PDL Inside the μ-calculus: A Syntactic and an Automata-theoretic Characterization

Facundo Carreiro [1] Yde Venema [2]

Institute for Logic, Language and Computation
Universiteit van Amsterdam, The Netherlands

Abstract

It is well known that Propositional Dynamic Logic (PDL) can be seen as a fragment of the modal μ-calculus. In this paper we provide an exact syntactic characterization of the fragments of the μ-calculus that correspond to PDL and to test-free PDL. In addition we give automata-theoretic characterizations for PDL, with and without tests, which shed light on the relation between these logics and the modal μ-calculus and provide a new framework for the development of the theory of PDL.

Keywords: propositional dynamic logic, automata theory, modal μ-calculus.

1 Introduction

The language now called Propositional Dynamic Logic was first investigated by Fisher and Ladner [3] 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".

The inductive structure of programs is made explicit in PDL's syntax, as complex programs are built out of atomic programs using four program constructors. Formally, the formulas of full PDL are given by a mutual induction:

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

where p is a proposition letter and d is an atomic action (or atomic program). Test-free PDL is a variant of full PDL excluding use of the test operator (?).

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 [7].

[1] E-mail: fcarreiro@dc.uba.ar

[2] E-mail: y.venema@uva.nl

Modal μ-calculus. The modal μ-calculus (μML) was introduced in its present form by Dexter Kozen [10]. It is highly expressive, corresponding to the bisimulation-invariant fragment of monadic second-order logic [9]. The modal μ-calculus is a very expressive language, subsuming a vast amount of dynamic and temporal logics such as PDL, CTL* and Game Logic. Yet μML is computationally well-behaved, and enjoys some excellent meta-logical properties, such as *uniform interpolation* [2].

The language of the modal μ-calculus on a set of propositions P and atomic actions D is given by the following grammar:

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

where $p, q \in \mathsf{P}$, $d \in \mathsf{D}$ and p is positive in φ (i.e., all occurrences of p are under an even number of negations).

The semantics of this language is completely standard. Observe that, given a Kripke model $\mathbb{S}$ and a formula φ with p free, the extension $[\![\varphi]\!]^{\mathbb{S}}$ of φ in $\mathbb{S}$ depends on the set of points where p holds. This dependence can be formalized as a map $\varphi_p^{\mathbb{S}} : \wp(S) \to \wp(S)$. The semantics of the least fixpoint operator is then given by interpreting $[\![\mu p.\varphi]\!]^{\mathbb{S}}$ as the least fixpoint of $\varphi_p^{\mathbb{S}}$.

Relative expressive power. It is well known that PDL can be translated to μML. However, to the best of our knowledge the exact fragment of μML that corresponds to PDL has not been characterized. As we will see in this article, the key notion leading to such a characterization is that of *complete additivity*. A formula φ is said to be completely p-additive if for any family of subsets $\{P_i\}_i \in I$ with $P_i \subseteq S$ it satisfies

$$\varphi_p^{\mathbb{S}}(\bigcup_i P_i) = \bigcup_i \varphi_p^{\mathbb{S}}(P_i). \tag{1}$$

An equivalent characterization of complete additivity is the requirement that $\varphi_p^{\mathbb{S}}(P) = \bigcup_{t\in P} \varphi_p^{\mathbb{S}}(\{t\})$, which implies that if φ is completely p-additive and true at a point s in a Kripke model, it will remain so if we restrict the valuation of p to a singleton. Complete additivity has been studied (under the name 'continuity') by van Benthem [15], in the context of operations on relations that are *safe for* (that is, preserve) bisimulations. Hollenberg [8] linked the notion to the syntax of PDL, showing that any completely p-additive formula in μML can be equivalently rewritten as $\langle\pi\rangle p$, where π belongs to the set of so-called μ-programs, which extend PDL-programs by admitting tests of arbitrary formulas in μML.

Fontaine and Venema [4,5] gave a different syntactic characterization of complete additivity. One of our two main contributions builds on this work, by showing that PDL exactly corresponds to the fragment μ_{ca}ML of μML where the fixpoint operator $\mu x.\varphi$ is restricted to formulas φ which belong to this syntactic fragment characterizing completely p-additivity. Similarly, test-free PDL corresponds to a smaller fragment μ_{rca}ML of μML.

Automata characterizations for μ-calculus and PDL. It is difficult to overstress the importance of automata-theoretic techniques in the theory of the μ-calculus: many of the fundamental results on μML are proved by means of the so-called μ-automata introduced by Janin and Walukiewicz [9].

In the case of PDL, the first automata-based result was by Streett [13,14] who translated PDL (with additional looping and converse operators) to deterministic two-way automata on infinite trees, and obtained decidability for the satisfiability problem. Vardi and Wolper [17] proved that PDL can be translated to Büchi (tree) automata, thus obtaining sharper complexity results. Muller et al. [12] showed that many dynamic and temporal logics can be uniformly represented using so-called weak alternating automata.

While the mentioned papers use translations of PDL-formulas to some kind of automata, none of them provides a precise automata-theoretic characterization of PDL. That is, the classes of automata under consideration contain automata that do not correspond to an equivalent PDL formula. The second main contribution of our article is to define two classes of alternating parity automata which exactly correspond, respectively, to PDL and its test-free variant PDL^{tf}. These two types of automata are easily seen to be subclasses of the *alternating* automata corresponding to μML [6].

Due to space constraints, most of our proofs have been moved to the Appendix.

2 Preliminaries

We assume that the reader is familiar with the syntax and semantics of PDL and the modal μ-calculus, and with parity games. We fix some notation and terminology, and discuss parity automata.

2.1 Structures and Languages

Throughout the article we fix a set of proposition symbols P and a set of atomic actions D. The structures that we are considering are multi-modal Kripke models, i.e., tuples $\mathbb{S} = \langle S, R_{d\in\mathsf{D}}, V\rangle$ where $R_d \subseteq S \times S$ and $V : \mathsf{P} \to \wp S$.

Definition 2.1 Propositional dynamic logic (in negation normal form) is given by mutual induction on formulas and programs:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle\pi\rangle\varphi \mid [\pi]\varphi$$
$$\pi ::= d \mid \pi;\pi \mid \pi \oplus \pi \mid \pi^* \mid \varphi?$$

where $p \in \mathsf{P}$, $d \in \mathsf{D}$. We denote this language by PDL(P,D) and drop P,D when clear from context. As an abuse of notation we write $\pi \in$ PDL(P,D) to mean that π is a program of PDL(P,D). Test-free propositional dynamic logic (denoted by PDL^{tf}) is PDL without the test operator (?). The set of proposition letters occurring in a formula φ (program π) is denoted by $\mathrm{Var}(\varphi)$ (respectively, $\mathrm{Var}(\pi)$).

The accessibility relation induced by a program π in a model $\mathbb{S}$ is denoted as $R^{\mathbb{S}}_\pi$, and the truth relation is denoted by $\Vdash$.

2.2 Parity Games

A *parity game* $\mathcal{G}$ consists of a partitioned board $G = G_\exists \uplus G_\forall$, a relation $E \subseteq G \times G$ indicating the available moves $E[u]$ from a position $u \in G$, and a parity map, i.e., a map $\Omega : G \to \mathbb{N}$ of finite range. A match of such a parity game consists of the two players, $\exists$ and $\forall$, moving a token from one position to another over the board; matches can be represented as paths through the graph (G, E). A player who gets stuck during the match by having arrived at a position with no admissible moves, immediately looses. If the match goes on forever then the parity map Ω is used to call a winner. The winner is $\exists$ if the maximum parity which occurs infinitely often in the match is even, otherwise $\forall$ wins. An *initialized parity game* $\mathcal{G}@u$ is a pair $(\mathcal{G}, u)$ where $\mathcal{G}$ is a parity game and $u \in G$ is the *initial position* of the game.

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 and therefore can be represented as a function $\sigma : G_\Pi \to G$. A match is *σ-guided* if for each position $u \in G_\Pi$ player Π chooses $\sigma(u)$ as the next position. We say that σ is *surviving* for Π if for each σ-guided match, the moves suggested by σ are always available to Π, and *winning* if in addition, Π wins each σ-guided match of the game. A winning position is one from which Π has a winning strategy. Finally, a player Π has a surviving strategy in an initialized game $\mathcal{G}@u$ *taking her/leading to position* p if Π has a strategy σ such that for every σ-guided match of $\mathcal{G}@u$ she either wins or gets to position p in finitely many steps.

2.3 Parity automata

Parity automata are finite devices operating on possibly infinite structures. In our paper, the automata classify pointed Kripke structures, and the question whether an automaton accepts or rejects such a structure is determined by a certain parity game. This acceptance game proceeds in a (possibly infinite) number of rounds, during each of which a certain *one-step formula* is the focus of attention.

