A brief overview in Dynamic Logics

Alexandre Madeira
Mathematics Dep, U. Aveiro

February 10, 2026,
Software Foundations, MAP-i 25/26
DMat, U.Aveiro

Outline

Outline

About the current computational systems

Andre Platzer

... there is probably no other area where the gap is more noticeable between the tremendous complexity of the systems we can built and the modest size of systems that we can analyse

Logical Analysis of Hybrid Systems, 2012

looking at the current practices

The quality and dependability of

most of the systems is assured by **test-oriented** development methodologies

looking at the current practices

The quality and dependability of

most of the systems is assured by **test-oriented** development methodologies

Edsger Dijkstra (Turing Award 1972)

program testing can be used to show the presence of bugs, but never to show their absence!

(1969)

looking to classic engineering ...

e.g. in Mechanics, mathematics is used

- as an **unambiguous language** to express requirements
- to support **rigorous modelling**
- to support the **validation and verification** tasks

... following the basic principles of engineering

Formal methods

mathematical and formal logic based techniques for the **specification**, **development** and **validation** of *computational* systems

Formal Logics in Computer Science

particularly, **Formal Logics** are valuable mathematical tools

For

- modelling
- reasoning on
- verify

Complex systems

Just an illustration

Equational Calculus in a rush

blackboard ...

Verification of classical imperative programs

Robert W. Floyd

ASSIGNING MEANINGS TO PROGRAMS[1]

Introduction. This paper attempts to provide an adequate basis for formal definitions of the meanings of programs in appropriately defined programming languages, in such a way that a rigorous standard is established for proofs about computer programs, including proofs of correctness, equivalence, and termination. The basis of our approach is the notion of an interpretation of a program: that is, an association of a proposition

An Axiomatic Basis for Computer Programming

C. A. R. Hoare
The Queen's University of Belfast, Northern Ireland*

Verification of classical imperative programs

Floyd-Hoare Logic

$$\{\phi\}\ \textbf{Prog}\ \{\varphi\}$$

Verification of classical imperative programs

Floyd-Hoare Logic

$$\{\phi\}\ \textbf{Prog}\ \{\varphi\}$$

Axioms:

$$\frac{}{\{\varphi[e/x]\}\, x := e\, \{\varphi\}} \qquad \frac{}{\{\phi\}\, \mathit{skip}\, \{\phi\}}$$

Verification of classical imperative programs

Floyd-Hoare Logic

$$\{\phi\}\ \textbf{Prog}\ \{\varphi\}$$

Axioms:

$$\frac{}{\{\varphi[e/x]\}\, x := e\, \{\varphi\}} \qquad \frac{}{\{\phi\}\, \mathit{skip}\, \{\phi\}}$$

Inference rules:

$$\frac{\phi \to \phi' \quad \{\phi'\}S\{\varphi'\} \quad \varphi' \to \varphi}{\{\phi\}S\{\varphi\}} \qquad \frac{\{\phi\}S\{\xi\} \quad \{\xi\}T\{\varphi\}}{\{\phi\}S;T\{\varphi\}}$$

$$\frac{\{\phi \wedge \alpha\}S_1\{\varphi\} \quad \{\phi \wedge \neg\alpha\}S_2\{\varphi\}}{\{\phi\}\textbf{if } \alpha \textbf{ then } S_1 \textbf{ else } S_2\{\varphi\}}$$

...

Verification of classical imperative programs

$$\{x = 1\}y := x + 1; z := y\{z = 2\}$$

Verification of classical imperative programs

$\{x = 1\}y := x + 1; z := y\{z = 2\}$

$$\frac{\overline{\{x=1\}y:=x+1\{y=2\}} \qquad \overline{\{y=2\}z:=y\{z=2\}}}{\{x = 1\}y := x + 1; z := y\{z = 2\}}$$

since $(y = 2)[x + 1/y] \Leftrightarrow x = 1$ and $(z = 2)[y/z] \Leftrightarrow y = 2$

Verification of classical imperative programs

$$\{x = 1\}y := x + 1; z := y\{z = 2\}$$

$$\frac{\overline{\{x=1\}y:=x+1\{y=2\}} \qquad \overline{\{y=2\}z:=y\{z=2\}}}{\{x = 1\}y := x + 1; z := y\{z = 2\}}$$

since $(y = 2)[x + 1/y] \Leftrightarrow x = 1$ and $(z = 2)[y/z] \Leftrightarrow y = 2$

$$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$$

Verification of classical imperative programs

$$\{x = 1\}y := x + 1; z := y\{z = 2\}$$

$$\frac{\overline{\{x{=}1\}y{:=}x{+}1\{y{=}2\}} \qquad \overline{\{y{=}2\}z{:=}y\{z{=}2\}}}{\{x = 1\}y := x + 1; z := y\{z = 2\}}$$

since $(y = 2)[x + 1/y] \Leftrightarrow x = 1$ and $(z = 2)[y/z] \Leftrightarrow y = 2$

$$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$$

$$\frac{\dfrac{\overline{\{x{=}1\}x{:=}x{+}1\{x{=}2\}}}{\{x{=}1 \wedge x{<}2\}x{:=}x{+}1\{x{=}2\}} \qquad \overline{\{x{=}1 \wedge x{\geq}2\}x{:=}x{*}x\{y{=}2\}}}{\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}}$$

Verification of classical imperative programs

$$\{x = 1\}y := x + 1; z := y\{z = 2\}$$

$$\frac{\overline{\{x{=}1\}y{:=}x{+}1\{y{=}2\}} \qquad \overline{\{y{=}2\}z{:=}y\{z{=}2\}}}{\{x = 1\}y := x + 1; z := y\{z = 2\}}$$

since $(y = 2)[x + 1/y] \Leftrightarrow x = 1$ and $(z = 2)[y/z] \Leftrightarrow y = 2$

$$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$$

$$\frac{\dfrac{\overline{\{x{=}1\}x{:=}x{+}1\{x{=}2\}}}{\{x{=}1 \wedge x{<}2\}x{:=}x{+}1\{x{=}2\}} \qquad \overline{\{x{=}1 \wedge x{\geq}2\}x{:=}x{*}x\{y{=}2\}}}{\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}}$$

since
$(x = 2)[x + 1/x] \Leftrightarrow x = 1$ and $x = 1 \wedge x < 2 \Leftrightarrow \mathit{false}$

Verification of classical imperative programs

$$\{x = 1\}y := x + 1; z := y\{z = 2\}$$

$$\frac{\overline{\{x=1\}y:=x+1\{y=2\}} \qquad \overline{\{y=2\}z:=y\{z=2\}}}{\{x = 1\}y := x + 1; z := y\{z = 2\}}$$

since $(y = 2)[x + 1/y] \Leftrightarrow x = 1$ and $(z = 2)[y/z] \Leftrightarrow y = 2$

$$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$$

$$\frac{\dfrac{\overline{\{x=1\}x:=x+1\{x=2\}}}{\{x=1\wedge x<2\}x:=x+1\{x=2\}} \qquad \overline{\{x=1\wedge x\geq 2\}x:=x*x\{y=2\}}}{\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}}$$

since
$(x = 2)[x + 1/x] \Leftrightarrow x = 1$ and $x = 1 \wedge x < 2 \Leftrightarrow \mathit{false}$

Exercise:
$\{y > 4 \wedge z > -1\}y := y + z\{y > 3\}$
$\{y > 4\}\textbf{if } (z > -1) \textbf{ then } y := y + z \textbf{ else } y := y - 1\{y > 3\}$

Verification of classical imperative programs

How to take advantage of formal logic methods and results?

- **Modal logic**, as logic of change, is the natural candidate to reason about programs

Verification of classical imperative programs

How to take advantage of formal logic methods and results?

- **Modal logic**, as logic of change, is the natural candidate to reason about programs

A modal logic to verify programs?

Vaughan Pratt, 76

SEMANTICAL CONSIDERATIONS ON FLOYD-HOARE LOGIC

Vaughan R. Pratt
Massachusetts Institute of Technology
Cambridge, MA 02139
August 1976

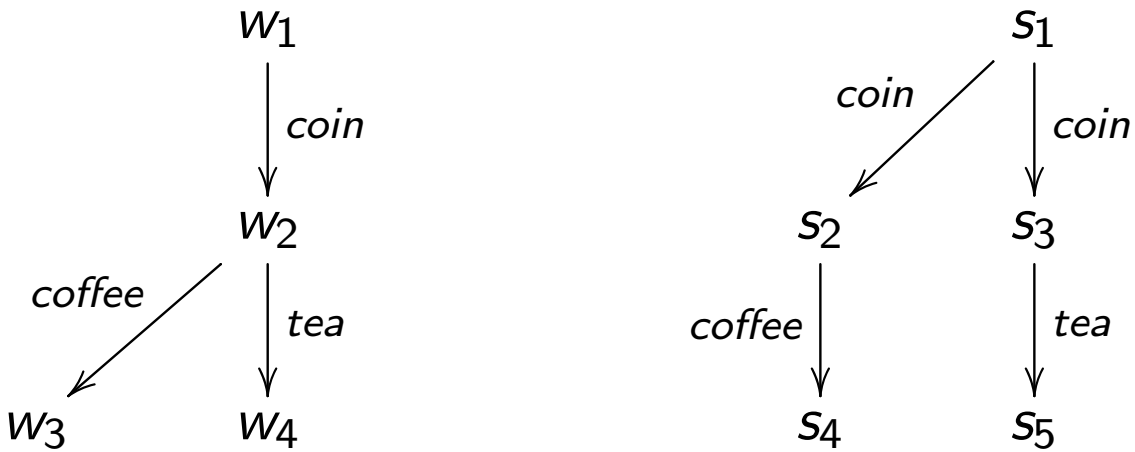
ABSTRACT

This paper deals with logics of programs. The objective is to formalize a notion of program description, and to give both plausible (semantic) and effective (syntactic) criteria for the

Outline

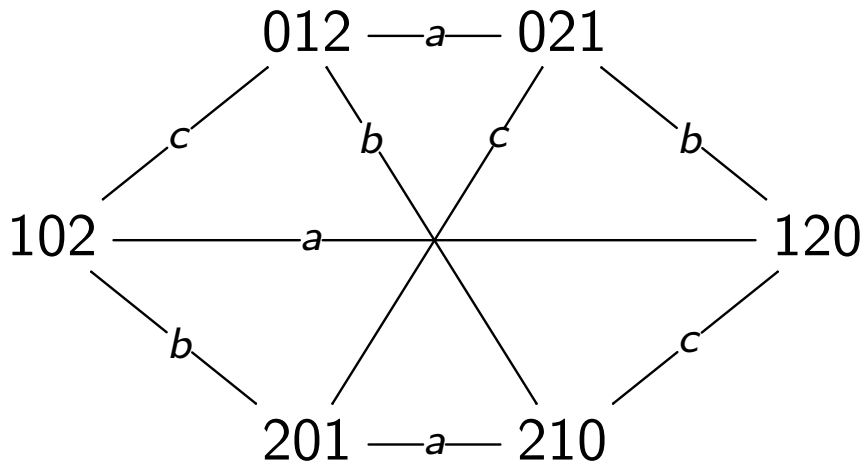
Processes are transition systems

Two coffee machines



Knowledge systems with multi-agents are transition systems

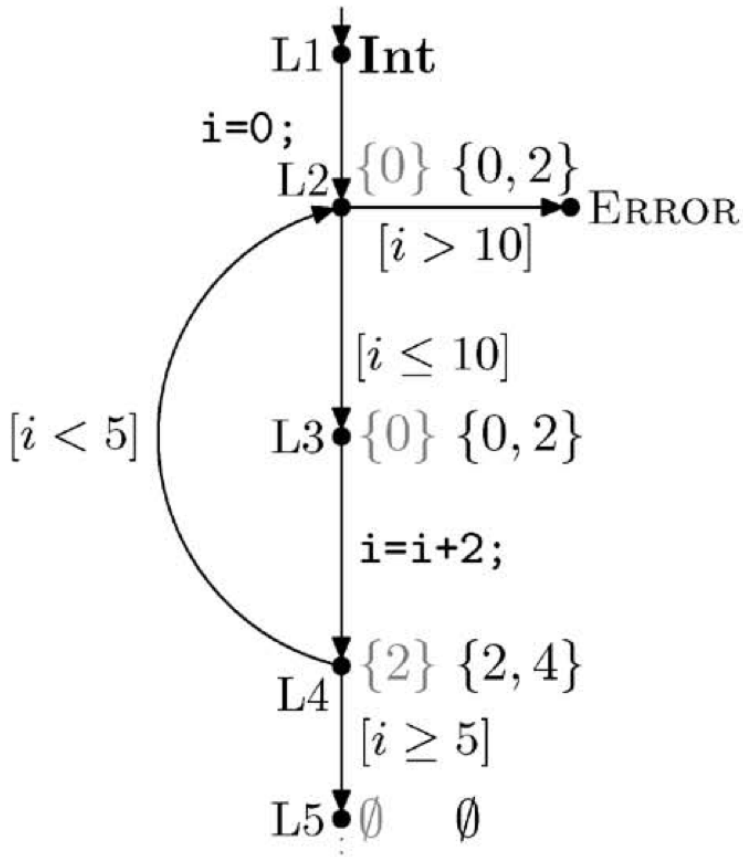
The envelop game



E.g. at state 012: Ana has an envelop with 0, Bob one with 1 and Clara with 2

Programs are transition systems

```
int i = 0;
do {
   assert(i <= 10);
   i = i+2;
} while (i < 5);
```



