winning strategy* for Π if (i) for each f-guided match, the moves suggested by f are always available to Π and (ii) Π wins each f-guided match of the game. A *winning position* is one from which Π has a winning strategy.

C. Parity automata

We recall the definition of a parity automaton, adapted to our setting. In order to correctly handle transition systems, which have many relations $R_{\ell \in \mathsf{D}}$, we will work with one-step models and languages which are *multi-sorted*. Each sort will intuitively correspond to one of the relations.

Definition 1. *Given a set A and sorts $\mathcal{S} = \{\mathtt{s}_1, \ldots, \mathtt{s}_n\}$, we define a* one-step model *to be a tuple $\mathbf{D} = (D_{\mathtt{s}_1}, \ldots, D_{\mathtt{s}_n}, V)$ consisting of a domain D and sets $D_{\mathtt{s}_1}, \ldots, D_{\mathtt{s}_n}$ such that $\bigcup_{\mathtt{s}} D_{\mathtt{s}} = D$, and a valuation $V : A \to \wp D$. A one-step model is called* strict *when the sets $D_{\mathtt{s} \in \mathcal{S}}$ are pairwise disjoint, that is, when $D_{\mathtt{s}_1}, \ldots, D_{\mathtt{s}_n}$ is a partition of D. Depending*

on context, elements of A will be called monadic predicates*,* names *or* propositional variables*. When the sets $D_{\mathsf{s}\in\mathcal{S}}$ are not relevant we will just write the model as (D, V).*

A (multi-sorted) one-step language *is a map $\mathcal{L}_1$ assigning to each set A and sorts $\mathcal{S}$, a set $\mathcal{L}_1(A,\mathcal{S})$ of objects called* one-step formulas *over A (on sorts $\mathcal{S}$). When the sorts are understood from context (or fixed) we simply write $\mathcal{L}_1(A)$. We denote one-step languages with a* subscript *number one.*

We assume that one-step languages come with a truth *relation: given a one-step model $\mathbf{D}$, a formula $\varphi \in \mathcal{L}_1$ is either* true *or* false *in $\mathbf{D}$, denoted by, respectively, $\mathbf{D} \models \varphi$ and $\mathbf{D} \not\models \varphi$. We also assume that $\mathcal{L}_1$ has a* positive fragment *$\mathcal{L}_1^+$ characterizing monotonicity in the sense that a formula $\varphi \in \mathcal{L}_1(A,\mathcal{S})$ is (semantically) monotone iff it is equivalent to a formula $\varphi' \in \mathcal{L}_1^+(A,\mathcal{S})$.*

The one-step languages featuring in this paper are all induced by well-known logics. Examples include (multi-sorted) monadic first-order logic (with and without equality) and fragments of these languages.

Definition 2. *A* parity automaton *based on the one-step language $\mathcal{L}_1$, actions D and alphabet $\wp(\mathsf{P})$ is a tuple $\mathbb{A} = \langle A, \Delta, \Omega, a_I\rangle$, where A is a finite set of states of the automaton, $a_I \in A$ is the initial state, $\Delta : A\times\wp(\mathsf{P}) \to \mathcal{L}_1^+(A,\mathsf{D})$ is the transition map, and $\Omega : A \to \mathbb{N}$ is the parity map. The collection of such automata will be denoted by $Aut(\mathcal{L}_1,\mathsf{P},\mathsf{D})$. For the rest of the article we fix the set of actions D and omit it in our notation, we also omit the set P when clear from context or irrelevant.*

Acceptance and rejection of a transition system by an automaton is defined in terms of the following parity game.

Definition 3. *Given an automaton $\mathbb{A} = \langle A, \Delta, \Omega, a_I\rangle$ in $Aut(\mathcal{L}_1)$ and a P-transition system $\mathbb{S} = \langle S, R_{\ell\in\mathsf{D}}, \kappa, s_I\rangle$, the* acceptance game *$\mathcal{A}(\mathbb{A},\mathbb{S})$ of $\mathbb{A}$ on $\mathbb{S}$ is the parity game defined according to the rules of Table I. A transition system $\mathbb{S}$ is* accepted *by $\mathbb{A}$ iff $\exists$ has a winning strategy in $\mathcal{A}(\mathbb{A},\mathbb{S})@(a_I, s_I)$.*

D. Propositional Dynamic Logic

Definition 4. *The formulas of Propositional Dynamic Logic (*PDL*) on propositions P and atomic actions D are given by mutual induction on formulas and programs:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi\vee\varphi \mid \langle\pi\rangle\varphi$$
$$\pi ::= \ell \mid \pi;\pi \mid \pi\oplus\pi \mid \pi^* \mid \varphi?$$

where $p \in \mathsf{P}$ and $\ell \in \mathsf{D}$.

Given a transition system $\mathbb{S} = \langle S, R_{\ell\in\mathsf{D}}, \kappa, s_I\rangle$, the semantics of PDL is then given as usual on the Boolean operators and as follows on modal operators and propositions.

$$\mathbb{S} \Vdash p \quad \text{iff} \quad s_I \in \kappa^\natural(p)$$
$$\mathbb{S} \Vdash \langle\pi\rangle\varphi \quad \text{iff} \quad \exists t \in S \text{ s.t. } R_\pi^{\mathbb{S}}(s,t) \text{ and } \mathbb{S}[s_I \mapsto t] \Vdash \varphi$$

where $R_\pi^{\mathbb{S}}$ is the relation on $\mathbb{S}$ induced by the program π.

E. The modal μ-calculus

The language of the modal μ-calculus on P and D is given by the following grammar:

$$\varphi ::= q \mid \neg\varphi \mid \varphi\vee\varphi \mid \langle\ell\rangle\varphi \mid \mu p.\varphi$$

where $p, q \in \mathsf{P}$, $\ell \in \mathsf{D}$ and p is positive in φ (i.e., p is under an even number of negations). We use the standard convention that no variable is both free and bound in a formula and that every bound variable is fresh.

The semantics of this language is completely standard. Let $\mathbb{S} = \langle S, R_{\ell\in\mathsf{D}}, \kappa, s_I\rangle$ be a transition system and $\varphi \in \mu\mathrm{ML}$. We inductively define the *meaning* $[\![\varphi]\!]^{\mathbb{S}}$ which includes the following clause for the fixpoint operator:

$$[\![\mu p.\psi]\!]^{\mathbb{S}} := \bigcap\{X \subseteq S \mid X \supseteq F_p^\psi(X)\}$$

We say that φ is *true* in $\mathbb{S}$ (notation $\mathbb{S} \Vdash \varphi$) iff $s_I \in [\![\varphi]\!]^{\mathbb{S}}$.

F. Bisimulation

Bisimulation is a notion of behavioral equivalence between processes. For transition systems, it is defined as follows.

Definition 5. *Let $\mathbb{S}$ and $\mathbb{S}'$ be two transition systems. A* bisimulation *is a relation $Z \subseteq S \times S'$ such that for all $(t,t') \in Z$ and $\ell \in \mathsf{D}$ the following holds:*

(atom) *$p \in \kappa(t)$ iff $p \in \kappa'(t')$ for all $p \in \mathsf{P}$;*
(forth$_\ell$) *for all $s \in R_\ell[t]$ there is $s' \in R'_\ell[t']$ s.t. $(s,s') \in Z$;*
(back$_\ell$) *for all $s' \in R'_\ell[t']$ there is $s \in R_\ell[t]$ s.t. $(s,s') \in Z$.*

Two transition systems $\mathbb{S}$ and $\mathbb{S}'$ are bisimilar *(denoted $\mathbb{S} \leftrightarrows \mathbb{S}'$) if there is a bisimulation $Z \subseteq S \times S'$ containing (s_I, s'_I).*

Fact 6. *$\mathbb{S}$ and its unravelling $\hat{\mathbb{S}}$ are bisimilar, for all LTS $\mathbb{S}$.*

As observed in the introduction, a key concept in this paper is that of bisimulation invariance. Formally, it is defined as follows for an arbitrary language $\mathcal{L}$:

Definition 6. *A formula $\varphi \in \mathcal{L}$ is* bisimulation-invariant *if $\mathbb{S} \leftrightarrows \mathbb{S}'$ implies that $\mathbb{S} \Vdash \varphi$ iff $\mathbb{S}' \Vdash \varphi$, for all $\mathbb{S}$ and $\mathbb{S}'$.*

Fact 7. *Every $\varphi \in \mu\mathrm{ML}$ (and* PDL*) is bisimulation-invariant.*

G. Weak chain logic

As mentioned in the introduction, in this paper we will work with weak chain logic which, over trees, has quantifies over finite chains. On the other hand, we also want to consider this logic on the class of *all* transition systems. In this broader class, we will interpret the quantification over generalized finite chains. Formally, weak chain logic is given as follows.

