Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 305 (2014) 5–18

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Bisimilar and Logically Equivalent Programs in PDL

Mario R. F. Benevides[1,2]

Computer Science Department and Systems and Computer Engineering Program
Federal University of Rio de Janeiro
Brazil

Abstract

In standard Propositional Dynamic Logic (PDL) literature [5,16,4] the semantics is given by Labeled Transition Systems, where for each program π we a associate a binary relation R_π. Process Algebras [1,8,10,2] also give semantics to process (terms) by means of Labeled Transition Systems. In both formalisms, PDL and Process Algebra, the key notion to compare processes is bisimulation. In PDL, we also have the notion of logic equivalence, that can be used to prove that two programs π_1 and π_2 are logically equivalent $\vdash \langle\pi_1\rangle\varphi \leftrightarrow \langle\pi_2\rangle\varphi$. Unfortunately, logic equivalence and bisimulation do not match in PDL. Bisimilar programs are logic equivalent but the converse does not hold.
This paper proposes a semantics and an axiomatization for PDL that makes logically equivalent programs also bisimilar. We prove soundness, completeness and the finite model property.

Keywords: Bisimulation, Propositional Dynamic Logic, Modal Logic

1 Motivation

In standard PDL literature [5,16,4] the semantics is given by Labeled Transition Systems, where for each program π we associate a binary relation R_π. The sequential composition and non-deterministic choice operators are defined as the composition and union of relations respectively.

$$R_{\pi_1;\pi_2} = R_{\pi_1} \circ R_{\pi_2} \quad R_{\pi_1 \cup \pi_2} = R_{\pi_1} \cup R_{\pi_2}$$

Process Algebras [1,8,10,2] also give semantics to process (terms) by means of Labeled Transition Systems. In both formalisms, PDL and Process Algebra, the key notion to compare processes is bisimulation. In PDL, we also have the notion of logic equivalence, that can be used to prove that two programs π_1 and π_2 are logically equivalent $\vdash \langle\pi_1\rangle\varphi \leftrightarrow \langle\pi_2\rangle\varphi$ (where $\langle\pi_i\rangle\varphi$ means that after the execution

[1] This work was supported by the Brazilian research agencies CNPq and CAPES.

[2] Email: mario@cos.ufrj.br

http://dx.doi.org/10.1016/j.entcs.2014.06.002

of program π_i formula φ holds). Unfortunately, logic equivalence and bisimulation do not match in PDL. Bisimilar programs are logic equivalent but the converse does not hold. For instance, take programs $\pi_1 = a;(\pi_3 \cup \pi_4)$ and $\pi_2 = a;\pi_3 \cup a;\pi_4$

$$\begin{aligned}
\langle a;(\pi_3 \cup \pi_4)\rangle\varphi &\leftrightarrow \langle a\rangle\langle(\pi_3 \cup \pi_4)\rangle\varphi \\
&\leftrightarrow \langle a\rangle(\langle\pi_3\rangle\varphi \vee \langle\pi_4\rangle\varphi) \\
&\leftrightarrow \langle a\rangle\langle\pi_3\rangle\varphi \vee \langle a\rangle\langle\pi_4\rangle\varphi \\
&\leftrightarrow \langle a;\pi_3\rangle\varphi \vee \langle a;\pi_4\rangle\varphi \\
&\leftrightarrow \langle a;\pi_3 \cup a;\pi_4\rangle\varphi
\end{aligned}$$

But it is not difficult to see that π_1 and π_2 are non-bisimilar programs, for after first a step on π_1 it arrives at $\pi_3 \cup \pi_4$, and this is matched by neither of the two possibilities on π_2: π_3 or π_4.

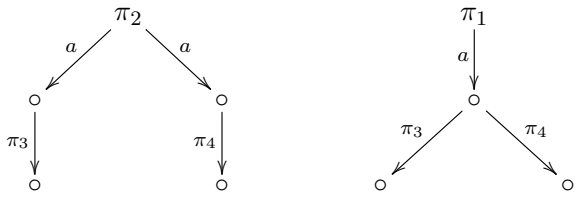


Fig. 1. Non-Bisimilar Programs

One interesting discussion on trace semantics appears on [2]. They define and compare various semantics for concrete sequential processes and provide algebraic axiomatization and semantical modal characterization (no modal axiomatization) for them.

The main motivation of this work is to propose a new semantics for PDL, based on traces with context, which matches the notion of logic equivalence and with bisimulation, i. e., two programs π_1 and π_2 are logically equivalent ($\vdash \langle\pi_1\rangle\varphi \leftrightarrow \langle\pi_2\rangle\varphi$) if and only if they are bisimilar. We provide an axiomatization and prove completeness w.r.t this new semantics. The proof completeness yields finite model property and decidability.

It is important to notice that our contribution is on PDL and not on process theory.

2 Propositional Dynamic Logic

In this section, we present the syntax and semantics of PDL.

Definition 2.1 The PDL language consists of a set Φ of countably many proposition symbols, a set Π of countably many basic programs, the boolean connectives $\neg$ and $\wedge$, the program constructors ;, $\cup$ and * and a modality $\langle\pi\rangle$ for every program

π. The formulas are defined as follows:

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle\pi\rangle\varphi, \text{ with } \pi ::= a \mid \pi_1;\pi_2 \mid \pi_1 \cup \pi_2 \mid \pi^\star,$$

where $p \in \Phi$ and $a \in \Pi$.

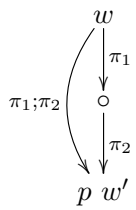
In all the logics that appear in this paper, we use the standard abbreviations $\bot \equiv \neg\top$, $\varphi \vee \phi \equiv \neg(\neg\varphi \wedge \neg\phi)$, $\varphi \to \phi \equiv \neg(\varphi \wedge \neg\phi)$ and $[\pi]\varphi \equiv \neg\langle\pi\rangle\neg\varphi$.

Definition 2.2 A *frame* for PDL is a tuple $\mathcal{F} = (W, \mathbf{R})$ where
- W is a non-empty set of states;
- $\mathbf{R} = \{R_a \mid a \in \Pi\}$, R_a are binary relations over W, for each basic program $a \in \Pi$;
- We can inductively define a binary relation R_π, for each non-basic program π, as follows
  - $R_{\pi_1;\pi_2} = R_{\pi_1} \circ R_{\pi_2}$,
  - $R_{\pi_1\cup\pi_2} = R_{\pi_1} \cup R_{\pi_2}$,
  - $R_{\pi^\star} = R_\pi^\star$, where $R_\pi^\star$ is the reflexive transitive closure of R_π.

Definition 2.3 A *model* for PDL is a pair $\mathcal{M} = (\mathcal{F}, \mathbf{V})$, where $\mathcal{F}$ is a PDL frame and $\mathbf{V}$ is a valuation function $\mathbf{V} : \Phi \to 2^W$.

Definition 2.4 Let $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ be a model. The notion of *satisfaction* of a formula φ in a model $\mathcal{M}$ at a state w, notation $\mathcal{M}, w \Vdash \varphi$, can be inductively defined as follows:
- $\mathcal{M}, w \Vdash p$ iff $w \in \mathbf{V}(p)$;
- $\mathcal{M}, w \Vdash \top$ always;
- $\mathcal{M}, w \Vdash \neg\varphi$ iff $\mathcal{M}, w \nVdash \varphi$;
- $\mathcal{M}, w \Vdash \varphi_1 \wedge \varphi_2$ iff $\mathcal{M}, w \Vdash \varphi_1$ and $\mathcal{M}, w \Vdash \varphi_2$;
- $\mathcal{M}, w \Vdash \langle\pi\rangle\varphi$ iff there is $w' \in W$ s.t. $wR_\pi w'$ and $\mathcal{M}, w' \Vdash \varphi$.