Definition 2.2 Let A be a set of names. The set $\mathrm{ML}_1(A)$ of one-step modal formulas is given by the following grammar.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle d \rangle a \mid [d]a,$$

where $p \in \mathsf{P}$, $d \in \mathsf{D}$, $a \in A$.

Clearly, in order to interpret one-step formulas in Kripke models, we need, besides the valuation, which takes care of the proposition letters in P, an interpretation for the variables in A. It will be convenient to use *markings* for this purpose, that is, maps $m : S \to \wp A$. Such a marking can also be presented as a valuation, or as a relation $Z_m \subseteq A \times S$, defined by $Z_m := \{(a, s) \mid s \in S, a \in m(s)\}$. We use these perspectives interchangeably.

Definition 2.3 A *(modal) parity automaton* is a tuple $\mathbb{A} = \langle A, \Delta, \Omega \rangle$ such that A is a finite set of states of the automaton; $\Delta : A \to \mathrm{ML}_1(A)$ is the transition

map; and $\Omega : A \to \mathbb{N}$ is the parity map. An *initialized automaton* is a pair $(\mathbb{A}, a_I)$ where $a_I \in A$ is the initial state.

The acceptance game associated with a parity automaton $\mathbb{A}$ and a Kripke model $\mathbb{S}$ is given as follows. 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 $(a, s) \in A \times S$, the players consider the sentence $\Delta(a) \in \mathrm{ML}_1(A)$. At this position $\exists$ has to come up with a marking $m : S \to \wp A$, such that the formula $\Delta(a)$ is true at $\mathbb{S}, s$ under m. After that, $\forall$ chooses an element of Z_m to continue the match.

Definition 2.4 Given a model $\mathbb{S}$ and an automaton $\mathbb{A}$ we define the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ as the parity game given by the following table:

Position	Player	Admissible moves	Parity
$(a, s) \in A \times S$	$\exists$	$\{m : S \to \wp A \mid \mathbb{S}, m, s \Vdash \Delta(a)\}$	$\Omega(a)$
$m : S \to \wp A$	$\forall$	$\{(a, s) \mid s \in S, a \in m(s)\}$	0

Positions of the form $(a, s) \in A \times S$ will be called *basic*.

If (a_I, s) is a winning position for $\exists$ in the game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ we say that $(\mathbb{A}, a_I)$ *accepts* the pointed model $(\mathbb{S}, s)$, notation: $\mathbb{S}, s \Vdash (\mathbb{A}, a_I)$.

We say that an initialized automaton $(\mathbb{A}, a_I)$ is equivalent to a formula φ if $\mathbb{S}, s \Vdash (\mathbb{A}, a_I) \iff \mathbb{S}, s \Vdash \varphi$, for all $\mathbb{S}, s$. More generally, we use the symbol $\equiv$ to denote equivalence between automata or formulas. The following fact lies behind the automata-theoretic approach towards the modal μ-calculus.

Fact 2.5 ([18,6]) *There are effective procedures transforming a formula of* μML *into an equivalent parity automaton, and vice versa.*

We now turn to the definition of *weak* parity automata.

Definition 2.6 Given $\mathbb{A} = \langle A, \Delta, \Omega \rangle$, we define the relation $\rightsquigarrow \subseteq A \times A$ by putting $a \rightsquigarrow b$ if b occurs in the formula $\Delta(a)$; we let $\prec$ and $\preceq$ denote, respectively, the transitive and reflexive-transitive closure of $\rightsquigarrow$. A *strongly connected component* or *SCC* of $\mathbb{A}$ is a subset $C \subseteq A$ such that for every $b, c \in C$ we have $b \preceq c$ and $c \preceq b$. An SCC is called *maximal*, or an *MSCC*, if none of its proper supersets is an SCC.

Definition 2.7 A parity automaton $\mathbb{A} = \langle A, \Delta, \Omega \rangle$ is *weak* if Ω satisfies

(weakness) if $a \preceq a'$ and $a' \preceq a$ then $\Omega(a) = \Omega(a')$.

Since in this case all states of a strongly connected component C have the same parity we may speak of the *parity of* C and denote it by $\Omega(C)$.

Remark 2.8 Any weak parity automaton $\mathbb{A}$ is equivalent to a weak parity automaton $\mathbb{A}'$ with $\Omega : A' \to \{0, 1\}$. From now on we assume such a parity map for weak parity automata.

3 Syntactic characterization of PDL and PDL^{tf}

In this section we will provide a precise characterization of the fragments of μML that correspond to full and test-free PDL. It will be convenient for us

to work with a version of PDL that includes the empty program ϵ (or `skip`), which is interpreted as the identity relation in any Kripke model. Observe that in full PDL, the role of ϵ can be taken by the test program $\top?$.

Definition 3.1 The formulas and programs of the language $\mathrm{PDL}^{tf\epsilon}$ is given by the following grammar:

$$\begin{array}{l} \varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\pi\rangle\varphi \\ \pi ::= d \mid \epsilon \mid \pi \oplus \pi' \mid \pi;\pi' \mid \pi^* \end{array}$$

Remark 3.2 It is not difficult to show that adding the skip program does not add expressive power to PDL^{tf}. To see this, think of the programs of PDL^{tf} and $\mathrm{PDL}^{tf\epsilon}$ as the sets of regular expressions over the set D that may and may not use the empty string symbol ϵ, respectively. Let $\equiv_\ell$ denote the relation of language equivalence between regular expressions, that is, write $\pi \equiv_\ell \pi'$ if π and π' denote the same regular language over D. One may show, by induction on programs, that for any $\pi \in \mathrm{PDL}^{tf\epsilon}$ either (a) $\pi \equiv_\ell \epsilon$, or there is a program $\overline{\pi} \in \mathrm{PDL}^{tf\epsilon}$ such that either (b) $\pi \equiv_\ell \overline{\pi}$ or (c) $\pi \equiv_\ell \epsilon \oplus \overline{\pi}$. Based on this observation we may inductively define a truth-preserving translation from $\mathrm{PDL}^{tf\epsilon}$-formulas to PDL^{tf}-formulas; the key clause of this translation uses that $\langle\pi\rangle\varphi$ is equivalent to either (a) φ, (b) $\langle\overline{\pi}\rangle\varphi$ or (c) $\varphi \vee \langle\overline{\pi}\rangle\varphi$.

Definition 3.3 Given a set X of propositional variables, we define the *completely additive fragment* with respect to X, notation: $\mathrm{CAF}(X)$, as follows:

$$\varphi ::= x \in X \mid \psi \mid \psi \wedge \varphi \mid \varphi \vee \varphi \mid \langle d\rangle\varphi \mid \mu y.\varphi'$$

Here we require that ψ belongs to the X-free fragment of the modal μ-calculus (i.e., none of the variables in X occurs freely in ψ), and $\varphi' \in \mathrm{CAF}(X \cup \{y\})$.

The *restricted completely additive fragment* with respect to X, notation: $\mathrm{RAF}(X)$, is defined, similarly, by:

$$\varphi ::= x \in X \mid \psi \mid \varphi \vee \varphi \mid \langle d\rangle\varphi \mid \mu y.\varphi'$$

We define $\mu_{ca}\mathrm{ML}$ and $\mu_{rca}\mathrm{ML}$ to be the fragments of the modal μ-calculus where the use of the least fixpoint operator is restricted to these fragments.

Remark 3.4 The fragment $\mathrm{CAF}(X)$ provides a syntactic characterization of a (minor) variant of complete additivity, where we require (1) to hold only for *non-empty* index sets I. It is proved in [5] that a formula $\varphi \in \mu\mathrm{ML}$ satisfies this property for each $x \in X$ iff φ is equivalent to a formula in the $\mathrm{CAF}(X)$.

Definition 3.5 Formulas of the fragment $\mu_{ca}\mathrm{ML}$ ($\mu_{rca}\mathrm{ML}$, respectively), are given by the following induction:

$$\alpha ::= p \mid \neg\alpha \mid \alpha \vee \alpha \mid \langle d\rangle\alpha \mid \mu x.\varphi,$$

where $\varphi \in \mu_{ca}\mathrm{ML} \cap \mathrm{CAF}(x)$ ($\varphi \in \mu_{rca}\mathrm{ML} \cap \mathrm{RAF}(x)$, respectively).

Theorem 3.6 PDL *and test-free* PDL *are effectively equivalent to the fragments* $\mu_{ca}\mathrm{ML}$ *and* $\mu_{rca}\mathrm{ML}$*, respectively.*

Proof of Theorem 3.6. The theorem follows directly from Propositions 3.8 and 3.9 below. □

We start with the direction from PDL and PDL^{tf} to μ_{ca}ML and μ_{rca}ML.