Definition 7. *The one-sorted* weak chain logic *on a set of predicates P and actions D is given by*

$$\varphi ::= \Downarrow p \mid p \sqsubseteq q \mid R_\ell(p,q) \mid \neg\varphi \mid \varphi\vee\varphi \mid \exists_{wc} p.\varphi$$

where $p, q \in \mathsf{P}$ and $\ell \in \mathsf{D}$. We denote this logic by $\mathrm{WCL}(\mathsf{P},\mathsf{D})$ *and omit P and D when clear from context.*

Position	Player	Admissible moves	Parity
$(a,s) \in A \times S$	$\exists$	$\{V : A \to \wp(R[s]) \mid (R_{\ell_1}[s], \dots, R_{\ell_n}[s], V) \models \Delta(a, \kappa(s))\}$	$\Omega(a)$
$V : A \to \wp(S)$	$\forall$	$\{(b,t) \mid t \in V(b)\}$	$\max(\Omega[A])$

Table I
RULES FOR THE ACCEPTANCE GAME $\mathcal{A}(\mathbb{A}, \mathbb{S})$.

Definition 8. *Let $\mathbb{S} = \langle S, R_{\ell \in \mathsf{D}}, \kappa, s_I \rangle$ be a LTS. The semantics of* WCL *is defined as usual for the Booleans, and:*

$$\begin{aligned} \mathbb{S} \models \Downarrow p \quad & \textit{iff} \quad \kappa^\natural(p) = \{s_I\} \\ \mathbb{S} \models p \sqsubseteq q \quad & \textit{iff} \quad \kappa^\natural(p) \subseteq \kappa^\natural(q) \\ \mathbb{S} \models R_\ell(p,q) \quad & \textit{iff} \quad \forall s \in \kappa^\natural(p)\ \exists t \in \kappa^\natural(q)\ \textit{s.t.}\ sR_\ell t \\ \mathbb{S} \models \exists_{wc} p.\varphi \quad & \textit{iff} \quad \mathbb{S}[p \mapsto X] \models \varphi \textit{ for some } \text{g.f.c. } X \subseteq S. \end{aligned}$$

The reader may have expected a more standard two-sorted language for second-order logic, for example given by

$$\varphi ::= p(x) \mid R_\ell(x,y) \mid x \approx y \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\varphi \mid \exists_{wc} p.\varphi$$

where $p \in \mathsf{P}$, $\ell \in \mathsf{D}$ and $x, y \in \mathsf{iVar}$ (individual variables). We call this language 2WCL. This semantics of this language is completely standard, with $\exists x$ denoting first-order quantification and $\exists_{wc} p$ denoting second-order quantification (that is, in this case, quantification over generalized finite chains). Both definitions can be proved to be equivalent, however, Definition 8 will be better suited to work with automata.

H. Fixpoint extension of first-order logic

First-order logic with equality (FOE) on predicates P, actions D and individual variables iVar is given by:

$$\varphi ::= q(x) \mid R_\ell(x,y) \mid x \approx y \mid \exists x.\varphi \mid \neg\varphi \mid \varphi \vee \varphi$$

where $q \in \mathsf{P}$, $\ell \in \mathsf{D}$ and $x, y \in \mathsf{iVar}$. We use the standard definition of free variables of φ and denote them by $FV(\varphi)$.

In one of the sections of this paper, we will also need to use a fixpoint extension of first-order logic. Because of the individual variables, the syntax and semantics of the extension of FOE with a fixpoint operator is considerably more involved than for the μ-calculus.

Definition 9. *The* First-order logic with equality and unary fixpoints (μFOE) *extends* FOE *with a fixpoint operator $[\mathsf{LFP}_{p:x}.\varphi(p,x)](z)$, where $p \in \mathsf{P}$ and $x, z \in \mathsf{iVar}$.*

Observe that z is free in the fixpoint clause and the fixpoint operator binds the designated variable x and predicate p.

As usual with (extensions of) first-order logic, μFOE will be interpreted over models with an assignment. The semantics of the fixpoint formula $[\mathsf{LFP}_{p:x}.\varphi(p,x)](z)$ is the expected one [18]. Given a model $\mathbb{M}$, and assignment $g : \mathsf{iVar} \to M$ the map $F^{\varphi}_{p:x} : \wp(M) \to \wp(M)$ is defined as

$$F^{\varphi}_{p:x}(Y) := \{t \in M \mid \mathbb{M}[p \mapsto Y], g[x \mapsto t] \models \varphi(p,x)\}.$$

We set $\mathbb{M}, g \models [\mathsf{LFP}_{p:x}.\varphi(p,x)](z)$ iff $g(z) \in \mathsf{LFP}(F^{\varphi}_{p:x})$.

III. ONE-STEP LANGUAGES

In this section we study the properties of several one-step languages. These languages will be later used in Section IV to formally define additive-weak automata.

Definition 10. *The multi-sorted first-order one-step language $\mathrm{FOE}_1(A, \mathcal{S})$ is given by the sentences generated by:*

$$\varphi ::= a(x) \mid x \approx y \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x{:}\mathtt{s}.\varphi(x)$$

where $x, y \in \mathsf{iVar}$, $a \in A$ and $\mathtt{s} \in \mathcal{S}$. The positive fragment $\mathrm{FOE}^+_1(A, \mathcal{S})$ is given by the sentences generated by:

$$\varphi ::= a(x) \mid x \approx y \mid x \not\approx y \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathcal{Q}x{:}\mathtt{s}.\varphi(x)$$

where $x, y \in \mathsf{iVar}$, $a \in A$, $\mathtt{s} \in \mathcal{S}$ and $\mathcal{Q} \in \{\exists, \forall\}$.

$\mathrm{FO}_1(A, \mathcal{S})$ is defined as $\mathrm{FOE}_1(A, \mathcal{S})$ but without equality, and the positive fragment $\mathrm{FO}^+_1(A, \mathcal{S})$ is defined analogously.

As required by Definition 1, the fragments FOE^+_1 and FO^+_1 can be shown to characterize monotonicity for FOE_1 and FO_1, respectively. For each language $\mathcal{L}_1$ that we consider, we focus on $\mathcal{L}^+_1$, as it is the relevant fragment for defining the transition map of parity automata (*cf.* Definition 2). However, all our results can be extended to the full language $\mathcal{L}_1$, and to arbitrary one-step models (as opposed to strict one-step models).

A. Normal Form

Given a set of names A and $S \subseteq A$, we call the formula $\tau^+_S(x) := \bigwedge_{a \in S} a(x)$ a *positive A-type*. We usually blur the distinction between $\tau^+_S(x)$ and the set S and call S a positive A-type as well.

Many of the results in this article are based on transformations of the one-step formulas in the automata. The following normal form theorem will be pivotal, since it gives us a precise representation of the information in such formulas.

Theorem 3. *Every $\varphi \in \mathrm{FOE}^+_1(A)$ is equivalent (over strict models) to a sentence in the strict normal form $\bigvee \bigwedge_{\mathtt{s}} \nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi)_{\mathtt{s}}$ where each $\nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi)_{\mathtt{s}}$ is defined as*

$$\exists \overline{\mathbf{x}}{:}\mathtt{s}.\big(\mathrm{diff}(\overline{\mathbf{x}}) \wedge \bigwedge_i \tau^+_{T_i}(x_i) \wedge \forall z{:}\mathtt{s}.(\mathrm{diff}(\overline{\mathbf{x}}, z) \to \bigvee_{S \in \Pi} \tau^+_S(z))\big)$$

where $\overline{\mathbf{T}} \in \wp(A)^k$ for some k, $\mathtt{s} \in \mathcal{S}$ and $\Pi \subseteq \overline{\mathbf{T}}$. The predicate $\mathrm{diff}(\overline{\mathbf{y}})$ states that the elements $\overline{\mathbf{y}}$ are different.

