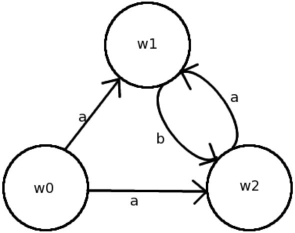
Propositional Dynamic Logic in ACL2

Karl Pichotta
CS 389R
Autumn 2010
UT Austin

November 29, 2010

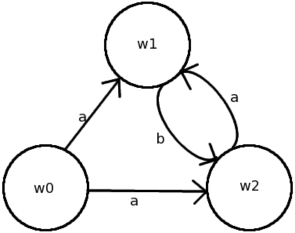
PDL: Frames

Frame: directed graph with labeled edges.



PDL: Frames

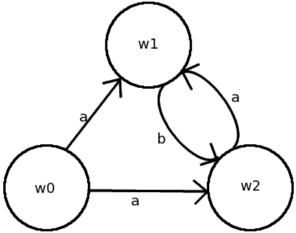
Frame: directed graph with labeled edges.



- Vertices are called **worlds**.

PDL: Frames

Frame: directed graph with labeled edges.



- Vertices are called **worlds**.
- If two worlds are connected, we write $w_0 R_a w_1$.

PDL: Programs

Programs are, semantically, relations constructed from frames, defined recursively:

$$\pi ::= \pi_a \mid \pi_1 \cup \pi_2 \mid \pi_1 ; \pi_2 \mid {\pi_1}^*$$

PDL: Programs

Programs are, semantically, relations constructed from frames, defined recursively:

$$\pi ::= \pi_a \mid \pi_1 \cup \pi_2 \mid \pi_1 ; \pi_2 \mid {\pi_1}^*$$

- **Atomic program** π_a: e.g. all edges labeled with "a".

PDL: Programs

Programs are, semantically, relations constructed from frames, defined recursively:

$$\pi ::= \pi_a \mid \pi_1 \cup \pi_2 \mid \pi_1 ; \pi_2 \mid {\pi_1}^*$$

- **Atomic program** π_a: e.g. all edges labeled with "*a*".
- **Choice** $\pi_1 \cup \pi_2$: union of sets of π_1 edges and π_2 edges.

PDL: Programs

Programs are, semantically, relations constructed from frames, defined recursively:

$$\pi ::= \pi_a \mid \pi_1 \cup \pi_2 \mid \pi_1 ; \pi_2 \mid {\pi_1}^*$$

- **Atomic program** π_a: e.g. all edges labeled with "a".
- **Choice** $\pi_1 \cup \pi_2$: union of sets of π_1 edges and π_2 edges.
- **Composition** $\pi_1 ; \pi_2$: $uR_{\pi_1;\pi_2}v$ iff $\exists w.\, uR_{\pi_1}w \wedge wR_{\pi_2}v$.

PDL: Programs

Programs are, semantically, relations constructed from frames, defined recursively:

$$\pi ::= \pi_a \mid \pi_1 \cup \pi_2 \mid \pi_1 ; \pi_2 \mid {\pi_1}^*$$

- **Atomic program** π_a: e.g. all edges labeled with "a".
- **Choice** $\pi_1 \cup \pi_2$: union of sets of π_1 edges and π_2 edges.
- **Composition** $\pi_1 ; \pi_2$: $uR_{\pi_1;\pi_2}v$ iff $\exists w.\, uR_{\pi_1}w \wedge wR_{\pi_2}v$.
- **Iteration** ${\pi_1}^*$: reflexive transitive closure of π_1.

PDL: Syntax

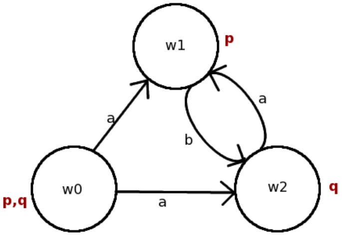
Modal logic with an infinite number of modalities:

$$\varphi ::= p \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid [\pi]\varphi_1$$

- p ranges over atomic proposition variables.
- π must be a valid PDL program.