The language

Signatures

Signatures are pairs $(\mathrm{Prop}, \mathrm{Act})$ where Prop and Act are (disjoint) sets of symbols of **proposition** and of **actions**

The language

Signatures

Signatures are pairs $(\mathrm{Prop}, \mathrm{Act})$ where Prop and Act are (disjoint) sets of symbols of **proposition** and of **actions**

Formulas

Let $(\mathrm{Prop}, \mathrm{Act})$ a signature. The set of formulas for $(\mathrm{Prop}, \mathrm{Act})$, in symbols $\mathrm{Fm}(\mathrm{Prop})$, is defined as follows:

$$\varphi \ ::= \ p \mid \bot \mid \varphi \to \varphi \mid [a]\varphi$$

where $p \in \mathrm{Prop}$ and $a \in \mathrm{Act}$

The language

Signatures

Signatures are pairs $(\mathrm{Prop}, \mathrm{Act})$ where Prop and Act are (disjoint) sets of symbols of **proposition** and of **actions**

Formulas

Let $(\mathrm{Prop}, \mathrm{Act})$ a signature. The set of formulas for $(\mathrm{Prop}, \mathrm{Act})$, in symbols $\mathrm{Fm}(\mathrm{Prop})$, is defined as follows:

$$\varphi \ ::= \ p \mid \bot \mid \varphi \to \varphi \mid [a]\varphi$$

where $p \in \mathrm{Prop}$ and $a \in \mathrm{Act}$

Abbreviates

- $\neg\varphi := \varphi \to \bot$
- $\varphi_1 \vee \varphi_2 := \neg\varphi_1 \to \varphi_2$
- $\varphi_1 \wedge \varphi_2 := \neg(\neg\varphi_1 \to \neg\varphi_2)$
- $\langle a \rangle \phi := \neg[a]\neg\phi$
- $\ldots$

Models

Models

A **model** for the signature $(\mathrm{Prop}, \mathrm{Act})$ is a pair $M = \langle F, V \rangle$, where

- $F = \langle W, (R_a)_{a \in \mathrm{Act}} \rangle$ is a **Kripke structure**, i.e.
  - W is a non empty set (of **states**)
  - $(R_a)_{a \in \mathrm{Act}}$ is a family of binary relations $R_a \subseteq W \times W$, for each modality symbol $a \in \mathrm{Act}$.
- $V : \mathrm{Prop} \to \mathcal{P}(W)$ is a **valuation**.

Multimodal satisfaction relation

Satisfaction for a model M in a state w

- $M, w \models p$ if $w \in V(p)$
- $M, w \models \bot$ is always false
- $M, w \models \varphi_1 \to \varphi_2$ if it is false that $M, w \models \varphi_1$ or $M, w \models \varphi_2$
- $M, w \models [a]\varphi$ if for any $v \in W$, if $(w, v) \in R_a$ then $M, v \models \varphi$

Multimodal satisfaction relation

Satisfaction for a model M in a state w

- $M, w \models p$ if $w \in V(p)$
- $M, w \models \bot$ is always false
- $M, w \models \varphi_1 \to \varphi_2$ if it is false that $M, w \models \varphi_1$ or $M, w \models \varphi_2$
- $M, w \models [a]\varphi$ if for any $v \in W$, if $(w, v) \in R_a$ then $M, v \models \varphi$

Corollary

- $M, w \models \neg\varphi$ if is false that $M, w \models \varphi$
- $M, w \models \varphi_1 \wedge \varphi_2$ if $M, w \models \varphi_1$ and $M, w \models \varphi_2$
- $M, w \models \varphi_1 \vee \varphi_2$ if $M, w \models \varphi_1$ or $M, w \models \varphi_2$
- $M, w \models \langle a \rangle \varphi$ if there exists a $v \in W$ such that $(w, v) \in R_a$ and $M, v \models \varphi$

Multimodal satisfaction

Satisfaction

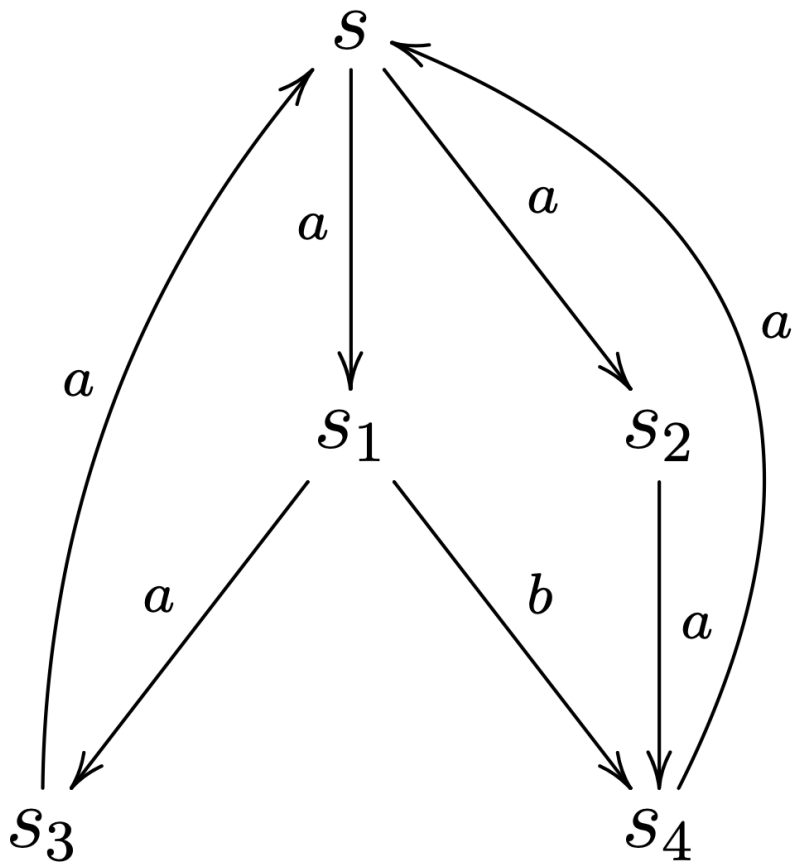
A formula $\phi \in \mathrm{Fm}(\mathrm{Prop})$ is

- **satisfiable in** M if it is satisfied in some state w of M
- **globally satisfied** in M ($M \models \phi$) if it is satisfied in all teh states of M
- **valid** ($\models \phi$) if it is globally satisfied in all the models for $(\mathrm{Prop}, \mathrm{Act})$
- **is a semantical consequence** of a set of formulas Γ ($\Gamma \models \phi$) i for any model M, and for every state w of M, if $M, w \models \Gamma$ then $M, w \models \phi$

Exercise

Verify that :

1. $M, s \models \langle a \rangle \top$
2. $M, s \models [a] \bot$
3. $M, s \models \langle b \rangle \top$
4. $M, s \models [b] \bot$
5. $M, s \models [a] \langle b \rangle \top$
6. $M, s \models \langle a \rangle \langle b \rangle \bot$
7. $M, s \models [a] \langle a \rangle [a] [b] \bot$
8. $M, s \models \langle a \rangle (\langle a \rangle \top \wedge \langle b \rangle \top)$
9. $M, s \models [a] (\langle a \rangle \top \vee \langle b \rangle \top)$
10. $M, s \models \langle a \rangle ([b][a] \bot \wedge \langle b \rangle \top)$



Exercício

Find a model M for $(\{\}, \{a, b, c\})$ with a state w such that that simultaneously verifies:

- $M, w \models \langle a \rangle (\langle b \rangle \langle c \rangle \top \wedge \langle c \rangle \top)$
- $M, w \models \langle a \rangle \langle b \rangle ([a] \bot \wedge [b] \bot \wedge [c] \bot)$
- $M, w \models [a] \langle b \rangle ([c] \bot \wedge \langle a \rangle \top)$

Outline

Programs as modalities

Programs:

Assuming a set of atomic programs Π:

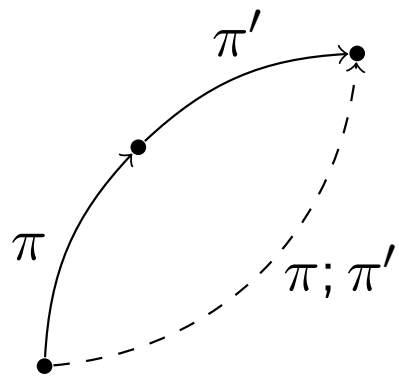
$$\pi ::= \pi_0 \mid \pi + \pi \mid \pi; \pi \mid \pi^* \mid ?\chi$$

for $\pi_0 \in \Pi$ and χ a predicate

Programs interpretation

Sequential program

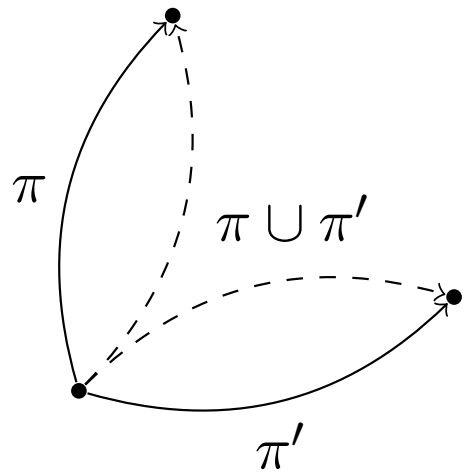
- $Pr_{\pi;\pi'} = Pr_{\pi} \circ Pr_{\pi'}$



Programs interpretation

Non deterministic choice

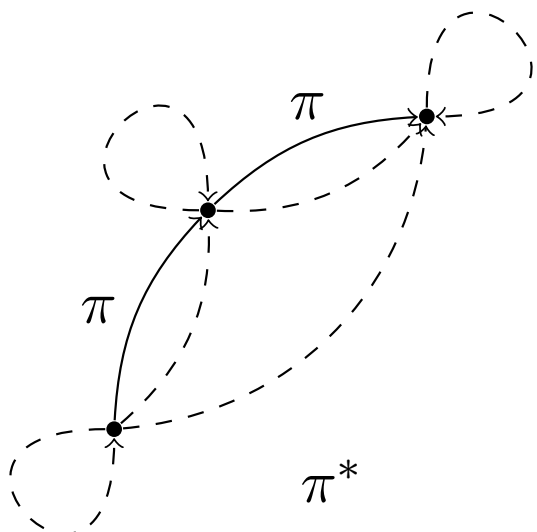
- $Pr_{\pi+\pi'} = Pr_{\pi} \cup Pr_{\pi'}$



Programs interpretation

Iterative closure

- $Pr_{\pi^*} = (Pr_\pi)^*$, for $(Pr_\pi)^* = \bigcup_{n \leq 0} (Pr_\pi)^n$, where
  - $(w, w') \in (Pr_\pi)^0$ if $w = w'$
  - $(w, w') \in (Pr_\pi)^{n+1}$ if $(w, w') \in (Pr_\pi) \circ (Pr_\pi)^n$



Programs interpretation

Tests

- $Pr_{\varphi?} = \{(w, w) \mid M, w \models \varphi\}$

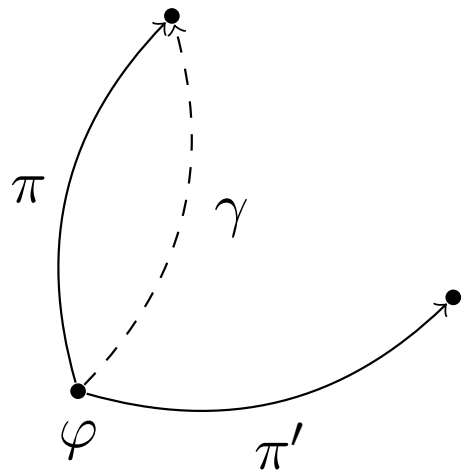
Exercise:

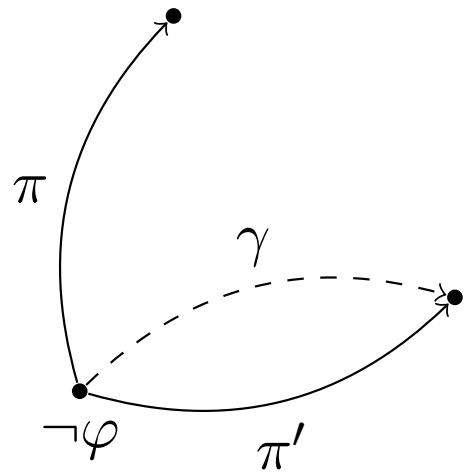
Express the following standard commands as terms of out program algebra:

- **if** φ **then** π **else** π'
- **while** φ **do** π **od**
- **repeat** π **until** φ

Programs interpretation

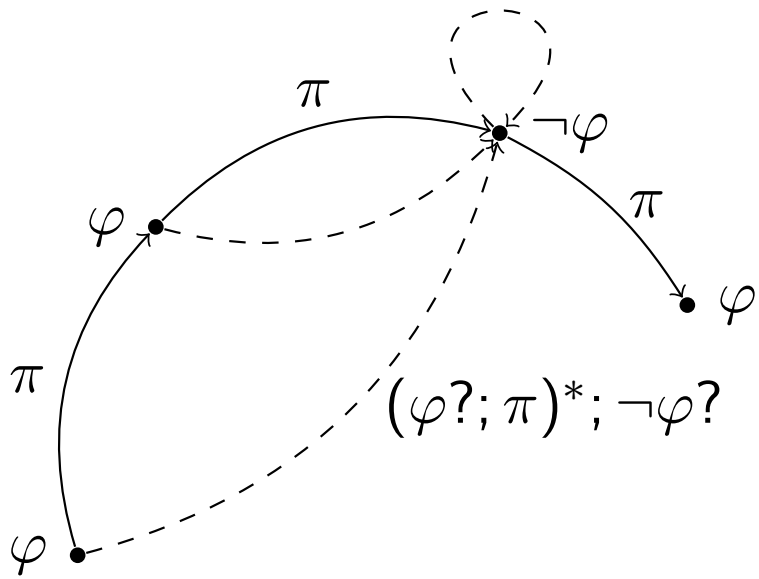
if φ **then** π **else** π' $\equiv (\varphi?;\pi) + (\neg\varphi?;\pi')$





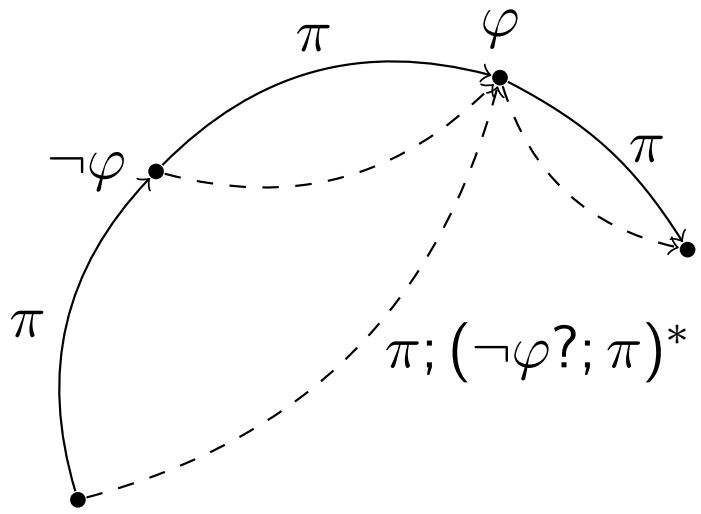
Programs interpretation

while φ **do** π **od** $\equiv$ $(\varphi?;\pi)^*;\neg\varphi?$



Programs interpretation

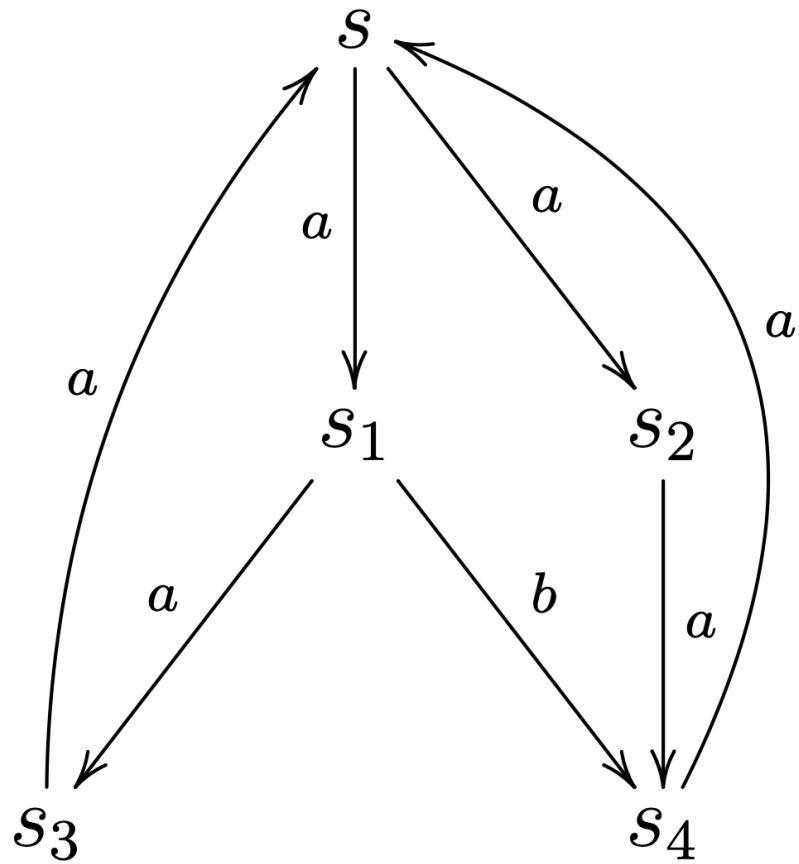
repeat π **until** $\varphi \equiv \quad \pi;(\neg\varphi?;\pi)^*$



Exercise

Consider the $(\{a, b\}, \{p, q\})$-model Mrepresented in the left, such that $V(p) = \{s_1, s_3\}$ e $V(q) = \{s, s_2, s_4\}$.
Interpret the following programs in M:

- $a; b$
- $b; a$
- $a + b$
- $(a; b) + b$
- a^*
- $(p?); a$
- $(q?); a + (\neg q?)b$
- $(a + b)^*$
- $(p \wedge q)?$



Propositional dynamic logic (in a rush)

Signatures Are pairs (Prop, Π) where Prop and Π are disjoint sets of **propositions**, and atomic programs

Propositional dynamic logic (in a rush)

Signatures Are pairs (Prop, Π) where Prop and Π are disjoint sets of **propositions**, and atomic programs

Sentences $\varphi ::= p \mid \langle\pi\rangle\varphi \mid [\pi]\varphi \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$

$\pi ::= \pi_0 \mid \pi + \pi \mid \pi;\pi \mid \pi^* \mid ?\varphi$

and $p \in \mathrm{Prop}$

Propositional dynamic logic (in a rush)

Signatures Are pairs (Prop, Π) where Prop and Π are disjoint sets of **propositions**, and atomic programs

Sentences $\varphi ::= p \mid \langle\pi\rangle\varphi \mid [\pi]\varphi \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$
$\pi ::= \pi_0 \mid \pi + \pi \mid \pi;\pi \mid \pi^* \mid ?\varphi$
and $p \in \mathrm{Prop}$

Models Models are Kripke structures, i.e. tuples (W, V, R)

- W is a set
- $V : \mathrm{Prop} \to \mathcal{P}(W)$ is a function
- $R = (R_\pi \subseteq W \times W)_{\pi \in \Pi}$ is an Π-family of binary relations

Propositional dynamic logic

Satisfaction

- $M, w \models \langle \pi \rangle \varphi$ iff there is a $w' \in W$ such that $(w, w') \in \overline{R}_\pi$ and $M, w' \models \varphi$;
- ...

Program interpretation

$\overline{R}(\pi_0) = R_{\pi_0}$

$\overline{R}(\chi?) = \{(w, w) | M, w \models \chi\}$

$\overline{R}(\pi + \pi') = \overline{R}(\pi) \cup \overline{R}(\pi')$

$\overline{R}(\pi; \pi') = \overline{R}(\pi) \cdot \overline{R}(\pi')$

$\overline{R}(\pi^*) = \overline{R}(\pi)^\star = \bigcup_{n \in \mathbb{N}} \overline{R}(\pi^n)$, where $\pi^{n+1} = \pi; \pi^n$

Exercício

Considere o $(\{a, b, c\}, \{p, q\})$-modelo $M = (W, R, V)$, com $W = \{w_1, w_2, w_3, w_4, w_5\}$ e tal que:

- $V(p) = \{w_1, w_3\}$ e $V(q) = W$,
- $R_a = \{(w_1, w_3), (w_1, w_4), (w_1, w_5), (w_2, w_3), (w_5, w_3)\}$
- $R_b = \{(x, y) \in W^2 | x = y\}$,
- $R_c = \{(w_1, x) | x \in W\}$

Verifique se:

- a) $M, w_1 \models [(a; b)]p \vee [b^* + c]q$
- b) $M, w_3 \models [q?; b]p \rightarrow [c]\neg q$

Exercício

Considere o $(\{a, b, c\}, \{p, q\})$-modelo $M = (W, R, V)$, com $W = \{-2, -1, 0, 1, 2\}$ e tal que:

- $V(p) = \{x \in W | x > 0\}$ e $V(q) = \{x \in W | x \leq 1\}$,
- $R_a = \{(x, y) \in W^2 | x \leq 0, y \geq 0\}$
- $R_b = \{(x, y) \in W^2 | x = y\}$
- $R_c = \{(0, x) | x \in W\}$

Verifique se:

- a) $M, 0 \models [(a + b)]p \vee [b^* + c]q$
- b) $M, 2 \models [(p \rightarrow q)?; b]p \rightarrow [c]\neg q$

Exercício

Verifique que as seguintes propriedades são válidas em PDL

- $[\alpha;\beta]\varphi \leftrightarrow [\alpha][\beta]\varphi$
- $[\alpha + \beta]\varphi \leftrightarrow [\alpha]\varphi \wedge [\beta]\varphi$
- $[\alpha^*]\varphi \rightarrow \varphi \wedge [\alpha][\alpha]^*\varphi$
- $[\alpha^*](\varphi \rightarrow [\alpha]\varphi) \rightarrow (\varphi \rightarrow [\alpha^*]\varphi$
- $[\varphi?]\psi \leftrightarrow (\varphi \rightarrow \psi)$

From propositional to first order DL

Programs:

$\alpha, \beta \ni x := \theta \mid \alpha + \beta \mid \alpha; \beta \mid \alpha^* \mid ?\chi$

States:

the space of variables valuations $\mathcal{S} = \mathbb{R}^{\mathcal{V}}$

Programs interpretation

The accessibility relation $\rho(\alpha) \subseteq \mathcal{S} \times \mathcal{S}$ is recursively defined by:

- $\rho(x := \theta) = \{(u, v) | v(x) = \theta$ and for any $y \in \mathcal{V} \setminus \{x\}, u(y) = v(y)\}$
- $\ldots$

By considering first order assignments:

verification of
$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$

$x = 1 \rightarrow [(x < 2)?; x := x + 1 + (\neg (x < 2))?; x := x * x]x = 2$

By considering first order assignments:

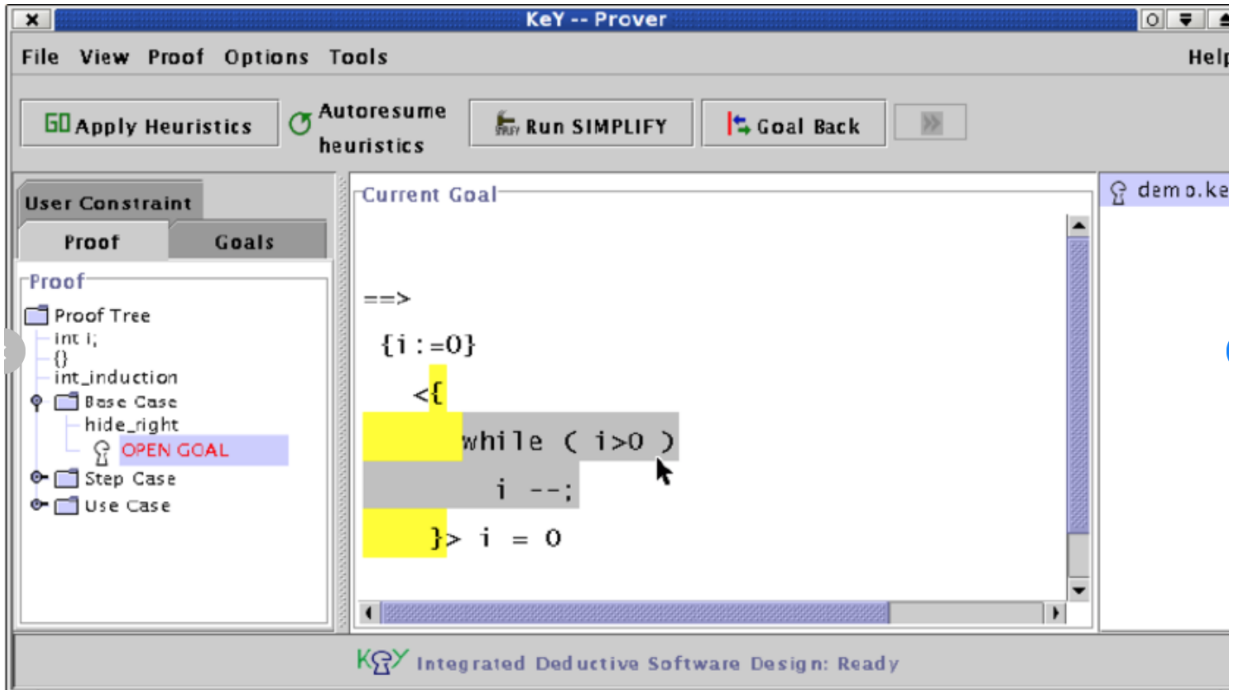
verification of
$\{x = 1\}\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x\{x = 2\}$

$x = 1 \rightarrow [(x < 2)?; x := x + 1 + (\neg (x < 2))?; x := x * x]x = 2$

$x = 1 \rightarrow [\textbf{if } x < 2 \textbf{ then } x := x + 1 \textbf{ else } x := x * x]x = 2$

Supporting tools?

KeY project



- A semi-automatic theorem prover based in dynamic logics
- see also KeY-Hoare

Reasoning about imperative programs with DL

Now, the verification of the Hoare triple

$$\{\text{Pre}\}\ \text{Prog}\ \{\text{Post}\}$$

corresponds to the dynamic logic validation of

$$\text{Pre} \rightarrow [\text{Prog}]\ \text{Post}$$

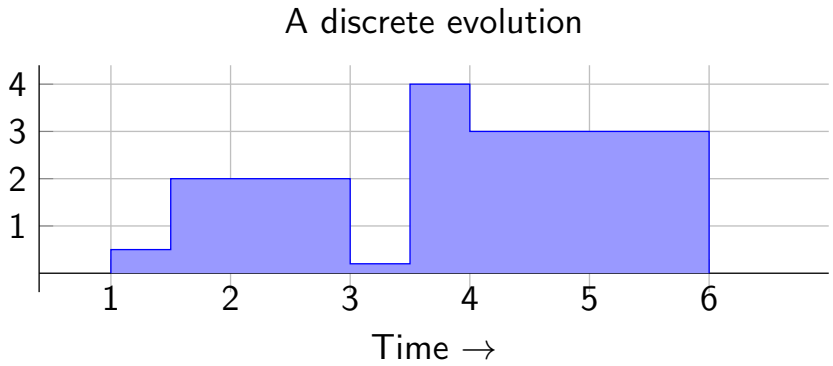
Outline

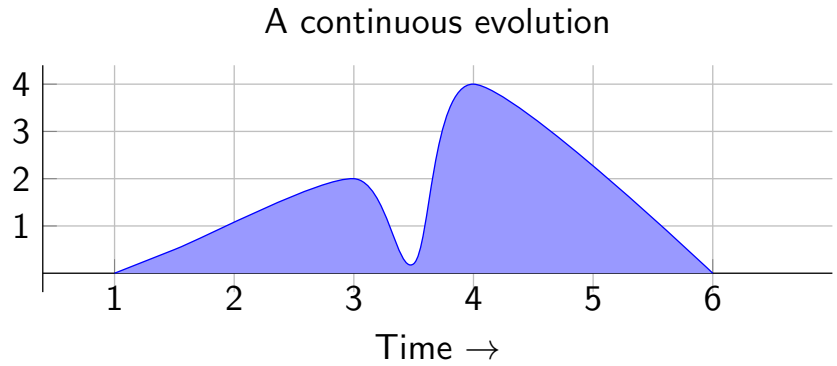
Dynamic Logic for hybrid systems?

Platzer's differential dynamic logic $d\mathcal{L}$

- logic developed for **specifying** and **verifying properties** of hybrid systems
- with a **'relative complete calculus'** i.e., we can prove properties of hybrid systems exactly as good as properties of differential equations can be proved
- a powerful **computational tool support** — KeYmaera

Discrete vs. Continuous evolutions





Hybrid = discrete + **continuous**

- digital controller actions, discrete event interaction, etc
- **physics entities, analogic controller actions, etc**

Platzer's $d\mathcal{L}$ – syntax

Hybrid Programs

$$\alpha, \beta \ni\ x := \theta \mid x' = \theta \,\&\, \chi \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid ?\chi$$

Platzer's $d\mathcal{L}$ – syntax

Hybrid Programs

$$\alpha, \beta \ni x := \theta \mid x' = \theta \,\&\, \chi \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid ?\chi$$

$d\mathcal{L}$-formulas

$$\phi, \psi \ni \theta_1 = \theta_2 \mid \theta_1 \leq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid [\alpha]\phi$$

where θ, θ_1 and θ_2 are terms

Platzer's $d\mathcal{L}$ – semantics

States:

are functions $\mathcal{V} \to \mathbb{R}$

Programs interpretation

The relation $\rho(\alpha) \subseteq \mathcal{S} \times \mathcal{S}$ is defined as for first order DL with

- $\rho(x := \theta) = \{(u, v) | v(x) = \theta$ and for any $y \in \mathcal{V} \setminus \{x\}, u(y) = v(y)\}$
- $\rho(x' = \theta \& \chi) = \{(\varphi(0), \varphi(r)) | \varphi(t) \models \chi, 0 \leq t \leq r,$ for $\varphi : [0, r] \to \mathcal{S}$ a solution of any duration $r\}$
- $\rho(\alpha \cup \beta) = \rho(\alpha) \cup \rho(\beta)$
- $\rho(\alpha; \beta) = \rho(\alpha) \circ \rho(\beta)$
- $\rho(\alpha^*) = \bigcup_{n \in \mathbb{N}} \rho(\alpha^n)$, where $\alpha^0 = id$ and $\alpha^{n+1} = \alpha; \alpha^n$
- $\rho(?\chi) = \{(v, v) | v \models \chi\}$

Platzer's $d\mathcal{L}$ – satisfaction

- $v \models (\theta_1 = \theta_2)$ iff $v_{\theta_1} = v_{\theta_2}$
- $v \models \neg\rho$ iff $v \not\models \rho$
- $v \models \rho \wedge \rho'$ iff $v \models \rho$ and $v \models \rho'$
- $v \models \rho \vee \rho'$ iff $v \models \rho$ or $v \models \rho'$
- $v \models [\alpha]\rho$ iff for any $(v, w) \in \rho(\alpha)$, $w \models \rho$
- $v \models \langle\alpha\rangle\rho$ iff there is a $(v, w) \in \rho(\alpha)$, such that $w \models \rho$

Platzer's $d\mathcal{L}$ – axiomatisation

Some deduction rules examples

$$\frac{\rho(\theta)}{[x := \theta]\rho(x)} \qquad \frac{\chi \to \rho}{[?\chi]\rho}$$

$$\frac{[\alpha][\beta]\rho}{[\alpha;\beta]\rho}$$

$$\frac{[\alpha^*](\rho \to [\alpha]\rho)}{\rho \to [\alpha^*]\rho}$$

$$\frac{\forall t \geq 0 [x := y(t)]\rho}{[x' = \theta]\rho}, \text{ for } y'(t) = \theta$$

...

Some deduction Rules

$$(ax)\frac{}{\varphi \vdash \varphi} \qquad (\neg\mathrm{r})\frac{\varphi \vdash}{\vdash \neg\varphi} \qquad (\neg\mathrm{l})\frac{\vdash \varphi}{\neg\varphi \vdash} \qquad (\vee\mathrm{r})\frac{\vdash \varphi, \psi}{\vdash \varphi \vee \psi} \qquad (\vee\mathrm{l})\frac{\varphi \vdash \quad \psi \vdash}{\varphi \vee \psi \vdash}$$

$$(cut)\frac{\vdash \varphi \quad \varphi \vdash}{\vdash} \qquad (\to\mathrm{r})\frac{\psi \vdash \varphi}{\vdash \psi \to \varphi} \qquad (\to\mathrm{l})\frac{\vdash \varphi \quad \psi \vdash}{\varphi \to \psi \vdash} \qquad (\wedge\mathrm{r})\frac{\psi \vdash \quad \varphi \vdash}{\vdash \varphi \wedge \psi} \qquad (\wedge\mathrm{l})\frac{\varphi, \psi \vdash}{\varphi \wedge \psi \vdash}$$

$$(\langle;\rangle)\frac{\langle\alpha\rangle\langle\beta\rangle\varphi}{\langle\alpha;\beta\rangle\varphi} \qquad (\langle\cup\rangle)\frac{\langle\alpha\rangle\varphi \vee \langle\beta\rangle\varphi}{\langle\alpha \cup \beta\rangle\varphi} \qquad (\langle?\rangle)\frac{\chi \wedge \varphi}{\langle?\chi\rangle\varphi} \qquad (\langle^{*n}\rangle)\frac{\varphi \vee \langle\alpha\rangle\langle\alpha^*\rangle\varphi}{\langle\alpha^*\rangle\varphi}$$

$$([;])\frac{[\alpha][\beta]\varphi}{[\alpha;\beta]\varphi} \qquad ([\cup])\frac{[\alpha]\varphi \wedge [\beta]\varphi}{[\alpha \cup \beta]\varphi} \qquad ([?])\frac{\chi \to \varphi}{[?\chi]\varphi} \qquad ([^{*n}])\frac{\varphi \wedge [\alpha][\alpha^*]\varphi}{[\alpha^*]\varphi}$$

$$(\langle:=\rangle)\frac{\varphi^{\theta_1 \ldots \theta_n}_{x_1 \ldots x_n}}{\langle x_1 := \theta_1, \ldots, x_n := \theta_n\rangle\varphi} \qquad (\langle'\rangle)\ \frac{\exists t \geq 0((\forall 0 \leq t' \leq t,\ \langle \mathfrak{S}_{t'}\rangle\chi) \wedge \langle \mathfrak{S}_t\rangle\varphi)}{\langle x_1' = \theta_1, .., x_n' = \theta_n\ \&\ \chi\rangle\varphi}$$

$$([:=])\frac{\langle x_1 := \theta_1, \ldots, x_n := \theta_n\rangle\varphi}{[x_1 := \theta_1, \ldots, x_n := \theta_n]\varphi} \qquad ([']\,)\frac{\forall t \geq 0((\forall 0 \leq t' \leq t,\ \langle \mathfrak{S}_{t'}\rangle\chi) \to \langle \mathfrak{S}_t\rangle\varphi)}{[x_1' = \theta_1, .., x_n' = \theta_n\ \&\ \chi]\varphi}$$

Some deduction Rules

$$(\forall \mathrm{r})\frac{\vdash \varphi(s(X_1,...,X_n))}{\vdash \forall x\varphi(x)} \qquad (\exists \mathrm{r})\frac{\vdash \varphi(X)}{\vdash \exists x\varphi(x)} \qquad (\mathrm{i}\forall)\frac{\vdash \mathrm{QE}(\forall X(\Phi(X) \vdash \Psi(X)))}{\Phi(s(X_1,...,X_n)) \vdash \Psi(s(X_1,...,X_n))}$$

$$(\exists \mathrm{l})\frac{\varphi(s(X_1,...,X_n)) \vdash}{\exists x(\varphi(x)) \vdash} \qquad (\forall \mathrm{l})\frac{\varphi(X) \vdash}{\forall x\varphi(x) \vdash} \qquad (\mathrm{i}\exists)\frac{\vdash \mathrm{QE}(\exists X \bigwedge_i(\Phi_i \vdash \Psi_i))}{\Phi_1 \vdash \Psi_1 \ \ldots \ \Phi_n \vdash \Psi_n}$$

$$([]gen)\frac{\vdash \forall^{\alpha}(\varphi \to \psi)}{[\alpha]\varphi \vdash [\alpha]\psi} \qquad (\langle\rangle gen)\frac{\vdash \forall^{\alpha}(\varphi \to \psi)}{\langle\alpha\rangle\varphi \vdash \langle\alpha\rangle\psi}$$

$$(ind)\frac{\vdash \forall^{\alpha}(\varphi \to [\alpha]\varphi)}{\varphi \vdash [\alpha^*]\varphi} \qquad (con)\frac{\vdash \forall^{\alpha}\forall v > 0(\varphi(v) \to \langle\alpha\rangle\varphi(v-1))}{\exists v\varphi(v) \vdash \langle\alpha^*\rangle\exists v \leq 0(\varphi(v))}$$

Case Study from Biology

Controller in a biological system.