Example 2.5 $\mathcal{M}, w \Vdash \langle(\pi_1;\pi_2)\rangle p$ iff there is $w' \in W$ s.t. $wR_{\pi_1;\pi_2}w'$ and $\mathcal{M}, w' \Vdash p$, iff there is $w' \in W$ s.t. $wR_{\pi_1} \circ R_{\pi_2}w'$ and $\mathcal{M}, w' \Vdash p$



3 Process Calculus

In this section, we propose a very small process (program) calculus for the PDL programs presented in the previous section 2. We prove that two processes are bisimilar if and only if they have the same set of finite possible runs with context. It is inspired on [2].

$\alpha \xrightarrow{\alpha} \surd$	$\alpha.\pi \xrightarrow{\alpha} \pi$	$\frac{\pi \xrightarrow{\alpha} \pi'}{\pi;\tau \xrightarrow{\alpha} \pi';\tau}$
$\frac{\pi \xrightarrow{\alpha} \pi'}{\pi+\tau \xrightarrow{\alpha} \pi'}$	$\frac{\tau \xrightarrow{\beta} \tau'}{\pi+\tau \xrightarrow{\beta} \tau'}$	$\frac{\pi \xrightarrow{\alpha} \pi'}{\pi^* \xrightarrow{\alpha} \pi';\pi^*}$

Table 1
Transition Relation

Let $\mathcal{N} = \{a, b, c, \ldots\}$ be a set of names or actions, denoted by $\alpha, \beta, \ldots$. The language can be defined as follows.

$$\pi ::= \alpha \mid \pi_1;\pi_2 \mid \pi_1 + \pi_2 \mid \pi^\star, \ \textit{where } \alpha \in \mathcal{N}$$

We use π and τ to denote processes (programs) and α, β and γ to denote actions.

We write $\pi \xrightarrow{\alpha} \pi'$ to express that the process π can perform the action α and after that behave as π'. We write $\pi \xrightarrow{\alpha} \surd$ to express that the process π successfully finishes after performing the action α. A process finishes when there is no possible action left for it to perform. For example, $\beta \xrightarrow{\beta} \surd$.

The semantics of our process calculus can be given by the transition rules presented in the table below.

The concept of bisimulation is a key notion in any process algebra. It is an equivalence relation between processes which have mutaly similar behavior. The intuition is that two bisimilar processes cannot be distinguished by an external observer.

Definition 3.1 Let $\mathcal{P}$ be the set of all processes. A set $Z \subseteq \mathcal{P} \times \mathcal{P}$ is a *strong bisimulation* if $(\pi, \tau) \in Z$ implies the following:

- If $\pi \xrightarrow{\alpha} \pi'$, then there is $\tau' \in \mathcal{P}$ such that $\tau \xrightarrow{\alpha} \tau'$ and $(\pi', \tau') \in Z$;
- If $\tau \xrightarrow{\alpha} \tau'$, then there is $\pi' \in \mathcal{P}$ such that $\pi \xrightarrow{\alpha} \pi'$ and $(\pi', \tau') \in Z$;
- $\pi \xrightarrow{\alpha} \surd$ if and only if $\tau \xrightarrow{\alpha} \surd$.

Definition 3.2 Two process π and τ are *strongly bisimilar* (or simply *bisimilar*), denoted by $\pi \sim \tau$, if there is a strong bisimulation Z such that $(\pi, \tau) \in Z$.

Proposition 3.3 $\pi_1;(\pi_2 + \pi_3) \not\sim \pi_1;\pi_2 + \pi_1;\pi_3$

See figue 1.

3.1 Runs with Context

In this section, we introduce the key concept of *finite possible runs with context* of a process. This concept plays a central role in the semantics of our logics.

Definition 3.4 A sequence of action with context, denoted by $\overrightarrow{\alpha}^c$, is a sequence of actions and finite sets of actions of the form

$$\alpha_1\{\beta_1^1 \cdots \beta_{k_1}^1\}.\alpha_2\{\beta_1^2 \cdots \beta_{k_2}^2\}, \cdots .\alpha_n\{\beta_1^n \cdots \beta_{k_n}^n\}.\cdots,$$

where $\alpha_i \notin \{\beta_1^i \cdots \beta_{k_i}^i\}$, for $1 \leq i \leq n$.

If $\overrightarrow{\alpha}^c = \alpha_1 C_1.\alpha_2 C_2 \cdots .\alpha_n C_n$ is a finite sequence of action with context, we say that the length of $\overrightarrow{\alpha}^c$ is n.

Definition 3.5 Let $\overrightarrow{\alpha}^c = \alpha_1 C_1.\alpha_2 C_2 \cdots .\alpha_n C_n$ and $\overrightarrow{\beta}^c = \beta_1 D_1.\beta_2 D_2 \cdots .\beta_n D_n$ sequences of action with context of length n. We define a strict partial order over sequences of actions with context as follows

$$\overrightarrow{\alpha}^c \prec \overrightarrow{\beta}^c \text{ iff for all } i,\ 1 \leq i \leq n, \quad \alpha_i = \beta_i,\ C_i \subseteq D_i,\ \text{for at least one } i,\ C_i \subset D_i.$$

Definition 3.6 Let $\overrightarrow{\alpha}^c = \alpha_1\{\beta_1^1 ... \beta_{k_1}^1\}\alpha_2\{\beta_1^2 ... \beta_{k_2}^2\}\ ...\alpha_n\{\beta_1^n \cdots \beta_{k_n}^n\}$ be a sequence of action with context. We say that $\overrightarrow{\alpha}^c$ *matches a process* π_0 if $\pi_0 \stackrel{\alpha_1}{\rightarrow} \pi_1 \stackrel{\alpha_2}{\rightarrow} \pi_2 \cdots \pi_{n-1} \stackrel{\alpha_n}{\rightarrow} \pi_n$ and for all i, $0 \leq i < n$, $\{\alpha_i, \beta_1^i \cdots \beta_{k_i}^i\}$ are all the actions that π_i can perform.

We write $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \pi'$ to express that $\overrightarrow{\alpha}^c$ matches π and the process π may perform the sequence of actions $\overrightarrow{\alpha}^c$ and after that behave as π'. We write $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ to express that $\overrightarrow{\alpha}^c$ matches π and the process π may successfully finish after performing the sequence of actions $\overrightarrow{\alpha}^c$ (this, in particular, implies that $\overrightarrow{\alpha}^c$ is finite).

Definition 3.7 We define the set of finite possible runs with context of a process π, denoted by $\overrightarrow{\mathcal{R}_f^c}(\pi)$, as $\overrightarrow{\mathcal{R}_f^c}(\pi) = \{\overrightarrow{\alpha}^c : \pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd\}$.

In order to obtain the desired relation between bisimulation and logic equivalence, we introduce the concept of *finite* possible runs with context of processes, i.e., situations in which the processes successfully finish. Thus, we present some useful results about finite possible runs with context. It is important to notice that in our process calculus all processes, at any state in their execution, can only perform a finite set of actions, i.e., they are *image finite*.

Definition 3.8 Let R and S be sets of finite sequences of actions with context. We can define the following operations on these sets:

(i) $R \circ S = \{\overrightarrow{\alpha}^c.\overrightarrow{\beta}^c : \overrightarrow{\alpha}^c \in R \text{ and } \overrightarrow{\beta}^c \in S\}$;
(ii) $R \cup S = \{\overrightarrow{\alpha}^c : \overrightarrow{\alpha}^c \in R \text{ or } \overrightarrow{\alpha}^c \in S\}$;
(iii) $R^0 = \{\overrightarrow{\varepsilon}\}$, $R^n = R \circ R^{n-1} (n \geq 1)$;
(iv) $R^* = \bigcup_{n \in \mathbb{N}} R^n$.