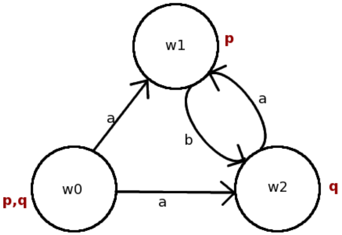
PDL: Semantics

A **model** is a frame with proposition letters at each world:



PDL: Semantics

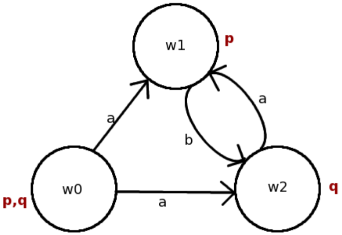
A **model** is a frame with proposition letters at each world:



- Formulas are evaluated at a world in a model: $\mathcal{M}, w \models \varphi$.

PDL: Semantics

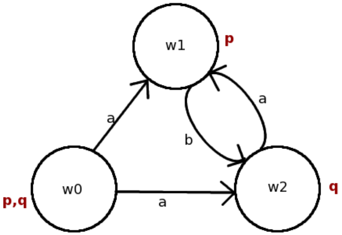
A **model** is a frame with proposition letters at each world:



- Formulas are evaluated at a world in a model: $\mathcal{M}, w \models \varphi$.
- **Propositional formulas** are easy. ($\mathcal{M}, w_0 \models p \wedge q$, $\mathcal{M}, w_1 \models \neg q$).

PDL: Semantics

A **model** is a frame with proposition letters at each world:



- Formulas are evaluated at a world in a model: $\mathcal{M}, w \models \varphi$.
- **Propositional formulas** are easy. ($\mathcal{M}, w_0 \models p \wedge q$, $\mathcal{M}, w_1 \models \neg q$).
- **Modal formulas**: $\mathcal{M}, w \models [\pi]\varphi$ iff φ is true in every world reachable with the relation for π from world w. ($\mathcal{M}, w_1 \models [\pi_b]q$, $\mathcal{M}, w_0 \models [\pi_a]p \vee q$, $\mathcal{M}, w_0 \not\models [\pi_a]p$).

Encoding in ACL2 (1/2)

Formulas:

p : `p`

Encoding in ACL2 (1/2)

Formulas:

p	:	`p`
$p \vee \neg q$	:	`'(v p (~ q))`

Encoding in ACL2 (1/2)

Formulas:

p	:	`p`
$p \vee \neg q$	:	`'(v p (~ q))`
$[\pi_a]p$	:	`'(box a p)`

Encoding in ACL2 (1/2)

Formulas:

p	:	`p`
$p \vee \neg q$	:	`'(v p (~ q))`
$[\pi_a]p$	:	`'(box a p)`
$[\pi_a \cup \pi_b]p \vee q$	:	`'(box (union a b) (v p q))`

Encoding in ACL2 (1/2)

Formulas:

p	:	`p`
$p \vee \neg q$	:	`'(v p (~ q))`
$[\pi_a]p$	:	`'(box a p)`
$[\pi_a \cup \pi_b]p \vee q$	:	`'(box (union a b) (v p q))`

Predicates: `framep`, `modelp`, `pdl-programp`, `pdl-formulap`.

Encoding in ACL2 (2/2)

Satisfiability: `(pdl-satisfies m w f)` takes model, world index, formula, returns `t` or `nil`.

```
(encapsulate
 ()
 (set-well-founded-relation l<)
 (mutual-recursion
  (defun pdl-satisfies-mutual (m w f worlds)
    (declare (xargs :measure (list (acl2-count f) (acl2-count worlds))))
    (cond ((symbolp f)
           (pdl-satisfies-symbol m w f))
          ((equal (len f) 2)
           (not (pdl-satisfies-mutual m w (second f) worlds)))
          ((equal (len f) 3)
           (cond ((equal (first f) 'v)
                  (or (pdl-satisfies-mutual m w (second f) worlds)
                      (pdl-satisfies-mutual m w (third f) worlds)))
                 ((equal (first f) 'box)
                  (pdl-satisfies-box-mutual
                   m
                   (prog-accessible-worlds m w (second f))
                   (third f)))))
          (t nil)))
  (defun pdl-satisfies-box-mutual (m p-accessible-worlds f)
    (declare (xargs :measure (list (acl2-count f)
                                   (acl2-count p-accessible-worlds))))
    (if (consp p-accessible-worlds)
        (and (pdl-satisfies-mutual m (car p-accessible-worlds) f nil)
             (pdl-satisfies-box-mutual m (cdr p-accessible-worlds) f))
      t))))
```