Example

$\begin{cases} x' = 5 - x \\ y' = 6 - y + u \end{cases}$ $x < 3 \wedge y \geq 2$	$\begin{cases} x' = 6 - x \\ y' = 1 - y + u \end{cases}$ $x \geq 3 \wedge y \geq 2$
$\begin{cases} x' = -x \\ y' = 5 - y + u \end{cases}$ $x < 3 \wedge y < 2$	$\begin{cases} x' = 1 - x \\ y' = -y + u \end{cases}$ $x \geq 3 \wedge y < 2$

D. Figueiredo, Manuel Martins and M. Chaves.
Applying differential dynamic logic to reconfigurable biological networks,
Mathematical Biosciences, vol. 291, 10-20, 2017.

Biological Examples.

Controller in a biological system.

we look for steady states

i.e., the values of x and y for which the system tends

Control:

- $u = 2$, if $x \geq 3$ and $t \geq 2$
- $u = 0$, otherwise.

Using numeric method we can obtain $(x, y) = (6, 3)$ as steady state candidate.

Biological Examples

Controller in a biological system.

$\begin{cases} x' = 5 - x \\ y' = 6 - y + u \end{cases}$ $x < 3 \wedge y \geq 2$	$\begin{cases} x' = 6 - x \\ y' = 1 - y + u \end{cases}$ $x \geq 3 \wedge y \geq 2$
$\begin{cases} x' = -x \\ y' = 5 - y + u \end{cases}$ $x < 3 \wedge y < 2$	$\begin{cases} x' = 1 - x \\ y' = -y + u \end{cases}$ $x \geq 3 \wedge y < 2$

- $\alpha_1 \equiv (?x < 3 \wedge y < 2; u := 0;$
  $(x' = -x, y' = 5 - y + u, \tau' = 1 \,\&\, x \leq 3 \wedge y \leq 2))$
- ...
- $\alpha_4 \equiv (?x \geq 3 \wedge y \geq 2; u := 2;$
  $(x' = 6 - x, y' = 1 - y + u, \tau' = 1 \,\&\, x \geq 3 \wedge y \geq 2))$

Example in Biology

Controller in a biological system.

the evolution of the entire biological system can be described by:

$$\alpha \equiv \alpha_1 \cup \alpha_2 \cup \alpha_3 \cup \alpha_4$$

Example in Biology

Controller in a biological system.

the evolution of the entire biological system can be described by:

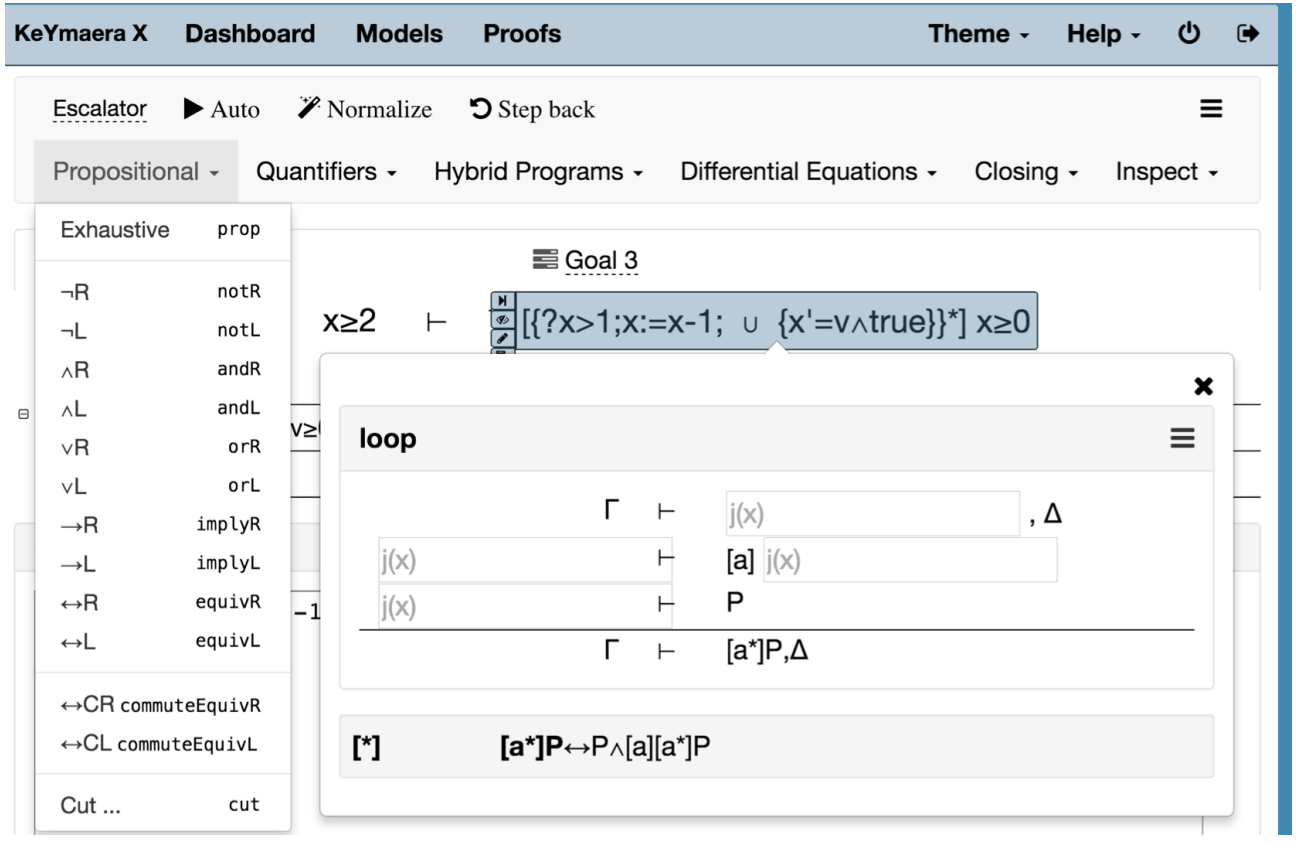
$$\alpha \equiv \alpha_1 \cup \alpha_2 \cup \alpha_3 \cup \alpha_4$$

$(x, y) = (6, 3)$ is a steady state:

$$\exists c > 0 (\forall\, 0 < k < c((x-6)^2 + (y-3)^2 = k \wedge\ \tau = 0 \rightarrow [\alpha^*](\tau = 0 \vee (x-6)^2 + (y-3)^2 < k)))$$

Supporting tools?

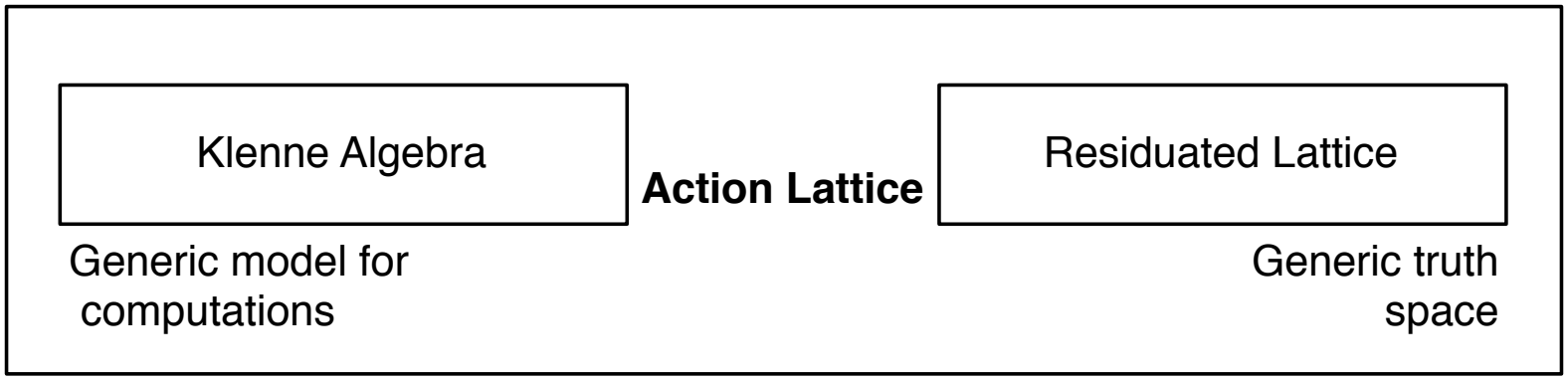
KeYmaera X



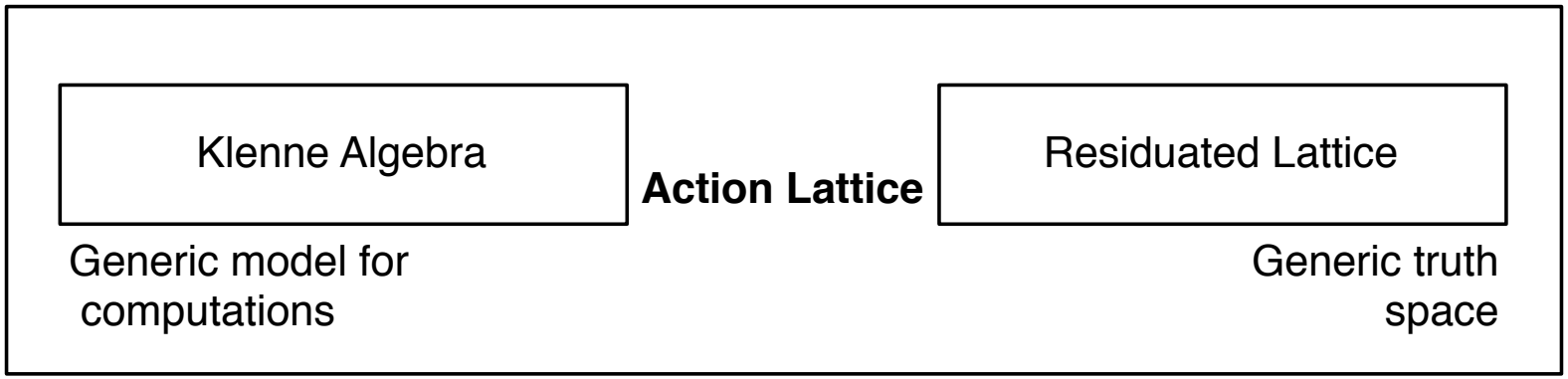
- A semi-automatic theorem prover to analyse cyber-physical systems based in Key system

Outline

Construction parameter



Construction parameter



Action lattice (Pratt 90, Kozen 91)

$$A = (A, +, ;, 0, 1, *, \rightarrow, \cdot)$$

- $(A, +, ;, 0, 1, *)$ is a **Kleene algebra**;
- $\rightarrow$ is a **residue wrt** ;
- $(A, +, .)$ is a **lattice** wrt relation $a \leq b \equiv a + b = b$