Definition 3.7 By a simultaneous induction on PDL-formulas and -programs, we define, for each PDL-program π, a function $f_\pi : \mu\mathrm{ML} \to \mu\mathrm{ML}$ on the set of modal fixpoint formulas, and a map $(\cdot)^t$ from PDL to μML:

$$\begin{array}{ll} f_d(\alpha) & := \langle d\rangle\alpha \\ f_{\varphi?}(\alpha) & := \varphi \wedge \alpha \\ f_{\pi\oplus\pi'}(\alpha) & := f_\pi(\alpha) \vee f_{\pi'}(\alpha) \\ f_{\pi;\pi'}(\alpha) & := f_\pi(f_{\pi'}(\alpha)) \\ f_{\pi^*}(\alpha) & := \mu x.\alpha \vee f_\pi(x) \end{array} \qquad \begin{array}{ll} p^t & := p \\ (\neg\varphi)^t & := \neg\varphi^t \\ (\varphi_0 \vee \varphi_1)^t & := \varphi_0^t \vee \varphi_1^t \\ (\langle\pi\rangle\varphi)^t & := f_\pi(\varphi^t) \end{array}$$

where, in the clause for f_{π^*}, x is some canonically chosen fresh variable.

Proposition 3.8 *For any* PDL*-formula φ, φ^t belongs to the fragment μ_{ca}ML, and is equivalent to φ. If φ belongs to* PDL^{tf}*, then $\varphi^t \in \mu_{rca}$ML.*

The translation in the other direction is provided by the following proposition.

Proposition 3.9

(i) There is an effective procedure rewriting any modal fixpoint formula $\alpha \in \mu_{ca}$ML into an equivalent PDL*-formula α^s. Moreover, if $\alpha \in \mu_{rca}$ML then $\alpha^s \in$ PDL$^{tf\epsilon}$.*

(ii) There is an effective procedure that, given a formula $\varphi \in \mu_{ca}$ML and a set X of variables such that $\varphi \in \mathrm{CAF}(X)$, returns an X-free PDL*-formula ψ, a subset $Y \subseteq X$, and a collection $\{\pi_x \mid x \in Y\}$ of X-free* PDL*-programs, such that*

$$\varphi \equiv \psi \vee \bigvee_{x\in Y} \langle\pi_x\rangle x. \tag{2}$$

If $\varphi \in \mu_{rca}\mathrm{ML} \cap \mathrm{RAF}(X)$, then ψ and each π_x belong to PDL$^{tf\epsilon}$.

4 A characterization of PDL^{tf} by automata

In this section we give the automata-theoretic characterization of test-free PDL. In μ-automata, $\rightsquigarrow$-cycles naturally correspond to fixpoint operators being unfolded. When considering (test-free) PDL, in the light of the previous section it is obvious that cycles have to be restricted so that they can only occur when induced by programs involving the iteration operator. The slogan that drives the definition of PDL- and PDL^{tf}-automata is that *maximal strongly connected components correspond to programs.*

The automata corresponding to PDL^{tf} will be weak parity automata where the transition map is subject to an additional constraint for all elements belonging to the same SCC. We begin by defining additional one-step languages needed for these automata. Recall that $\mathrm{ML}_1(A)$ is the set of one-step modal formulas in A.

Definition 4.1 Let $C, O \subseteq A$ be such that $C \cap O = \varnothing$. The sets $\mathrm{ADD}_1^{tf}(O, C)$ and $\mathrm{MUL}_1^{tf}(O, C)$ of one-step formulas that are *test-free additive*, resp. *test-free multiplicative* in C, are defined, respectively, by the following grammars:

$$\varphi ::= \varphi \vee \varphi \mid \langle d \rangle c \mid \psi \qquad \text{and} \qquad \varphi ::= \varphi \wedge \varphi \mid [d]c \mid \psi$$

where $c \in C$, $\psi \in \mathrm{ML}_1(O)$.

Definition 4.2 A PDL^{tf}*-automaton* is a weak parity automaton $\mathbb{A}$ satisfying, for every strongly connected component C, the following constraint:

(tf-additivity) If $\Omega(C) = 1$ then $\Delta(c) \in \mathrm{ADD}_1^{tf}(A \setminus C, C)$ for each $c \in C$.
In case $\Omega(C) = 0$ then $\Delta(c) \in \mathrm{MUL}_1^{tf}(A \setminus C, C)$ for each $c \in C$.

The main theorem of this section states that these PDL^{tf}-automata characterize PDL^{tf}. It will be proved in the two following subsections.

Theorem 4.3

(i) For every PDL^{tf}*-formula* φ *one can compute an equivalent initialized* PDL^{tf}*-automaton* $(\mathbb{A}_\varphi, a_\varphi)$.
(ii) For every initialized PDL^{tf}*-automaton* $(\mathbb{A}, a_I)$ *one can compute an equivalent* PDL^{tf}*-formula* $\varphi_{\mathbb{A}, a_I}$.

4.1 From formulas to automata

In this subsection we will prove Theorem 4.3(i), that is, given a PDL^{tf}-formula φ we construct an equivalent PDL^{tf}-automaton $(\mathbb{A}_\varphi, a_\varphi)$. To begin, we focus on formulas of the form $\varphi = \langle \pi \rangle \alpha$ and, for the moment, assume that we already have an automaton $(\mathbb{A}, a_A) \equiv \alpha$. It is of our interest to understand how the operation $\langle \pi \rangle$ changes $(\mathbb{A}, a_A)$ to get an automaton for $\varphi \langle \pi \rangle \alpha$. Our idea is to represent π itself as a PDL^{tf}-automaton, which we will then combine with $\mathbb{A}$.

In this subsection we will briefly use non-deterministic finite-state automata (NFA). Recall that an NFA is a tuple $\underline{\mathrm{A}} = \langle A, \delta, F, a_I \rangle$ where $\delta : A \times \mathsf{D} \to \wp A$ is the transition map, $F \subseteq A$ are the final states and $a_I \in A$ is the initial state. Given a model $\mathbb{S}$, an NFA denotes a set of *paths* through $\mathbb{S}$. We formalize the acceptance of a path with the following game.

Definition 4.4 Given a model $\mathbb{S}$ and an NFA $\underline{\mathrm{A}} = \langle A, \delta, F, a_I \rangle$ we define the *acceptance game* $\underline{\mathcal{A}}(\underline{\mathrm{A}}, \mathbb{S})$ having as basic positions pairs $(a, s) \in A \times S$.

Position	Pl'r	Admissible moves
$(a, s) \in (A \setminus F) \times S$	$\exists$	$\{(b, t) \mid \exists d \in \mathsf{D}.\ b \in \delta(a, d)\ \&\ R_d(s, t)$
$(f, s) \in F \times S$	$\exists$	$\{\mathtt{end}\} \cup \{(b, t) \mid \exists d \in \mathsf{D}.\ b \in \delta(f, d)\ \&\ R_d(f, t)\}$
`end`	$\forall$	$\varnothing$

Finite matches are lost by the player who gets stuck, infinite matches are all won by $\forall$. A path $\vec{s}$ in $\mathbb{S}$ is accepted by $\underline{\mathrm{A}}$ iff $\exists$ has a winning strategy σ for the initialized game $\underline{\mathcal{A}}(\underline{\mathrm{A}}, \mathbb{S})@(a_I, s)$ such that every σ-guided match visits precisely (and in order) the states of $\vec{s}$.

Using this game-theoretic approach towards NFAs we can easily prove the following lemma.

Lemma 4.5 *For every π there exists an initialized* PDL^{tf}*-automaton $(\mathbb{P}_\pi, a_\pi)$ and a set $F \subseteq |\mathbb{P}_\pi|$ such that for all $\mathbb{S}$ and $s,t \in |\mathbb{S}|$ we have $R_\pi(s,t)$ iff $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s)$ taking her to (f,t) for some $f \in F$.*

The above lemma gives us an automaton $(\mathbb{P}_\pi, a_\pi, F) = \langle P, \Delta_P, \Omega_P, a_\pi, F\rangle$ which works as a representation of π. We now combine this automaton with the representation of α given by $(\mathbb{A}, a_A) = \langle A, \Delta_A, \Omega_P, a_A\rangle$ yielding an automaton $(\mathbb{A}_\varphi, a_\varphi)$ for $\varphi = \langle\pi\rangle\alpha$. Define $(\mathbb{A}_\varphi, a_\varphi) := \langle A \uplus P, \Delta, \Omega, a_\varphi\rangle$ where $a_\varphi := a_\pi$, $\Omega := \Omega_A \cup \Omega_P$, and the transition map is defined as

$$\Delta(e) := \begin{cases} \Delta_P(e) & \text{if } e \in P \setminus F \\ \Delta_P(e) \vee \Delta(a_A) & \text{if } e \in F \\ \Delta_A(e) & \text{if } e \in A. \end{cases}$$

Remark 4.6 Observe that the construction has the following properties

(i) $\mathbb{A}_\varphi$ is a well-defined PDL^{tf}-automaton,
(ii) You can only go from the $\mathbb{P}_\pi$ part to the $\mathbb{A}$ part from a state in $F \subseteq P$,
(iii) Once you leave the $\mathbb{P}_\pi$ part you cannot come back.