Lemma 3.9 *If $\pi \sim \tau$, then, for every $\overrightarrow{\alpha}^c$, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ iff $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$.*

A proof of this lemma can be found in appendix A. This proof is based in similar one presented in [3].

Lemma 3.10 *If for every $\overrightarrow{\alpha}^c$, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ if and only if $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$, then $\pi \sim \tau$*

Proof. Suppose $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ if and only if $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ and $\pi \not\sim \tau$. Then there exists α_1 such that $\pi \stackrel{\alpha_1}{\rightarrow} \pi_1$ and for all τ_1 either $\tau \stackrel{\alpha_1}{\not\rightarrow} \tau_1$ (1) or $\pi_1 \not\sim \tau_1$ (2). But (1) cannot be true,

because it contradicts the hypothesis that π and τ are able to perform the same set of actions, because if α_1 is the first action in some sequence of action $\overrightarrow{\alpha}^c$, then its context contains all the actions that π and τ can perform. The only remaining possibility is (2), $\pi_1 \not\sim \tau_1$. If we apply the same reasoning for $\pi_2 \not\sim \tau_2$ and so on for $\pi_i \not\sim \tau_i$ and π_i and τ_i must be able to perform the same set of actions. As all processes eventually terminate, we must eventually reach a pair π_n and τ_n such that $\pi_n \not\sim \tau_n$ and π_n and τ_n must be able to perform the same set of actions $\gamma_n^1, \cdots \gamma_n^{k_n}$ and either $\pi_n = \surd$ or $\tau_n = \surd$ or both $\pi_n = \tau_n = \surd$. The first two cases are not possible because π_n and τ_n must be able to perform the same set of action and $\surd$ does not perform any action and any process different of $\surd$ must be able to perform at least one action. Thus, the only possibility is $\pi_n = \tau_n = \surd$, which yields that $\pi_n \sim \tau_n$, which is a contradiction. Therefore, $\pi \sim \tau$.

□

Theorem 3.11 *$\pi \sim \tau$ if and only if $\overrightarrow{\mathcal{R}_f^c}(\pi) = \overrightarrow{\mathcal{R}_f^c}(\tau)$.*

Proof. ($\Rightarrow$) Suppose that $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$. Then, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$. As $\pi \sim \tau$, this implies, by lemma 3.9, that $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$, which means that $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\tau)$. Thus, $\overrightarrow{\mathcal{R}_f^c}(\pi) \subseteq \overrightarrow{\mathcal{R}_f^c}(\tau)$. The proof that $\overrightarrow{\mathcal{R}_f^c}(\tau) \subseteq \overrightarrow{\mathcal{R}_f^c}(\pi)$ is entirely analogous.

($\Leftarrow$) Suppose that $\overrightarrow{\mathcal{R}_f^c}(\pi) = \overrightarrow{\mathcal{R}_f^c}(\tau)$. By the definition of $\overrightarrow{\mathcal{R}_f^c}(\pi)$ and $\overrightarrow{\mathcal{R}_f^c}(\tau)$, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ if and only if $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$. And by lemma 3.10, $\pi \sim \tau$.

□

Next, we present some equalities between sets of finite possible runs that are useful to the soundness proof of our axiomatization.

Definition 3.12 Let $\overrightarrow{\alpha}^c = \alpha_1 C_1 . \alpha_2 C_2 \cdots . \alpha_n C_n \in \overrightarrow{\mathcal{R}_f^c}(\pi_1)$, $\{\gamma_1, \cdots, \gamma_m\}$ be the set of all actions that π_2 can perform and $C_1' = C_1 \cup \{\gamma_1, \cdots, \gamma_m\}$ and $C_1'' = C_1 \setminus \{\gamma_1, \cdots, \gamma_m\}$. We define

- $\overrightarrow{\alpha}^c \mid^{+\pi_2} = \alpha_1 C_1' . \alpha_2 C_2 \cdots . \alpha_n C_n$
- $\overrightarrow{\alpha}^c \mid^{-\pi_2} = \alpha_1 C_1'' . \alpha_2 C_2 \cdots . \alpha_n C_n$
- $\overrightarrow{\mathcal{R}_f^c}(\pi_1) \mid^{+\pi_2} = \{\overrightarrow{\alpha}^c \mid^{+\pi_2} \;\mid\; \overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi_1)\}$
- $\overrightarrow{\mathcal{R}_f^c}(\pi_1) \mid^{-\pi_2} = \{\overrightarrow{\alpha}^c \mid^{-\pi_2} \;\mid\; \overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi_1)\}$

Theorem 3.13 *The following set equalities are true:*

(i) $\overrightarrow{\mathcal{R}_f^c}(\alpha) = \{\alpha\}$;

(ii) $\overrightarrow{\mathcal{R}_f^c}(\pi_1; \pi_2) = \overrightarrow{\mathcal{R}_f^c}(\pi_1) \circ \overrightarrow{\mathcal{R}_f^c}(\pi_2)$;

(iii) $\overrightarrow{\mathcal{R}_f^c}(\pi_1 + \pi_2) = \overrightarrow{\mathcal{R}_f^c}(\pi_1) \mid^{+\pi_2} \cup \overrightarrow{\mathcal{R}_f^c}(\pi_2) \mid^{+\pi_1}$;

(iv) $\overrightarrow{\mathcal{R}_f^c}(\pi^*) = (\overrightarrow{\mathcal{R}_f^c}(\pi))^*$.

Proof. The proof is straightforward from table 1. □

4 PDL+

In this section we present the language, semantics and an axiomatic system of our Propositional Dynamic Logic with a non-deterministic choice operator.

4.1 Language and Semantics

The language is similar to the one presented in definition 2.1, where we replace + for $\cup$.

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle\pi\rangle\varphi, \text{ with } \pi ::= a \mid \pi_1;\pi_2 \mid \pi_1 + \pi_2 \mid \pi^\star,$$

where $p \in \Phi$ and $a \in \mathcal{N}$.

Definition 4.1 A *frame* for PDL+ is a tuple $\mathcal{F} = (W, R_a)$ where
- W is a non-empty set of states;
- R_a is a binary relation over W, for each basic program $a \in \Pi$;
- We can inductively define a binary relation R_π, for each non-basic program π, as follows

- $R_{\pi_1;\pi_2} = R_{\pi_1} \circ R_{\pi_2}$,
- $R_{\pi_1+\pi_2} = \{(s,t) \mid [(s,t) \in R_{\pi_1} \textit{ and } \exists r(s,r) \in R_{\pi_2}] \textit{ or } [(s,t) \in R_{\pi_2} \textit{ and } \exists r\ (s,r) \in R_{\pi_1}]\}$
- $R_{\pi^\star} = R^\star_\pi$, where $R^\star_\pi$ is the reflexive transitive closure of R_π.

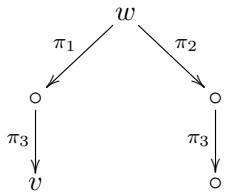
The semantical notion of PDL+ model and satisfaction for PDL+ is as defined for PDL in definitions 2.3 and 2.4

If $\mathcal{M}, w \Vdash \varphi$ for every state w, we say that φ is *globally satisfied* in the model $\mathcal{M}$, notation $\mathcal{M} \Vdash \varphi$. If φ is globally satisfied in all models $\mathcal{M}$ of a frame $\mathcal{F}$, we say that φ is *valid* in $\mathcal{F}$, notation $\mathcal{F} \Vdash \varphi$. Finally, if φ is valid in all frames, we say that φ is valid, notation $\Vdash \varphi$. Two formulas φ and ψ are *semantically equivalent* if $\Vdash \varphi \leftrightarrow \psi$.