Proof: This theorem is proved with a multi-sorted variant of Ehrenfeucht-Fraïssé games. Each disjunct in the normal form provides a full description of a class of one-step models where φ is true. This description is given by specifying the properties of the elements of each sort. A proof for the unsorted case can be found in [19, Section 3.1.2]. ∎

B. Complete additivity and multiplicativity

First we extend the notion of complete additivity explained in the introduction to complete additivity *in the product*, which will be applicable to maps $F : \wp(S)^n \to \wp(S)$.

Definition 11. *Given $\overline{\mathbf{X}} \in \wp(S)^n$ we say that $\overline{\mathbf{Y}} \in \wp(S)^n$ is an atom of $\overline{\mathbf{X}}$ iff $\overline{\mathbf{Y}} = (\varnothing, \dots, \{x_i\}, \dots, \varnothing)$ for some element $x_i \in X_i$ standing at some coordinate i. We say that $\overline{\mathbf{Q}}$ is a* quasi-atom *(q.a.) if it is an atom or $\overline{\mathbf{Q}} = (\varnothing, \dots, \varnothing)$.*

In this terminology, we can formulate the concept of complete additivity in the product by asking that F restricts to quasi-atoms; i.e., for every $\overline{\mathbf{P}} \in \wp(S)^n$, it should satisfy:

$$F(\overline{\mathbf{P}}) = \bigcup \{F(\overline{\mathbf{Q}}) \mid \overline{\mathbf{Q}} \text{ is a quasi-atom of } \overline{\mathbf{P}}\}.$$

Another way to read this last definition is that every $s \in F(\overline{\mathbf{P}})$ depends on at most one singleton on one of the coordinates.

Next we give a syntactic characterization of complete additivity and multiplicativity for both FO_1^+ and FOE_1^+. This is instrumental in a proper implementation of the **(additivity)** condition on the automata that we seek to define.

Definition 12. *We say that $\varphi \in \mathrm{FOE}_1(A)$ is* completely additive in $\{\overline{\mathbf{a}}\} \subseteq A$ *if φ is monotone in every a_i and, for every one-step model (D, V) and assignment $g : \mathsf{iVar} \to D$,*

If $(D, V), g \models \varphi$ then
$(D, V[\overline{\mathbf{a}} \mapsto \overline{\mathbf{Q}}]), g \models \varphi$ for some quasi-atom $\overline{\mathbf{Q}}$ of $V(\overline{\mathbf{a}})$.

The notion of complete multiplicativity is given dually. Note that these definitions also apply for $\mathrm{FO}_1^+(A) \subseteq \mathrm{FOE}_1^+(A)$.

Theorem 4. *A sentence of $\mathrm{FOE}_1^+(A, \mathcal{S})$ is completely additive in $A' \subseteq A$ iff it is equivalent to a sentence given by:*

$$\varphi ::= \psi \mid a(x) \mid \exists x{:}\mathtt{s}.\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \psi$$

where $a \in A'$, $\mathtt{s} \in \mathcal{S}$ and $\psi \in \mathrm{FOE}_1^+(A \setminus A', \mathcal{S})$. We denote this fragment as $\mathrm{FOE}_1^+\mathsf{ADD}_{A'}(A, \mathcal{S})$. The fragment $\mathrm{FOE}_1^+\mathsf{MUL}_{A'}(A, \mathcal{S})$ is defined dually, by switching $\exists/\forall$ and $\wedge/\vee$, and characterizes complete multiplicativity in A'.

Proof: The soundness of the fragment is easily proved by induction. For the challenging direction (i.e., completeness) it is possible to give a translation $(-)^\triangle : \mathrm{FOE}_1^+(A, \mathcal{S}) \to \mathrm{FOE}_1^+\mathsf{ADD}_{A'}(A, \mathcal{S})$ such that a formula $\varphi \in \mathrm{FOE}_1^+(A, \mathcal{S})$ is completely additive in A' if and only if $\varphi \equiv \varphi^\triangle$.

We only discuss the definition of such a translation for a special case. To define the translation it is useful to assume that φ is in normal form. Intuitively, the translation performs syntactic replacements in φ, removing the parts which cause it to fall outside the target fragment. The idea is that, if the formula is truly completely additive, these changes shouldn't change its meaning. For example, consider a formula $\nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi)_{\mathtt{s}}$ which is claimed to be completely additive in A'. Then, it should never be the case that

$$\exists a, b \in T_i \cap T_j \text{ for distinct } a, b \in A' \text{ or distinct } i, j \quad (!)$$

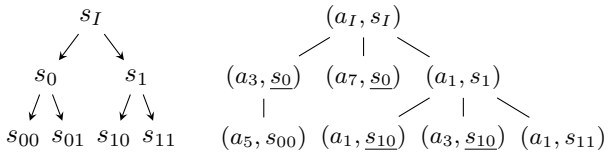

Figure 2. A tree $\mathbb{T}$ and $\mathbb{T}_f$ for a fixed strategy f for $\exists$.

since this would require that more than one element is colored with A'. For such formulas the translation is defined as:

$$\nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi)^\triangle_{\mathtt{s}} := \begin{cases} \bot & \text{if (!),} \\ \nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi^{\times}_{A'})_{\mathtt{s}} & \text{otherwise,} \end{cases}$$

where $\Pi^{\times}_{A'} := \{S \in \Pi \mid A' \cap S = \varnothing\}$. ∎

IV. Additive-weak parity automata

In this section we formally define the additive-weak automata introduced in Section I and discuss its closure under Boolean operations and (finite chain) projection.

Definition 13. *The class $Aut_{wa}(\mathrm{FOE}_1)$ of additive-weak automata is given by the automata $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ of $Aut(\mathrm{FOE}_1)$ such that for every* maximal *strongly connected component $C \subseteq A$ and states $a, b \in C$:*

(weakness) *$\Omega(a) = \Omega(b)$,*
(additivity) *for every color $c \in \wp(\mathsf{P})$:*
If $\Omega(a)$ is odd then $\Delta(a, c) \in \mathrm{FOE}_1^+\mathsf{ADD}_C(A)$, otherwise
if $\Omega(a)$ is even then $\Delta(a, c) \in \mathrm{FOE}_1^+\mathsf{MUL}_C(A)$.

The *weakness* condition has the effect of dividing the automaton in clusters of connected components with the same parity. If we think about trees, the *additivity* condition has the following effect: while a run of an additive automaton stays inside a connected component with odd parity, we can assume without loss of generality that the nodes of the tree coloured with some state of C form a *path* in the tree. The reason being that at each step –because of complete additivity– player $\exists$ can play a valuation where at most one node is colored with C. Therefore, a repetition of this step will define a path.

For the automata used in this article we need to combine the constraints for the horizontal and vertical dimensions, yielding automata with both the weakness and additivity constraints. The intuition is that we want to define a class of automata that works with *finite paths*.

A. Simulation theorem

One of the main technical results for parity automata is the so-called "Simulation Theorem", which was used to prove the closure of $Aut(\mathrm{FOE}_1)$ under projection:

Theorem 5 ([14], [20])**.** *Every $\mathbb{A} \in Aut(\mathrm{FOE}_1)$ is equivalent to a* non-deterministic *automaton $\mathbb{A}' \in Aut(\mathrm{FOE}_1)$.*

Very informally, an automaton $\mathbb{A}$ is called non-deterministic when in every acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$, if $\forall$ can choose to play both (a, s) and (b, s) at a given moment, then $a = b$. That is, $\forall$'s power boils down to being a *pathfinder* in $\mathbb{S}$.

Observe that if we fix a strategy f for $\exists$ for the game $\mathcal{A}(\mathbb{A}, \mathbb{T})$ then the whole game can be represented by a tree, whose nodes are the different admissible moves for $\forall$. We assume that the automaton is clear from context and denote such a tree by $\mathbb{T}_f$. Figure 2 shows the move-tree for $\forall$ for some fixed strategy f for $\exists$. Each branch of $\mathbb{T}_f$ represents a possible f-guided match. In this figure the chosen strategy for $\exists$ is *not* non-deterministic. The admissible moves which violate this condition are underlined.