Action lattice axiomatisation

$$
\begin{array}{rcll}
a + (b + c) & = & (a + b) + c & (1) \\
a + b & = & b + a & (2) \\
a + a & = & a & (3) \\
a + 0 & = & 0 + a = a & (4) \\
a; (b; c) & = & (a; b); c & (5) \\
a; 1 & = & 1; a = a & (6) \\
a; (b + c) & = & (a; b) + (a; c) & (7) \\
(a + b); c & = & (a; c) + (b; c) & (8) \\
a; 0 & = & 0; a = 0 & (9) \\
1 + a + (a^*; a^*) & \leq & a^* & (10) \\
a; x \leq x & \Rightarrow & a^*; x \leq x & (11)
\end{array}
$$

$$
x; a \leq x \quad \Rightarrow \quad x; a^* \leq x \quad (12)
$$

Action lattice axiomatisation

$$
\begin{array}{rcll}
a + (b + c) & = & (a + b) + c & (1) \\
a + b & = & b + a & (2) \\
a + a & = & a & (3) \\
a + 0 & = & 0 + a = a & (4) \\
a; (b; c) & = & (a; b); c & (5) \\
a; 1 & = & 1; a = a & (6) \\
a; (b + c) & = & (a; b) + (a; c) & (7) \\
(a + b); c & = & (a; c) + (b; c) & (8) \\
a; 0 & = & 0; a = 0 & (9) \\
1 + a + (a^*; a^*) & \leq & a^* & (10) \\
a; x \leq x & \Rightarrow & a^*; x \leq x & (11)
\end{array}
$$

$$
\begin{array}{rcll}
x; a \leq x & \Rightarrow & x; a^* \leq x & (12) \\
a; x \leq b & \Leftrightarrow & x \leq a \to b & (13) \\
a \to b & \leq & a \to (b + c) & (14) \\
(x \to x)^* & = & x \to x & (15) \\
x & \leq & a \to (a; x) & (16)
\end{array}
$$

Action lattice axiomatisation

$$a + (b + c) = (a + b) + c \quad (1)$$
$$a + b = b + a \quad (2)$$
$$a + a = a \quad (3)$$
$$a + 0 = 0 + a = a \quad (4)$$
$$a;(b;c) = (a;b);c \quad (5)$$
$$a;1 = 1;a = a \quad (6)$$
$$a;(b + c) = (a;b) + (a;c) \quad (7)$$
$$(a + b);c = (a;c) + (b;c) \quad (8)$$
$$a;0 = 0;a = 0 \quad (9)$$
$$1 + a + (a^*;a^*) \leq a^* \quad (10)$$
$$a;x \leq x \Rightarrow a^*;x \leq x \quad (11)$$
$$x;a \leq x \Rightarrow x;a^* \leq x \quad (12)$$
$$a;x \leq b \Leftrightarrow x \leq a \to b \quad (13)$$
$$a \to b \leq a \to (b + c) \quad (14)$$
$$(x \to x)^* = x \to x \quad (15)$$
$$x \leq a \to (a;x) \quad (16)$$
$$a \cdot (b \cdot c) = (a \cdot b) \cdot c \quad (17)$$
$$a \cdot b = b \cdot a \quad (18)$$
$$a \cdot a = a \quad (19)$$
$$a + (a \cdot b) = a \quad (20)$$
$$a \cdot (a + b) = a \quad (21)$$
$$a;(a \to b) \leq b \quad (22)$$

Examples

2 - linear two-values lattice

$$\mathbf{2} = (\{\top, \bot\}, \vee, \wedge, \bot, \top, *, \rightarrow, \wedge)$$

$\vee$	$\bot$	$\top$
$\bot$	$\bot$	$\top$
$\top$	$\top$	$\top$

$\wedge$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$

$\rightarrow$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\top$

$*$	
$\bot$	$\top$
$\top$	$\top$

Examples

2 - linear two-values lattice

$$\mathbf{2} = (\{\top, \bot\}, \vee, \wedge, \bot, \top, *, \rightarrow, \wedge)$$

$\vee$	$\bot$	$\top$
$\bot$	$\bot$	$\top$
$\top$	$\top$	$\top$

$\wedge$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$

$\rightarrow$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\top$

$*$	
$\bot$	$\top$
$\top$	$\top$

3 - linear three-value lattice

$$\mathbf{3} = (\{\top, u, \bot\}, \vee, \wedge, \bot, \top, *, \rightarrow, \wedge)$$

$\vee$	$\bot$	u	$\top$
$\bot$	$\bot$	u	$\top$
u	u	u	$\top$
$\top$	$\top$	$\top$	$\top$

$\wedge$	$\bot$	u	$\top$
$\bot$	$\bot$	$\bot$	$\bot$
u	$\bot$	u	u
$\top$	$\bot$	u	$\top$

$\rightarrow$	$\bot$	u	$\top$
$\bot$	$\top$	$\top$	$\top$
u	$\bot$	$\top$	$\top$
$\top$	$\bot$	u	$\top$

$*$	
$\bot$	$\top$
u	$\top$
$\top$	$\top$

Examples

Ł - the Łukasiewicz arithmetic lattice

$$\mathbf{Ł} = ([0,1], max, \odot, 0, 1, *, \rightarrow, min)$$

where

- $x \odot y = max\{0, y + x - 1\}$,
- $x \rightarrow y = min\{1, 1 - x + y\}$ and
- $*$ maps each point of $[0,1]$ to 1.

Examples

FW - the Floyd-Warshall algebra

$$\mathbb{N}^{+}_{\perp\top} = (\{\perp, 0, 1, \ldots, \top\}, \mathit{max}, +, \perp, 0, *, \sim, \mathit{min})$$

- $+$ extends addition on $\mathbb{N}$ by considering $\perp$ as its absorbent
- *max* and *min* wrt the order $\perp < 0 < \cdots < \top$

$$a \sim b = \begin{cases} \top, & \text{if } a = \perp \text{ or } b = \top \\ b - a, & \text{if } b \geq a \text{ and } a, b \in \mathbb{N} \\ 0, & \text{if } a > b \text{ and } a, b \in \mathbb{N} \\ \perp & \text{otherwise} \end{cases}$$

$*$	
$\perp$	0
0	0
i	$\top$
$\top$	$\top$

Parametric construction

Let us construct

$$\mathcal{GDL}(\mathbf{A})$$

for a fixed action lattice

$$\mathbf{A} = (A, +, ;, 0, 1, *, \to, \cdot)$$

Parametric construction

Let us construct

$$\mathcal{GDL}(\mathbf{A})$$

for a fixed action lattice

$$\mathbf{A} = (A, +, ;, 0, 1, *, \to, \cdot)$$

$\mathcal{GDL}(\mathbf{A})$-signatures

are propositional dynamic logic signatures, i.e. pairs

$$(\mathrm{Prop}, \Pi)$$

$\mathcal{GDL}(\mathbf{A})$ – formulæ

The **set of programs** $\mathit{Prog}(\Pi)$:

$$\pi \ni \pi_0 \,|\, \pi;\pi \,|\, \pi + \pi \,|\, \pi^*$$

for $\pi_0 \in \Pi$.

$\mathcal{GDL}(\mathbf{A})$ – formulæ

The **set of programs** $\mathit{Prog}(\Pi)$:

$$\pi \ni \pi_0 \mid \pi;\pi \mid \pi + \pi \mid \pi^*$$

for $\pi_0 \in \Pi$. The set of formulas $\mathrm{Fm}^{\Gamma(\mathbf{A})}(\Pi, \mathrm{Prop})$:

$$\rho \ni \top \mid \bot \mid p \mid \rho \vee \rho \mid \rho \wedge \rho \mid \rho \to \rho \mid \langle\pi\rangle\rho \mid [\pi]\rho$$

for $p \in \mathrm{Prop}$ and $\pi \in \mathit{Prog}(\Pi)$.

$\mathcal{GDL}(\mathbf{A})$ – models

Based on [Conway 71] we consider the Kleene algebra

$$\mathbb{M}_n(\mathbf{A}) = (M_n(\mathbf{A}), +, ;, \mathbf{0}, \mathbf{1}, *)$$

- $M_n(\mathbf{A})$ is the space of $(n \times n)$-matrices over $\mathbf{A}$
- $M = A+B$ defined by $M_{i,j} = A_{i,j} + B_{i,j}$, $i, j \leq n$.
- $M = A \,;\, B$ defined by $M_{i,j} = \sum_{k=1}^{n}(A_{i,k}; B_{k,j})$ for any $i, j \leq n$.
- $\mathbf{1}$ and $\mathbf{0}$ the identity and 0 matrices
- $M = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$

  $M^* = \left[\begin{array}{c|c} F^* & F^* \,;\, B \,;\, D^* \\ \hline D^*; C; F^* & D^* + (D^* \,;\, C \,;\, F^* \,;\, B \,;\, D^*) \end{array}\right]$ where

  $F = A + B \,;\, D^* \,;\, C.$

$\mathcal{GDL}(\mathbf{A})$ – models

$\mathcal{GDL}(\mathbf{A})$-models for (Prop, Π)

$$\mathcal{A} = (W, V, (\mathcal{A}_\pi)_{\pi \in \Pi})$$

where

- W is a set (of states),
- $V : \mathrm{Prop} \times W \to \mathbf{A}$ is a function,
- and $\mathcal{A}_\pi \in \mathbb{M}_n(\mathbf{A})$, with n standing for the cardinality of W.

Examples

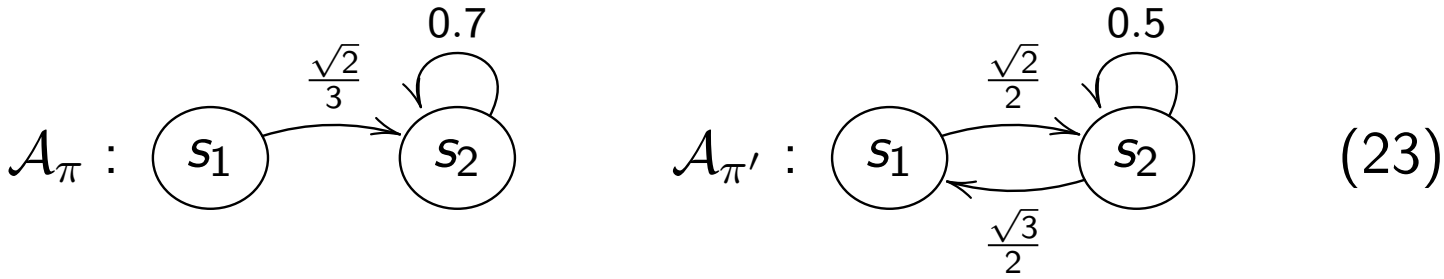
For a classic *PDL* semantics

$$\mathbf{2} = (\{\top, \bot\}, \vee, \wedge, \bot, \top, *, \leftarrow, \rightarrow, \wedge)$$

$$\mathcal{A}_{\pi;\pi'} = \begin{bmatrix} \bot & \top \\ \bot & \top \end{bmatrix}; \begin{bmatrix} \bot & \top \\ \bot & \bot \end{bmatrix} =$$

$$\begin{bmatrix} (\bot \wedge \bot) \vee (\top \wedge \bot) & (\bot \wedge \top) \vee (\top \wedge \bot) \\ (\bot \wedge \bot) \vee (\top \wedge \bot) & (\bot \wedge \top) \vee (\top \wedge \bot) \end{bmatrix} = \begin{bmatrix} \bot & \bot \\ \bot & \bot \end{bmatrix}$$

Examples



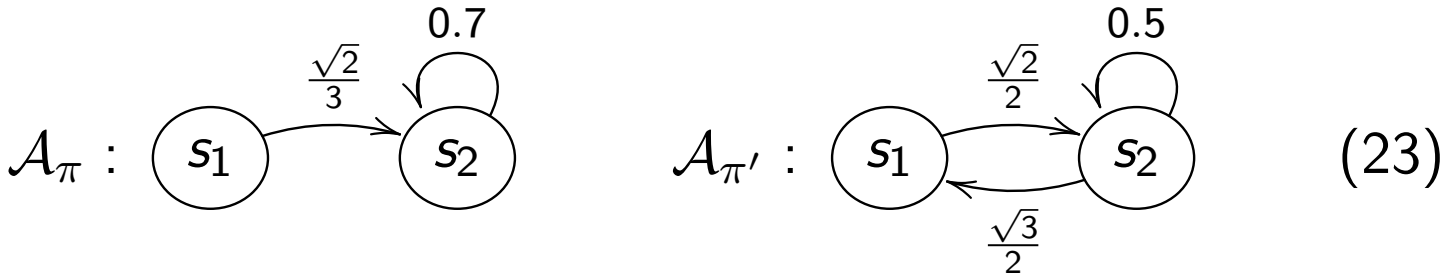
(23)