Now we prove that $(\mathbb{A}_\varphi, a_\varphi)$ is an automaton representation of $\varphi = \langle\pi\rangle\alpha$.

Proposition 4.7 *$\mathbb{S}, s \Vdash \langle\pi\rangle\alpha$ iff $\exists$ has a winning strategy in $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(a_\varphi, s)$.*

This finishes the proof of Theorem 4.3(i) for the particular case of $\varphi = \langle\pi\rangle\alpha$ when we already have an automaton for α. The general case can be proved by induction on $\varphi \in$ PDL^{tf}. The propositional and Boolean cases are easy and Proposition 4.7 gives us the required automaton for $\varphi = \langle\pi\rangle\alpha$. If $\varphi = [\pi]\alpha$ the construction and proofs are dual to the diamond case.

4.2 From automata to formulas

In this subsection we prove Theorem 4.3(ii), that is, for every initialized PDL^{tf}-automaton $(\mathbb{A}, a_I)$ we give an equivalent PDL^{tf}-formula $\varphi_{\mathbb{A},a_I}$. The key idea underlying our construction is to turn MSCCs of $\mathbb{A}$ into sets of PDL^{tf}-equations, and use the properties of PDL^{tf}-automata to solve these equations *inside* PDL^{tf}. In more detail, for every MSCC C and entry point $b \in C$ we will show how to get an equivalent formula $\varphi_{C,b} \in$ PDL$^{tf}(\mathsf{P} \uplus O, \mathsf{D})$ where the propositional variables in $O := A \setminus C$ correspond to the states outside C.

Definition 4.8 A *set of B-incomplete τ-equations* is a tuple $\mathbf{E} = (E, \xi, \tau)$ where E is a non-empty, finite set of equations specified by the map $\xi : E \to$ PDL$^{tf}(\mathsf{P} \uplus B \uplus E, \mathsf{D})$ and $\tau \in \{\mu, \nu\}$ is the type of $\mathbf{E}$. We sometimes specify a set of equations using the notation $\mathbf{E} := \{e_1 \approx \psi_1, \ldots, e_n \approx \psi_n\}_\tau$.

Definition 4.9 Given a model $\mathbb{S}$ and a set of B-incomplete τ-equations $\mathbf{E} = (E, \xi, \tau)$ we define the *solution game* $\mathcal{S}(\mathbf{E}, \mathbb{S})$ having as basic positions pairs $(x, s) \in (E \cup B) \times S$.

Position	Player	Admissible moves
$(e, s) \in E \times S$	$\exists$	$\{m : S \to \wp(E \cup B) \mid \mathbb{S}, m, s \Vdash \xi(e)\}$
$m : S \to \wp(E \cup B)$	$\forall$	$\{(x, s) \mid s \in S, x \in m(s)\}$

Whenever a position of the form $(b, s) \in B \times S$ is reached, the match is declared a tie; finite matches not ending in a tie are lost by the player that got stuck and infinite matches are won by $\exists$ if $\tau = \nu$, and by $\forall$ if $\tau = \mu$.

Let C be an MSCC of $\mathbb{A}$. First we consider the case where the parity of C is 1. We turn the information of C into a set of O-incomplete μ-equations $\mathbf{C} = (C, \xi, \mu)$ given by $\xi(c) := \Delta(c)$ for all $c \in C$. Observe that, by construction, the set of equations satisfies

$$\xi(c) = \alpha \vee \bigvee_{u \in U} \langle \pi_u \rangle u \quad \text{for } U \subseteq C \text{ and } \alpha, \pi_u \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus O, \mathsf{D}). \qquad (*)$$

This set of equations is equivalent to C in the following sense:

Proposition 4.10 *Let $b \in C$, $o \in O$, and $s, t \in |\mathbb{S}|$, the following are equivalent*

(i) $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{A}, \mathbb{S})@(b, s)$ taking her to (o, t),
(ii) $\exists$ has a surviving strategy in $\mathcal{S}(\mathbf{C}, \mathbb{S})@(b, s)$ taking her to (o, t).

Proof. Straightforward from the definition of the games. □

For a moment, we forget about MSCCs and focus on sets of equations. We show that if a set of equations satisfies $(*)$ we can solve it inside PDL^{tf}. The proof is basically a game-theoretic version of the one found in [16], which is also reminiscent of the transformation of linear grammars into regular expressions.

Lemma 4.11 *Let $\mathbf{E} = (E, \xi, \mu)$ be a set of B-incomplete μ-equations satisfying $(*)$. For all $e \in E$ there exists $\varphi_{E,e} \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus B, \mathsf{D})$ such that for all $b \in B$ and $s, t \in |\mathbb{S}|$ the following are equivalent:*

(i) $\exists$ has a surviving strategy in $\mathcal{S}(\mathbf{E}, \mathbb{S})@(e, s)$ taking her to (b, t),
(ii) $\mathbb{S}, m, s \Vdash \varphi_{E,e}$ where $m : S \to \wp B$ is such that $Z_m = \{(b, t)\}$.

It is only left to apply the above results to $\mathbf{C}$ to get the required formula.

Corollary 4.12 *For every MSCC C and $b \in C$ there is $\varphi_{C,b} \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus O, \mathsf{D})$ such that for all $o \in O$ and $s, t \in |\mathbb{S}|$ the following are equivalent:*

(i) $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{A}, \mathbb{S})@(b, s)$ taking her to (o, t),
(ii) $\mathbb{S}, m, s \Vdash \varphi_{C,b}$ where $m : S \to \wp O$ is such that $Z_m = \{(o, t)\}$.

Proof. Combination of Proposition 4.10 and Lemma 4.11 applied to $\mathbf{C}$. □

The above corollary provides a formula $\varphi_{C,b}$ when the parity of the connected component C is 1. The case where the parity of C is 0 is solved in a dual but completely similar way.

Now that we can get a formula for every point of an MSCC we turn to the general case. In order to create a formula from an initialized automaton we introduce the following concept.

Definition 4.13 Given a PDL^{tf}-automaton $\mathbb{A}$, the *DAG of connected components of* $\mathbb{A}$ is the pair $\mathrm{DCC}(\mathbb{A}) = (G, E)$ where G is the set of $\preceq$-MSCCs of $\mathbb{A}$ and $(C_1, C_2) \in E$ if $C_1 \neq C_2$ and $a \rightsquigarrow b$ for some $a \in C_1$ and $b \in C_2$.

Remark 4.14 Observe that a node of $\mathrm{DCC}(\mathbb{A})$ is either a $\prec$-connected component or a single element $a \in A$ which does not belong to any $\prec$-cycle. Another observation is that, even though $\mathrm{DCC}(\mathbb{A})$ may not be a tree, it certainly contains no cycles. Therefore E is well-founded and, given $C \in G$, we can associate a notion of height to the subgraph generated by C.

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

Proof of Theorem 4.3(ii). For every initialized PDL^{tf}-automaton $(\mathbb{A}, a_I)$ we give an equivalent PDL^{tf}-formula $\varphi_{\mathbb{A},a_I}$. The proof will be done by induction on the height of the subgraph of $\mathrm{DCC}(\mathbb{A})$ generated by a_I.

If the height of the subgraph is 1, then it is composed of a single MSCC C and $a_I \in C$. By Corollary 4.12, we get a formula $\varphi_{C,a_I} \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus O, \mathsf{D})$. We only have to observe that, because C is not connected to any other MSCC then $O = \varnothing$. Therefore $\varphi_{C,a_I} \in \mathrm{PDL}^{tf}(\mathsf{P}, \mathsf{D})$ and is equivalent to $(\mathbb{A}, a_I)$.

Suppose the height of the subgraph is n and $a_I \in C$ for some MSCC C. Again by Corollary 4.12 we get a formula $\varphi_{C,a_I} \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus O, \mathsf{D})$ where $O = \{o_1, \ldots, o_k\}$ and $o_i \in C_i$ for some MSCCs C_i. By inductive hypothesis, we get a formula $\varphi_{\mathbb{A},o_i} \in \mathrm{PDL}^{tf}(\mathsf{P}, \mathsf{D})$ for each o_i. It is straightforward to check that $\varphi_{\mathbb{A},a_I} := \varphi_{C,a_I}[o_i \mapsto \varphi_{\mathbb{A},o_i} \mid i \leq k]$ is equivalent to $(\mathbb{A}, a_I)$. □

5 A characterization of PDL by automata

This section concerns the automata-theoretic characterization of full PDL. Since our approach here is an adaptation of the one taken in the previous section we will be a bit more sketchy.