Unfortunately, the transformation performed by Theorem 5 does not preserve the weakness and additivity conditions (see [21, Remark 3.5]), and therefore does not give us non-deterministic automata for the class $Aut_{wa}(\mathrm{FOE}_1)$.

However, we provide, for every $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1)$ and $p \in \mathsf{P}$, an automaton $\mathbb{A}^{\div}_p \in Aut_{wa}(\mathrm{FOE}_1)$ which, although not being fully non-deterministic, is specially tailored to prove the closure of $Aut_{wa}(\mathrm{FOE}_1)$ under *finite chain* projection. The state space of $\mathbb{A}^{\div}_p$ is the disjoint union of two parts:

- A *non-deterministic* part based on $\wp(A)$; and
- An *alternating* part based on A.

The non-deterministic part is basically the powerset construction of $\mathbb{A}$, and contains the initial state of the automaton. It will have very nice properties enforced by construction:

- It will behave non-deterministically,
- The parity will be 1, trivially satisfying **(weakness)**; and
- The transition map will be completely additive in $\wp(A)$.

The alternating part is a copy of $\mathbb{A}$ modified such that it cannot be used to read nodes colored with the propositional variable p. The automaton $\mathbb{A}^{\div}_p$ is therefore based both on states from A and on "macro-states" from $\wp(A)$. Moreover, the transition map of $\mathbb{A}^{\div}_p$ is defined such that if a match goes from the non-deterministic part to the alternating part, then it cannot come back. Successful runs of $\mathbb{A}^{\div}_p$ will have the property of processing only a *finite* amount of the input being in a macro-state and all the rest behaving exactly as $\mathbb{A}$ (but without reading any node colored with p). Figure 3 shows an schematic view of this construction.

Definition 14. *Given $\mathbb{A} \in Aut(\mathcal{L})$, a subset $B \subseteq A$ of the states of $\mathbb{A}$, and a tree $\mathbb{T}$; a strategy f for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{T})$ is:*

- Functional in B, *if for each node $s \in \mathbb{T}$ there is at most one $b \in B$ such that (b, s) belongs to $\mathbb{T}_f$.*
- Non-branching in B, *if all the nodes of $\mathbb{T}_f$ with a state from B belong to the same branch of $\mathbb{T}_f$.*
- Well-founded in B, *if the set of nodes of $\mathbb{T}_f$ with a state from B are all contained in a well-founded subtree of $\mathbb{T}_f$.*

We will construct $\mathbb{A}^{\div}_p$ such that it satisfies the properties of the above definition (for every tree and strategy) for the set of states $\wp(A)$. That is, for the non-deterministic part.

In order to define the non-deterministic part of $\mathbb{A}^{\div}_p$ we need to do two things: (1) first, lift the state space A of $\mathbb{A}$ to $\wp(A)$ and adapt the formulas in the transition map Δ of $\mathbb{A}$ accordingly; (2) ensure that the formulas of the non-deterministic part are completely additive in $\wp(A)$.

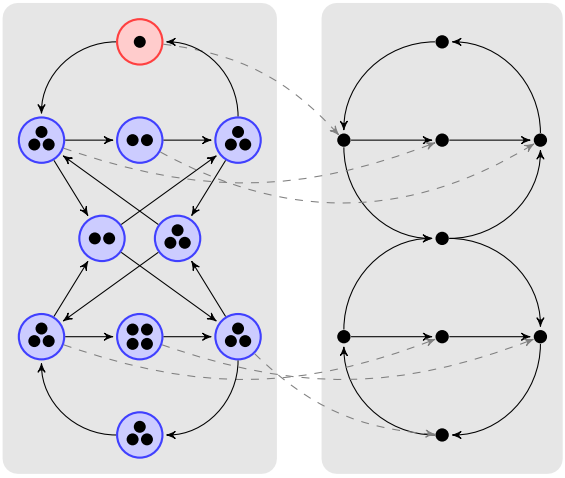

Figure 3. Two-part construction, initial state in red (illustrative)

The first step is standard: for $Q \in \wp(A)$ and $c \in \wp(\mathsf{P})$ one first considers the formula $\Phi_{Q,c} = \bigwedge_{a \in Q} \Delta(a, c)$. Since the macro-state Q is supposed to encode concurrent matches $\pi_{a \in Q}$ of the acceptance game, then it is natural that the transition map of Q, c should contain the moves for every such state. It is then customary to consider $\Phi_{Q,c}$ in normal form, and perform the required lifting to $\wp(A)$. For example, suppose that $\Phi_{Q,c}$ contains a conjunct $\alpha = \nabla^{+}_{\mathrm{FOE}}(\{a, b\}\{a\}, \{a\})_{\mathtt{s}}$ belonging to $\mathrm{FOE}^{+}_1(A)$. The usual approach is to define the lifted version $\alpha' := \nabla^{+}_{\mathrm{FOE}}(\{\{a, b\}\}\{\{a\}\}, \{\{a\}\})_{\mathtt{s}}$ which now belongs to $\mathrm{FOE}^{+}_1(\wp(A))$. The problem with this formula, is that it is not be completely additive in $\wp(A)$.

To overcome this problem, we will only perform a partial liftings on α. That is, we lift α in such a way that we obtain a formula which is completely additive in $\wp(A)$. For α, there are three ways to do this (changes are underlined):

- $\alpha'_1 := \nabla^{+}_{\mathrm{FOE}}(\{a, b\}\{a\}, \{a\})_{\mathtt{s}}$
- $\alpha'_2 := \nabla^{+}_{\mathrm{FOE}}(\underline{\{\{a, b\}\}}\{a\}, \{a\})_{\mathtt{s}}$
- $\alpha'_3 := \nabla^{+}_{\mathrm{FOE}}(\{a, b\}\underline{\{\{a\}\}}, \{a\})_{\mathtt{s}}$

We call these liftings 'non-branching liftings' of α and then define $\alpha' := \bigvee_i \alpha'_i$. The main intuition behind this definition is the 'non-branching' condition of Definition 14. What we want, is that at any given point of an acceptance game, at most one branching of $\mathbb{T}_f$ can stay in the non-deterministic part and the rest of the branches should go to the alternating part. That is, $\exists$ should never be required to colour more than one element with a state of $\wp(A)$. That is what we obtain by restricting the occurrence of $\wp(A)$ to at most one set, and only in the existential part of the normal form.

This construction can be done effectively for every α, and by extension for every $\Phi_{Q,c}$ by considering it in normal form and processing each part. We denote the resulting formula as $\Psi_{Q,c} \in \mathrm{FOE}_1(\wp(A) \cup A)$. We are now ready to define the two-part construction.

Definition 15. *Let the automaton $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ belong to $Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ and let p be a propositional variable.*

We define the two-part construct of $\mathbb{A}$ with respect to p *as the automaton* $\mathbb{A}^{\div}_p = \langle A^F, \Delta^F, \Omega^F, a^F_I \rangle \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ *given as follows:*

$$\begin{array}{lcllcl} A^F & := & A \cup \wp(A) & \Omega^F(Q) & := & 1 \\ a^F_I & := & \{a_I\} & \Omega^F(a) & := & \Omega(a) \end{array}$$

$$\Delta^F(Q,c) := \Psi_{Q,c}$$

$$\Delta^F(a,c) := \begin{cases} \bot & \text{if } p \in c, \\ \Delta(a,c) & \text{otherwise.} \end{cases}$$

The main properties of $\mathbb{A}^{\div}_p$ are stated in the following theorem.

Theorem 6. *For every* $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ *and* $p \in \mathsf{P}$ *there is an automaton* $\mathbb{A}^{\div}_p \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ *such that:*

1. *Every winning strategy for* $\exists$ *in* $\mathcal{A}(\mathbb{A}^{\div}_p, \mathbb{T})@(a^F_I, s_I)$ *can be assumed to satisfy all properties of Def. 14 for* $B = \wp(A)$.
2. *For every* P'*-tree* $\mathbb{T}$ *we have that* $\mathbb{A}^{\div}_p$ *accepts* $\mathbb{T}$ *iff* $\mathbb{A}$ *accepts* $\mathbb{T}[p \mapsto X_p]$ *for some finite chain* $X_p \subseteq T$.

