# Propositional Dynamic Logic for Planning

**Mario R. F. Benevides**[1,2], **Anna C. C. M. Oliveira**[2]

[1]Instituto de Computação
Universidade Federal Fluminense (UFF)
Niterói - RJ

[2]PESC/COPPE
Universidade Federal Rio de Janeiro (UFRJ)
Rio de Janeiro - RJ

mario@ic.uff.br, acoliveira@cos.ufrj.br

***Abstract.*** *This paper presents an on going work on Propositional Dynamic Logic PDL in which atomic programs are STRIPS actions. We think that this new framework is appropriate to reasoning about actions and plans when dealing with planning problem. Unlike PDL atomic programs, STRIPS actions have pre-conditions and post-conditions. We propose a novel operator of action composition that takes into account the features of STRIPS actions. We propose an axiomatization and prove its soundness. Completeness, decidability and computational complexity are left as future work.*

***Resumo.*** *Este artigo apresenta um trabalho em andamento sobre Lógica Dinâmica Proposicional PDL em que programas atômicos são ações do STRIPS. Pensamos que esse novo arcaboço é apropriado para raciocinar sobre ações e planos ao lidar com problemas de planejamento. Ao contrário dos programas atômicos PDL, as ações do STRIPS têm pré-condições e pós-condições. Propomos um novo operador de composição de ações que leva em consideração os recursos das ações do STRIPS. Propomos uma axiomatização e provamos sua correção. Completude, decidibilidade e complexidade computacional são deixadas como trabalho futuro.*

## 1. Background

This section presents a brief overview of two topics on which the later development is based on. First, we make a brief review of the syntax and semantics of [Harel et al. 2000]. Second, we present the classical planning problem. Finally, we present a brief introduction to STRIPS.

### 1.1. Propositional Dynamic Logic

In this section, we present the syntax and semantics of the most used dynamic logic called PDL for regular programs.

**Definition 1.1** *The PDL language consists of a set $\Phi$ of countably many proposition symbols, a set $\Pi$ of countably many basic programs, the boolean connectives $\neg$ and $\wedge$, the program constructors ; (sequential composition), $\cup$ (non-deterministic choice) and $^\star$ (iteration) and a modality $\langle\pi\rangle$ for every program $\pi$. The formulas are defined as follows:*

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

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

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

Each program $\pi$ corresponds to a modality $\langle\pi\rangle$, where a formula $\langle\pi\rangle\varphi$ means that there is a run of $\pi$ after which $\varphi$ is true, considering that $\pi$ halts. There is also the possibility of using $[\pi]\varphi$ (as an abbreviation for $\neg\langle\pi\rangle\neg\varphi$) indicating that the property denoted by $\varphi$ holds after every possible run of $\pi$.

The semantics of PDL is normally given using a transition diagram, which consists of a set of states and binary relations (one for each program) indicating the possible execution of each program at each state. In PDL literature a transition diagram is called a frame.

**Definition 1.2** *A* frame *for PDL is a tuple* $\mathcal{F} = \langle W, R_\pi\rangle$ *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* $R_\pi$*, for each non-basic program* $\pi$*, as follows*
  - $R_{\pi_1;\pi_2} = R_{\pi_1} \circ R_{\pi_2}$,
  - $R_{\pi_1\cup\pi_2} = R_{\pi_1} \cup R_{\pi_2}$,
  - $R_{\varphi?} = \{(w, w) \mid \mathcal{M}, w \Vdash \varphi\}$,
  - $R_{\pi^\star} = R^\star_\pi$*, where* $R^\star_\pi$ *denotes the reflexive transitive closure of* $R_\pi$.

**Definition 1.3** *A* model *for PDL is a pair* $\mathcal{M} = \langle\mathcal{F}, \mathbf{V}\rangle$*, where* $\mathcal{F}$ *is a PDL frame and* $\mathbf{V}$ *is a valuation function* $\mathbf{V} : \Phi \rightarrow 2^W$.