For systems with uncertainty

$$\mathbf{Ł} = ([0,1], max, \odot, 0, 1, *, \rightarrow, min)$$

$\mathcal{A}_{\pi+\pi'} =$
$\max(\mathcal{A}_\pi, \mathcal{A}_{\pi'}) =$

Examples



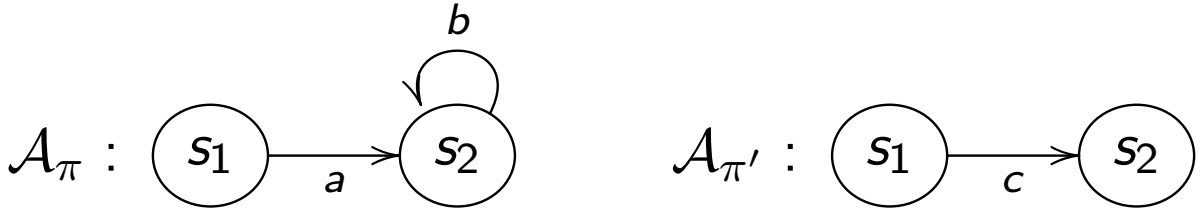
(23)

For systems with uncertainty

$$\mathbf{Ł} = ([0,1], max, \odot, 0, 1, *, \rightarrow, min)$$

$$\mathcal{A}_{\pi+\pi'} =$$
$$\max(\mathcal{A}_\pi, \mathcal{A}_{\pi'}) =$$
$$\max\left(\begin{bmatrix} 0 & \frac{\sqrt{2}}{3} \\ 0 & 0.7 \end{bmatrix}, \begin{bmatrix} 0 & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{3}}{2} & 0.5 \end{bmatrix}\right) = \begin{bmatrix} 0 & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{3}}{2} & 0.7 \end{bmatrix}$$

Examples



For cost transitions systems

$$\mathbb{N}^+_{\perp\top} = (\{\perp, 0, 1, \ldots, \top\}, max, +, \perp, 0, *, \sim, min)$$

$$\mathcal{A}_{\pi^*} = \begin{bmatrix} \perp & a \\ \perp & b \end{bmatrix} = \begin{bmatrix} f^* & f^* + a + b^* \\ \perp & max\{b^*, b^* + \perp + \perp^* + a + b^*\} \end{bmatrix} =$$

$$\begin{bmatrix} 0 & a + b^* \\ \perp & b^* \end{bmatrix}, \text{ where } f = max\{\perp, a + b^* + \perp\}$$

$\mathcal{GDL}(\mathbf{A})$ satisfaction

$$\models : W \times \mathrm{Fm}^{\Gamma(\mathbf{A})}(\Pi, \mathrm{Prop}) \to A$$

$\mathcal{GDL}(\mathbf{A})$ satisfaction

$$\models : W \times \mathrm{Fm}^{\Gamma(\mathbf{A})}(\Pi, \mathrm{Prop}) \to A$$

- $(w \models \top) = \top$
- $(w \models \bot) = \bot$
- $(w \models p) = V(p, w)$, for any $p \in \mathrm{Prop}$
- $(w \models \rho \wedge \rho') = (w \models \rho) \cdot (w \models \rho')$
- $(w \models \rho \vee \rho') = (w \models \rho) + (w \models \rho')$
- $(w \models \rho \to \rho') = (w \models \rho) \to (w \models \rho')$
- $(w \models \rho \leftrightarrow \rho') = (w \models \rho \to \rho'); (w \models \rho' \to \rho)$
- $(w \models \langle \pi \rangle \rho) = \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); (w' \models \rho) \right)$
- $(w \models [\pi] \rho) = \prod_{w' \in W} \left(\mathcal{A}_\pi(w, w') \to (w' \models \rho) \right)$

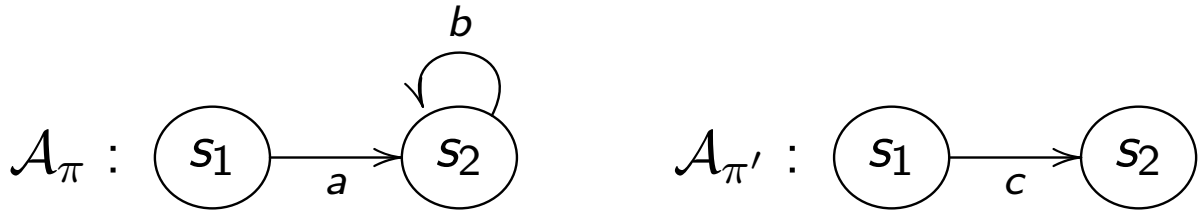
Example — $\mathcal{GDL}(\mathbf{2})$

with $V(p, s_1) = \bot$ and $V(p, s_2) = \top$

$$
\begin{aligned}
(s_1 \models \langle \pi^* \rangle p) = \\
&= \sum_{w' \in W} \{\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p)\} \\
&= \left(\mathcal{A}_{\pi^*}(s_1, s_1) \wedge (s_1 \models p)\right) \vee \left(\mathcal{A}_{\pi^*}(s_1, s_2) \wedge (s_2 \models p)\right) \\
&= (\top \wedge V(p, s_1)) \vee (\top \wedge V(p, s_2)) \\
&= (\top \wedge \bot) \vee (\top \wedge \top) \\
&= \top
\end{aligned}
$$

we can achieve at a state satisfying p from s_1 through π^*

Example — $\mathcal{GDL}(\mathbb{N}^{+}_{\perp\top})$

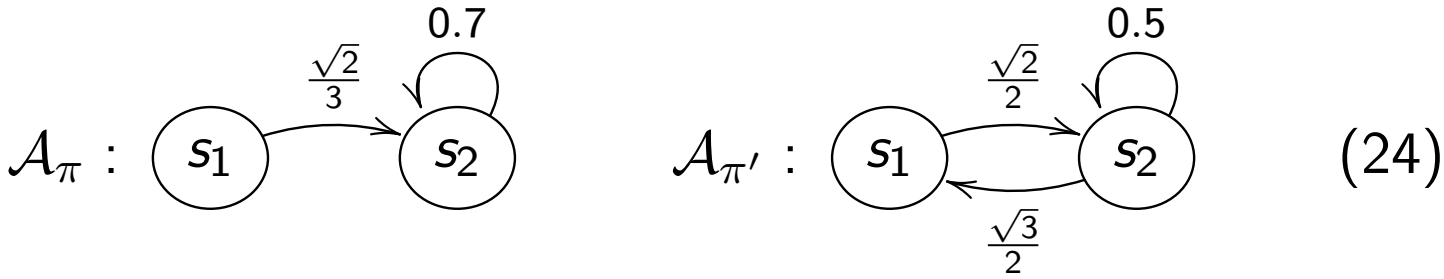


with $V(s_1, p) = \perp$ and $V(s_2, p) = 0$

$$
\begin{aligned}
(s_1 \models \langle \pi^* \rangle p) = \\
&= \sum_{w' \in W} \{\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p)\} \\
&= max\{\mathcal{A}_{\pi^*}(s_1, s_1) + (s_1 \models p), \mathcal{A}_{\pi^*}(s_1, s_2) + (s_2 \models p)\} \\
&= max\{0 + \perp, a + b^* + 0\} \\
&= a + b^*
\end{aligned}
$$

we can achieve at a state satisfying p from s_1 through π^* consuming $a + b^*$ cost unities

Example — $\mathcal{GDL}(Ł)$



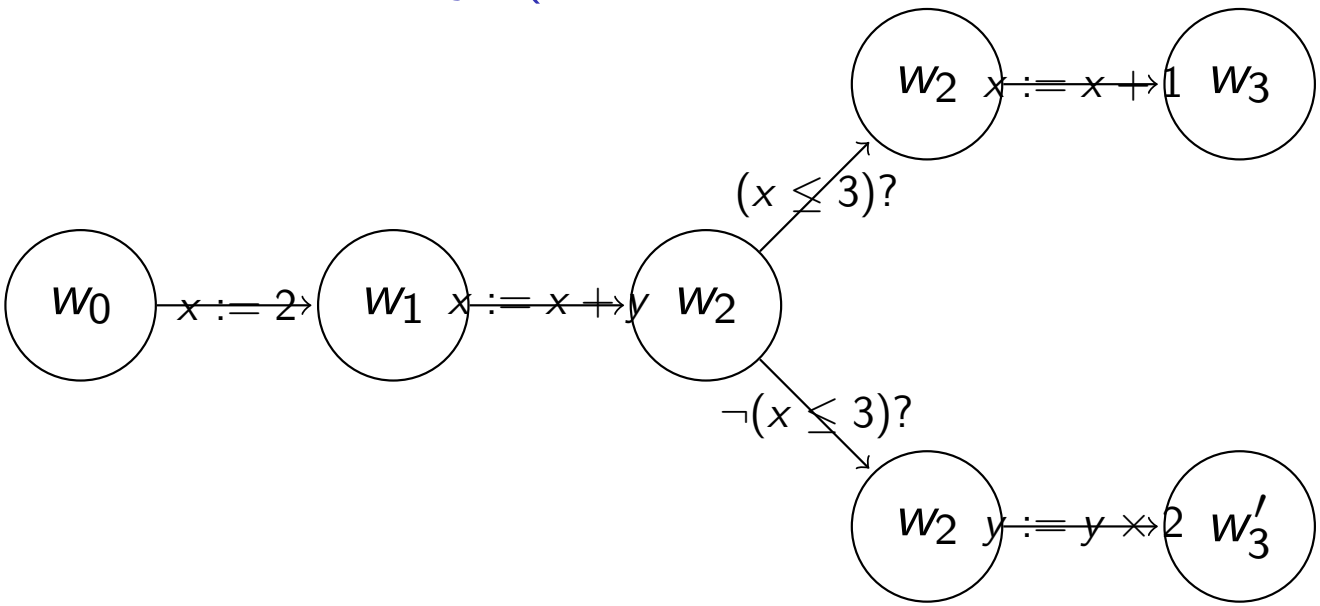
(24)

with $V(p, s_1) = 0.1$, $V(q, s_1) = 0.5$, $V(p, s_2) = \frac{\pi}{4}$ and $V(q, s_2) = 0.75$

$$
\begin{aligned}
s_1 \models \langle \pi + \pi' \rangle (p \to q)) \\
&= \max(0 \odot (0.1 \to 0.5), \tfrac{\sqrt{2}}{2} \odot (0.75 \to \tfrac{\pi}{4})) \\
&= \tfrac{\sqrt{2}}{2} \odot (0.75 \to \tfrac{\pi}{4}) \\
&= \tfrac{\sqrt{2}}{2} \odot \min(1, 1 - 0.75 + \tfrac{\pi}{4}) \\
&= \tfrac{\sqrt{2}}{2}
\end{aligned}
$$

Logic for imperative weighted programs?

$x := 2; x := x + y; (\textbf{if } x \leq 3 \textbf{ then } x := x + 1 \textbf{ else } y := y \times 2)$



On the Generation of Equational Dynamic Logics for Weighted Imperative Programs. Leandro Gomes, Alexandre Madeira, Manisha Jain, Luís Soares Barbosa. ICFEM 2019

Syntax of $\Gamma(\mathbf{A})$

$$\mathbf{A} = (A, +, ;, 0, 1, *, \to, \cdot)$$

Signatures of of $\Gamma(\mathbf{A})$

are pairs (Σ, Π), with

- Σ is a FOL signature
- $\Pi = \{x := t \mid x \in X \text{ and } t \in T_\Sigma(X)\}$

Programs

$\pi ::= \pi_0 \mid \phi? \mid \pi; \pi \mid \pi + \pi \mid \pi^*, \;\; \pi_0 \in \Pi$

Formulas of $\Gamma(\mathbf{A})$

$\varphi ::= \top \mid \bot \mid P(t_0, \ldots, t_n) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \to \varphi \mid \langle \pi \rangle \varphi \mid [\pi] \varphi$

Interpretation of Atomic Programs

States are functions

$$w : X \times \mathbb{R} \to A$$

where A is the carrier of action lattice $\mathbf{A}$

(Σ, Π)-Models of $\Gamma(\mathbf{A})$

are structures

$$M = (W, E)$$

where

- $W \subseteq A^{X \times \mathbb{R}}$ is a set of states;
- $E : \Pi \times (W \times W) \to A$ is a program grading function.