Historical remarks and related results. The idea to use a two-part automata to preserve the weakness condition was first explicitly introduced in [21], [22]. The second critical element of this section is the enforcing of the additivity condition on the non-deterministic component. The core of this idea was developed in [23], [19] where it is applied for another notion called "continuity" (which is not complete additivity.)

B. Closure properties

The closure under Boolean operators is straightforward: the difficult case is the closure under complementation, where we crucially use that the one-step language FOE_1 is closed under Boolean duals. For the closure under finite chain projection we proceed as follows.

Definition 16. *Let* $\mathbb{A}$ *belong to* $Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}' \uplus \{p\})$. *We define the* chain projection of $\mathbb{A}$ over p *as the automaton* $\exists_{wc}p.\mathbb{A} := \langle A \cup \wp(A), \Delta^{\exists}, \Omega^{\exists}, \{a_I\} \rangle \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ *given as follows:*

$$\begin{array}{lcllcl} \Omega^{\exists}(a) & := & \Omega(a) & \Delta^{\exists}(a,c) & := & \Delta(a,c) \\ \Omega^{\exists}(Q) & := & 1 & \Delta^{\exists}(Q,c) & := & \Psi_{Q,c} \vee \Psi_{Q,c \cup \{p\}} \end{array}$$

for every $c \in \wp(\mathsf{P}')$, *where* $\Psi_{Q,c}$ *is as in Definition 15.*

The key observation to be made about the above definition is that $\exists_{wc}p.\mathbb{A}$ is actually defined based on the two-part construction $\mathbb{A}^{\div}_p$ (see Definition 15). The main change is that the non-deterministic part has been projected with respect to p. This can be observed in the definition of $\Delta^{\exists}(Q,c)$.

The closure under (finite chain) projection is relatively easy to obtain once we have a simulation theorem like Theorem 6. The following lemma is proved similar to the usual projection construction (*cf.* [21], [23]), crucially using the two-part construction and Theorem 6.

Lemma 1. *For each automaton* $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}' \uplus \{p\})$ *there is an automaton* $\exists_{wc}p.\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1, \mathsf{P}')$ *such that for every* P'*-tree* $\mathbb{T}$ *we have that* $\exists_{wc}p.\mathbb{A}$ *accepts* $\mathbb{T}$ *iff* $\mathbb{A}$ *accepts* $\mathbb{T}[p \mapsto X_p]$ *for some finite chain* $X_p \subseteq T$.

V. Logical characterizations of $Aut_{wa}(\mathrm{FOE}_1)$

The main objective of this section is to prove that $Aut_{wa}(\mathrm{FOE}_1)$ is effectively equivalent to WCL on trees. To do it, we will need to go through an intermediate first-order language $\mu_a\mathrm{FOE}^{\gg}$, which is a fragment of the fixpoint logic $\mu\mathrm{FOE}$. Namely, this fragment is invariant under generated submodels and the formulas under the (least) fixpoints are completely additive. Intuitively, $\mu_a\mathrm{FOE}^{\gg}$ is a 'modal' fragment of $\mu\mathrm{FOE}$, which additionally has the additivity restriction in the fixpoint construction (as $\mu_a\mathrm{ML}$).

By giving effective translations, we show that the following formalisms are expressively equivalent:

(i) WCL,
(ii) $Aut_{wa}(\mathrm{FOE}_1)$,
(iii) $\mu_a\mathrm{FOE}^{\gg}$.

As a corollary we obtain Theorem 2.

A. Forward-looking and additive fragment of $\mu\mathrm{FOE}$

First, we provide a definition of a completely additive fragment of $\mu\mathrm{FOE}$. This fragment can be seen as the first order equivalent of the completely additive fragment studied in [12], [13] for the modal μ-calculus.

Definition 17. *Let* $\mathsf{Q} \subseteq \mathsf{P}$ *be a set of monadic predicates. The fragment* $\mu\mathrm{FOEADD}_{\mathsf{Q}}(\mathsf{P}, \mathsf{D})$ *of* $\mu\mathrm{FOE}$ *is defined as follows:*

$$\varphi ::= \psi \mid q(x) \mid \exists x.\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \psi \mid [\mathsf{LFP}_{p:x}.\xi(p,x)](z)$$

where $q \in \mathsf{Q}$, $\psi \in \mu\mathrm{FOE}(\mathsf{P} \setminus \mathsf{Q}, \mathsf{D})$, $p \in \mathsf{P} \setminus \mathsf{Q}$ *and the formula* $\xi(p,x)$ *belongs to* $\mu\mathrm{FOEADD}_{\mathsf{Q} \cup \{p\}}(\mathsf{P}, \mathsf{D})$.

The fragment $\mu_a\mathrm{FOE}$ *of* $\mu\mathrm{FOE}$ *is given by restricting the least fixpoint operator to the completely additive fragment.*

Observe that the atomic formulas given by equality and relations are taken into account in the ψ clause.

Proposition 2. *Every* $\varphi \in \mu\mathrm{FOEADD}_{\mathsf{Q}}$ *is comp. add. in* Q.

It is easy to see that parity automata 'restrict to descendants.' That is, whenever the game $\mathcal{A}(\mathbb{A}, \mathbb{T})$ is at some basic position (a,s), the match can only continue to positions of the form (b,t) where $t \in R^*[s]$. Moreover, the game can never go back towards the root of the tree. Therefore, it is to be expected that formulas that correspond to parity automata also restrict to descendants. This concept is formalized as follows.

Definition 18. *The* forward-looking fragment $\mu\mathrm{FOE}^{\gg}$ *of* $\mu\mathrm{FOE}$ *is the smallest collection of formulas containing all atomic formulas, closed under Boolean connectives, and:*

- *If* $\varphi(\overline{\mathbf{x}}, y) \in \mu\mathrm{FOE}^{\gg}$ *and* $\overline{\mathbf{x}} = (x_1, \ldots, x_m)$ *are individual variables such that* $FV(\varphi) \subseteq \{\overline{\mathbf{x}}, y\}$ *then the formulas*

  $$\exists y.(R_\ell(x_j, y) \wedge \varphi(\overline{\mathbf{x}}, y)) \quad \text{and} \quad \forall y.(R_\ell(x_j, y) \to \varphi(\overline{\mathbf{x}}, y))$$

  are in $\mu\mathrm{FOE}^{\gg}$ *for all* $1 \leq j \leq m$ *and* $\ell \in \mathsf{D}$.
- *If* $\varphi(q, y) \in \mu\mathrm{FOE}^{\gg}$ *is monotone in* q *and* $FV(\varphi) \subseteq \{y\}$ *then* $[\mathsf{LFP}_{q:y}.\varphi(q,y)](x)$ *is in* $\mu\mathrm{FOE}^{\gg}$ *for all* $q \in \mathsf{P}$.

The forward-looking fragment of $\mu_a\mathrm{FOE}$ *is then defined as* $\mu_a\mathrm{FOE}^{\gg} := \mu_a\mathrm{FOE} \cap \mu\mathrm{FOE}^{\gg}$.

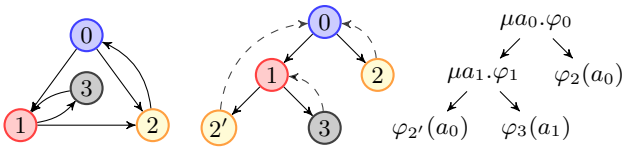

Figure 4. Automata, finite unravelling and formula structure.

Definition 19. *Let $\varphi \in \mu\mathrm{FOE}^{\nearrow}$ be such that $FV(\varphi) \subseteq X$. We say that φ* restricts *to descendants if for every model $\mathbb{M}$, assignment g and $p \in \mathsf{P}$ the following holds:*

$$\mathbb{M}, g \models \varphi \quad \textit{iff} \quad \mathbb{M}[p \mapsto R^*[X]], g \models \varphi$$

where $R^*[X] := \bigcup_{x \in X} R^*[g(x)]$.

The following lemma is easily proved by induction.

Lemma 2. *Every $\varphi \in \mu\mathrm{FOE}^{\nearrow}$ restricts to descendants.*

B. Translations

We now prove the main result of this section, giving effective translations between the languages.