The semantical notion of satisfaction for PDL is defined as follows:

**Definition 1.4** *Let* $\mathcal{M} = \langle\mathcal{F}, \mathbf{V}\rangle$ *be a model. The notion of* satisfaction *of a formula* $\varphi$ *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* $w' \in W$ *such that* $wR_\pi w'$ *and* $\mathcal{M}, w' \Vdash \varphi$.

For more details on PDL see [Harel et al. 2000].

### 1.2. Planning Problem

The planning problem consists of an automated process to check if a goal is achievable, given the starting state and the defined actions. With the information acquired in the process, it's possible to construct the possibility graph [Luger 2008, Russell and Norvig 2003].

This is interesting for the field of artificial intelligence because it results in a tree of possibilities given a set of actions and an initial state, with which one can find a path to reach the intended state. This can be applied in robot and games programming, for performance analysis, and in some decision making processes.

**Definition 1.5** *The classic planning problem is a tuple* $\langle S, Ac, s_0, S_G\rangle$*, where* $S$ *is the set of all possible states,* $Ac$ *a set of actions,* $s_0 \in S$ *is the initial state, and* $S_G \subseteq S$ *is a set containing the possible goals.*

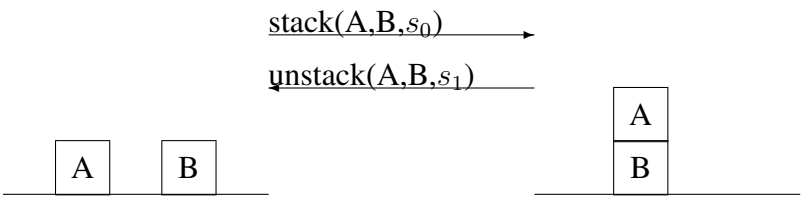
### 1.3. Example

In order to illustrate each method, the Blocks World problem is used. In this example, the world is supposed to be a table with labeled wood blocks on it. All blocks have the same size, and each one can be either on the table or on top of just one other block. Each configuration of the blocks corresponds to a different state of the Blocks World.

State $s_0$ State $s_1$


There are two actions that can be performed in this example. The first one is the action of stacking a block on top of another. And the second is the action of unstacking a block from top of another. The unstacked block is put on the table.

State $s_0$ State $s_1$




The operators descriptions are:

$stack(x, y)$:

PRECONDITION $Table(x) \wedge Clear(x) \wedge Clear(x) \wedge x \neq y$
POSCONDITION $On(x, y) \wedge \neg Clear(y) \wedge \neg Table(x)$

$unstack(x, y)$:

PRECONDITION $Clear(x) \wedge On(x, y)$
POSCONDITION $Clear(y) \wedge Table(x) \wedge \neg On(x, y)$

### 1.4. STRIPS (Stanford Research Institute Problem Solver)

The Stanford Research Institute Problem Solver is a problem solver developed by Fikes and Nilsson in 1971 [Fikes and Nilsson 1971, Fikes et al. 1998]. The language was created with the goal of implementing efficient operators. It was developed to be a planner for the first mobile robot controlled by artificial inteligence, "Shakey the robot".

In Strips, the problem space is formed by an initial state, a set of operators with their effects, and goal conditions. The search space is a set of all possible worlds that are transversed to locate a goal. It applies the operators which change the current state, until it reaches the goal conditions. An operator consists of a set of preconditions and effects, which can be in a delete list, or an add list.

The description of each operator follows the schemata below.

$< operator >$:

| | |
|---|---|
| PRECONDITION | $< formula >$ |
| ADD LIST | $< list - of - formulas >$ |
| DELETE LIST | $< list - of - formulas >$ |

A Blocks World representation, in STRIPS, is presented below.

The operators descriptions are:

$stack(x, y)$:

| | |
|---|---|
| PRECONDITION | $Table(x) \wedge Clear(x) \wedge Clear(x) \wedge x \neq y$ |
| ADD LIST | $On(x, y)$ |
| DELETE LIST | $Clear(y), Table(x)$ |

$unstack(x, y)$:

| | |
|---|---|
| PRECONDITION | $Clear(x) \wedge On(x, y)$ |
| ADD LIST | $Clear(y), Table(x)$ |
| DELETE LIST | $On(x, y)$ |

## 2. PDL with STRIPS Actions

### 2.1. Language

The language is a standard PDL-language with composition, choice, test and iteration. Let $\mathcal{A}ct = \{a, b, c, \cdots\}$ be the set of action names and $\alpha$ denote an element of $\mathcal{A}ct$.