Proposition 4.2 $\not\Vdash \langle\pi_1;(\pi_2+\pi_3)\rangle p \leftrightarrow \langle\pi_1;\pi_2+\pi_1;\pi_3\rangle p$

Proof. Let $\mathcal{M}$ be a model based on the frame bellow with $V(p) = \{v\}$. It is easy to verify that

$\mathcal{M}, w \not\Vdash \langle\pi_1;(\pi_2+\pi_3)\rangle p$ and
$\mathcal{M}, w \Vdash \langle\pi_1;\pi_2+\pi_1;\pi_3\rangle p$



□

Next definition and lemma relate our semantics with possible runs with context.

Definition 4.3 Let $\mathcal{F} = (W, R_\alpha)$ be a frame, $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, be a finite path in $\mathcal{F}$ and $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$, of length n, be a sequence of action with context of a process π. We say that $\overrightarrow{\alpha}^c$ *matches* path $(v_0, v_1, \ldots, v_n)$ for process π iff for all i, $1 \leq i \leq n$, $(\overrightarrow{\alpha}^c)_i = \alpha_i\{\beta_1^i ... \beta_{k_i}^i\}$ and $(v_{i-1}, v_i) \in R_{\alpha_i}$ and for all β_j, $1 \leq j \leq k_i$, there exists a w such that $(v_{i-1}, w) \in R_{\beta_j}$. We say that $\overrightarrow{\alpha}^c$ matches exactly path $(v_0, v_1, \ldots, v_n)$, if there exists a unique w such that $(v_{i-1}, w) \in R_\gamma$, for all $\gamma \in \{\beta_1^i \cdots \beta_{k_i}^i\}$ and $1 \leq i \leq n$.

A frame $\mathcal{F}$ *matches a process* π *at state* w iff for all $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$, there exists a path ρ, in $\mathcal{F}$, such that $\overrightarrow{\alpha}^c$ matches ρ.

Lemma 4.4 *$\mathcal{M}, w \Vdash \langle\pi\rangle\varphi$ iff $\mathcal{F}$ matches π at w, there is a finite path $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, such that $v_0 = w$, $\mathcal{M}, v_n \Vdash \varphi$ and there is $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$ of length n such that $\overrightarrow{\alpha}^c$ matches the path $(v_0, \ldots, v_n)$.*

A proof of this lemma can be found in appendix B.

Bellow, we present the main theorem of this section, it establishes the equivalence between bisimilar and logically equivalent programs.

Theorem 4.5 *$\overrightarrow{\mathcal{R}_f^c}(\pi) = \overrightarrow{\mathcal{R}_f^c}(\tau)$ if and only if $\Vdash \langle\pi\rangle p \leftrightarrow \langle\tau\rangle p$.*

Proof. ($\Rightarrow$) Suppose that $\overrightarrow{\mathcal{R}_f^c}(\pi) = \overrightarrow{\mathcal{R}_f^c}(\tau)$, but $\not\Vdash \langle\pi\rangle p \leftrightarrow \langle\tau\rangle p$. Then, we may assume, without loss of generality, that there is a model $\mathcal{M}$ and a state v_0 in this model such that $\mathcal{M}, v_0 \Vdash \langle\pi\rangle p$ (1), but $\mathcal{M}, v_0 \not\Vdash \langle\tau\rangle p$ (2). By lemma 4.4, (1) implies that there is a path $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, in $\mathcal{M}$ such that $\mathcal{M}, v_n \Vdash p$ (3) and there is $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$ that matches this path. But as $\overrightarrow{\mathcal{R}_f^c}(\pi) = \overrightarrow{\mathcal{R}_f^c}(\tau)$, then $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\tau)$. This and (3) imply, by definition 4.4, that $\mathcal{M}, v_0 \Vdash \langle\tau\rangle p$, which contradicts (2).

($\Leftarrow$) Suppose that $\Vdash \langle\pi\rangle p \leftrightarrow \langle\tau\rangle p$ (1), but $\overrightarrow{\mathcal{R}_f^c}(\pi) \neq \overrightarrow{\mathcal{R}_f^c}(\tau)$. Then, we may assume, without loss of generality, that there is $\overrightarrow{\alpha}^c$ such that $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$, but $\overrightarrow{\alpha}^c \notin \overrightarrow{\mathcal{R}_f^c}(\tau)$ and there is no $\overrightarrow{\beta}^c \in \overrightarrow{\mathcal{R}_f^c}(\tau)$ such that $\overrightarrow{\beta}^c \prec \overrightarrow{\alpha}^c$.

Let us build a frame $\mathcal{F}$, that matches π, which consists of a finite tree and has a path $(v_0, \ldots, v_n)$, $n \geq 1$, such that $\overrightarrow{\alpha}^c$ matches exactly the path $(v_0, \ldots, v_n)$ for process π.

Let $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ be a model, such that $\mathbf{V}(p)$ is a singleton which the only element is v_n, $v_n \in \mathbf{V}(p)$. Then, we have a path $(v_0, \ldots, v_n)$ such that $\mathcal{M}, v_n \Vdash p$ and $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}_f^c}(\pi)$ matches this path. By lemma 4.4, $\mathcal{M}, v_0 \Vdash \langle\pi\rangle p$.

As $\overrightarrow{\alpha}^c \notin \overrightarrow{\mathcal{R}_f^c}(\tau)$ and there is no $\overrightarrow{\beta}^c \in \overrightarrow{\mathcal{R}_f^c}(\tau)$ such that $\overrightarrow{\beta}^c \prec \overrightarrow{\alpha}^c$, so as $\overrightarrow{\alpha}^c$ matches exactly the path $(v_0, \ldots, v_n)$, then $(v_0, \ldots, v_n)$ is not matched by any other sequence for process τ. Besides that, there is no other path $(v_0, \ldots, v_m)$, $m \geq 1$, in $\mathcal{M}$ such that $\mathcal{M}, v_m \Vdash p$, because $\mathcal{F}$ is a tree. Thus, by lemma 4.4, $\mathcal{M}, v_0 \not\Vdash \langle\tau\rangle p$, which contradicts (1). □

Corollary 4.6 *$\pi \sim \tau$ if and only if $\Vdash \langle\pi\rangle p \leftrightarrow \langle\tau\rangle p$.*

Proof. It follows directly from theorems 3.11 and 4.5.

□

4.2 Axiomatization

We use the standard boolean abbreviations $\bot$, $\vee$, $\rightarrow$ and $\leftrightarrow$ and the following abbreviations for the duals: $[\pi]\varphi := \neg\langle\pi\rangle\neg\varphi$.

The axiomatization presented below is the standard PDL proof theory extended with a new axiom for non-deterministic choice.

Axioms

1. *All tautologies*,
2. $[\pi](\varphi \rightarrow \psi) \rightarrow ([\pi]\varphi \rightarrow [\pi]\psi)$,
3. $[\pi_1;\pi_2]\varphi \leftrightarrow [\pi_1][\pi_2]\varphi$,
4. $\langle\pi_1 + \pi_2\rangle p \leftrightarrow (\langle\pi_1\rangle p \vee \langle\pi_2\rangle p) \wedge (\langle\pi_1\rangle\top \wedge \langle\pi_2\rangle\top)$,
5. $[\pi^*]\varphi \leftrightarrow \varphi \wedge [\pi][\pi^*]\varphi$,
6. $[\pi^*](\varphi \rightarrow [\pi]\varphi) \rightarrow ([\pi]\varphi \rightarrow [\pi^*]\varphi)$,

Inference Rules

M.P. $\varphi, \varphi \rightarrow \psi/\psi$ U.G. $\varphi/[\pi]\varphi$ SUB. $\varphi/\sigma\varphi$

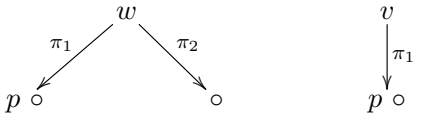
where σ is a map uniformly substituting formulas for propositional variables.