Definition 5.1 Let $C, O \subseteq A$ be such that $C \cap O = \varnothing$. The sets $\mathrm{ADD}_1(O, C)$ and $\mathrm{MUL}_1(O, C)$ of one-step formulas that are additive, resp. multiplicative in C, are defined, respectively, by the grammars

$$\varphi ::= \varphi \vee \varphi \mid \langle d \rangle c \mid \psi \mid \varphi \wedge \psi \qquad \text{and} \qquad \varphi ::= \varphi \wedge \varphi \mid [d]c \mid \psi \mid \varphi \vee \psi$$

where $c \in C$, $\psi \in \mathrm{ML}_1(O)$.

Note that the difference with the sets one-step formulas for PDL^{tf}-automata given in Definition 4.1 lies in the fact that here, in order to take care of tests, we allow conjunctions with C-free formulas.

Definition 5.2 A PDL-*automaton* is a weak parity automaton $\mathbb{A}$ satisfying, for every strongly connected component C, the following constraint:

(additivity) If $\Omega(C) = 1$ then $\Delta(c) \in \mathrm{ADD}_1(A \setminus C, C)$ for each $c \in C$. In case $\Omega(C) = 0$ then $\Delta(c) \in \mathrm{MUL}_1(A \setminus C, C)$ for each $c \in C$.

The main theorem of this section states that PDL-automata characterize PDL.

Theorem 5.3

(i) For every PDL*-formula φ one can compute an equivalent initialized automaton $(\mathbb{A}_\varphi, a_\varphi)$.*

(ii) For every initialized automaton $(\mathbb{A}, a_I)$ one can compute an equivalent PDL*-formula $\varphi_{\mathbb{A},a_I}$.*

5.1 From formulas to automata

In this section we will prove Theorem 5.3(i), that is, given a PDL-formula φ we create an equivalent PDL-automaton $(\mathbb{A}_\varphi, a_\varphi)$. We give a proof by induction on φ. If φ is a test-free formula (that is, $\varphi \in \mathrm{PDL}^{tf}$) we can get the corresponding PDL-automaton from Theorem 4.3 by observing that every PDL^{tf}-automaton is also a PDL-automaton. It is also easy to check that the class of PDL-automata is closed under the Boolean operators.

The interesting case, therefore, is where $\varphi = \langle\pi\rangle\alpha$, with $\alpha \equiv (\mathbb{A}_\alpha, a_\alpha)$ and π involving tests. To treat this case we use the following strategy: first we will consider tests as additional atomic actions and get an NFA for π, similar to what we did in Section 4; after that we show how to convert this NFA to a PDL-automaton and merge it with the automata for the tested formulas to get a PDL-automaton $\mathbb{P}_\pi$ for π. To finish, we combine $\mathbb{P}_\pi$ and $\mathbb{A}_\alpha$ to get an automaton for φ.

In the process of creating a PDL-automaton for π we encounter new complexities because of the presence of tests. To be able to properly define a merging operation we need to introduce the following concepts.

Definition 5.4 Let B be a set of names such that $A \cap B = \varnothing$ and $\mathsf{P} \cap B = \varnothing$. A *$B$-incomplete* PDL-*automaton* $\mathbb{A}$ is a PDL-automaton based on the set of propositions $\mathsf{P} \cup B$ such that the elements of B occur only positively in the transition map of $\mathbb{A}$. The acceptance games of Definition 2.4 are extended to B-incomplete automata with the intention to interpret the elements of B as names (as opposed to propositions). Basic positions are then taken from $(A \cup B) \times S$ and markings are of the type $m : S \to \wp(A \cup B)$. Whenever a position from $B \times S$ is reached, the match is declared a tie.

Definition 5.5 The *completion of a B-incomplete automaton $\mathbb{A}$ with a* PDL-*automaton* $\mathbb{A}' = \langle A', \Delta', \Omega'\rangle$ is defined as $(\mathbb{A} \rtimes \mathbb{A}') = \langle C, \Delta_C, \Omega_C\rangle$ where $C := A \uplus A'$, $\Omega_C := \Omega \cup \Omega'$ and the transition map is given by

$$\Delta_C(c) := \begin{cases} \Delta'(c) & \text{if } c \in A', \\ \Delta(c)[\mathsf{b} \mapsto \Delta'(\mathsf{b}) \mid \mathsf{b} \in B \cap A'] & \text{if } c \in A. \end{cases}$$

Note that the completion can be partial if $B \not\subseteq A'$, in this case the outcome will be $(B \setminus A')$-incomplete. If $B \subseteq A'$, the outcome will be a (complete) PDL-automaton. Also observe that a completion cannot generate new cycles.

Definition 5.6 Given $\pi \in \mathrm{PDL}(\mathsf{P}, \mathsf{D})$ we use $\pi^\flat \in \mathrm{PDL}^{tf}(\mathsf{P}, \mathsf{D} \cup \mathsf{T})$ to denote the version of π where its top-level tests T are considered as atomic actions. The T-extension of a model $\mathbb{S} = \langle S, R_{d\in\mathsf{D}}, V\rangle$ is defined as $\mathbb{S}^{\mathsf{T}} = \langle S, R_{d\in\mathsf{D}}, R_{\chi\in\mathsf{T}}, V\rangle$ where $R_\chi := \{(s,s) \in S \times S \mid \mathbb{S}, s \Vdash \chi\}$.

Lemma 5.7 *For every $\pi \in$ PDL there exists an $\mathtt{x}$-incomplete initialized* PDL-*automaton $(\mathbb{P}_\pi, a_\pi)$ such that for all models $\mathbb{S}$ and $s, t \in |\mathbb{S}|$ we have that $R^{\mathbb{S}}_\pi(s,t)$ iff $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s)$ taking her to $(\mathtt{x}, t)$.*

Proof. Let T be the top-level tests appearing in π. We claim the following:

CLAIM 1 For every model $\mathbb{S}$ and $s, t \in |\mathbb{S}|$ we have that $R^{\mathbb{S}}_{\pi}(s,t)$ iff $R^{\mathbb{S}^{\mathsf{T}}}_{\pi^\flat}(s,t)$.

As in Section 4, we can construct an NFA $\underline{\mathrm{A}}_\pi = \langle A, \delta, F, a_I\rangle$ which recognizes $\pi^\flat$. By definition of $\underline{\mathrm{A}}_\pi$ recognizing $\pi^\flat$, we have the following claim.

CLAIM 2 For every model $\mathbb{S}$, $\underline{\mathrm{A}}_\pi$ accepts the path $s, \ldots, t$ in $\mathbb{S}^{\mathsf{T}}$ iff $R^{\mathbb{S}^{\mathsf{T}}}_{\pi^\flat}(s,t)$.

CLAIM 3 Without loss of generality we can assume these properties on $\underline{\mathrm{A}}_\pi$:

(i) Each state has either exiting action transitions or test transitions (but not both), and will accordingly be called *action state* or *test state*.

(ii) Every cycle contains at least one action state.

(iii) The initial state has no incoming transitions.

(iv) Test transitions always arrive into an action state.

These properties are reminiscent of the work by Kozen [11]. For reasons of space limitations we have to omit the proofs.

Let $T = \{\mathtt{a}_\chi \mid \chi \in \mathsf{T}\} \cup \{\mathtt{x}\}$ be a set of names. From $\underline{\mathrm{A}}_\pi$ we define a T-incomplete initialized PDL-automaton $\mathbb{A}_\pi := \langle A_\pi, \Delta_\pi, a_\pi\rangle$ by setting $A_\pi := A$, $\Omega(a) := 1$ for all $a \in A$,

$$\Delta_\pi(a) := \begin{cases} \bigvee\{\langle d\rangle b \mid d \in \mathsf{D}, b \in \delta(a,d)\} & \text{if } a \notin F \text{ is an action state,} \\ \mathtt{x} \vee \bigvee\{\langle d\rangle b \mid d \in \mathsf{D}, b \in \delta(a,d)\} & \text{if } a \in F \text{ is an action state,} \\ \bigvee\{\mathtt{a}_\chi \wedge \Delta_\pi(b) \mid \chi \in \mathsf{T}, b \in \delta(a,\chi)\} & \text{if } a \notin F \text{ is a test state,} \\ \mathtt{x} \vee \bigvee\{\mathtt{a}_\chi \wedge \Delta_\pi(b) \mid \chi \in \mathsf{T}, b \in \delta(a,\chi)\} & \text{if } a \in F \text{ is a test state.} \end{cases}$$

Note that this is well defined since we first define $\Delta_\pi(a)$ for action states and then for test states.

CLAIM 4 $\mathbb{A}_\pi$ is a well-defined T-incomplete PDL-automaton.

Let $(\mathbb{A}_\chi, a_\chi)_{\chi\in\mathsf{T}}$ be the family of PDL-automata for $\mathsf{T} = \{\chi_1, \ldots, \chi_k\}$, provided by the inductive hypothesis. To finish the construction define $(\mathbb{P}_\pi, a_\pi) := (\mathbb{A}_\pi \rtimes \mathbb{A}_{\chi_1} \rtimes \cdots \rtimes \mathbb{A}_{\chi_k}, a_\pi)$.