From WCL ***to*** $Aut_{wa}(\mathrm{FOE}_1)$**.** This translation is given inductively using the closure properties of $Aut_{wa}(\mathrm{FOE}_1)$ proved in Section IV-B. It is easy to give the right automata for the atomic formulas, and the existential (finite chain) quantifier is taken care of by the closure under finite chain projection on the automata side.

From $Aut_{wa}(\mathrm{FOE}_1)$ ***to*** $\mu_a\mathrm{FOE}^{\nearrow}$**.** With each automaton $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1)$ we associate a formula $\varphi_{\mathbb{A}}(x) \in \mu_a\mathrm{FOE}^{\nearrow}$ with exactly one free variable, such that

$$\exists \text{ wins } \mathcal{A}(\mathbb{A}, \mathbb{T})@(a_I, s) \quad \text{iff} \quad \mathbb{T} \models \varphi_{\mathbb{A}}(s),$$

for every tree $\mathbb{T}$ and $s \in T$. We use a technique of partial unraveling of automata, similar to the one in Janin's habilitation thesis [24, Section 3.1].

The first step is to perform a *partial* unraveling on $\mathbb{A}$, and obtain an equivalent automaton $\mathbb{A}^u$ whose induced graph is a *tree with back edges*. Intuitively, the tree part of $\mathbb{A}^u$ is used to define the formula structure of $\varphi_{\mathbb{A}}$. On top of that, the nodes which are the target of back-edges will correspond to binding definitions of fixpoint variables. Fig. 4 shows an illustration of this process, where the target formula is taken to be in the μ-calculus. This is done for illustrative reasons; in our case we will actually have binding definitions given by the first-order fixpoint operator $[\mathsf{LFP}_{a_i:y}.\varphi_i(y)](x)$.

As we are working with additive-weak automata, it is important to observe that the partial unraveling preserves the properties of weakness and additivity. In other words,

Proposition 3. *If $\mathbb{A} \in Aut_{wa}(\mathcal{L})$ then $\mathbb{A}^u \in Aut_{wa}(\mathcal{L})$.*

The formula $\varphi_{\mathbb{A}}$ is then defined inductively on the tree structure of $\mathbb{A}^u$. The main idea is to define a formula $\xi_a(x) \in \mu_a\mathrm{FOE}^{\nearrow}$ for every state $a \in A$. We first take the first-order one-step formulas and replace the quantifiers $\exists y{:}\mathtt{s}.\alpha$ with $\exists y.R_{\mathtt{s}}(x,y) \wedge \alpha$ and then inductively substitute the predicates $a(y)$ in α by the formulas $\chi_a(y)$. The last step is to add the necessary fixpoint bindings.

From $\mu_a\mathrm{FOE}^{\nearrow}$ ***to*** WCL**.** In this case we will make use of the correspondence between the one-sorted WCL and the two-sorted 2WCL. The translation is given inductively, from $\mu_a\mathrm{FOE}^{\nearrow}$ to 2WCL. It is clear that the interesting part is the simulation of the fixpoint operator using the weak chain quantifier.

Let $\varphi = [\mathsf{LFP}_{p:y}.\psi(p,y)](x) \in \mu_a\mathrm{FOE}^{\nearrow}$ be such that $FV(\psi) \subseteq \{y\}$. By induction hypothesis we know that there is a formula $\psi'(p,y) \in 2\mathrm{WCL}$, which is equivalent to $\psi(p,y)$.

Proposition 4. *The formula $\psi'(p,y)$ restricts to descendants and is completely additive in p.*

Proof: By hypothesis $\psi(p,y)$ belongs to $\mu\mathrm{FOEADD}_p$. Therefore, by Proposition 2 it is completely additive in p. As ψ also belongs to $\mu\mathrm{FOE}^{\nearrow}$, by Lemma 2 it restricts to successors. It only remains to observe that as ψ' is equivalent to ψ then these properties also hold for ψ'. ∎

From this proposition we can deduce that the map $F_{\psi'}$ induced by ψ' is completely additive and restrict to descendants as well, in the following sense.

Definition 20. *A map $G : \wp(S)^n \to \wp(S)$ on a model $\mathbb{M}$* restricts to descendants *if for every $s \in S$ and $\overline{\mathbf{X}} \in \wp(S)^n$ we have that $s \in G(\overline{\mathbf{X}})$ iff $s \in G(\overline{\mathbf{X}} \cap R^*[s])$.*

The following theorem, which will be critical for the translation, gives a characterization of the fixpoint of maps that are both completely additive and restrict to descendants.

Theorem 7. *Let $F : \wp(S) \to \wp(S)$ be completely additive and restrict to descendants. For every $s \in S$ we have that $s \in \mathsf{LFP}(F)$ if and only if $s \in \mathsf{LFP}(F_{\restriction Y})$ for some generalized finite chain Y.*

Proof: To prove this theorem it is useful to recall that an element belongs to the least fixpoint of a monotone map iff it belongs to one of its approximants. For completely additive maps, it is also enough to consider the finite approximants. The (finite) approximants of F are defined as $F^0(\varnothing) = \varnothing$ and $F^{i+1}(\varnothing) = F(F^i(\varnothing))$. The key observation to obtain this theorem is that, as F is completely additive, an element $s \in F(F^i(\varnothing))$ iff $s \in F(\{t\})$ for some $t \in F^i(\varnothing)$. Moreover, because F restricts to descendants, t can be chosen to be among the succesors of s. Repeating this process, we get a finite chain of elements. ∎

To finish, define the translation $([\mathsf{LFP}_{p:y}.\psi(p,y)](x))^t$ as

$$\exists_{wc} Y.\big(\forall_{wc} X \subseteq Y. X \in \mathsf{PRE}(F_Y) \to x \in X\big)$$

where $X \in \mathsf{PRE}(F_Y) := \forall v.\psi'(X,v) \wedge v \in Y \to v \in X$.

Proposition 5. *$\varphi \equiv \varphi^t$ (on trees), for all $\varphi \in \mu_a\mathrm{FOE}^{\nearrow}$.*

Proof: To justify the translation we proceed as follows: first recall that the standard translation of $[\mathsf{LFP}_{p:y}.\psi(p,y)](x)$ into MSO (not WCL) is given by:

$$\forall W.\big(W \in \mathsf{PRE}(F^{\psi}) \to x \in W\big) \qquad (*)$$

where $W \in \mathsf{PRE}(F^\psi)$ expresses that W is a prefixpoint of the map $F^\psi : \wp(S) \to \wp(S)$. This translation is based on the following fact about fixpoints of monotone maps:

$$s \in \mathsf{LFP}(F^\psi) \quad \text{iff} \quad s \in \bigcap\{W \subseteq S \mid W \in \mathsf{PRE}(F^\psi)\}.$$

In our translation, however, we cannot make use of the arbitrary set quantifier $\exists W$, since we are dealing with WCL. The crucial observation is that, as $F^\psi : \wp(T) \to \wp(T)$ is completely additive and restricts to descendants, then we can use Theorem 7 to prove that we can restrict to finite chains, in the following sense:

Claim. For all $s \in T$ we have $s \in \mathsf{LFP}(F^\psi)$ if and only if $s \in \bigcap\{W \subseteq Y \mid W \in \mathsf{PRE}(F^\psi_{\restriction Y})\}$ for some finite chain Y.

Therefore, our translation basically expresses the same as $(*)$ but relativized to a finite chain Y. ∎

VI. Expressiveness modulo bisimilarity

In this section, we characterize the bisimulation-invariant fragment of the main formalisms that we have been using throughout the article. Our final objective is to show that

$$\mathrm{PDL} \equiv \mathrm{WCL}/\underline{\leftrightarrow}.$$

That is, we will prove Theorem 1. Moreover, we show that the equivalence is effective. First, we give a connection between $Aut_{wa}(\mathrm{FOE}_1)$ and $Aut_{wa}(\mathrm{FO}_1)$ which will be crucial for the bisimulation-invariance result.