Axioms 1, 2, 3, 5 and 6 and the inference rules are standard in PDL for regular programs [5,16,4]. Axiom 4 deserves some explanation. It can be re-written as

$$\langle\pi_1 + \pi_2\rangle p \leftrightarrow (\langle\pi_1\rangle p \wedge \langle\pi_2\rangle\top) \vee (\langle\pi_1\rangle\top \wedge \langle\pi_2\rangle p)$$

The intuitive meaning is "whenever we perform a non-deterministic choice $\pi_1 + \pi_2$, we must be able to perform either π_1 or π_2, but both must be available for execution. This is what $\langle\pi_i\rangle\top$ $(i = 1, 2)$ assures, i.e., it is possible to perform π_i.

Example 4.7 $\mathcal{M}, w \Vdash \langle(\pi_1 + \pi_2)\rangle p$ and $\mathcal{M}, v \nVdash \langle(\pi_1 + \pi_2)\rangle p$



4.3 Soundness and Completeness

In order to prove soundness it is necessary to show both that every axiom is valid in this class of frames and the inference rules also preserve the validity. The validity of axioms 1, 2, 3, 5 and 6 and the inference rules are well-known from the PDL literature [5,16,4]. Below, we present the proof for axioms 4.

Lemma 4.8 *The following formula is valid:*

$$\Vdash \langle\pi_1 + \pi_2\rangle p \leftrightarrow (\langle\pi_1\rangle p \vee \langle\pi_2\rangle p) \wedge (\langle\pi_1\rangle\top \wedge \langle\pi_2\rangle\top)$$

A proof of this lemma can be found in appendix C.

Theorem 4.9 *(**Soundness**): PDL+ is sound.*

Theorem 4.10 *(**Completeness for Finite PDL+ Models**): Propositional Dynamic Logic PDL+ is complete with respect to the class of finite PDL+ models.*

A proof of this theorem can be found in appendix D.

4.4 Decidability and Complexity

Section 4.3 proves that PDL+ is complete with respect to the classes of finite PDL+ models. Hence, it has have the finite model property, and moreover, that every consistent formula ψ can be satisfied at a state of a model with at most $2^{|\psi|}$, where $|\psi|$ is the number of symbols of ψ. A naive decision procedure for the satisfiability problem of our logic could be: given a formula ψ, construct all Kripke models with at most $2^{|\psi|}$ states, verify if they belong to the appropriate class, and test if ψ is satisfied at some state of them. There are approximately $2^{2^{|\psi|}}$ such models. Therefore, this algorithm establishes a double exponential time upper bound for the satisfiability problem of our logic.

The satisfiability problem for PDL is EXPTIME-complete [5]. This yields an exponential time lower bound for the satisfiability problem of our logic.

5 Conclusion

This paper presents a new semantics to PDL based on finite runs with context, as far as we know this is a new semantics and opens up new possibilities not only for PDL but for other modal logics as well. We propose an axiomatization and prove its soundness, completeness and finite model property. The main result is equivalence between bisimilar programs and logically equivalent programs.

We proved completeness with respect to the class of finite PDL+ and the complexity should be the same as for PDL.

PDL+ opens up possibilities to investigate new variants of PDL where programs are process terms from some process algebra. In [3], a Dynamic Logic for CCS programs was presented, the main criticism of this logic was the lack of equivalence between bisimilar processes and logically equivalent programs. This problem is completely solved with our new semantics. In [3], we also present a logic that uses recursion in the place of iteration. But, in order to keep decidability, we had to restrict the use of recursive equations. In the present work, we use iteration and finite runs, dealing only with terminating programs. We would like to extend this work with recursion and investigate more expressive semantics.

Another possibility for future work would be to establish the precise complexity of the satisfability problem for PDL+. We already have the EXPTIME-hardness due to PDL. We suspect it is EXPTIME-complete, as PDL, but we would like to provide an EXPTIME algorithm for the satisfability problem.

References

[1] J.A. Bergstra, A. Ponse and S.A. Smolka (editors), Handbook of Process Algebra, Elsevier, 2001.

[2] R. J. van Glabbleek, *The Linear Time - Branching Time Spectrum I: The Semantics of Concrete, Sequential Processes*. In Handbook of Process Algebra (J.A. Bergstra, A. Ponse and S.A. Smolka, eds.), Chapter 1, pp. 3-99, Elsevier, 2001.

[3] M. R. F. Benevides and L. M. Schechter. A propositional dynamic logic for CCS programs. In *Proceedings of the XV Workshop on Logic, Language, Information and Computation*, volume 5110 of *LNAI*, pages 83–97. Springer, 2008.

[4] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Theoretical Tracts in Computer Science. Cambridge University Press, 2001.

[5] D. Harel and D. Kozen and J. Tiuryn. *Dynamic Logics*. MIT Press, 2000.

[6] M. Dam. On the decidability of process equivalences for the pi-calculus. *Theoretical Computer Science*, 183(2):215–228, 1997.

[7] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194–211, 1979.

[8] W. J. Fokkink. *Introduction to Process Algebra*. Texts in Theoretical Computer Science. Springer, 2000.

[9] D. Harel and D. Raz. Deciding properties of nonregular programs. *SIAM Journal on Computing*, 22(4):857–874, 1993.

[10] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[11] R. Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999.

[12] R. Milner, J. Parrow, and D. Walker. Modal logics for mobile processes. *Theoretical Computer Science*, 114(1):149–171, 1993.

[13] D. Peleg. Communication in concurrent dynamic logic. *Journal of Computer and System Sciences*, 35(1):23–58, 1987.

[14] D. Peleg. Concurrent dynamic logic. *Journal of the Association for Computing Machinery*, 34(2):450–479, 1987.

[15] C. Stirling. *Modal and Temporal Properties of Processes*. Texts in Computer Science. Springer, 2001.

[16] R. Goldblatt. *Logics of Time and Computation*. CSLI Lecture Notes 7. Stanford, 1992.

A Proof of lemmas 3.10

Proof. We prove by induction on the length n of $\overrightarrow{\alpha}^c$.

• $n = 1$, then $\overrightarrow{\alpha}^c = \alpha\{\beta_1^1 \cdots \beta_{k_1}^1\}$, for some action α. Then, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd \Leftrightarrow \pi \stackrel{\alpha}{\rightarrow} \surd$. By the hypothesis that $\pi \sim \tau$ we have that $\{\alpha, \beta_1^1 \cdots \beta_{k_1}^1\}$ are the only actions π and τ can perform, and $\pi \stackrel{\alpha}{\rightarrow} \surd \Leftrightarrow \tau \stackrel{\alpha}{\rightarrow} \surd$. Finally, $\tau \stackrel{\alpha}{\rightarrow} \surd \Leftrightarrow \tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$.

• I. H.: suppose that the lemma holds for all $n < k$. Let $\overrightarrow{\alpha}^c$ be a sequence of length k. Let $\alpha\{\beta_1^1 \cdots \beta_{k_1}^1\}$ be the first action of the sequence and let $\overrightarrow{\gamma}^c$ be a sequence of length $k-1$ such that $\overrightarrow{\alpha}^c = \alpha\{\beta_1^1 \cdots \beta_{k_1}^1\}.\overrightarrow{\gamma}^c$. Then, $\pi \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$ if and only if there is a process π' such that $\pi \stackrel{\alpha}{\rightarrow} \pi'$ and $\pi' \stackrel{\overrightarrow{\gamma}^c}{\Rightarrow} \surd$. But if $\pi \stackrel{\alpha}{\rightarrow} \pi'$ and $\pi \sim \tau$, then there is a process τ' such that $\tau \stackrel{\alpha}{\rightarrow} \tau'$ and $\pi' \sim \tau'$. Moreover, as $\pi \sim \tau$, then $\{\alpha, \beta_1^1 \cdots \beta_{k_1}^1\}$ are the only actions that π and τ can perform. Now, $\overrightarrow{\gamma}^c$ is a sequence of length shorter than k, so by the induction hypothesis, as $\pi' \sim \tau'$ and $\pi' \stackrel{\overrightarrow{\gamma}^c}{\Rightarrow} \surd$, then $\tau' \stackrel{\overrightarrow{\gamma}^c}{\Rightarrow} \surd$. This means that $\tau \stackrel{\overrightarrow{\alpha}^c}{\Rightarrow} \surd$. □

B Proof of lemma 4.4

Proof. We prove by induction on the structure of π.