CLAIM 5 For every model $\mathbb{S}$ and $s, t \in |\mathbb{S}|$, the following are equivalent.

(i) $\exists$ has a surviving strategy in $\underline{\mathcal{A}}(\underline{\mathrm{A}}_\pi, \mathbb{S}^{\mathsf{T}})@(a_\pi, s)$ leading to $(f \in F, t)$.

(ii) $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s)$ taking her to $(\mathtt{x}, t)$.

Due to lack of space we have to omit the proof of this claim.

Finally, combining the above claims we find the following equivalences:

$$\begin{aligned} R^{\mathbb{S}}_{\pi}(s,t) &\iff R^{\mathbb{S}^{\mathsf{T}}}_{\pi^\flat}(s,t) && \text{(Claim 1)} \\ &\iff \underline{\mathrm{A}}_\pi \text{ accepts the path } s, \ldots, t \text{ in } \mathbb{S}^{\mathsf{T}} && \text{(Claim 2)} \\ &\iff \exists \text{ has a surviving strategy in} \\ &\qquad \mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s) \text{ leading to } (\mathtt{x}, t) && \text{(Claim 5)} \end{aligned}$$

This finishes the proof of the lemma. □

To conclude we have to give an automaton for $\varphi = \langle\pi\rangle\alpha$. Let $(\mathbb{A}_\alpha, \mathtt{x})$ be

the automaton for α, given by the inductive hypothesis. Define $(\mathbb{A}_\varphi, a_\varphi) := (\mathbb{P}_\pi \rtimes \mathbb{A}_\alpha, a_\pi)$. We prove that $(\mathbb{A}_\varphi, a_\varphi)$ is an automaton representation of φ.

Proposition 5.8 $\mathbb{S}, s \Vdash \langle\pi\rangle\alpha$ *iff* $\exists$ *has a winning strategy in* $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(a_\varphi, s)$.

Proof. The proof is similar to Proposition 4.7 but using Lemma 5.7. □

5.2 From automata to formulas

The proof is basically the same as for PDL^{tf}. The new challenge lies in showing that when solving the system of equations (i.e., an analogue of Lemma 4.11) we can provide a normal form which functions as $(*)$. The key observation is that one-step formulas in $\mathrm{ADD}_1(O, C)$ can be assumed to be in the form

$$\alpha \vee \bigvee_{u \in U} \langle\pi_u\rangle u \quad \text{with} \quad U \subseteq C;\ \alpha, \pi_u \in \mathrm{PDL}(\mathsf{P} \cup O, \mathsf{D})$$

This can be proved by a straightforward induction on the complexity of $\mathrm{ADD}_1(O, C)$-formulas, where in the inductive step we use that $\psi \wedge \varphi$ is equivalent to $\langle\psi?\rangle\varphi$. Further details are left to the reader.

6 Conclusions

In this paper we have clarified the relation between PDL and the modal μ-calculus, by (1) providing explicit syntactic translations between PDL and the fragment of μML to which it corresponds, and (2) giving an automata-theoretic characterization of PDL. Both results were obtained in versions for full and for test-free PDL, respectively.

Although we have treated the syntactic and the automata-theoretic characterizations separately, the two results are in fact closely related. This is witnessed, in the case of full PDL [3], by the close syntactic similarities between the completely additive fragment of μML (Definition 3.3) and the additive one-step formulas (Definition 5.1), in that both use the same set of operators. Using these similarities it would not be very hard to also give direct transformations between formulas in the fragment μ_{ca}ML and PDL-automata. Another similarity between the two characterizations surfaces in the proof: in both cases, the heart of the argument showing one direction of the equivalence lies in the fact that certain sets of fixpoint equations can be solved *inside* PDL. In fact, we could have proved our results for PDL in the form 'PDL-formulas $\longrightarrow$ PDL-automata $\longrightarrow$ μ_{ca}ML-formulas'. We chose to discuss the syntactic and the automata-theoretic result separately because we believe that the two proofs have some merits in their own right: the syntactic transformations are simple and straightforward, and the automata-theoretic proof sheds some light on the automata-theoretic nature of PDL-programs.

We hope that our characterizations of PDL provide new tools for proving results on PDL. In particular, it is an interesting question to find a natural logic of which PDL is the bisimulation-invariant fragment. In this light, note

[3] The case of PDL^{tf} is analogous.

that in a related paper [1] with Facchini and Zanasi we used similar automata to the ones in this paper to prove that a variant of μ_{rca}ML corresponds to the bisimulation-invariant fragment of weak monadic second-order logic. Other interesting questions would be to find semantic properties that set PDL apart as a fragment of μML, to prove the (un-)decidability of the question whether a given μ-calculus formula has an equivalent in PDL, to give a characterization of PDL-programs (cf. [8, p. 91]), and to give a constructive proof of the Craig interpolation property for PDL.

References

[1] Carreiro, F., A. Facchini, Y. Venema and F. Zanasi, *Weak MSO: Automata and expressiveness modulo bisimilarity*, in: *CSL/LICS 2014*, 2014, to appear.
[2] D'Agostino, G. and M. Hollenberg, *Logical Questions Concerning the μ-Calculus: Interpolation, Lyndon and Łoś-Tarski*, The Journal of Symbolic Logic **65** (2000), pp. 310–332.
[3] Fischer, M. J. and R. E. Ladner, *Propositional dynamic logic of regular programs*, J. Comput. Syst. Sci. **18** (1979), pp. 194–211.
[4] Fontaine, G., "Modal fixpoint logic: some model-theoretic questions," Ph.D. thesis, ILLC (University of Amsterdam) (2010).
[5] Fontaine, G. and Y. Venema, *Some model theory for the modal μ-calculus: syntactic characterizations of semantic properties* (2012), submitted.
[6] Grädel, E., W. Thomas and T. Wilke, editors, "Automata, Logics, and Infinite Games: A Guide to Current Research," Lecture Notes in Computer Science **2500**, Springer, 2002.
[7] Harel, D., J. Tiuryn and D. Kozen, "Dynamic Logic," MIT Press, Cambridge, MA, USA, 2000.
[8] Hollenberg, M., "Logic and Bisimulation," Ph.D. thesis, University of Utrecht (1998).
[9] Janin, D. and I. Walukiewicz, *On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic*, in: U. Montanari and V. Sassone, editors, *CONCUR*, Lecture Notes in Computer Science **1119** (1996), pp. 263–277.
[10] Kozen, D., *Results on the propositional mu-calculus*, Theor. Comput. Sci. **27** (1983), pp. 333–354.
[11] Kozen, D., *Automata on guarded strings and applications*, Technical report, Ithaca, NY, USA (2001).
[12] Muller, D. E., A. Saoudi and P. E. Schupp, *Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time*, in: *Proceedings of the Third Annual IEEE Symposium on Logic in Computer Science (LICS 1988)* (1988), pp. 422–427.
[13] Streett, R. S., *Propositional dynamic logic of looping and converse*, in: *STOC* (1981), pp. 375–383.
[14] Streett, R. S., *Propositional dynamic logic of looping and converse is elementarily decidable*, Information and Control **54** (1982), pp. 121 – 141.
[15] van Benthem, J., "Exploring Logical Dynamics," CSLI Publications, Stanford, California, 1996.
[16] van Benthem, J. and D. Ikegami, *Modal fixed-point logic and changing models*, in: A. Avron, N. Dershowitz and A. Rabinovich, editors, *Pillars of Computer Science*, Lecture Notes in Computer Science **4800** (2008), pp. 146–165.
[17] Vardi, M. Y. and P. Wolper, *Automata-theoretic techniques for modal logics of programs*, Journal of Computer and System Sciences **32** (1986), pp. 183 – 221.
[18] Venema, Y., *Lecture notes on the modal μ-calculus* (2011).

Appendix: Proofs

Proof of Proposition 3.8. By a simultaneous induction on formulas and programs, we prove that
(1) for any PDL-program π, and any formula $\alpha \in \mu\mathrm{ML}$:
  (1a) $f_\pi(\alpha)$ belongs to $\mu_{ca}\mathrm{ML}$ if $\alpha \in \mu_{ca}\mathrm{ML}$;
  (1b) $f_\pi(\alpha) \in \mathrm{CAF}(X)$ if $\alpha \in \mathrm{CAF}(X)$ and $\mathrm{Var}(\pi) \cap X = \varnothing$;
  (1c) $\langle\pi\rangle\alpha \equiv f_\pi(\alpha)$.
(2) for any PDL-formula α:
  (2a) $\alpha^t \in \mu_{ca}\mathrm{ML}$;
  (2b) $\alpha \equiv \alpha^t$.
Analogous statements can be proved for $\mathrm{PDL}^{tf\epsilon}$, $\mathrm{RAF}(X)$ and $\mu_{rca}\mathrm{ML}$.