Interpretation of Atomic Programs

Interpretation of terms $[\![t]\!]_w \colon T_\Sigma^F(X) \to A^{\mathbb{R}}$

- $[\![x]\!]_w(r) = w(x, r)$
- $[\![c]\!]_w(r) = 1$ if $r = c$ and $[\![c]\!]_w(r) = 0$ otherwise
- $[\![f(t_1, \ldots, t_n)]\!]_w(r) = \sum_{i \in I} \{\prod_{j=1}^{n} [\![t_j]\!]_w(r_j^i) \mid f(r_1^i, \ldots, r_n^i) = r\}$,
  I the cardinality of the set solutions of $f(r_1^i, \ldots, r_n^i) = r$ in $\mathbb{R}$

Interpretation of Atomic Programs

Example in $\Gamma(\mathbf{G})$

Let us consider a state w such that $w(x,1) = 0.5$, $w(x,2) = 0.2$, $w(y,1) = 0.1$, $w(y,2) = 0.4$ and 0 otherwise for state w.

$$\begin{aligned}
[\![x+y]\!]_w(3) =& [\![x]\!]_w(1); [\![y]\!]_w(2) + [\![x]\!]_w(2); [\![y]\!]_w(1) \\
=& w(x,1); w(y,2) + w(x,2); w(y,1) \\
=& max(min(0.5; 0.4), min(0.2; 0.1)) \\
=& 0.4
\end{aligned}$$

Interpretation of Atomic Programs

Interpretation of predicates $[\![p]\!]_w : T_\Sigma^P(X) \to A$

$$[\![p(t_1, \ldots, t_n)]\!]_w = \sum_{i \in I} \{ \prod_{j=1}^{n} [\![t_j]\!]_w(r_j^i) : p(r_1^i, \ldots, r_n^i) \textbf{ is true} \}$$

where I is the cardinality of the set of all possible values $(r_1^i, \ldots, r_n^i) \in \mathbb{R}^n$ satisfying $p(r_1^i, \ldots, r_n^i)$ in $\mathbb{R}$

Example in $\Gamma(\mathbf{G})$

$w(x, 2) = 0.3$, $w(x, 3) = 0.5$, $w(x, 4) = 0.5$, $w(x, r) = 0$ otherwise

$[\![x \leq 3]\!](w) = [\![x]\!](2); [\![3]\!](3) + [\![x]\!](3); [\![3]\!](3) =$

$max\{min\{0.3, 1\}, min\{0.2, 1\}\} = 0.3$

Interpretation of atomic programs

Interpretation of atomic programs

$$[\![\_]\!]^0 : \Pi \to A^{W \times W}$$

is the map defined by:

$$[\![x := t]\!]^0(w, w') = \begin{cases} E(x := t, (w, w')) & \text{if } (w, w') \in (\!|x := t|\!) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

with

$$(w, w') \in (\!|x := t|\!) \Leftrightarrow \begin{cases} w'(y, r) = w(y, r) & \text{if } y \neq x \\ w'(x, r) = [\![t]\!]_w(r) & \text{otherwise} \end{cases}$$

interpretation of (composed) programs

The algebra of *program grading functions*

for an action lattice $\mathbf{A} = (A, +, ;, 0, 1, *, \rightarrow, \cdot)$ and a set of states W, is the structure

$$\mathbf{E} = (Z(E), \cup, \circ, \varnothing, \chi, *)$$

where:

- $Z(E)$ is the universe of all the program grading functions
- $(E(\pi_1) \cup E(\pi_2))(w, w') = E(\pi_1, (w, w')) + E(\pi_2, (w, w'))$
- $(E(\pi_1) \circ E(\pi_2))(w, w') = \sum_{w'' \in W} E(\pi_1, (w, w'')); E(\pi_2, (w'', w'))$
- $\varnothing(w, w') = 0$
- $(E(\pi))^*(w, w') = \sum_{i \geq 0} (E(\pi))^i(w, w') =$
  $(E(\pi))^0(w, w') + (E(\pi))^1(w, w') + (E(\pi))^2(w, w') + \ldots$

interpretation of (composed) programs

The interpretation of a composed program

in a model is a map

$$[\![-]\!] : \mathrm{Prg}(\Sigma, X) \to A^{W \times W}$$

where

- $[\![\pi_0]\!] = [\![\pi_0]\!]^0$, for each $\pi_0 \in \mathrm{Prg}_0(\Delta)$
- $[\![\pi ; \pi]\!] = [\![\pi]\!] \circ [\![\pi']\!]$
- $[\![\pi + \pi]\!] = [\![\pi]\!] \cup [\![\pi']\!]$
- $[\![\pi^*]\!] = [\![\pi]\!]^*$

where, for $r \in A^{W \times W}$, $r^*(w, w') = \sum_{k \geq 0} r^k(w, w')$.

Satisfaction

The graded Satisfaction relation

for a model $M \in \mathrm{Mod}^{\Gamma(\mathbf{A})}(\Delta)$, consists of a function

$$\models_{\Gamma(\mathbf{A})} : W \times \mathrm{Fm}^{\Gamma(\mathbf{A})}(\Delta) \to A$$

recursively defined by

- $(w \models_{\Gamma(\mathbf{A})} \top) = 1$
- $(w \models_{\Gamma(\mathbf{A})} \bot) = 0$
- $(w \models_{\Gamma(\mathbf{A})} p(t_1, \ldots, t_n)) = [\![p(t_1, \ldots, t_n)]\!]_w$
- $(w \models_{\Gamma(\mathbf{A})} \varphi \to \varphi') = (w \models_{\Gamma(\mathbf{A})} \varphi) \to (w \models_{\Gamma(\mathbf{A})} \varphi')$
- $(w \models_{\Gamma(\mathbf{A})} \langle\pi\rangle\varphi) = \sum_{w' \in W} \left([\![\pi]\!](w, w'); (w' \models_{\Gamma(\mathbf{A})} \varphi)\right)$
- $(w \models_{\Gamma(\mathbf{A})} [\pi]\varphi) = \bigwedge_{w' \in W} \left([\![\pi]\!](w, w') \to (w' \models_{\Gamma(\mathbf{A})} \varphi)\right)$

Satisfaction

Interpretation of tests:

Classic interpretation

$$R_{\varphi?} = \{(w, w) | w \models \varphi\}$$

In this work

$$[\![\varphi?]\!](w, w') = \begin{cases} (w \models_{\Gamma(\mathbf{A})} \varphi) & \text{if } w = w' \\ 0 & \text{otherwise} \end{cases}$$

Illustration

if $x \leq 3$ **then** $x := x + 1$ **else** $y := y \times 2$

$\equiv [\![((x \leq 3)?; x := x + 1) + (((x \leq 3) \to \bot)?; y := y \times 2)]\!](w, v)$
$= [\![(x \leq 3)?; x := x + 1]\!](w, v) + [\![((x \leq 3) \to \bot)?; y := y \times 2]\!](w, v)$
$= [\![(x \leq 3)?]\!](w, w); [\![x := x + 1]\!]_0(w, v) + [\![((x \leq 3) \to \bot)?]\!](w, w); [\![y := y \times 2]\!]_0(w, v)$
$= (w \models x \leq 3); E(x := x + 1, (w, v)) + ((w \models x \leq 3) \to (w \models 0)); E(y := y \times 2, (w, v))$

$\Gamma(\mathbf{2})$ – classic programs

$w(x, 2) = \top$ and $w(x, r) = \bot$, $r \neq 2$ and $v(x, 3) = \top$ and $v(x, r) = \bot$, $r \neq 3$,
$(\top \wedge \top) \vee ((\top \to \bot) \wedge \top) = \top$

$\Gamma(\mathbf{G})$ – fuzzy programs

$max\{min\{0.3, 0.7\}, min\{0.3 \to 0,\ 0.09\}\} = 0.3$

$\Gamma(\mathbf{R})$ – resources dependent programs

$min\{3 + 7, 0 + 9\} = 9$

Outline

Dynamic Logics for Quantum Programs?

Quantum logics have a long tradition...

- (Von Neumann-Birkhoff, 36)/ (Mackey, 56)/ (Piron, 76)...
  "Logics for quantum mechanics"
  - unlike in Classic Mechanics, Quantum Mechanics requires giving up basic principles of classical proposition logic
    – **Orthocomplemented lattices**

Dynamic Logics for Quantum Programs?

Quantum logics have a long tradition...

- (Von Neumann-Birkhoff, 36)/ (Mackey, 56)/ (Piron, 76)...
  "Logics for quantum mechanics"
  - unlike in Classic Mechanics, Quantum Mechanics requires giving up basic principles of classical proposition logic – **Orthocomplemented lattices**

(Feynman,82) – seminal idea of quantum computing –

Dynamic Logics for Quantum Programs?

Quantum logics have a long tradition...

- (Von Neumann-Birkhoff, 36)/ (Mackey, 56)/ (Piron, 76)...
  "Logics for quantum mechanics"
  - unlike in Classic Mechanics, Quantum Mechanics requires giving up basic principles of classical proposition logic – **Orthocomplemented lattices**

(Feynman,82) – seminal idea of quantum computing –

- Challenge:
  logics for the specification and verification of quantum algorithms

Dynamic Logics for Quantum Programs?

Dynamic Logics are suitable to verify a wide class of computational systems

Quantum Computing is an exception?

Dynamic Logics for Quantum Programs?

Dynamic Logics are suitable to verify a wide class of computational systems

Quantum Computing is an exception?

- Hoare Logics for Quantum programs – (M. Ying, 12), (Kakutani, 09), ...

Dynamic Logics for Quantum Programs?

Dynamic Logics are suitable to verify a wide class of computational systems

Quantum Computing is an exception?

- Hoare Logics for Quantum programs – (M. Ying, 12), (Kakutani, 09), ...
- Dynamic turn in quantum logic of Baltag-Smets
  - Quantum Logic as **'Dynamic logic of Quantum Measurements and Quantum Evolutions'**
  - evolved to fit the **verification of quantum algorithms**

Baltag - Smets Quantum Dynamic Logics

Since 2004,

- LQM - logic of the quantum measurements
  - single quantum systems
- LQA - logic of quantum actions
  - unitary transformations (quantum-gates) as atomic programs
- LQP - logic of compound quantum systems
  - $\otimes$-composition of $\mathcal{H}$-subspaces and spatial modalities
- PLPQ - probabilistic quantum programs
  - probabilistic modalities
- ...

Principles of the approach

Let us fix and Hilbert space $\mathcal{H}$ and a signature (Prop, U)

- **Syntax** is the classic one – atomic actions are quantum gates (unitary transformations)
- **Quantum Kripke frame** $M = (W, S, U)$:
  - W is the one-dimensional subspaces of $\mathcal{H}$ (i.e. the rays)
  - S is a set of testable properties (i.e. st $S = S^{\perp\perp}$, for $S^{\perp} = \{t \in W | t \perp s, s \in S\}$)
  - for each $u \in U$, $R_u : W \to W$ is an unitary transformation (a quantum gate)

Principles of the approach

(Classic)Tests Vs Measurements (evolutions)

- **classic case:** $?\varphi$ means that – "φ holds in the tested state"
  - $\overline{R}_{?\varphi} = \{(w, w) | w \models \varphi\}$
  - $M, w \models [?\varphi]\psi$ **iff** $M, w \models \varphi$ **implies** $M, w \models \psi$

Principles of the approach

(Classic)Tests Vs Measurements (evolutions)

- **classic case:** $?\varphi$ means that – "φ holds in the tested state"
  - $\overline{R}_{?\varphi} = \{(w, w) | w \models \varphi\}$
  - $M, w \models [?\varphi]\psi$ **iff** $M, w \models \varphi$ **implies** $M, w \models \psi$
- **quantum case:**, $?\varphi$ means that – "φ holds after the test"
  - $\overline{R}_{?\varphi} = \{(s, t) | Proj_{\varphi}(v) = t, v \in s\}$, for $Proj_{\varphi} : \mathcal{H} \to \mathcal{H}$ is the projection onto the closed linear subspace that the set of states satisfying φ generates
  - $M, w \models [?\varphi]\psi$
    iff for all $v \in w$**,** $M, Proj_{\varphi}(v) \models \varphi$ **implies** $M, Proj_{\varphi}(v) \models \psi$

(There are more slides to put here ...)

A brief overview in Dynamic Logics

Alexandre Madeira
Mathematics Dep, U. Aveiro

February 10, 2026,
Software Foundations, MAP-i 25/26
DMat, U.Aveiro