The base case is for $|\ \pi\ | = 1$, for atomic program it is straightforward.

Suppose it holds for $|\ \pi\ | \leq n$, so we have three possibilties.

• Suppose $\mathcal{M}, w \Vdash \langle \pi^* \rangle \varphi$, iff iff there is $v \in W$ s. t. $wR_{\pi^*}v$ and $\mathcal{M}, v \Vdash \varphi$. But we know that $R_{\pi^*} = R^*_\pi$. Then we get we have a path $wR_\pi v_1 R_\pi ... R_\pi v$. As R^*_π is transitive $wR_\pi v$ and $\mathcal{M}, v \Vdash \varphi$, but this is iff $\mathcal{M}, w \Vdash \langle \pi \rangle \varphi$. By the induction Hypothesis there is a finite path $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, such that $v_0 = w, v_n = v$, $\mathcal{M}, v_n \Vdash \varphi$ and there is $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi)$ of length n such that $\overrightarrow{\alpha}^c$ matches the path $(v_0, \ldots, v_n)$. But by theorem 3.13 we know that $\overrightarrow{\mathcal{R}^c_f}(\pi) \subseteq (\overrightarrow{\mathcal{R}^c_f}(\pi))^* = \overrightarrow{\mathcal{R}^c_f}(\pi^*)$

• The cases for $\pi = \pi_1 + \pi_2$ and $\pi = \pi_1; \pi_2$ are analogous to the previous case.

□

C Proof of lemma 4.8

Proof.

($\Rightarrow$) Suppose that, for some model $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ and some state w in this model, $\mathcal{M}, w \Vdash \langle \pi_1 + \pi_2 \rangle p$. Then, by lemma 4.4, $\mathcal{F}$ matches $\pi_1 + \pi_2$ at w, there is a finite path $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, such that $v_0 = w$, $\mathcal{M}, v_n \Vdash p$ and a sequence $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_1 + \pi_2)$ that matches this path.

Now, by the third equality in theorem 3.13, either $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)\ |^{+\pi_2}$ or $\overrightarrow{\alpha}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_2)\ |^{+\pi_1}$, It follows directly from definition 3.12 that $\overrightarrow{\alpha}^c\ |^{-\pi_2} \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)$ or $\overrightarrow{\alpha}^c\ |^{-\pi_1} \in \overrightarrow{\mathcal{R}^c_f}(\pi_2)$.

Besides, $\overrightarrow{\alpha}^c\ |^{-\pi_2}$ and $\overrightarrow{\alpha}^c\ |^{-\pi_1}$ match path $(v_0, v_1, \ldots, v_n)$., which implies that $\mathcal{M}, w \Vdash \langle \pi_1 \rangle p$ or $\mathcal{M}, w \Vdash \langle \pi_2 \rangle p$. Thus, $\mathcal{M}, w \Vdash \langle \pi_1 \rangle p \vee \langle \pi_2 \rangle p$ (1).

As $\mathcal{F}$ matches $\pi_1 + \pi_2$ at w, so $\mathcal{F}$ matches π_1 and π_2 at w. Then, there exist $\overrightarrow{\alpha_1}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)$ that matches a path $(w_0, v_1, \ldots, w_k)$ and $\overrightarrow{\alpha_2}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_2)$ that matches a path $(u_0, v_1, \ldots, u_l)$ and $w = w_0 = u_0$. Which implies that $\mathcal{M}, w \Vdash \langle \pi_1 \rangle \top$ and $\mathcal{M}, w \Vdash \langle \pi_2 \rangle \top$. Thus, $\mathcal{M}, w \Vdash \langle \pi_1 \rangle \top \wedge \langle \pi_2 \rangle \top$ (2).

From (1) and (2), we have

$\mathcal{M}, w \Vdash (\langle \pi_1 \rangle p \vee \langle \pi_2 \rangle p) \wedge (\langle \pi_1 \rangle \top \wedge \langle \pi_2 \rangle \top)$

($\Leftarrow$) Suppose $\mathcal{M}, w \Vdash \langle \pi_1 \rangle p \vee \langle \pi_2 \rangle p$ (1) and $\mathcal{M}, w \Vdash \langle \pi_1 \rangle \top \wedge \langle \pi_2 \rangle \top$ (2).

From (2) we have that $\mathcal{F}$ matches π_1 and π_2 at w.

From (1) we have that $\mathcal{M}, w \Vdash \langle \pi_1 \rangle p$ (3) or $\mathcal{M}, w \Vdash \langle \pi_2 \rangle p$ (4).

(3) implies that $\mathcal{F}$ matches π_1 at w and there is a finite path $(v_0, v_1, \ldots, v_n)$, $n \geq 1$, such that $v_0 = w$, $\mathcal{M}, v_n \Vdash p$ and a sequence $\overrightarrow{\alpha_1}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)$ that matches this path.

As $\overrightarrow{\alpha_1}^c \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)$, so $\overrightarrow{\alpha_1}^c\ |^{+\pi_2} \in \overrightarrow{\mathcal{R}^c_f}(\pi_1)\ |^{+\pi_2} \subseteq \overrightarrow{\mathcal{R}^c_f}(\pi_1)\ |^{+\pi_2} \cup \overrightarrow{\mathcal{R}^c_f}(\pi_2)\ |^{+\pi_1} = \overrightarrow{\mathcal{R}^c_f}(\pi_1 + \pi_2)$ (Using theorem 3.13(3.)). Hence, $\overrightarrow{\alpha_1}^c\ |^{+\pi_2} \in \overrightarrow{\mathcal{R}^c_f}(\pi_1 + \pi_2)$.

From (2), we have that $\overrightarrow{\alpha_1}^c\ |^{+\pi_2}$ matches path $(v_0, v_1, \ldots, v_n)$. Thus,
$\mathcal{M}, w \Vdash \langle \pi_1 + \pi_2 \rangle p$.

Analogously, from (4) we also obtain $\mathcal{M}, w \Vdash \langle \pi_1 + \pi_2 \rangle p$.

□

D Completeness Proof for PDL+

The canonical model construction is the standard one used for PDL [5,4,16].

Definition D.1 (**Fischer and Ladner Closure**): Let Γ be a set of formulas. The **closure** of Γ, notation $C_{FL}(\Gamma)$, is the smallest set of formulas satisfying the following conditions:

1. $C_{FL}(\Gamma)$ is closed under subformulas,
2. if $\langle \pi^* \rangle \varphi \in C_{FL}(\Gamma)$, then $\langle \pi \rangle \langle \pi^* \rangle \varphi \in C_{FL}(\Gamma)$,
3. if $\langle \pi_1; \pi_2 \rangle \varphi \in C_{FL}(\Gamma)$, then $\langle \pi_1 \rangle \langle \pi_2 \rangle \varphi \in C_{FL}(\Gamma)$,
4. if $\langle \pi_1 \cup \pi_2 \rangle \varphi \in C_{FL}(\Gamma)$, then $\langle \pi_1 \rangle \varphi \vee \langle \pi_2 \rangle \varphi \in C_{FL}(\Gamma)$,
5. if $\langle \pi_1 \cup \pi_2 \rangle \varphi \in C_{FL}(\Gamma)$, then $\langle \pi_1 \rangle \top$ and $\langle \pi_2 \rangle \top \in C_{FL}(\Gamma)$,
6. if $\varphi \in C_{FL}(\Gamma)$ and φ is not of the form $\neg \psi$, then $\neg \varphi \in C_{FL}(\Gamma)$.