For the proof of (1), we confine our attention to the case where $\pi = \rho^*$. Take an arbitrary formula $\alpha \in \mu\mathrm{ML}$. Recall that $f_\pi(\alpha)$ is of the form $\mu x.\alpha \vee f_\rho(x)$, where x does not occur in either α or ρ. For (1a), suppose that $\alpha \in \mu_{ca}\mathrm{ML}$. By the inductive hypothesis (1b), applied to ρ and the formula $x \in \mathrm{CAF}(x)$, we have that $f_\rho(x) \in \mathrm{CAF}(x)$. Thus we see that $\alpha \vee f_\rho(x) \in \mathrm{CAF}(x)$ as well, and so $\mu x.\alpha \vee f_\rho(x)$ belongs to the set $\mu_{ca}\mathrm{ML}$ indeed. For (1b), let X be a set of variables that do not occur in ρ, and are such that $\alpha \in \mathrm{CAF}(X)$. Since x does not occur in α this means that $\alpha \in \mathrm{CAF}(X \cup \{x\})$. Since x, as a formula, also belongs to the set $\mathrm{CAF}(X \cup \{x\})$, and $\mathrm{Var}(\pi) \cap (X \cup \{x\}) = \varnothing$, an application of the inductive hypothesis shows that $f_\rho(x) \in \mathrm{CAF}(X \cup \{x\})$ as well. Hence the disjunction $\alpha \vee f_\rho(x) \in \mathrm{CAF}(X \cup \{x\})$, and from this we may conclude that, indeed, $f_\pi(\alpha)$ belongs to the set $\mathrm{CAF}(X)$. For (1c), it is obvious that $\langle\rho^*\rangle\alpha \equiv \mu x.\alpha \vee \langle\rho\rangle x \equiv \mu x.\alpha \vee f_\rho(x) \equiv f_\pi(\alpha)$.

For the proof of (2), we only consider the inductive case where α is of the form $\langle\pi\rangle\beta$. Inductively, we may assume that $\beta^t \in \mu_{ca}\mathrm{ML}$, and that $\beta \equiv \beta^t$. But then it follows from the inductive hypothesis, applied to the program π, that $\alpha^t = f_\pi(\beta^t)$ belongs to $\mu_{ca}\mathrm{ML}$ as well (1a), and that $\alpha^t = f_\pi(\beta^t)$ is equivalent to the formula $\alpha = \langle\pi\rangle\beta$. This suffices to prove the proposition. □

Proof of Proposition 3.9. We prove the proposition via a mutual induction on the fragments $\mu_{ca}\mathrm{ML}$ and CAF. The stronger statements concerning formulas in the restricted fragments follow from an easy inspection.

We first consider item (i). Leaving the other cases as exercises for the reader, we focus on the interesting case of the inductive step, where we are dealing with a formula $\mu x.\varphi$, with $\varphi \in \mu_{ca}\mathrm{ML} \cap \mathrm{CAF}(x)$. In order to find the right translation for this formula, we use the induction hypothesis of item (ii). That is, we assume that φ has been rewritten as an equivalent disjunction $\psi \vee \langle\pi_x\rangle x$, where x does not occur in either ψ or π_x. Hence, if we put

$$(\mu x.\varphi)^s := \langle\pi_x^*\rangle\psi,$$

it is easy to verify that this definition satisfies the required properties.

This leaves the proof of item (ii). In case $\varphi = x$, simply take $\psi := \bot$,

$Y := \{x\}$, and $\pi_x := \top?$. (In the case $\varphi \in \mu_{rca}\mathrm{ML} \cap \mathrm{RAF}(X)$ and we need to land in test-free PDL, we put $\pi_x := \epsilon$. This is the reason for adding the skip program ϵ to PDL^{tf}.) Clearly $\varphi \equiv \bot \vee \langle d \rangle x$. In case φ is X-free, inductively we may assume that we have applied item 1 to φ, obtaining the equivalent PDL-formula φ^s. Take $\psi := \varphi^s$, and put $Y := \varnothing$. Clearly then $\bigvee_{x \in Y} \langle \pi_x \rangle x \equiv \bot$, so that indeed we find $\varphi \equiv \varphi^s \vee \bigvee_{x \in Y} \langle \pi_x \rangle x$.

We leave the inductive cases where $\varphi = \psi \wedge \varphi'$, $\varphi = \varphi' \vee \varphi''$ and $\varphi = \langle d \rangle \varphi'$ as exercises, and turn to the case where $\varphi = \mu z.\varphi'$. We may apply the inductive hypothesis with respect to φ' and $X \cup \{z\}$, which gives an $X \cup \{z\}$-free (and hence, X-free) formula ψ', a subset $Y \subseteq X \cup \{z\}$, and a program π'_y for each $y \in Y$, such that

$$\varphi' \equiv \psi' \vee \bigvee_{y \in Y} \langle \pi'_y \rangle y.$$

Now distinguish cases. If $z \notin Y$ then $Y \subseteq X$ and the formula $\psi' \vee \bigvee_{y \in Y} \langle \pi'_y \rangle y$ is z-free, so that $\varphi = \mu z.\varphi' \equiv \psi' \vee \bigvee_{y \in Y} \langle \pi'_y \rangle y$ and we are done. If, on the other hand, $z \in Y$, then define $Y' := Y \setminus \{z\}$, so that we have $Y' \subseteq X$ and

$$\varphi' \equiv \psi' \vee \bigvee_{y \in Y'} \langle \pi'_y \rangle y \vee \langle \pi'_z \rangle z.$$

From this it is immediate that

$$\mu z.\varphi' \equiv \langle (\pi'_z)^* \rangle (\psi' \vee \bigvee_{y \in Y'} \langle \pi'_y \rangle y),$$

and so we find that

$$\mu z.\varphi' \equiv \langle (\pi'_z)^* \rangle \psi' \vee \bigvee_{y \in Y'} \langle (\pi'_z)^*;\pi'_y \rangle y,$$

from which we can read off the formula $\psi := \langle (\pi'_z)^* \rangle \psi'$ and the programs $\pi_y := (\pi'_z)^*;\pi'_y$, for each $y \in Y'$. □

Proof of Lemma 4.5. A PDL^{tf}-program π is nothing but a regular expression over D. Regular expressions over D can be given semantics over Kripke models such that they denote a set of paths. Using this approach, we know that there is a non-deterministic finite-state automaton (NFA) which recognizes the same language as π. Let $\underline{\mathrm{A}}_\pi = \langle A, \delta, F, a_I \rangle$ be such an automaton. Since $\underline{\mathrm{A}}_\pi$ accepts the language denoted by the regular expression π, it is straightforward to verify that $\underline{\mathrm{A}}_\pi$ accepts the path $s, \ldots, t$ iff $R_\pi(s,t)$.

Next we define $\mathbb{P}_\pi = \langle A, \Delta, \Omega \rangle$ where for all $a \in A$ the transition map is

$$\Delta(a) := \bigvee_{d \in \mathsf{D}, b \in \delta(a,d)} \langle d \rangle b \ ,$$

and the parity map is $\Omega(a) := 1$ for every element. It is not difficult to see that $\mathbb{P}_\pi$ is a well-defined PDL^{tf}-automaton: it satisfies the PDL^{tf} restrictions

for cycles because every state appears under a diamond and there are only disjunctions in the transition map, and the other conditions (e.g., weakness) are trivially satisfied.

Furthermore, it is clear from the definition of $\mathbb{P}_\pi$ that the following are equivalent, for any $(a, s) \in A \times S$:

(i) (b, t) is an admissible move for $\exists$ in $\underline{\mathcal{A}}(\underline{\mathrm{A}}_\pi, \mathbb{S})@(a, s)$,
(ii) $\{(b, t)\}$ is an admissible move for $\exists$ in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a, s)$.

Now consider the triple $(\mathbb{P}_\pi, a_\pi, F)$ consisting of the initialized automaton $(\mathbb{P}_\pi, a_\pi)$ where $a_\pi := a_I$ and F is the set of final states. Combining the above claims we get that $R_\pi(s, t)$ iff $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s)$ taking her to (f, t) for some $f \in F$. This finishes the proof of the lemma. □

Proof of Proposition 4.7. ($\Rightarrow$) Suppose $\mathbb{S}, s \Vdash \langle\pi\rangle\alpha$. By definition there is $t \in S$ such that $R_\pi(s, t)$ and $\mathbb{S}, t \Vdash \alpha$. Using Lemma 4.5 we know that therefore $\exists$ has a surviving strategy in $\mathcal{A}(\mathbb{P}_\pi, \mathbb{S})@(a_\pi, s)$ taking her to (f, t) for some $f \in F$. Now $\exists$ can use that strategy to play a match in $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(a_\varphi, s)$ and get to the same position (f, t). By inductive hypothesis (as $\mathbb{S}, t \Vdash \alpha$) we know that $\exists$ has a winning strategy in $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_A, t)$. Because of the way the transition map Δ is defined, she can use that same strategy to win $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(f, t)$.