Theorems proved (propositional semantics - 1/2)

Negation:

$$\mathcal{M}, w \models \neg\varphi \quad \textit{iff} \quad \mathcal{M}, w \not\models \varphi$$

```
(iff (pdl-satisfies m w '(~ f))
     (not (pdl-satisfies m w f)))
```

Theorems proved (propositional semantics - 1/2)

Negation:

$$\mathcal{M}, w \models \neg\varphi \quad \textit{iff} \quad \mathcal{M}, w \not\models \varphi$$

```
(iff (pdl-satisfies m w '(~ f))
     (not (pdl-satisfies m w f)))
```

```
(defthm negation-semantics-correct
  (implies (and (pdl-formulap f
                              (get-prop-atoms m)
                              (get-prog-atoms m))
                (equal (first f) '~))
           (equal (pdl-satisfies m w (second f))
                  (not (pdl-satisfies m w f)))))
```

Theorems proved (propositional semantics - 2/2)

Disjunction:

$$\mathcal{M}, w \models \varphi \vee \psi \quad \textit{iff} \quad \mathcal{M}, w \models \varphi \textit{ or } \mathcal{M}, w \models \psi$$

```
(defthm disjunction-semantics-correct
  (implies (and (equal (len f) 3)
                (equal (first f) 'v))
           (equal (pdl-satisfies m w f)
                  (or (pdl-satisfies m w (second f))
                      (pdl-satisfies m w (third f))))))
```

Theorems proved (Program correctness)

- **Length**: $\pi_1 \cup \pi_2$, $\pi_1 ; \pi_2$, and ${\pi_1}^*$ are all the correct length

Theorems proved (Program correctness)

- **Length**: $\pi_1 \cup \pi_2$, $\pi_1 ; \pi_2$, and ${\pi_1}^*$ are all the correct length
- **Choice correctness**: $uR_{\pi_1 \cup \pi_2} v$ iff $uR_{\pi_1} v$ or $uR_{\pi_2} v$.

Theorems proved (Program correctness)

- **Length**: $\pi_1 \cup \pi_2$, $\pi_1 ; \pi_2$, and ${\pi_1}^*$ are all the correct length
- **Choice correctness**: $uR_{\pi_1 \cup \pi_2} v$ iff $uR_{\pi_1} v$ or $uR_{\pi_2} v$.
- **Half of composition correctness**: If $uR_{\pi_1} v$ and $vR_{\pi_2} w$, then $uR_{\pi_1 ; \pi_2} w$.

Theorems proved (Program satisfiability)

- **Half of box correctness**:

$$\mathcal{M}, w \models [\pi]\varphi \Rightarrow \left(wR_\pi v \ \Rightarrow \ \mathcal{M}, v \models \varphi \right)$$

Theorems proved (Program satisfiability)

- **Half of box correctness**:

$$\mathcal{M}, w \models [\pi]\varphi \Rightarrow \left(wR_\pi v \;\Rightarrow\; \mathcal{M}, v \models \varphi \right)$$

- **Half of choice correctness**:

$$\mathcal{M}, w \models [\pi_1 \cup \pi_2]\varphi \Rightarrow \left(\left(wR_{\pi_1} v \vee wR_{\pi_2} v \right) \Rightarrow \mathcal{M}, v \models \varphi \right)$$

Theorems proved (Program satisfiability)

- **Half of box correctness**:

$$\mathcal{M}, w \models [\pi]\varphi \Rightarrow \left(wR_{\pi}v \;\Rightarrow\; \mathcal{M}, v \models \varphi \right)$$

- **Half of choice correctness**:

$$\mathcal{M}, w \models [\pi_1 \cup \pi_2]\varphi \Rightarrow \left(\left(wR_{\pi_1}v \vee wR_{\pi_2}v \right) \Rightarrow \mathcal{M}, v \models \varphi \right)$$

- **Half of composition correctness**:

$$\mathcal{M}, w \models [\pi_1 ; \pi_2]\varphi \Rightarrow \left(\left(wR_{\pi_1}u \wedge uR_{\pi_2}v \right) \Rightarrow \mathcal{M}, v \models \varphi \right)$$

Theorems not proved

- Half of box satisfiability correctness
- Half of choice, composition satisfiability correctness
- Iteration correctness.
- A slew of other interesting statements!