A. Connecting $Aut_{wa}(\mathrm{FOE}_1)$ and $Aut_{wa}(\mathrm{FO}_1)$

We define a construction $(-)^\bullet : Aut_{wa}(\mathrm{FOE}_1) \to Aut_{wa}(\mathrm{FO}_1)$ such that for every automaton $\mathbb{A}$ and LTS $\mathbb{S}$:

$$\mathbb{A}^\bullet \text{ accepts } \mathbb{S} \quad \text{iff} \quad \mathbb{A} \text{ accepts } \mathbb{S}^\omega. \tag{$\dagger$}$$

As we shall see, this map is completely determined at the one-step level, i.e., by a connection between FOE_1 and FO_1.

Definition 21. *We define the (partial) one-step translation $(-)^\bullet_1$ which transforms formulas of $\mathrm{FOE}^+_1(A, \mathcal{S})$ which are in strict normal form into formulas of $\mathrm{FO}^+_1(A, \mathcal{S})$, as follows:*

$$(\nabla^+_{\mathrm{FOE}}(\overline{\mathbf{T}}, \Pi)_{\mathtt{s}})^\bullet_1 := \bigwedge_i \exists x_i{:}\mathtt{s}.\tau^+_{T_i}(x_i) \wedge \forall x{:}\mathtt{s}. \bigvee_{S \in \Pi} \tau^+_S(x).$$

The key property of this one-step translation is the following.

Proposition 6. *For every one-step model $(D_1, \ldots, D_n, V)$ and $\alpha \in \mathrm{FOE}^+_1(A)$ we have that $(D_1, \ldots, D_n, V) \models \alpha^\bullet$ if and only if $(D_1 \times \{1\} \times \omega, \ldots, D_n \times \{n\} \times \omega, V_\pi) \models \alpha$, where $V_\pi(a) := \{(d, i, k) \mid d \in V(a), 1 \leq i \leq n, k \in \omega\}$.*

The one-step model on the left (call it $\mathbf{D}$) is an arbitrary (i.e., not necessarily strict) one-step model, whereas the one on the right (call it $\mathbf{D}_\omega$) is *strict*. The strictness is obtained by tagging elements of D_i. Also, observe that $\mathbf{D}_\omega$ has ω-many copies of each element of $\mathbf{D}$.

The following proposition will be crucial for the development of this section. It states that, on strict trees, we can assume the transition map of our automata to be in strict normal form. This is easily achieved by transforming the transition map using Theorem 3.

Proposition 7. *For every $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1)$ we can effectivelly construct an automaton $\mathbb{A}' \in Aut_{wa}(\mathrm{FOE}_1)$ such that (1) $\mathbb{A} \equiv \mathbb{A}'$ over strict trees and (2) the transition map of $\mathbb{A}'$ is in strict normal form (*cf. *Theorem 3.)*

Finally, we can give the main definition of this section. That is, we define $\mathbb{A}^\bullet$ for every $\mathbb{A}$. This definition will be tailored to satisfy (†). Therefore, it is worth reminding that as $\mathbb{A}$ is run on $\mathbb{S}^\omega$ (which is a strict tree) we can assume that the transition map is in *strict* normal form.

Definition 22. *Let $\mathbb{A} = \langle A, \Delta, \Omega, a_I\rangle$ be an automaton in $Aut(\mathrm{FOE}_1)$. Using Proposition 7 we assume that the transition map is in strict normal form. We define the automaton $\mathbb{A}^\bullet := \langle A, \Delta^\bullet, \Omega, a_I\rangle$ in $Aut(\mathrm{FO}_1)$ by putting:*

$$\Delta^\bullet(a, c) := (\Delta(a, c))^\bullet_1.$$

for each $(a, c) \in A \times \wp(\mathsf{P})$.

First, it needs to be checked that the construction $(-)^\bullet$, which has been defined for arbitrary automata in $Aut(\mathrm{FOE}_1)$, transforms the additive-weak automata of $Aut_{wa}(\mathrm{FOE}_1)$ into automata in the right class, that is, $Aut_{wa}(\mathrm{FO}_1)$.

Proposition 8. $\mathbb{A}^\bullet \in Aut_{wa}(\mathrm{FO}_1)$ *for* $\mathbb{A} \in Aut_{wa}(\mathrm{FOE}_1)$.

Proof: This can be verified by a straightforward inspection, at the one-step level: if a formula $\alpha \in \mathrm{FOE}^+_1(A)$ belongs to the fragment $\mathrm{FOE}^+_1\mathsf{ADD}_{A'}(A)$, then $\alpha^\bullet_1$ lands in $\mathrm{FO}^+_1\mathsf{ADD}_{A'}(A)$, i.e., the fragment of $\mathrm{FOE}^+_1\mathsf{ADD}_{A'}(A)$ without equality. ∎

We are now ready to prove the main lemma of this section.

Lemma 3. $\mathbb{A}^\bullet$ *accepts* $\mathbb{S}$ *iff* $\mathbb{A}$ *accepts* $\mathbb{S}^\omega$*, for every LTS* $\mathbb{S}$*.*

Proof: The proof of this lemma is based on a fairly routine comparison of the acceptance games $\mathcal{A}(\mathbb{A}^\bullet, \mathbb{S})$ and $\mathcal{A}(\mathbb{A}, \mathbb{S}^\omega)$, using the fact that $\mathbb{S}^\omega$ is a strict tree. A similar proof for the unsorted case can be found in [3], [21], [22]. ∎

Remark 2. *For each automaton $\mathbb{A}$ it is possible to give a number k such that $\mathbb{A}^\bullet$ accepts $\mathbb{S}$ iff $\mathbb{A}$ accepts $\mathbb{S}^k$, by taking $k := \max\{n \mid \mathtt{diff}(x_1, \ldots, x_n)$ in $\mathbb{A}\} + 1$. Therefore our results transfer to the class of finitely branching trees as well.*

B. Bisimulation-invariant fragment of WCL

In this section we prove that $\mathrm{PDL} \equiv \mathrm{WCL}/\underline{\leftrightarrow}$. Moreover, we prove that the equivalence is effective.

One of the inclusions is given by a translation from PDL to WCL. We prove this through a detour via the modal μ-calculus. By Fact 2 we know that PDL is equivalent to the fragment $\mu_a\mathrm{ML}$ where the fixpoint operator $\mu p.\varphi$ is restricted to formulas which are completely additive in p. We will therefore give a translation $\mathrm{ST}^{wc}_x : \mu_a\mathrm{ML} \to 2\mathrm{WCL}$ which proves that $\mathrm{PDL} \leq \mathrm{WCL}$. The idea is to use almost the same translation as in Section V-B where we prove that $\mu_a\mathrm{FOE} \leq \mathrm{WCL}$ on trees.

The only interesting case of the translation is the fixpoint operator. Let $\varphi = \mu p.\psi(p)$ where ψ is completely additive in p. We state the following claim

Claim. The formula $\psi \in \mu_a\mathrm{ML}$ is completely additive in p and restricts to descendants.

This is clear because the formula belongs to $\mu\mathrm{ML}$. These formulas are invariant under generated submodels, in particular, they restrict to descendants. To finish, define the translation of the fixpoint as follows:

$$\mathrm{ST}_x^{wc}(\mu p.\psi) := \exists_{wc} Y.\big(\forall_{wc} Z \subseteq Y.Z \in \mathsf{PRE}(F_Y) \to x \in Z\big)$$
$$Z \in \mathsf{PRE}(F_Y) := \forall v.\mathrm{ST}_v^{wc}(\psi)[p \mapsto Z] \wedge v \in Y \to v \in Z.$$

The correctness of this translation is justified by Theorem 7 and a minor modification of the proof of Proposition 5.

For the other inclusion we state the following stronger lemma.

Lemma 4. *There is an effective translation* $(-)_\blacktriangledown : \mathrm{WCL} \to \mathrm{PDL}$ *such that for every* $\varphi \in \mathrm{WCL}$ *we have that* $\varphi \equiv \varphi_\blacktriangledown$ *iff* φ *is bisimulation-invariant.*

Proof: The translation $(-)_\blacktriangledown : \mathrm{WCL} \to \mathrm{PDL}$ is defined as follows: given a formula $\varphi \in \mathrm{WCL}$ we first construct an automaton $\mathbb{A}_\varphi \in Aut_{wa}(\mathrm{FOE}_1)$ using Theorem 2. Next, we compute the automaton $\mathbb{A}^\bullet_\varphi \in Aut_{wa}(\mathrm{FO}_1)$ using Lemma 3. To finish, we use Fact 4 to get a formula $\varphi_\blacktriangledown \in \mathrm{PDL}$.