($\Leftarrow$) Suppose that $\exists$ has a winning strategy in $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(a_\varphi, s)$. As the parity of $\mathbb{P}_\pi$ is 1 for every element this means that $\exists$ plays finitely many moves in $\mathbb{P}_\pi$ which get her to some position (f, t) and then makes a move which takes her to the $\mathbb{A}$ part of the automaton. Observe that this can only happen if $f \in F$. Using Lemma 4.5 we get that $R_\pi(s, t)$. As $\exists$ has a winning strategy in $\mathcal{A}(\mathbb{A}_\varphi, \mathbb{S})@(f, t)$ and because of how Δ is defined, she can use that same strategy to win the game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(f, t)$ and thus by inductive hypothesis we get that $\mathbb{S}, t \Vdash \alpha$. By definition, this means that $\mathbb{S}, s \Vdash \langle\pi\rangle\alpha$. □

Proof of Lemma 4.11. By induction on $|E|$, we solve this set of equations while preserving $(*)$ and finally get a formula in $\mathrm{PDL}^{tf}(\mathsf{P} \uplus B, \mathsf{D})$.

For the base case let $E = \{e\}$, we have to consider two cases: if $e \notin \xi(e)$ then $\xi(e) = \alpha$ with $\alpha \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus B, \mathsf{D})$ and we are done. Otherwise, the equation should be of the form $\xi(e) = \alpha \vee \langle\pi\rangle e$. Let $\varphi_{E,e} := \langle\pi^*\rangle\alpha$, it is easy to see that the formula belongs to the right fragment. The following claim states that $\varphi_{E,e}$ is equivalent to E.

Claim 1 The following are equivalent:

- $\exists$ has a surviving strategy in $\mathcal{S}(\{e \approx \alpha \vee \langle\pi\rangle e\}_\mu, \mathbb{S})@(e, s)$ taking her to (b, t).
- $\mathbb{S}, m, s \Vdash \langle\pi^*\rangle\alpha$ where $m : S \to \wp B$ is such that $Z_m = \{(b, t)\}$.

Proof of Claim. ($\Rightarrow$) As the set of equations is of type μ this means that $\exists$ plays only a finite number of moves, otherwise she would lose. Because of the shape of the set of equations she has to play markings $m_1, \ldots, m_k$ such that

$e \in m_i(s_i)$ for some s_i and in each turn $\forall$ chooses (e, s_i). After that $\exists$ plays a marking m such that $e \in m(t)$ and $\forall$ must choose (b, t). It is clear to observe that the first k rounds induce a π^*-path $s, s_1, \ldots, s_k$ and the last round implies that $\mathbb{S}, m, s_k \Vdash \alpha$. It is only left to observe that as $\exists$ can *force* $\forall$ to choose (b, t) then it must be the case that $Z_m = \{(b, t)\}$.

($\Leftarrow$) Assume $\mathbb{S}, m, s \Vdash \langle \pi^* \rangle \alpha$, then by definition there is an s_k such that $R^*_\pi(s, s_k)$ and $\mathbb{S}, m, s_k \Vdash \alpha$. Moreover this means that there are $s_1, \ldots, s_k$ such that $R_\pi(s_i, s_{i+1})$. We can give a surviving strategy for $\exists$ as follows: first she plays, in order, markings $m_1, \ldots, m_k$ such that $Z_{m_i} = \{(e, s_i)\}$. These markings constitute legitimate moves for $\exists$ and constrain $\forall$ to follow the path $s, s_1, \ldots, s_k$. Finally, she plays the marking m which by hypothesis makes α true at s_k and leaves $\forall$ only one choice, namely (b, t). ◀

For the inductive case let $E = \{e, e_1, \ldots, e_n\}$ with $n > 0$. If $e \notin \xi(e)$ we skip to the next step, otherwise we need to treat this equation first. Let $\xi(e) = \alpha \vee \langle \pi \rangle e \vee \bigvee_{u \in U} \langle \pi_u \rangle u$ be such that $e \notin U$. In order to eliminate e from $\xi(e)$ we create a slightly modified version of $\mathbf{E}$.

Claim 2 Let $\mathbf{E}' := (E, \xi', \mu)$ with $\xi'(e) := \langle \pi^* \rangle \alpha \vee \bigvee_{u \in U} \langle \pi^*; \pi_u \rangle u$ and let $\xi'(e_i) := \xi(e_i)$ for all i. For all $s, t \in |\mathbb{S}|$ and $b \in B$, the following are equivalent,

(i) $\exists$ has a surviving strategy in $\mathcal{S}(\mathbf{E}, \mathbb{S})@(e, s)$ taking her to (b, t),
(ii) $\exists$ has a surviving strategy in $\mathcal{S}(\mathbf{E}', \mathbb{S})@(e, s)$ taking her to (b, t).

Proof of Claim. As the two sets only differ on e, it will be enough to show that, given a strategy for $\exists$, we can simulate the moves made by $\exists$ (when standing at e) in one set of equations using the other set of equations.

($\Rightarrow$) The type of $\mathbf{E}$ is μ, therefore $\exists$ will only play a finite amount of moves. Assume $\exists$ plays, in order, markings $m_1, \ldots, m_k$ such that $\forall$ chooses (e, s_i) on each round and finally plays a marking m such that $\forall$ chooses (x, s') with $x \neq e$. It is easy to check that in $\mathbf{E}'$ she can play m and will also get to (x, s').

($\Leftarrow$) Suppose $\exists$ plays a marking m such that it actually makes $\langle \pi^* \rangle \alpha$ true (the case for $\langle \pi^*; \pi_u \rangle u$ is analogous) and $\forall$ chooses (x, s'). This means that there is an R_{π^*} path $s, s_1, \ldots, s_k$ and a marking m_α with $\mathbb{S}, m_\alpha, s_k \Vdash \alpha$. She can simulate this play in $\mathbf{E}$ by playing as follows: first she plays, in order, markings m_i such that $Z_{m_i} = \{(e, s_i)\}$; after that she plays m_α. ◀

Having removed e from $\xi(e)$, we still have a formula where other elements of E may occur. We first substitute $\xi'(e)$ into the other equations, setting $\xi'(e_i) := \xi(e_i)[e \mapsto \xi'(e)]$ for all i. It is easy to see that this substitution preserves the behaviour of $\mathbf{E}'$.

Using the distribution laws of the diamond and PDL^{tf} identities the new formulas can be taken to the normal form in ($*$). To illustrate the process suppose $\xi(e_i) = \alpha \vee \langle \pi_e \rangle e \vee \bigvee_{u \in U} \langle \pi_u \rangle u$ with $e \notin U$ and $\xi'(e) = \alpha' \vee \bigvee_{u \in U} \langle \pi'_u \rangle u$.[4]

[4] To simplify the presentation we assume that U is the same in $\xi(e_i)$ and $\xi(e)$. This need not be this way but the process can be easily adjusted to work for the general case.

The formula $\xi'(e_i)$ is then obtained as follows:

$$\alpha \vee \langle \pi_e \rangle e \vee \bigvee_{u \in U} \langle \pi_u \rangle u \qquad \text{(before replacement)}$$

$$\alpha \vee \langle \pi_e \rangle \big(\alpha' \vee \bigvee_{u \in U} \langle \pi'_u \rangle u\big) \vee \bigvee_{u \in U} \langle \pi_u \rangle u \qquad \text{(after replacement)}$$

$$\big(\alpha \vee \langle \pi_e \rangle \alpha'\big) \vee \bigvee_{u \in U} \langle \pi_e \rangle \langle \pi'_u \rangle u \vee \langle \pi_u \rangle u \qquad \text{(distribution of diamonds, regrouping)}$$

$$\big(\alpha \vee \langle \pi_e \rangle \alpha'\big) \vee \bigvee_{u \in U} \langle \pi_e ; \pi'_u \oplus \pi_u \rangle u \qquad \text{(program identities)}$$

We inductively solve the smaller set of equations $\mathbf{E}'' := (E \setminus \{e\}, \xi', \mu)$ and get formulas ψ_u for every $u \in E \setminus \{e\}$. Finally we give a solution for e setting $\varphi_{E,e} := \xi'(e)[u \mapsto \psi_u \mid u \in E \setminus \{e\}]$. Observe that $\varphi_{E,e} \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus B, \mathsf{D})$ because it is of the form $\alpha \vee \bigvee_{u \in U} \langle \pi_u \rangle \psi_u$ where (by induction and hypothesis) we have $\alpha, \psi_u \in \mathrm{PDL}^{tf}(\mathsf{P} \uplus B, \mathsf{D})$. □