**Definition 2.1** *An action is a triple $\langle \alpha, pre(\alpha), pos(\alpha) \rangle$, where $\alpha$ is an action name and $pre(\alpha)$ and $pos(\alpha)$ are PDL formulas.*

**Definition 2.2** *The* **dynamic modal language** *is a multi-modal language consisting of a set $\Phi$ of countably many propositional symbols (the elements of $\Phi$ are denoted by p, q, ..), the booleans connectives $\neg$ and $\wedge$ and a family of modal operators $\langle\pi\rangle$, one for each program $\pi$. Formulas and programs are defined as follows:*

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle\pi\rangle\varphi$$

$$\pi ::= \alpha \mid \pi_1;\pi_2 \mid \pi_1 + \pi_2 \mid \pi^* \mid \varphi?, \text{ where } \alpha \in \mathcal{A}.$$

**Definition 2.3** *The* **pre-condition** *of a program $\pi$, $pre(\pi)$, and the* **post-condition** *of a program $\pi$, $pos(\pi)$, can be inductively defined as follows*

- *if $\pi = \alpha$, then $pre(\pi) = pre(\alpha)$ and $pos(\pi) = pos(\alpha)$,*
- *if $\pi = \pi_1;\pi_2$, then $pre(\pi) = pre(\pi_1)$ and $pos(\pi) = pos(\pi_2)$,*
- *if $\pi = \pi_1 + \pi_2$, then $pre(\pi) = pre(\pi_1) \vee pre(\pi_2)$ and $pos(\pi) = pos(\pi_1) \vee pos(\pi_2)$,*
- *if $\pi = \varphi?$, then $pre(\pi) = pos(\pi) = \varphi$,*
- *if $\pi = \pi_1^*$, then $pre(\pi) = pre(\pi_1)$ and $pos(\pi) = pos(\pi_1)$.*

## 2.2. Semantics

The definitions of *frame* and *models* are exactly the same as Definition 1.2 and Definition 1.3 for PDL.

**Definition 2.4** *A* **proper SPDL model** *is a model $\mathcal{M} =< \mathcal{F}, \mathbf{V} >$, satisfying the following conditions:*

1. *for all actions $\alpha$ and for all $u, v \in W$: $(u, v) \in R_\alpha$ iff $\mathcal{M}, u \Vdash pre(\alpha)$ and $\mathcal{M}, v \Vdash pos(\alpha)\}$,*
2. *for all programs $\pi$: $\forall u \in W,\ if\ \mathcal{M}, u \Vdash pre(\pi),\ then\ \exists v (u, v) \in R_\pi$.*

The definition of *satisfaction and validity in proper models* remains exactly the same for regular models (definition 1.4).

**Lemma 2.1** *Let $\mathcal{M} =< \mathcal{F}, \mathbf{V} >$ be a proper SPDL model. Then,*

$$R_\pi = \{(u, v) \in W \times W \mid \mathcal{M}, u \Vdash pre(\pi) \text{ and } \mathcal{M}, v \Vdash pos(\pi)\}.$$