Claim. $\varphi \equiv \varphi_\blacktriangledown$ iff φ is invariant under bisimulation.

The left to right direction is trivial because $\varphi_\blacktriangledown \in \mathrm{PDL}$, therefore if $\varphi \equiv \varphi_\blacktriangledown$ it also has to be invariant under bisimulation. The opposite direction is obtained by the following chain of equivalences:

$$\begin{aligned}
\mathbb{S} \models \varphi \quad &\text{iff} \quad \mathbb{S}^\omega \models \varphi && (\varphi \text{ bisimulation invariant})\\
&\text{iff} \quad \mathbb{S}^\omega \models \mathbb{A}_\varphi. && (\text{Theorem 2})\\
&\text{iff} \quad \mathbb{S} \Vdash \mathbb{A}^\bullet_\varphi && (\text{Lemma 3})\\
&\text{iff} \quad \mathbb{S} \Vdash \varphi_\blacktriangledown. && (\text{Fact 4})
\end{aligned}$$

This finishes the proof of the lemma. ∎

As a corollary of this lemma we get Theorem 1.

VII. Conclusions and future work

In this article we proved two characterization results: one is the automata characterization of WCL on trees, and the other is the characterization of the bisimulation-invariant fragment of WCL as PDL. In order to obtain these results, we also had to develop many secondary results that are of independent interest: among others, we gave characterizations of fixpoints of completely additive maps which restrict to successors (Theorem 7), normal forms and completely additive fragments for FOE_1 (Section III).

Selected open questions. There are many old an new open problems related to the content of this paper. In the following non-exhaustive list we mention a few of them:

The confusion conjecture: In [11] Bojańczyk defines a notion of 'confusion' and conjectures that a regular language (i.e., MSO definable) of finite trees is definable in CL iff it contains no confusion. A remarkable property of the notion of confusion is that it is *decidable* whether a language has it or not. As the results of our paper transfer to finite trees (and $\mathrm{CL} \equiv \mathrm{WCL}$ in that class) the conjecture implies that a language definable in the mu-calculus (on finite trees) is definable in PDL iff it contains no confusion. It is a major open problem whether we can decide if an arbitrary formula of $\mu\mathrm{ML}$ is equivalent to some formula in PDL. Therefore, it would be important to check the confusion conjecture.

Automata for Monadic Path Logic: Moller and Rabinovich [25] show that the full computation tree logic (CTL^*) corresponds, on trees, to the bisimulation-invariant fragment of monadic path logic (MPL). These last logic is a variant of MSO which quantifies over full (finite or infinite) paths. In contrast, WCL quantifies over subsets of finite paths. The authors do not use an automata-theoretic approach and leave this approach as an open question. Given the similarities, it would be interesting to see if our approach can be applied to get an automata characterization of MPL and characterize its bisimulation-invariant fragment on the class of all models.

PDL *and* $\mathrm{FO}(\mathrm{TC}^1)$: The logic $\mathrm{FO}(\mathrm{TC}^1)$ is an extension of first-order logic with (unary) transitive closure. The bisimulation-invariant fragment of this logic has not been characterized and it has long been suggested that it should be PDL. The logics WCL and $\mathrm{FO}(\mathrm{TC}^1)$ can be shown to be incomparable in expressive power, but it is possible that the tools developed in this article could be useful to obtain a characterization result for $\mathrm{FO}(\mathrm{TC}^1)$.

Acknowledgements

The author is grateful to Yde Venema for reading and commenting on *many* previous versions of this document and to Alessandro Facchini, Sumit Sourabh, Fabio Zanasi and Balder ten Cate for fruitful discussions on the topics of this article. Additionally, the author would like to thank the anonymous referees for their comments.

References

[1] J. van Benthem, "Modal correspondence theory," Ph.D. dissertation, Universiteit van Amsterdam, 1977.

[2] A. Dawar and M. Otto, "Modal characterisation theorems over special classes of frames," *Ann. Pure Appl. Logic*, vol. 161, no. 1, pp. 1–42, 2009.

[3] D. Janin and I. Walukiewicz, "On the expressive completeness of the propositional μ-calculus with respect to monadic second order logic," in *CONCUR '96*. London, UK: Springer-Verlag, 1996, pp. 263–277.

[4] M. Hollenberg, "Logic and bisimulation," Ph.D. dissertation, University of Utrecht, 1998.

[5] J. van Benthem, "Program constructions that are safe for bisimulation," *Studia Logica*, vol. 60, no. 2, pp. 311–330, 1998.

[6] ——, *Exploring Logical Dynamics*. Center for the Study of Language and Information, 1996.

[7] M. J. Fischer and R. E. Ladner, "Propositional dynamic logic of regular programs," *J. Comput. Syst. Sci.*, vol. 18, no. 2, pp. 194–211, 1979.

[8] D. Harel, J. Tiuryn, and D. Kozen, *Dynamic Logic*. Cambridge, MA, USA: MIT Press, 2000.

[9] W. Thomas, "Logical aspects in the study of tree languages," in *Proc. Of the Conference on Ninth Colloquium on Trees in Algebra and Programming*. New York, NY, USA: Cambridge University Press, 1984, pp. 31–49.

[10] ——, "Languages, automata, and logic," in *Handbook of Formal Languages*. Springer, 1996, pp. 389–455.

[11] M. Bojańczyk, "Decidable properties of tree languages," Ph.D. Thesis, University of Warsaw, 2004.

[12] G. Fontaine, "Modal fixpoint logic: some model-theoretic questions," Ph.D. dissertation, ILLC, University of Amsterdam, 2010.

[13] F. Carreiro and Y. Venema, "PDL inside the μ-calculus: a syntactic and an automata-theoretic characterization," in *AiML 2014*. College Publications, 2014, pp. 74–93.

[14] I. Walukiewicz, "Monadic second order logic on tree-like structures," in *STACS*, ser. LNCS, vol. 1046. Springer, 1996, pp. 401–413.

[15] G. Fontaine, R. Leal, and Y. Venema, "Automata for coalgebras: An approach using predicate liftings," in *Automata, Languages and Programming*, ser. LNCS. Springer Berlin Heidelberg, 2010, vol. 6199, pp. 381–392.

[16] G. D'Agostino and M. Hollenberg, "Logical Questions Concerning the μ-Calculus: Interpolation, Lyndon and Łoś-Tarski," *The Journal of Symbolic Logic*, vol. 65, no. 1, pp. 310–332, 2000.

[17] E. A. Emerson and C. S. Jutla, "Tree automata, μ-calculus and determinacy (extended abstract)," in *FOCS*. IEEE Computer Society, 1991, pp. 368–377.

[18] A. Chandra and D. Harel, "Structure and complexity of relational queries," *J. of Comp. and Sys. Sci.*, vol. 25, no. 1, pp. 99–128, 1982.

[19] F. Carreiro, A. Facchini, Y. Venema, and F. Zanasi, "Weak MSO: Automata and expressiveness modulo bisimilarity," *CoRR*, vol. abs/1401.4374, 2014, online: http://arxiv.org/abs/1401.4374.

[20] I. Walukiewicz, "Monadic second-order logic on tree-like structures," *Theor. Comput. Sci.*, vol. 275, no. 1-2, pp. 311–346, 2002.

[21] F. Zanasi, "Expressiveness of monadic second order logics on infinite trees of arbitrary branching degree," Master's thesis, ILLC, Universiteit van Amsterdam, the Netherlands, 2012.

[22] A. Facchini, Y. Venema, and F. Zanasi, "A characterization theorem for the alternation-free fragment of the modal μ-calculus," in *LICS*. IEEE Computer Society, 2013, pp. 478–487.

[23] F. Carreiro, A. Facchini, Y. Venema, and F. Zanasi, "Weak MSO: Automata and expressiveness modulo bisimilarity," in *CSL-LICS '14*. ACM, 2014, pp. 27–37.

[24] D. Janin, "Contributions to formal methods: games, logic and automata," 2006, habilitation thesis.

[25] F. Moller and A. Rabinovich, "On the expressive power of CTL*," *Logic in Computer Science*, vol. 0, p. 360, 1999.