The proof that if Γ is a finite set of formulas, then the closure $C_{FL}(\Gamma)$ of Γ is also finite. We assume Γ to be finite from now on.

Definition D.2 Let Γ be a set of formulas. A set of formulas $\mathcal{A}$ is said to be an **atom of** Γ if it is a maximal consistent subset of $C_{FL}(\Gamma)$. The set of all atoms of Γ is denoted by $At(\Gamma)$.

Lemma D.3 *Let Γ be a set of formulas. If $\varphi \in C_{FL}(\Gamma)$ and φ is consistent then there exists an atom $\mathcal{A} \in At(\Gamma)$ such that $\varphi \in \mathcal{A}$.*

Proof. We can construct the atom $\mathcal{A}$ as follows. First, we enumerate the elements of $C_{FL}(\Gamma)$ as $\phi_1, \cdots, \phi_n$. We start the construction making $\mathcal{A}_1 = \{\varphi\}$, then for $1 < i < n$, we know that $\vdash \bigwedge \mathcal{A}_i \leftrightarrow (\bigwedge \mathcal{A}_i \wedge \phi_{i+1}) \vee (\bigwedge \mathcal{A}_i \wedge \neg\phi_{i+1})$ is a tautology and therefore either $\mathcal{A}_i \wedge \phi_{i+1}$ or $\mathcal{A}_i \wedge \neg\phi_{i+1}$ is consistent. We take $\mathcal{A}_{i+1}$ as the union of $\mathcal{A}_i$ with the consistent member of the previous disjunction. At the end, we make $\mathcal{A} = \mathcal{A}_n$. □

Definition D.4 Let Γ be a set of formulas. The **canonical relations over** Γ S^{Γ}_{π} on $At(\Gamma)$ are defined as follows:

$\mathcal{A}S^{\Gamma}_{\pi}\mathcal{B}$ iff $\bigwedge \mathcal{A} \wedge \langle\pi\rangle \bigwedge \mathcal{B}$ is consistent.

Definition D.5 Let Γ be a set of formulas. The **canonical model over** Γ is a tuple $\mathcal{M}^{\Gamma} =< At(\Gamma), S^{\Gamma}_{\pi}, \mathbf{V}^{\Gamma} >$, where for all propositional symbols p and for all atoms $\mathcal{A} \in At(\Gamma)$ we have

- $\mathbf{V}^{\Gamma}(p) = \{\mathcal{A} \in At(\Gamma) \mid p \in \mathcal{A}\}$ is called canonical valuation;
- S^{Γ}_{π} and $S^{\Gamma+}_{\pi}$ are the canonical relations. [3]

Lemma D.6 *Let $\mathcal{A} \in At(\Gamma)$. Then, for all basic programs α,*

$\langle\alpha\rangle\varphi \in \mathcal{A}$ iff there exists $\mathcal{B} \in At(\Gamma)$ such that $\mathcal{A}S_{\alpha}\mathcal{B}$ and $\varphi \in \mathcal{B}$.

Proof.

$\Rightarrow$: Suppose $\langle\alpha\rangle\varphi \in \mathcal{A}$. By definition D.2, we have that $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle\varphi$ is consistent. Using the tautology $\vdash \varphi \leftrightarrow ((\varphi \wedge \phi) \vee (\varphi \wedge \neg\phi))$, we have that either $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle(\varphi \wedge \phi)$ is consistent or $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle(\varphi \wedge \neg\phi)$ is consistent. So, by the appropriate choice of ϕ, for all formulas $\phi \in C_{FL}$, we can construct an atom $\mathcal{B}$ such that $\varphi \in \mathcal{B}$ and $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle(\varphi \wedge \bigwedge \mathcal{B})$ is consistent and by definition D.4 $\mathcal{A}S_{\alpha}\mathcal{B}$.

$\Leftarrow$: Suppose there is $\mathcal{B}$ such that $\varphi \in \mathcal{B}$ and $\mathcal{A}S_{\alpha}\mathcal{B}$. Then $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle \bigwedge \mathcal{B}$ is consistent and also $\bigwedge \mathcal{A} \wedge \langle\alpha\rangle\varphi$ is consistent. But $\langle\alpha\rangle\varphi \in C_{FL}$ and by maximality $\langle\alpha\rangle\varphi \in \mathcal{A}$.

□

Lemma D.7 *Let $\mathcal{A}, \mathcal{B} \in At(\Gamma)$. Then if $\mathcal{A}S_{\pi*}\mathcal{B}$ then $\mathcal{A}S^{*}_{\pi}\mathcal{B}$.*

Proof. Suppose $\mathcal{A}S_{\pi*}\mathcal{B}$. Let $\mathbf{C} = \{\mathcal{C} \in At(\Gamma) \mid \mathcal{A}S_{\pi}*\mathcal{C}\}$. We want to show that $\mathcal{B} \in \mathbf{C}$. Let $\mathbf{C}^{\wedge}_{\vee} = (\bigwedge \mathcal{C}_1 \vee \cdots \vee \bigwedge \mathcal{C}_n)$.

It is not difficult to see that $\mathbf{C}^{\wedge}_{\vee} \wedge \langle\pi\rangle\neg\mathbf{C}^{\wedge}_{\vee}$ is inconsistent, otherwise for some $\mathcal{D}$ not reachable from $\mathcal{A}$, $\mathbf{C}^{\wedge}_{\vee} \wedge \langle\pi\rangle \bigwedge \mathcal{D}$ would be consistent, and for some $\mathcal{C}_i$, $\bigwedge \mathcal{C}_i \wedge \langle\pi\rangle \bigwedge \mathcal{D}$ was also consistent, which would mean that $\mathcal{D} \in \mathbf{C}$, which is not the case. From a similar reasoning we know that $\bigwedge \mathcal{A} \wedge \langle\pi\rangle\neg\mathbf{C}^{\wedge}_{\vee}$ is also inconsistent and hence $\vdash \bigwedge \mathcal{A} \rightarrow [\pi]\mathbf{C}^{\wedge}_{\vee}$ is a theorem.

As $\mathbf{C}^{\wedge}_{\vee} \wedge \langle\pi\rangle\neg\mathbf{C}^{\wedge}_{\vee}$ is inconsistent, so its negation is a theorem $\vdash \neg(\mathbf{C}^{\wedge}_{\vee} \wedge \langle\pi\rangle\neg\mathbf{C}^{\wedge}_{\vee})$ and also $\vdash (\mathbf{C}^{\wedge}_{\vee} \rightarrow [\pi]\mathbf{C}^{\wedge}_{\vee})$ (1), applying generalization $\vdash [\pi^*](\mathbf{C}^{\wedge}_{\vee} \rightarrow [\pi]\mathbf{C}^{\wedge}_{\vee})$. Using Segerberg axiom (axiom 6), we have $\vdash ([\pi]\mathbf{C}^{\wedge}_{\vee} \rightarrow [\pi^*]\mathbf{C}^{\wedge}_{\vee})$ and by (1) we obtain $\vdash (\mathbf{C}^{\wedge}_{\vee} \rightarrow [\pi^*]\mathbf{C}^{\wedge}_{\vee})$. As $\vdash \bigwedge \mathcal{A} \rightarrow [\pi]\mathbf{C}^{\wedge}_{\vee}$ is a theorem, then $\vdash \bigwedge \mathcal{A} \rightarrow [\pi*]\mathbf{C}^{\wedge}_{\vee}$. By supposition, $\bigwedge \mathcal{A} \wedge \langle\pi^*\rangle \bigwedge \mathcal{B}$ is consistent and so is $\bigwedge \mathcal{B} \wedge \mathbf{C}^{\wedge}_{\vee}$. Therefore, for at least one $\mathcal{C} \in \mathbf{C}$, we know that $\bigwedge \mathcal{B} \wedge \bigwedge \mathcal{C}$ is consistent. By maximality, we have that $\mathcal{B} = \mathcal{C}$. And by the definition of $\mathbf{C}^{\wedge}_{\vee}$, we have $\mathcal{A}S_{\pi}*\mathcal{B}$.