**Proof:** By induction on the construction of $\pi$.

- $\pi = \alpha$ (basic action): it follows straightforward from definition 2.4.
- H.I.: suppose it holds for programs $|\pi| < k$. We have to prove that it holds for programs $\pi = \pi_1;\pi_2,\ \pi_1 + \pi_2,\ \pi_1^*,\ \varphi?$ of length $|\pi| = k$.
  - $\pi = \pi_1;\pi_2$. By definition 1.2, $uR_{\pi_1;\pi_2}v$ iff $uR_{\pi_1} \circ R_{\pi_2}v$. But this implies the there exists $w$ s.t. $uR_{\pi_1}w$ and $wR_{\pi_2}v$. By the I.H. $\mathcal{M}, u \Vdash pre(\pi_1)\ and\ \mathcal{M}, v \Vdash pos(\pi_2)\}$. Using definition 2.3, $R_{\pi_1;\pi_2} = \{(u, v) \in W \mid \mathcal{M}, u \Vdash pre(\pi_1;\pi_2)\ and\ \mathcal{M}, v \Vdash pos(\pi_1;\pi_2)\}$.
  - $\pi = \pi_1 + \pi_2,\ \pi_1^*,\ \varphi?$. These cases are analogous to the previous case. They follow straightforward from definitions 2.3 and 1.2.

$\triangle$

### 2.2.1. Axiomatization

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

The axiomatization presented below is the standard one of PDL extended with the axiom for the parallel operator.

**Axioms**

1. *All tautologies*,
   $[\pi](p \rightarrow q) \rightarrow ([\pi]p \rightarrow [\pi]q)$,
2. $[\pi_1;\pi_2]p \leftrightarrow [\pi_1][\pi_2]p$,
3. $[\pi_1 + \pi_2]p \leftrightarrow [\pi_1] \wedge [\pi_2]p$,
4. $[\pi^*]p \leftrightarrow p \wedge [\pi][\pi^*]p$,
5. $[\pi^*](p \rightarrow [\pi]p) \rightarrow ([\pi]p \rightarrow [\pi^*]p)$,
6. $[p?]q \leftrightarrow p \rightarrow q$,
7. $pre(\pi) \leftrightarrow \langle\pi\rangle pos(\pi)$
8. $pre(\pi) \rightarrow \langle\pi\rangle\top$

**Inference Rules**

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

where $\sigma$ is a map uniformly substituting formulas for propositional variables.

Axioms 1, 2, 3, 4, 5, and 6 and the inference rules are standard in PDL for regular programs [Harel et al. 2000, Goldblatt 1992, Blackburn et al. 2001]. Axiom 7 says that, if the precondition of a program is satisfied, then there exists a state reached after the execution of the program where the post condition must hold. Axiom 8 asserts that if the pre-condition of a program holds, then it must be the case that the program can be executed.

### 2.2.2. Soundness

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, 3, 4, 5 and 6 and the inference rules are well-known from the PDL literature [Harel et al. 2000, Goldblatt 1992, Blackburn et al. 2001]. Due to the pre-conditions and post-conditions of programs, axiom 2 must be revisited. Thus, we only prove the validity of axioms 2, 7 and 8.

**Lemma 2.2** *Axiom 2 is valid, i.e.,* $\Vdash [\pi_1;\pi_2]p \leftrightarrow [\pi_1][\pi_2]p$.

**Proof:** *For the sake of clarity, we prove the validity of the dual of axiom 2. The soundness of axiom 2 follows straightforward.*

$\Rightarrow$ *Suppose that, for some model* $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ *and some state* $u$ *in this model,* $\mathcal{M}, u \Vdash \langle\pi_1;\pi_2\rangle p$ *(1),*
*(1) iff* $\exists v$, $uR_{\pi_1;\pi_2}v$ *and* $\mathcal{M}, v \Vdash p$,
$uR_{\pi_1;\pi_2}v$ *iff* $\exists w,\ uR_{\pi_1}w$ *and* $wR_{\pi_2}v$ *(2).*
*By (1) and (2), we have* $\mathcal{M}, w \Vdash \langle\pi_2\rangle p$. *And using (2) again we obtain* $\mathcal{M}, u \Vdash \langle\pi_1\rangle\langle\pi_2\rangle p$