□

Definition D.8 Let Γ be a set of formulas. The **PDL+ model over** Γ is a tuple $\mathcal{M} =< At(\Gamma), R_{\pi}, \mathbf{V} >$, where for all propositional symbols p and for all atoms $\mathcal{A} \in At(\Gamma)$ we have

- $\mathbf{V}(p) = \{\mathcal{A} \in At(\Gamma) \mid p \in \mathcal{A}\}$;
- $R_{\alpha} = S_{\alpha}$, for all basic programs α
- R_{π} is inductively defined as in definition 4.1.

Lemma D.9 $S_{\pi} \subseteq R_{\pi}$.

Proof. Induction on the strauctures of π.

Base case is straightforward as $R_{\alpha} = S_{\alpha}$, for basic programs α.

Suppose it holds for programs π such that $\mid \pi \mid \leq n$. We only prove the case where $\pi = \pi_1 + \pi_2$. The case for $\pi = \pi_1; \pi_2$ and π^* are standard in PDL literature.

Suppose $\mathcal{A}S_{\pi_1+\pi_2}\mathcal{B}$, iff $\bigwedge \mathcal{A} \wedge \langle\pi_1 + \pi_2\rangle \bigwedge \mathcal{B})$ is consistent. By axiom 4. $\bigwedge \mathcal{A} \wedge ((\langle\pi_1\rangle \bigwedge \mathcal{B} \wedge \langle\pi_2\rangle\top) \vee (\langle\pi_2\rangle \bigwedge \mathcal{B} \wedge \langle\pi_1\rangle\top))$ is consistent. Either

$\bigwedge \mathcal{A} \wedge ((\langle\pi_1\rangle \bigwedge \mathcal{B} \wedge \langle\pi_2\rangle\top)$ is consistent (1) or

$(\langle\pi_2\rangle \bigwedge \mathcal{B} \wedge \langle\pi_1\rangle\top))$ is consistent (2).

From (1) $\bigwedge \mathcal{A} \wedge ((\langle\pi_1\rangle \bigwedge \mathcal{B})$ is consistent (3) and

$\bigwedge \mathcal{A} \wedge (\langle\pi_2\rangle\top)$ is consistent (4)

From (3 and (4) we get $\mathcal{A}S_{\pi_1}\mathcal{B}$ and there exists an atom $\mathcal{C}$ s.t. $\mathcal{A}S_{\pi_1}\mathcal{C}$.

By the Induction Hypothesis

$\mathcal{A}R_{\pi_1}\mathcal{B}$ and there exists $\mathcal{C}$ s.t. $\mathcal{A}R_{\pi_1}\mathcal{C}$ (5).

[3] For the sake of clarity we avoid using the Γ subscripts

Analogously, from (2) we can obtain
$\mathcal{A}R_{\pi_1}\mathcal{B}$ and there exists $\mathcal{C}$ s.t. $\mathcal{A}R_{\pi_1}\mathcal{C}$, which together with (5) allows us to conclude $\mathcal{A}R_{\pi_1+\pi_2}\mathcal{B}$.

□

Lemma D.10 Exixtence Lemma: *Let $\mathcal{A} \in At(\Gamma)$. Then,*
$\langle\pi\rangle\varphi \in \mathcal{A}$ iff there exists $\mathcal{B} \in At(\Gamma)$ such that $\mathcal{A}R_\pi\mathcal{B}$ and $\varphi \in \mathcal{B}$.

Proof.
⇒: This direction follows is analogous to the one presented for basic programs in lemma D.6 and the previous lemma that states that $S_\pi \subseteq R_\pi$.
⇐: Induction on the strauture of π.
Base case is straightforward from lemma D.6, for basic programs α.
Suppose it holds for programs π such that $| \pi | \leq n$. We only prove the case where $\pi = \pi_1 + \pi_2$. The case for $\pi = \pi_1;\pi_2$ and π^* are standard in PDL literature.
Suppose $\mathcal{A}R_{\pi_1+\pi_2}\mathcal{B}$ (1) and $\varphi \in \mathcal{B}$ (2). That means that either
$\mathcal{A}R_{\pi_1}\mathcal{B}$ and there exists $\mathcal{C}$ s.t. $\mathcal{A}R_{\pi_2}\mathcal{C}$ (3) or
$\mathcal{A}R_{\pi_2}\mathcal{B}$ and there exists $\mathcal{C}$ s.t. $\mathcal{A}R_{\pi_1}\mathcal{C}$ (4)
From (2), (3 and (4) and the Induction Hypothesis we have that either
$\langle\pi_1\rangle\varphi \in \mathcal{A}$ and $\langle\pi_2\rangle\top \in \mathcal{A}$ (5) or $\langle\pi_2\rangle\varphi \in \mathcal{A}$ and $\langle\pi_1\rangle\top \in \mathcal{A}$ (6)
By (5) and (6) and axiom 4., we have that $\bigwedge\mathcal{A} \wedge \langle\pi_1 + \pi_2\rangle\varphi$ is consistent. And by maximality $\langle\pi_1 + \pi_2\rangle\varphi \in \mathcal{A}$

□

Lemma D.11 Truth Lemma: *Let $\mathcal{M} = (W, S_\pi, \mathbf{V})$ be a finite canonical model for ϕ. For all atoms $\mathcal{A}$ and all $\varphi \in C_{FL}(\phi)$, $\mathcal{M}, \mathcal{A} \models \varphi$ iff $\varphi \in \mathcal{A}$.*

Proof. : The proof is by induction on the construction of φ.
• Atomic formulas and Boolean operators: the proof is straightforward from the definition of $\mathbf{V}$.
• Modality $\langle x\rangle$, for $x \in \{\alpha, \pi_1;\pi_2, \pi_1 + \pi_2, \pi^*\}$.
• ⇒: Suppose $\mathcal{M}, \mathcal{A} \models \langle x\rangle\varphi$, then there exists $\mathcal{A}'$ such that $\mathcal{A}S_x\mathcal{A}'$ and
$\mathcal{M}, \mathcal{A}' \models A$. By the induction hypothesis we know that $\varphi \in \mathcal{A}'$, and by lemma D.10 we have $\langle x\rangle\varphi \in \mathcal{A}$.
• ⇐: Suppose $\mathcal{M}, \mathcal{A} \not\models \langle x\rangle\varphi$, by the definition of satisfaction we have
$\mathcal{M}, \mathcal{A} \models \neg\langle x\rangle\varphi$. Then for all $\mathcal{A}'$, $\mathcal{A}S_x\mathcal{A}'$ implies $\mathcal{M}, \mathcal{A}' \not\models \varphi$. By the induction hypothesis we know that $\varphi \notin \mathcal{A}'$, and by lemma D.10 we have $\langle x\rangle\varphi \notin \mathcal{A}$.

□

Theorem D.12 *(**Completeness for Finite PDL+ Models**): Propositional Dynamic Logic PDL+ is complete with respect to the class of finite PDL+ models.*

Proof. For every consistent formula φ we can build a canonical finite PDL+ model $\mathbf{M}_\varphi$. By lemma D.3, there exist an atom $\mathcal{A} \in At(\varphi)$ such that $\varphi \in \mathcal{A}$, and by the truth lemma D.11 $\mathcal{M}, \mathcal{A} \models \varphi$. Therefore, our modal system is complete with respect to the class of finite PDL+ models.

□