$\Leftarrow$ *Suppose that, for some model* $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ *and some state* $u$ *in this model,* $\mathcal{M}, u \Vdash \langle \pi_1 \rangle \langle \pi_2 \rangle p$ *(1),*
*(1) iff* $\exists w,\ uR_{\pi_1}w$ *and* $\mathcal{M}, w \Vdash \langle \pi_2 \rangle p$ *(2),*
*(2) iff* $\exists v,\ wR_{\pi_2}v$ *and* $\mathcal{M}, v \Vdash p$ *(3),*
*From (2) and (3),* $\exists w,\ uR_{\pi_1}w$ *and* $wR_{\pi_2}v$ *and* $\mathcal{M}, v \Vdash p$*,*
*iff* $\exists w,\ uR_{\pi_1}w \circ wR_{\pi_2}v$ *and* $\mathcal{M}, v \Vdash p$*,*
*iff* $\exists w,\ uR_{\pi_1;\pi_2}v$ *and* $\mathcal{M}, v \Vdash p$*,*
*Thus,* $\mathcal{M}, u \Vdash \langle \pi_1; \pi_2 \rangle p$*.*

$\triangle$

**Lemma 2.3** *Axiom 7 is valid, i.e.,* $\Vdash pre(\pi) \leftrightarrow \langle \pi \rangle pos(\pi)$.

**Proof:**

$\Rightarrow$ Suppose that, for some model $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ and some state $u$ in this model, $\mathcal{M}, u \Vdash pre(\pi)$ (1) and $\mathcal{M}, u \nVdash \langle \pi \rangle pos(\pi)$ (2)
(2) iff $\forall v$, $uR_\pi v \ \Rightarrow\ \mathcal{M}, v \Vdash \neg pos(\pi)$. (3)
Using (1) and definition 2.4, of proper models, $\exists v,\ uR_\pi v$. From (3), we have $\mathcal{M}, v \Vdash \neg pos(\pi)$.
But this is a contradiction with lemma 2.1.

$\Leftarrow$ Suppose that, for some model $\mathcal{M} = (\mathcal{F}, \mathbf{V})$ and some state $u$ in this model, $\mathcal{M}, u \nVdash pre(\pi)$ (1) and $\mathcal{M}, u \Vdash \langle \pi \rangle pos(\pi)$ (2)
(2) iff $\exists v$, $uR_\pi v$ and $\mathcal{M}, v \Vdash \neg pos(\pi)$,
But this is a contradiction with lemma 2.1.

$\triangle$

**Lemma 2.4** *Axiom 8 is valid, i.e.,* $pre(\pi) \rightarrow \langle \pi \rangle \top$.

**Proof:** It follows straightforward from definition 2.4.

$\triangle$

**Theorem 2.1** *(***Soundness***): STRIPS-PDL is sound with respect to the class of SPDL proper models.*

**Proof:** *The proof of soundness is analogous to the proof of soundness for dynamic logic, it is not difficult to see that every SPDL proper model is a model for the axioms and the inference rules preserve validity.*

$\triangle$

## 3. Conclusions

In this work we introduce a new dynamic logic called STRIPS- PDL. Its aim is to reason about actions and planning in artificial intelligence scenario. We propose an axiomatic system and prove its soundness.

As a future work, we would like to prove completeness and decidability of STRIPS-PDL and also establish its computational complexity. We also would like to investigate new extension of this logic with other operator like: parallel composition and epsitemic actions. Finally, it would interesting to develop a planner based on our framework.

## Referências


Blackburn, P., de Rijke, M., and Venema, Y. (2001). *Modal Logic*. Cambridge University Press, UK.

Fikes, R. E., Nillson, N. J., and Cocosco, C. A. (1998). A review of "strips: A new approach to the application of theorem proving to problem solving by r.e. fikes, n.j. nillson, 1971".

Fikes, R. E. and Nilsson, N. J. (1971). Strips: A new approach to the application of theorem proving to problem solving. In *Proceedings of the 2Nd International Joint Conference on Artificial Intelligence*, IJCAI'71, pages 608–620, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Goldblatt, R. (1992). *Logics of Time and Computation*. CSLI Lecture Notes 7. CSLI, Stanford.

Harel, D., Kozen, D., and Tiuryn, J. (2000). *Dynamic Logics*. MIT Press.

Luger, G. F. (2008). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Addison-Wesley Publishing Company, USA, 6th edition.

Russell, S. J. and Norvig, P. (2003). *Artificial intelligence - a modern approach, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall.