Programming Cyber-Physical Systems With Logic

André Platzer

Carnegie Mellon University

Symposium on Principles of Programming Languages 2019 TutorialFest
http://keymaeraX.org/
Outline

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
Outline (Introduction to CPS)

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
Which control decisions are safe for aircraft collision avoidance?

Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.
CPSs Promise Transformative Impact!

Prospects: Safe & Efficient

Driver assistance
Autonomous cars

Pilot decision support
Autopilots / UAVs

Train protection
Robots near humans

Prerequisite: CPSs need to be safe

How do we make sure CPSs make the world a better place?
Can you trust a computer to control physics?

Depends on how it has been programmed
And on what will happen if it malfunctions

Rationale
Safety guarantees require analytic foundations.
A common foundational core helps all application domains.
Foundations revolutionized digital computer science & our society.
Need even stronger foundations when software reaches out into our physical world.

CPSs deserve proofs as safety evidence!

André Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL’19
Can you trust a computer to control physics?

1. Depends on how it has been programmed
2. And on what will happen if it malfunctions

Rationale

1. Safety guarantees require analytic foundations.
2. A common foundational core helps all application domains.
3. Foundations revolutionized digital computer science & our society.
4. Need even stronger foundations when software reaches out into our physical world.

CPSs deserve proofs as safety evidence!
CPSs are Multi-Dynamical Systems

**CPS Dynamics**

CPS are characterized by multiple facets of dynamical systems.

**CPS Compositions**

CPS combines multiple simple dynamical effects.

**Tame Parts**

Exploiting compositionality tames CPS complexity.

---

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 5 / 66
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[[\alpha]\varphi \Rightarrow \varphi\]

\(\alpha \neq m \land b > 0 \Rightarrow \)

\(\text{init} \rightarrow [\text{if} \ (SB \ x, m)] a := -b ; x' = v, v' = a \)

\(*\)

\(\alpha \neq m \Rightarrow \)

\(\text{post} \)

\(a\)

\(v\)

\(x\)

\(m\)

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL’19 6 / 66
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\([\alpha]\varphi \quad \alpha \quad \varphi\]

\(x \neq m\)

\(\forall t \in \mathbb{R}^+ \quad x(t) \neq m(t)\)

\(\neg \exists t \in \mathbb{R}^+ \quad x(t) = m(t)\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta x(t)}{\Delta t} > 0\)

\([\alpha]\varphi \quad \alpha \quad \varphi\]

\(\forall t \in \mathbb{R}^+ \quad x_{\text{post}}(t) \neq x_{\text{init}}(t)\)

\(\forall t \in \mathbb{R}^+ \quad x(t) \neq m(t)\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta x(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta m(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta v(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta a(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta x(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta v(t)}{\Delta t} > 0\)

\(\forall t \in \mathbb{R}^+ \quad \frac{\Delta a(t)}{\Delta t} > 0\)
Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \quad \varphi \]

\[ \Box x \neq m \rightarrow x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ a \]

\[ v \]

\[ x \]

\[ m \]
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \Rightarrow \varphi \]

\[ [x \neq m] \Rightarrow \neg [x \neq m] \]

\[ a \]

\[ v \]

\[ x \]

\[ m \]

\[ \text{Programming Cyber-Physical Systems With Logic} \]

André Platzer (CMU)
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \]

\[ \alpha \]

\[ \varphi \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x' = v, v' = a \]

\[ \text{ODE} \]
Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \]

\[ x \neq m \]

\[ [\alpha] x \neq m \]

\[ a := -b \]

\[ x' = v, v' = a \]

### ODE

- \( a \)
- \( x \)
- \( m \)

### Assignments

- \( a \) assignments
- \( x \) values
- \( m \) values

### Time

- \( t \) values

André Platzer (CMU)

Programming Cyber-Physical Systems With Logic

POPL'19 6 / 66
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \quad \alpha \quad \varphi \]

\[ [x \neq m] \quad x \neq m \quad x \neq m \]

\[
\begin{align*}
    (\text{if}(\text{SB}(x, m)) & \quad a := -b) \\
    x' &= v, v' &= a
\end{align*}
\]

\[
\begin{align*}
    \text{test} & \quad \text{assign} \\
    \text{ODE} & \quad \text{ODE}
\end{align*}
\]
Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

$$[\alpha] \varphi \rightarrow \varphi$$

$$(\text{if}(SB(x, m)) \quad a := -b) \ ; \ x' = v, v' = a$$

ODE
assign
seq.
compose
test
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \implies \varphi \]

\[
(( \text{if}(SB(x, m)) \ a := -b) ; \ x' = v, v' = a)^*
\]

seq. compose

nondet. repeat

assign

test

ODE

---

André Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL’19 6 / 66
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \quad \varphi \alpha \]

\[ [x \neq m] \quad x \neq m \quad x \neq m \]

\[ \left[ ((\text{if}(SB(x, m)) \ a := -b) \ ; \ x' = v, v' = a) \right]^* \]

all runs

\[ x \neq m \quad \text{post} \]
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ \lbrack \alpha \rbrack \varphi \rightarrow \varphi \]

\[ x \not= m \land b > 0 \rightarrow \lbrack ((\text{if}(SB(x, m)) \ a := -b) ; \ x' = v, v' = a)^* \rbrack \not= m \]

\[ \text{init} \]

\[ \text{post} \]

\[ \text{all runs} \]

Andre Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL’19  6 / 66
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \rightarrow \varphi \]

\[ x \neq m \land b > 0 \rightarrow \left[ ((? \neg SB(x, m) \cup a := -b) \land x = v, v' = a)^* \right] x \neq m \]

nondet. choice

\[
\begin{align*}
\text{init} & : x \neq m \\
\text{post} & : x \neq m
\end{align*}
\]
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ [\alpha] \varphi \]

\[ x \neq m \]

\[ (\neg \text{SB}(x, m) \cup a := -b) ; x' = v, v' = a)^* \]

\[ x \neq m \]

\[ \text{test} \]

\[ \text{nondet. choice} \]

\[ x \neq m \]

\[ \text{init} \]

\[ \text{post} \]

\[ x \neq m \]

\[ \text{init} \]

\[ \text{post} \]

\[ x \neq m \]

\[ \text{test} \]

\[ \text{nondet. choice} \]
CPS Analysis

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ \begin{align*}
\left[ \alpha \right] \varphi & \quad \alpha \quad \varphi \\
\end{align*} \]

\[ \begin{align*}
x & \not\equiv m \\
\left[ \varphi \right] x & \not\equiv m \\
\end{align*} \]

\[ x \not\equiv m \land b > 0 \implies \left[ \left( \neg \neg SB(x, m) \cup a := -b \right) ; x' = v, v' = a \right]^\ast \]

hybrid program dynamics

\[ a \]

\[ v \]

\[ m \]
Application Highlights

Obstacle Avoidance + Ground Navigation

Airborne Collision Avoidance (ACAS X)

Train Control Brakes

Ship Cooling

\[ x' = f(x) \]

\[ y' = g(x, y) \]

\[ \text{inv} \]

Andre Platzer (CMU)

Programming Cyber-Physical Systems With Logic

POPL’19 7 / 66
CPS are Multi-Dynamical Systems

CPS Programs
- Syntax
- Semantics
- Examples

Differential Dynamic Logic
- Syntax
- Semantics
- Example: Car Control Design

Dynamic Axioms for Dynamical Systems
- Axiomatics
- Safe CPS Programming & Proving in KeYmaera X

Differential Invariants for Differential Equations

Applications

Verified Compilation of CPS Programs

Summary
Playing with Acceleration and Braking

Example (Speedy the point)

\( \{ x' = v, v' = a \} \)

Purely continuous dynamics

What about the cyber?

\[ a = a + 1; \]
\[
\begin{align*}
x' &= v, \\
v' &= a
\end{align*}
\]
Example (Speedy the point)

\[ a := a + 1 \]

Purely discrete dynamics

How do both meet?
Example (Speedy the point)

\[ a := a + 1; \{ x' = v, v' = a \} \]

Hybrid dynamics, i.e., composition of continuous and discrete dynamics
Here: sequential composition first;second
Example (Speedy the point)

\[
\begin{align*}
    a &:= -2; \quad \{ x' = v, v' = a \}; \\
    a &:= 0.25; \quad \{ x' = v, v' = a \}; \\
    a &:= -2; \quad \{ x' = v, v' = a \}; \\
    a &:= 0.25; \quad \{ x' = v, v' = a \}; \\
    a &:= -2; \quad \{ x' = v, v' = a \}; \\
    a &:= 0.25; \quad \{ x' = v, v' = a \};
\end{align*}
\]
Example (Speedy the point)

\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]
\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]
\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]

André Platzer (CMU)  Programming Cyber-Physical Systems With Logic  POPL’19 8 / 66
Playing with Acceleration and Braking

Example (Speedy the point)

\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]
\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]
\[ a := -2; \{ x' = v, v' = a \}; \]
\[ a := 0.25; \{ x' = v, v' = a \}; \]

How to check conditions before actions?
Playing with Acceleration and Braking

Example (Speedy the point)

\[
\begin{align*}
\text{if}(v < 4) & \quad a := a + 1 \quad \text{else} \quad a := -b; \\
\{x' = v, v' = a\}
\end{align*}
\]

Velocity-dependent control
Example (Speedy the point)

\[
\text{if}(x - m > s) \ a := a + 1 \ \text{else} \ a := -b; \\
\{ x' = v, \ v' = a \}
\]

Distance-dependent control for obstacle \( m \)
Example (Speedy the point)

\[
\text{if} (x - m > s \land v < 4) \ a := a + 1 \ \text{else} \ a := -b; \\
\{x' = v, v' = a\}
\]

Velocity and distance-dependent control

**Iterative Design**

Start as simple as possible, then add challenges once basics are correct.
Playing with Acceleration and Braking

Example (Speedy the point)

\[
\text{if}(x - m > s \land v < 4 \land \text{efficiency}) \ a := a + 1 \ \text{else} \ a := -b;
\]
\[
\{x' = v, \ v' = a\}
\]

Also only accelerate if it’s efficient to do so
Example (Speedy the point)

if \((x - m > s \land v < 4 \land \text{efficiency})\) \(a := a + 1\) else \(a := -b\);

\(\{x' = v, v' = a\}\)

Exact models are unnecessarily complex. Not all features are safety-critical.
### Example (Speedy the point)

\[
(a := a + 1 \cup a := -b); \\
\{x' = v, v' = a\}
\]

Nondeterministic choice \( \cup \) allows either side to be run, arbitrarily.

**Power of Abstraction**

Only include relevant aspects, elide irrelevant detail. The model and its analysis become simpler. And apply to more systems.
Playing with Acceleration and Braking

Example (Speedy the point)

\[(a := a + 1 \cup a := -b); \quad \{x' = v, v' = a\}\]

Nondeterministic choice \(\cup\) allows either side to be run, arbitrarily.

Oops, now it got too simple! Not every choice is always acceptable.
Example (Speedy the point)

\[(?v < 4; a := a + 1 \cup a := -b);\]
\[\{x' = v, v' = a\}\]

Test \(?Q\) checks if formula \(Q\) is true in current state
Example (Speedy the point)

\[
(?v < 4; a := a + 1 \cup a := -b); \\
\{x' = v, v' = a\}
\]

Test \(Q\) checks if formula \(Q\) is true in current state, otherwise run fails.

Discarding failed runs and backtracking

System runs that fail tests are discarded and not considered further.

\[
?v < 4; \ v := \ v + 1 \quad \text{only runs if} \\
v := \ v + 1; \ ?v < 4 \quad \text{only runs if}
\]

Broader significance of nondeterminism

Nondeterminism is a tool for abstraction to focus on critical aspects. Nondeterminism is essential to describe imperfectly known environment.
Playing with Acceleration and Braking

Example (Speedy the point)

\((?v < 4; a := a + 1 \cup a := -b); \{x' = v, v' = a\}\)

Test \(\mathcal{Q}\) checks if formula \(Q\) is true in current state, otherwise run fails.

**Discarding failed runs and backtracking**

System runs that fail tests are discarded and not considered further.

\(?v < 4; v := v + 1\) only runs if \(v < 4\) initially true
\(v := v + 1; ?v < 4\) only runs if \(v < 3\) initially true

**Broader significance of nondeterminism**

Nondeterminism is a tool for abstraction to focus on critical aspects. Nondeterminism is essential to describe imperfectly known environment.
Example (Speedy the point)

\[
\begin{align*}
(?v < 4; &a := a + 1 \cup a := -b); \\
\{x' = v, v' = a\}; \\
(?v < 4; &a := a + 1 \cup a := -b); \\
\{x' = v, v' = a\}; \\
(?v < 4; &a := a + 1 \cup a := -b); \\
\{x' = v, v' = a\}
\end{align*}
\]

Repeated control needs longer programs, e.g., by copy&paste
Example (Speedy the point)

\[
((?v < 4; a := a + 1 \lor a := -b); \\
\{x' = v, v' = a\})^* 
\]

Nondeterministic repetition * repeats any arbitrary number of times
Definition (Syntax of hybrid program $\alpha$)

$$ \alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) \land Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^* $$
Definition (Syntax of hybrid program $\alpha$)

$$\alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$

Definition (Syntax of hybrid program $\alpha$)

$$\alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$


Like regular expressions. Everything nondeterministic
Hybrid Programs: Semantics

\[ \omega \xrightarrow{x := e} \nu \]

\[ \omega \xrightarrow{x' = f(x) \& Q} \nu \]

\[ \omega \xrightarrow{?Q} \nu \]
Hybrid Programs: Semantics

\[ x := e \]
\[ x' = f(x) \& Q \]

\[ \text{if } \nu(x) = \omega[e] \]
\[ \omega \text{ and } \nu(z) = \omega(z) \text{ for } z \neq x \]

\[ t \]

\[ \text{no change if } \omega \in [Q] \]
\[ \text{otherwise no transition} \]
Hybrid Programs: Semantics

1. $x := e$

2. $x' = f(x) \& Q$

3. $?Q$

---

$v$ if $v(x) = \omega[e]$ and $v(z) = \omega(z)$ for $z \neq x$

no transition otherwise

$t$

$q$

$x'$ if $\omega \in [Q]$

no change if $\omega \not\in [Q]$

$0$

$r$

André Platzer (CMU)
Hybrid Programs: Semantics

\[ x := e \]

\[ \omega \xrightarrow{x := e} \nu \]

\[ x' = f(x) \land Q \]

\[ \omega \xrightarrow{x' = f(x) \land Q} \nu \]

\[ \omega \xrightarrow{?Q} \nu \]

\[ \nu \text{ if } \nu(x) = \omega[e] \]

\[ \omega \text{ and } \nu(z) = \omega(z) \text{ for } z \neq x \]

\[ ?Q \text{ if } \omega \in [Q] \]

\[ \omega \notin [Q] \]

\[ x' = f(x) \land Q \text{ otherwise} \]
Hybrid Programs: Semantics

\[ x := e \]

\[ \nu(x) = \omega[e] \]

\[ \omega \text{ and } \nu(z) = \omega(z) \text{ for } z \neq x \]

\[ x' = f(x) \& Q \]

\[ Q \]

\[ Q \]

\[ ?Q \]

\[ \omega \]

\[ \nu \]

\[ t \]

\[ x \]

\[ f(x) \]

\[ r \]

\[ 0 \]

\[ x' = f(x) \& Q \]
Hybrid Programs: Semantics

\[ x := e \]

\[ \nu \]

\[ x' = f(x) \& Q \]

\[ \omega \]

\[ \nu \]

\[ \omega \]

\[ ?Q \]

\[ \nu \]

\[ \nu \] if \( \nu(x) = \omega[e] \)

\[ \omega \] and \( \nu(z) = \omega(z) \) for \( z \neq x \)

\[ x' = f(x) \& Q \]

\[ \omega \]

[Diagram of hybrid program semantics]
Hybrid Programs: Semantics

\[ x := e \]

\[ x' = f(x) \land Q \]

\[ ?Q \]

\[ \text{if } \omega \in \{Q\} \]

\[ \nu \text{ if } \nu(x) = \omega[e] \]

\[ \omega \text{ and } \nu(z) = \omega(z) \text{ for } z \neq x \]

\[ r \]

\[ x' = f(x) \land Q \]

\[ \omega \text{ no change if } \omega \in \{Q\} \]
Hybrid Programs: Semantics

\[ x := e \]

\[ x' = f(x) \& Q \]

?Q

\[ \text{if } \omega \in \llbracket Q \rrbracket \]

\[ \text{if } \omega \not\in \llbracket Q \rrbracket \]

\[ \nu \text{ if } \nu(x) = \omega[e] \]

\[ \omega \text{ and } \nu(z) = \omega(z) \text{ for } z \neq x \]

\[ t = 0 \]

\[ t = r \]

\[ x' = f(x) \& Q \]

\[ \text{no change if } \omega \in \llbracket Q \rrbracket \]

\[ \text{otherwise no transition} \]
Hybrid Programs: Semantics

\[ \omega \rightarrow^{\alpha} \nu_1 \]

\[ \beta \rightarrow^{\alpha \cup \beta} \nu_2 \]

\[ \omega \rightarrow^{\alpha; \beta} \nu \]

\[ \omega \rightarrow^{\alpha^*} \nu \]
Hybrid Programs: Semantics

\begin{align*}
\omega & \xrightarrow{\alpha} \nu_1 \\
\nu_2 & \xrightarrow{\beta} \nu_1 \\
\omega & \xrightarrow{\alpha \cup \beta} \nu_2 \\
\nu & \xrightarrow{\alpha ; \beta} \nu_2 \\
\omega & \xrightarrow{\alpha^*} \nu \\
\nu & \xrightarrow{\beta} \nu
\end{align*}
Hybrid Programs: Semantics

\begin{align*}
\omega & \xrightarrow{\alpha} \nu_1 \\
\omega & \xrightarrow{\alpha \cup \beta} \nu_2 \\
\omega & \xrightarrow{\alpha} \mu \xrightarrow{\beta} \nu \\
\omega & \xrightarrow{\alpha^*} \nu
\end{align*}

\begin{align*}
\nu_1 & \\
\nu_2 & \\
\nu
\end{align*}
Hybrid Programs: Semantics

\[ \begin{align*}
\omega &\xrightarrow{\alpha} \nu_1 \\
\nu_1 &\xrightarrow{\alpha \cup \beta} \nu_2 \\
\nu_2 &\xrightarrow{\beta} \nu \\
\omega &\xrightarrow{\alpha} \mu \xrightarrow{\beta} \nu \\
\omega &\xrightarrow{\alpha^*} \omega_1 \xrightarrow{\alpha} \omega_2 \xrightarrow{\alpha} \nu
\end{align*} \]
Plug-in for Semantics of Composed Hybrid Programs

\[ \alpha ; \beta \]

\[ \omega \xrightarrow{\alpha} \mu \xrightarrow{\beta} \nu \]

\[ \alpha^* \]

\[ \omega \xrightarrow{\alpha} \omega_1 \xrightarrow{\alpha} \omega_2 \xrightarrow{\alpha} \nu \]

\[ \alpha \cup \beta \]

\[ \omega \xrightarrow{\alpha} \nu_1 \xrightarrow{\alpha \cup \beta} \nu_2 \]
Plug-in for Semantics of Composed Hybrid Programs

\[ \omega; \beta \]

\[ \alpha^* \]

\[ \omega \rightarrow \omega_1 \rightarrow \omega_2 \rightarrow \nu \]

\[ (\alpha; \beta)^* \]

\[ (\alpha \cup \beta)^* \]

\[ (\alpha; \beta)^* \rightarrow (\alpha \cup \beta)^* \]

\[ (\alpha \cup \beta)^* \]

\[ (\omega \cup \beta)^* \]

\[ \omega \rightarrow \omega_1 \rightarrow \omega_2 \rightarrow \nu_1 \rightarrow \nu_2 \rightarrow \nu_3 \rightarrow \nu_4 \rightarrow \nu_5 \rightarrow \nu_6 \rightarrow \nu_7 \rightarrow \nu_8 \]

\[ (\alpha \cup \beta)^* \]
Hybrid Programs: Syntax & Semantics

Definition (Syntax of hybrid program $\alpha$)

$$\alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^*$$

Definition (Semantics of hybrid programs) \((\llbracket \cdot \rrbracket : \text{HP} \rightarrow \wp(S \times S))\)

$$\llbracket x := e \rrbracket = \{ (\omega, \nu) : \nu = \omega \text{ except } \nu[x] = \omega[e] \}$$
$$\llbracket ?Q \rrbracket = \{ (\omega, \omega) : \omega \in \llbracket Q \rrbracket \}$$
$$\llbracket x' = f(x) \rrbracket = \{ (\varphi(0), \varphi(r)) : \varphi \models x' = f(x) \text{ for some duration } r \geq 0 \}$$
$$\llbracket \alpha \cup \beta \rrbracket = \llbracket \alpha \rrbracket \cup \llbracket \beta \rrbracket$$
$$\llbracket \alpha ; \beta \rrbracket = \llbracket \alpha \rrbracket \circ \llbracket \beta \rrbracket = \{ (\omega, \nu) : (\omega, \mu) \in \llbracket \alpha \rrbracket \text{ and } (\mu, \nu) \in \llbracket \beta \rrbracket \}$$
$$\llbracket \alpha^* \rrbracket = \llbracket \alpha \rrbracket^* = \bigcup_{n \in \mathbb{N}} \llbracket \alpha^n \rrbracket \quad \alpha^n \equiv \underbrace{\alpha ; \alpha ; \alpha ; \ldots ; \alpha}_{n \text{ times}}$$

compositional
Hybrid Programs: Syntax & Semantics

Definition (Syntax of hybrid program $\alpha$)

$$\alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) \land Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$

Definition (Semantics of hybrid programs) \((\llbracket \cdot \rrbracket : HP \to \wp(S \times S))\)

- \(\llbracket x := e \rrbracket = \{ (\omega, \nu) : \nu = \omega \text{ except } \nu[x] = \omega[e] \}\)
- \(\llbracket ?Q \rrbracket = \{ (\omega, \omega) : \omega \in \llbracket Q \rrbracket \}\)
- \(\llbracket x' = f(x) \rrbracket = \{ (\varphi(0), \varphi(r)) : \varphi \models x' = f(x) \text{ for some duration } r \geq 0 \}\)
- \(\llbracket \alpha \cup \beta \rrbracket = \llbracket \alpha \rrbracket \cup \llbracket \beta \rrbracket\)
- \(\llbracket \alpha; \beta \rrbracket = \llbracket \alpha \rrbracket \circ \llbracket \beta \rrbracket\)
- \(\llbracket \alpha^* \rrbracket = \llbracket \alpha \rrbracket^* = \bigcup_{n \in \mathbb{N}} \llbracket \alpha^n \rrbracket\)

1. $\varphi(z)(x') = \frac{d\varphi(t)(x)}{dt}(z)$ exists at all times $0 \leq z \leq r$
2. $\varphi(z) \in \llbracket x' = f(x) \land Q \rrbracket$ for all times $0 \leq z \leq r$
3. $\varphi(z) = \varphi(0)$ except at $x, x'$

Andre Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 13 / 66
Example (Quantum the Bouncing Ball)
Example (Quantum the Bouncing Ball)

\[ \{ x' = v, v' = -g \} \]
Example (Quantum the Bouncing Ball)

\[ \{ x' = v, v' = -g \} \]
Example (Quantum the Bouncing Ball)

\[\{x' = v, v' = -g & x \geq 0\}\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{ x' = v, \ v' = -g & \land x \geq 0 \}; \\
\text{if}(x = 0) \quad & v := -cv
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\{(x' = v, v' = -g \& x \geq 0); \\
\text{if}(x = 0) \ v := -cv\}^*
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) \ v := -cv
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{ & x' = v, \ v' = -g & \& x \geq 0 \}; \\
\text{if} (x = 0) \ & v := -cv \}
\end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\{x' = v, v' = -g & x \geq 0\};
\]

\[
\text{if}(x = 0) \quad v := -cv
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{ & x' = v, \ v' = -g \ & \land \ & x \geq 0 \}; \\
& \text{if}(x = 0) \ & v := -cv \end{align*}
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{x' &= v, \ v' = -g & x \geq 0\}; \\
\text{if}(x = 0) \ (v := -cv \cup v := 0) & \ 
\end{align*}
\]

\[
\text{if}(Q) \ \alpha \ \text{else} \ \beta \equiv (Q; \alpha) \cup (\lnot Q; \beta)
\]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\{x' &= v, \ v' = -g & & \text{if } x \geq 0 \\
\text{if}(x = 0) \ (c := \ast; \ ?c \geq 0; \ v := -cv)\}^* 
\end{align*}
\]
Runaround Robot with Dubins Paths

\[(x, y) \neq o \rightarrow \begin{cases} \omega := -1 \\ \omega := 1 \\ \omega := 0 \end{cases}; \begin{cases} x' = v \\ y' = w \\ v' = \omega w \\ w' = -\omega v \end{cases}\]
Example (Runaround Robot)

\[
((\omega := -1 \cup \omega := 1 \cup \omega := 0);
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
\left( (?Q_{-1}; \omega := -1 \cup ?Q_{1}; \omega := 1 \cup ?Q_{0}; \omega := 0); \right.
\left. \{ x' = v, y' = w, v' = \omega w, w' = -\omega v \} \right)^* 
\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[ (((\neg Q_1; \omega := -1 \cup Q_1; \omega := 1 \cup Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^* \]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
( (?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{ x' = v, y' = w, v' = \omega w, w' = -\omega v \})^*
\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_{1}; \omega := 1 \cup ?Q_{0}; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
(\text{(?Q}_{-1}; \omega := -1 \cup ?Q_{1}; \omega := 1 \cup ?Q_{0}; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^\ast
\]
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
(\langle ?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0 \rangle; \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v \})^*
\]
Example (Runarround Robot)

\[
\left( (\neg Q_{-1}; \omega := -1 \cup Q_1; \omega := 1 \cup Q_0; \omega := 0) \right)^* \ 
\{ x' = v, y' = w, v' = \omega w, w' = -\omega v \}
\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
\left( (\omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\} \right)^*\]
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
(((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^* \]

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 15 / 66
Example (Runaround Robot)

\[
((?Q_{-1}; \omega := -1 \cup ?Q_{1}; \omega := 1 \cup ?Q_{0}; \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[
(\{ ?Q_1; \omega := -1 \lor \omega := 1 \lor \omega := 0 \};
\{ x' = v, y' = w, v' = \omega w, w' = -\omega v \})^*
\]
Outline (Specifying CPS)

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
CPS Analysis

Concept (Differential Dynamic Logic) (JAR'08, LICS'12)

\[ [\alpha] \varphi \rightarrow \varphi \]

\[ x \neq m \land b > 0 \rightarrow \left( ((if(SB(x, m)) \ a := -b) ; x' = v, v' = a)^* \right) x \neq m \]

\[ \forall \text{ all runs} \]

---

Andre Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL'19 16 / 66
Example (Quantum the Bouncing Ball)

\[ \{x' = v, v' = -g \& x \geq 0\}; \]
\[ \text{if}(x = 0) \ v := -cv \]
Example (Quantum the Bouncing Ball)

ensures\((0 \leq x)\)

\[
\{x' = v, v' = -g \& x \geq 0\};
\]

if\((x = 0)\) \(v := -cv\)\(^*\)
Example (Quantum the Bouncing Ball)

\[
\text{ensures}(0 \leq x) \\
\text{ensures}(x \leq H) \\
\text{ensures}(x \geq 0) \\
\{x' = v, \ v' = -g \land x \geq 0\}; \\
\text{if}(x = 0) \ v := -cv)
\]
Example (Quantum the Bouncing Ball)

\[ \text{requires}(x = H) \]

\[ \text{ensures}(0 \leq x) \]

\[ \text{ensures}(x \leq H) \]

\[ \{ x' = v, v' = -g \& x \geq 0 \}; \]

\[ \text{if}(x = 0) v := -cv \]
Example (Quantum the Bouncing Ball)

\begin{align*}
\text{requires} & (x = H) \\
\text{requires} & (0 \leq H) \\
\text{ensures} & (0 \leq x) \\
\text{ensures} & (x \leq H) \\
\{x' = v, v' = -g & \land x \geq 0\};
\text{if}(x = 0) v := -cv
\end{align*}
Example (Quantum the Bouncing Ball)

\[ \text{requires} (x = H) \]
\[ \text{requires} (0 \leq H) \]
\[ \text{ensures} (0 \leq x) \]
\[ \text{ensures} (x \leq H) \]
\[ \{ x' = v, v' = -g \& x \geq 0 \}; \]
\[ \text{if} (x = 0) v := -c v \] *\@invariant* (x \geq 0)
Example (Quantum the Bouncing Ball)

\[
\text{requires}(x = H)
\]

\[
\text{requires}(0 \leq H)
\]

\[
\text{ensures}(0 \leq x)
\]

\[
\text{ensures}(x \leq H)
\]

\[
\left\{ x' = v, v' = -g \land x \geq 0 \right\}; \quad \text{if}(x = 0) \ v := -cv \]

\[
\text{@invariant}(x \geq 0)
\]
CPS contracts are crucial for CPS safety. We need to understand CPS programs and contracts and how we can convince ourselves that a CPS program respects its contract.

Contracts are at a disadvantage compared to full logic.

**Logic is for Specification and Reasoning**

1. Specification of a whole CPS program.
2. Analytic inspection of its parts.
3. Argumentative relations between contracts and program parts. “Yes, this CPS program meets its contract, and here’s why …”
Example (Quantum the Bouncing Ball)

\begin{align*}
\text{requires} & (x = H) \\
\text{requires} & (0 \leq H) \\
\text{ensures} & (0 \leq x) \\
\text{ensures} & (x \leq H) \\
\{ x' = v, \ v' = -g &\land x \geq 0 \} &; \\
\text{if} (x = 0) \ v := -cv \end{align*}
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
\text{requires}(x = H) \\
\text{requires}(0 \leq H) \\
\text{ensures}(0 \leq x) \\
\text{ensures}(x \leq H)
\end{align*}
\]

\[
\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) v := -cv^*
\end{align*}
\]

Precondition:

\[x = H \land 0 \leq H\] in FOL
Example (Quantum the Bouncing Ball)

\begin{align*}
\text{requires}(x = H) & \quad \text{Precondition:} \quad x = H \land 0 \leq H \text{ in FOL} \\
\text{requires}(0 \leq H) & \\
\text{ensures}(0 \leq x) & \quad \text{Postcondition:} \quad 0 \leq x \land x \leq H \text{ in FOL} \\
\text{ensures}(x \leq H) & \\
\{x' = v, v' = -g \& x \geq 0\} & \\
\text{if}(x = 0) v := -cv & 
\end{align*}
Example (Quantum the Bouncing Ball)

- \textbf{requires}(x = H)
- \textbf{requires}(0 \leq H)
- \textbf{ensures}(0 \leq x)
- \textbf{ensures}(x \leq H)

\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) v := -cv
\end{align*}

Precondition:
\(x = H \land 0 \leq H\) in FOL

Postcondition:
\(0 \leq x \land x \leq H\) in FOL

How to say post is true after all HP runs?
Example (Quantum the Bouncing Ball)

\begin{align*}
\text{requires}(&x = H) \\
\text{requires}(0 &\leq H) \\
\text{ensures}(0 &\leq x) \\
\text{ensures}(x &\leq H) \\
&\{x' = v, v' = -g & x \geq 0\}; \\
&\text{if}(x = 0) v := -cv
\end{align*}

- **Precondition:** \(x = H \land 0 \leq H\) in FOL
- **Postcondition:** \(0 \leq x \land x \leq H\) in FOL

\[\square x \leq H\]
Example (Quantum the Bouncing Ball)

- \text{requires}(x = H)
- \text{requires}(0 \leq H)
- \text{ensures}(0 \leq x)
- \text{ensures}(x \leq H)

\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\}; \\
\text{if}(x = 0) v := -cv^{*}
\end{align*}

Precondition:
\(x = H \land 0 \leq H\) in FOL

Postcondition:
\(0 \leq x \land x \leq H\) in FOL
Example (Quantum the Bouncing Ball)

requires \( x = H \)

requires \( 0 \leq H \)

ensures \( 0 \leq x \)

ensures \( x \leq H \)

\( \{ x' = v, v' = -g \& x \geq 0 \}; \)

if \( x = 0 \) \( v := -cv \)

Precondition:

\( x = H \& 0 \leq H \) in FOL

Postcondition:

\( 0 \leq x \land x \leq H \) in FOL
Example (Quantum the Bouncing Ball)

- `requires(x = H)`
- `requires(0 ≤ H)`
- `ensures(0 ≤ x)`
- `ensures(x ≤ H)`

\[
\{ x' = v, \quad v' = -g \land x \geq 0 \}; \\
\text{if}(x = 0) \ v := -cv
\]

Precondition:

\[ x = H \land 0 \leq H \text{ in FOL} \]

Postcondition:

\[ 0 \leq x \land x \leq H \text{ in FOL} \]
**Contracts for Quantum the Acrophobic Bouncing Ball**

\[
[\left(\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x = 0) v := -cv\right)^*]
\]

---

**Example (Quantum the Bouncing Ball)**

<table>
<thead>
<tr>
<th>requires (x = H)</th>
<th>Precondition: (x = H \land 0 \leq H) in FOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>requires (0 \leq H)</td>
<td>Postcondition: (0 \leq x \land x \leq H) in FOL</td>
</tr>
<tr>
<td>ensures (0 \leq x)</td>
<td></td>
</tr>
<tr>
<td>ensures (x \leq H)</td>
<td></td>
</tr>
<tr>
<td>({x' = v, v' = -g &amp; x \geq 0}; \text{if}(x = 0) v := -cv\right)^*</td>
<td></td>
</tr>
</tbody>
</table>
contracts for Quantum the Acrophobic Bouncing Ball

Example (Quantum the Bouncing Ball)

\[
[((\{x' = v, v' = -g \& x \geq 0\}; if(x=0) v := -cv)^*](x \leq H)
\]

Precondition:
\[x = H \land 0 \leq H\] in FOL

Postcondition:
\[0 \leq x \land x \leq H\] in FOL

- requires\(x = H\)
- requires\(0 \leq H\)
- ensures\(0 \leq x\)
- ensures\(x \leq H\)

\((\{x' = v, v' = -g \& x \geq 0\}; if(x = 0) v := -cv)^*\)
Example (Quantum the Bouncing Ball)

Available contracts for Quantum the Acrophobic Bouncing Ball

\[ ((\{x' = \nu, \nu' = -g & x \geq 0\}; \text{if}(x = 0, \nu := -c\nu))^*](0 \leq x) \\
[(((x' = \nu, \nu' = -g & x \geq 0); \text{if}(x = 0, \nu := -c\nu))^*](x \leq H) \\

Precondition: 
\[ x = H \land 0 \leq H \in \text{FOL} \]

Postcondition: 
\[ 0 \leq x \land x \leq H \in \text{FOL} \]
Example (Quantum the Bouncing Ball)

\[
\begin{align*}
&\text{requires}(x = H) \\
&\text{requires}(0 \leq H) \\
&\text{ensures}(0 \leq x) \\
&\text{ensures}(x \leq H) \\
&\{x' = v, v' = -g \land x \geq 0\}; \text{if}(x = 0) v := -cv)^* \\
&\text{Precondition:} \\
&\text{Postcondition:}
\end{align*}
\]

\[
\begin{align*}
&(0 \leq x) \\
&(x \leq H) \\
&(0 \leq x \land x \leq H) \in \text{FOL}
\end{align*}
\]
Example (Quantum the Bouncing Ball)

- **requires**($x = H$)
- **requires**($0 \leq H$)
- **ensures**($0 \leq x$)
- **ensures**($x \leq H$)

\[
\begin{align*}
\{x' = v, v' = -g & \land x \geq 0\} ; \text{if}(x=0) \ v := -cv^* \} (0 \leq x) \\
\wedge \{x' = v, v' = -g & \land x \geq 0\} ; \text{if}(x=0) \ v := -cv^* \} (x \leq H) \\
\leftrightarrow \{x' = v, v' = -g & \land x \geq 0\} ; \text{if}(x=0) \ v := -cv^* \} (0 \leq x \land x \leq H)
\end{align*}
\]

Precondition: $x = H \land 0 \leq H$ in FOL

Postcondition: $0 \leq x \land x \leq H$ in FOL
\[
[(\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x=0) \ v := -cv)^*](0 \leq x)
\]

Example (Quantum the Bouncing Ball)

- **requires** \(x = H\)
- **requires** \(0 \leq H\)
- **ensures** \(0 \leq x\)
- **ensures** \(x \leq H\)

\(\{x' = v, v' = -g \& x \geq 0\}; \text{if}(x = 0) \ v := -cv)^*\)

**Precondition:**
\[x = H \land 0 \leq H\] in FOL

**Postcondition:**
\[0 \leq x \land x \leq H\] in FOL
Example (Quantum the Bouncing Ball)

- \textbf{requires}(x = H)
- \textbf{requires}(0 \leq H)
- \textbf{ensures}(0 \leq x)
- \textbf{ensures}(x \leq H)

\[
(x = H \rightarrow \left[ (\{x' = v, v' = -g \land x \geq 0\}; \text{if}(x = 0) v := -cv)^* \right](0 \leq x))
\]

Precondition:
\(x = H \land 0 \leq H\) in FOL

Postcondition:
\(0 \leq x \land x \leq H\) in FOL
Example (Quantum the Bouncing Ball)

- \textbf{requires} (x = H)
- \textbf{requires} (0 \leq H)
- \textbf{ensures} (0 \leq x)
- \textbf{ensures} (x \leq H)

\[
\begin{align*}
\{(x' = v, v' = -g & x \geq 0); \text{if}(x=0) v := -cv\}^* & (0 \leq x)
\end{align*}
\]

Precondition:

\(x = H \land 0 \leq H\) in FOL

Postcondition:

\(0 \leq x \land x \leq H\) in FOL
Differential Dynamic Logic: Syntax

Definition (Syntax of differential dynamic logic)
The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P \]

\[ \]
Definition (Syntax of differential dynamic logic)

The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x \ P \mid \exists x \ P \mid [\alpha]P \mid \langle \alpha \rangle P \]

- **Not**
- **And**
- **Or**
- **Imply**
- **All reals**
- **Some real**
- **All runs**
- **Some runs**
Definition (dL Formulas)

\[ [\alpha]P \Rightarrow P \]

\[ \omega \]

\[ P \rightarrow P \rightarrow P \]

\[ [\alpha]P \rightarrow P \rightarrow P \rightarrow P \]

\[ \Rightarrow \]

\[ P \rightarrow P \rightarrow P \]

Differential Dynamic Logic dL: Semantics
Definition (dL Formulas)

\[ \langle \alpha \rangle P \]

\[ \omega \]

\[ P \]
Definition (dL Formulas)

\[ \omega \]
Definition (dL Formulas)

\[ [\alpha]P \]

\[ \langle \beta \rangle P \]

\[ \omega \]

\[ \alpha \text{-span} \]

\[ \beta \text{-span} \]
Definition (dL Formulas)
Definition (Syntax of differential dynamic logic)

The formulas of differential dynamic logic are defined by the grammar:

\[ P, Q ::= e \geq \bar{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle \alpha \rangle P \]

Definition (dL semantics)

\[
\begin{align*}
[e \geq \bar{e}] &= \{ \omega : \omega[e] \geq \omega[\bar{e}] \} \\
[\neg P] &= [P]^{c} = S \setminus [P] \\
[P \land Q] &= [P] \cap [Q] \\
[P \lor Q] &= [P] \cup [Q] \\
[P \rightarrow Q] &= [P]^{c} \cup [Q] \\
[\langle \alpha \rangle P] &= [\alpha] \circ [P] = \{ \omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha] \} \\
[[\alpha]P] &= [\neg \langle \alpha \rangle \neg P] = \{ \omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha] \} \\
[\exists x P] &= \{ \omega : \omega_{x}^{r} \in [P] \text{ for some } r \in \mathbb{R} \} \\
[\forall x P] &= \{ \omega : \omega_{x}^{r} \in [P] \text{ for all } r \in \mathbb{R} \} \\
\omega_{x}^{d}(y) &= \begin{cases} 
  d & \text{if } y = x \\
  \omega(y) & \text{if } y \neq x
\end{cases}
\end{align*}
\]
Differential Dynamic Logic: Syntax & Semantics

$[P]$ the set of states in which formula $P$ is true

$\omega \in [P]$ formula $P$ is true in state $\omega$, alias $\omega \models P$

$\models P$ formula $P$ is valid, i.e., true in all states $\omega$, i.e., $[P] = S$

**Definition (dL semantics)** $(\llbracket \cdot \rrbracket : \text{Fml} \to \wp(S))$

\[
\begin{align*}
\llbracket e \geq \bar{e} \rrbracket &= \{ \omega : \omega[e] \geq \omega[\bar{e}] \} \\
\llbracket \neg P \rrbracket &= \llbracket P \rrbracket^c = S \setminus \llbracket P \rrbracket \\
\llbracket P \land Q \rrbracket &= \llbracket P \rrbracket \cap \llbracket Q \rrbracket \\
\llbracket P \lor Q \rrbracket &= \llbracket P \rrbracket \cup \llbracket Q \rrbracket \\
\llbracket P \rightarrow Q \rrbracket &= \llbracket P \rrbracket^c \cup \llbracket Q \rrbracket \\
\llbracket \langle \alpha \rangle P \rrbracket &= \langle \alpha \rangle \circ \llbracket P \rrbracket = \{ \omega : \nu \in \llbracket P \rrbracket \text{ for some } \nu : (\omega, \nu) \in \llbracket \alpha \rrbracket \} \\
\llbracket [\alpha] P \rrbracket &= \llbracket \neg \langle \alpha \rangle \neg P \rrbracket = \{ \omega : \nu \in \llbracket P \rrbracket \text{ for all } \nu : (\omega, \nu) \in \llbracket \alpha \rrbracket \} \\
\llbracket \exists x P \rrbracket &= \{ \omega : \omega^r_x \in \llbracket P \rrbracket \text{ for some } r \in \mathbb{R} \} \\
\llbracket \forall x P \rrbracket &= \{ \omega : \omega^r_x \in \llbracket P \rrbracket \text{ for all } r \in \mathbb{R} \}
\end{align*}
\]
\([P]\) the set of states in which formula \(P\) is true

\(\omega \in [P]\) formula \(P\) is true in state \(\omega\), alias \(\omega \models P\)

\(\models P\) formula \(P\) is valid, i.e., true in all states \(\omega\), i.e., \([P]\) = \(S\)

\(\exists d \left[ x := 1; x' = d \right] x \geq 0\) and \(\left[ x := x + 1; x' = d \right] x \geq 0\) and \(\left< x' = d \right> x \geq 0\)

**Definition (dL semantics)**

\([e \geq \bar{e}]\) = \(\{ \omega : \omega[e] \geq \omega[\bar{e}] \}\)

\([\neg P]\) = \([P]^c = S \setminus [P]\)

\([P \land Q]\) = \([P] \cap [Q]\)

\([P \lor Q]\) = \([P] \cup [Q]\)

\([P \rightarrow Q]\) = \([P]^c \cup [Q]\)

\([\langle \alpha \rangle P]\) = \([\alpha] \circ [P] = \{ \omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha] \}\)

\([[[\alpha]P]\) = \([\neg \langle \alpha \rangle \neg P]\) = \(\{ \omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha] \}\)

\(\exists x P\) = \(\{ \omega : \omega^r_x \in [P] \text{ for some } r \in \mathbb{R} \}\)

\(\forall x P\) = \(\{ \omega : \omega^r_x \in [P] \text{ for all } r \in \mathbb{R} \}\)
Differential Dynamic Logic: Syntax & Semantics

\([P]\) the set of states in which formula \(P\) is true

\(\omega \in [P]\) formula \(P\) is true in state \(\omega\), alias \(\omega \models P\)

\(\models P\) formula \(P\) is valid, i.e., true in all states \(\omega\), i.e., \([P] = S\)

\(\models \exists d [x := 1; x' = d] x \geq 0\) and \(\not \models [x := x + 1; x' = d] x \geq 0\) and \(\not \models \langle x' = d \rangle x \geq 0\)

### Definition (dL semantics) \((\llbracket \cdot \rrbracket : \text{Fml} \to \wp(S))\)

\([e \geq \tilde{e}] = \{\omega : \omega[e] \geq \omega[\tilde{e}]\}\)

\([-P] = [P]^C = S \setminus [P]\)

\([P \land Q] = [P] \cap [Q]\)

\([P \lor Q] = [P] \cup [Q]\)

\([P \to Q] = [P]^C \cup [Q]\)

\(\langle \alpha \rangle P = \alpha \circ [P] = \{\omega : \nu \in [P] \text{ for some } \nu : (\omega, \nu) \in [\alpha]\}\)

\(\lbrack [\alpha] P \rbrack = \langle \neg \alpha \rangle \neg P = \{\omega : \nu \in [P] \text{ for all } \nu : (\omega, \nu) \in [\alpha]\}\)

\(\exists x P = \{\omega : \omega_x^r \in [P] \text{ for some } r \in \mathbb{R}\}\)

\(\forall x P = \{\omega : \omega_x^r \in [P] \text{ for all } r \in \mathbb{R}\}\)
Example (Bouncing Ball)

\[
\begin{align*}
\{ x' &= v, v' = -g \& x \geq 0 \}; \\
\text{if}(x = 0) \ v &:= -cv)
\end{align*}
\]
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\[ H = x \geq 0 \quad \rightarrow \quad \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \quad \text{if}(x = 0) \quad v := -cv \right]^* \]
\[ 0 \leq x \leq H \]
Example (Bouncing Ball)

\[ H = x \geq 0 \quad \rightarrow \quad [(\{x' = v, v' = -g \ & x \geq 0\};
\text{if}(x = 0) v := -cv)^*] \quad 0 \leq x \leq H \]
Example (Bouncing Ball)

\[ H = x \geq 0 \land g > 0 \rightarrow \left[ \left( \{ x' = v, v' = -g \land x \geq 0 \} ; \right. \right. \]
\[ \left. \left. \text{if}(x = 0) \; v := -cv \right) \right]_0 \leq x \leq H \]
Quantum the Acrophobic Bouncing Ball

Not if $c > 1$ for anti-damping

Example (Bouncing Ball)

$$H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \right. \left. \text{if}(x = 0) v := -cv \right]^* \] 0 \leq x \leq H$$
Example (Bouncing Ball)

\[ 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \left\{ x' = v, v' = -g \land x \geq 0 \right\}; \right. \\
\left. \text{if}(x = 0) v := -cv \right] \land 0 \leq x \leq H \]
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\[ 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \ & x \geq 0 \} ; \right. \]

\[ \left. \text{if} ( x = 0 ) v := -cv \right) \ast 0 \leq x \leq H \]
Example (Bouncing Ball)

\[ v \leq 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \\
\text{if}(x = 0) v := -cv \right]^* \] 
\[ 0 \leq x \leq H \]
Quantum the Acrophobic Bouncing Ball

Example (Bouncing Ball)

\[ v \leq 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, v' = -g \land x \geq 0 \}; \right. \\
\left. \text{if}(x = 0) \ v := -cv \right] ^\ast 0 \leq x \leq H \]

Not if \( v \ll 0 \) initial dribbling
Example (Bouncing Ball)

\[
v = 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \left\{ x' = v, v' = -g \land x \geq 0 \right\}; \right.
\]

\[
\text{if} (x = 0) \; v := -cv)^* \right] \; 0 \leq x \leq H
\]
Repeat control decisions

Example (Single car $car_s$)

\[
((a := A \cup a := -b); \{x' = v, v' = a\})^*
\]
How does this model brake?

Example (Single car $car_s$)

\[
((a := A \cup a := -b); \{x' = v, v' = a\})^*
\]
Velocity bound $v \geq 0$ in evolution domain

Example (Single car $cars$)

\[
((a := A \cup a := -b); \{x' = v, v' = a \& v \geq 0\})^*
\]
Ex: Car Control Programs

Acceleration not always safe

Example (Single car $car_s$)

\[
(( a := A \cup a := -b); \{ x' = v, v' = a \& v \geq 0 \})^*
\]
Ex: Car Control Programs

Acceleration condition $Q$

Example (Single car $car_s$)

$$(((Q; a := A) \cup a := -b); \{x' = v, v' = a \land v \geq 0\})^*$$
Ex: Car Control Properties

\[ Q \equiv \]

Example (Single car \( \text{car}_\varepsilon \) time-triggered)

\[ (((\text{?Q}; a := A) \cup a := -b); t := 0; \{ x' = v, v' = a, t' = 1 \& v \geq 0 \& t \leq \varepsilon \}^*) \]

Example (Safely stays before traffic light \( m \))

\[ A \geq 0 \& b > 0 \rightarrow [\text{car}_\varepsilon] x \leq m \]
Example (Single car \( \text{car}_\varepsilon \) time-triggered)

\[
((?Q; a := A) \cup a := -b); t := 0; \{ x' = v, v' = a, t' = 1 \& v \geq 0 \& t \leq \varepsilon \})^*
\]

Example (\( \square \) Safely stays before traffic light \( m \))

\[
v^2 \leq 2b(m - x) \land A \geq 0 \land b > 0 \rightarrow [\text{car}_\varepsilon] x \leq m
\]
Ex: Car Control Properties

\[ Q \equiv 2b(m - x) \geq v^2 + (A + b)(A\varepsilon^2 + 2\varepsilon v) \]

Example (Single car \( \text{car}_\varepsilon \) time-triggered)

\[ (((?Q; a := A) \cup a := -b); t := 0; \{ x' = v, v' = a, t' = 1 \& v \geq 0 \land t \leq \varepsilon})* \]

Example (\( \square \) Safely stays before traffic light \( m \))

\[ v^2 \leq 2b(m - x) \land A \geq 0 \land b > 0 \rightarrow [\text{car}_\varepsilon] x \leq m \]
Ex: Car Control Properties

\[ Q \equiv 2b(m - x) \geq v^2 + (A + b)(A\varepsilon^2 + 2\varepsilon v) \]

Example (Single car \textit{car}_\varepsilon \text{ time-triggered})

\[ (((?Q; a := A) \cup a := -b); t := 0; \{ x' = v, v' = a, t' = 1 \& v \geq 0 \land t \leq \varepsilon \})^* \]

Example (Live, can move everywhere)

\[ \varepsilon > 0 \land A > 0 \land b > 0 \rightarrow \forall p \exists m \langle \text{car}_\varepsilon \rangle x \geq p \]
Example (dL-based model-predictive control design)

\[\begin{align*}
\land v \geq 0 \land A \geq 0 \land b > 0 & \rightarrow \\
(( & \\
? & ; \\
a := A) \\
\cup a := -b); \\
t := 0; \{x' = v, v' = a, t' = 1 & v \geq 0 \land t \leq \varepsilon\})^* & \] x \leq m
\end{align*}\]
Example (dL-based model-predictive control design)

\[ \text{???, } v \geq 0 \land A \geq 0 \land b > 0 \implies \]

\[ \left[ ((
\text{? (}
\text{? (}
\text{? (}

\text{a := } A) \cup a := -b) ;

\text{t := 0; } \{ x' = v, v' = a, t' = 1 \land v \geq 0 \land t \leq \varepsilon \} \right)^* \right] x \leq m \]
Example (dL-based model-predictive control design)

\[
\begin{align*}
[x' = v, v' = -b] & \ x \leq m \ \land \ v \geq 0 \ \land \ A \geq 0 \ \land \ b > 0 \ \rightarrow \\
[(\ \\
\ ? \ \\
\ ^{a := A}) \ \\
\cup \ a := -b); \\
\ t := 0; \ \{x' = v, v' = a, t' = 1 \ \& \ v \geq 0 \ \land \ t \leq \varepsilon\}\}^* \ x \leq m
\end{align*}
\]
Example (dL-based model-predictive control design)

\[
[x' = v, v' = -b]x \leq m \land v \geq 0 \land A \geq 0 \land b > 0 \rightarrow \\
[(( ??
( ??

a := A)

\cup a := -b);

t := 0; \{x' = v, v' = a, t' = 1 \land v \geq 0 \land t \leq \varepsilon\})^*] x \leq m
\]
Example (\(dL\)-based model-predictive control design)

\[
[x' = v, v' = -b] x \leq m \land v \geq 0 \land A \geq 0 \land b > 0 \rightarrow \\
\left(\left( ??[t := 0; x' = v, v' = A, t' = 1 \land v \geq 0 \land t \leq \varepsilon] [x' = v, v' = -b] x \leq m \right) \cup a := A \right) \\
\quad \cup a := -b); \\
\quad t := 0; \{x' = v, v' = a, t' = 1 \land v \geq 0 \land t \leq \varepsilon\})^* \] x \leq m
Example (dL-based model-predictive control design)

\[ [x' = v, v' = -b] x \leq m \land v \geq 0 \land A \geq 0 \land b > 0 \rightarrow \]

\[
(((
(?[t := 0; x' = v, v' = A, t' = 1 \& v \geq 0 \land t \leq \varepsilon][x' = v, v' = -b] x \leq m ;

a := A) \cup a := -b);

[ t := 0; \{x' = v, v' = a, t' = 1 \& v \geq 0 \land t \leq \varepsilon\}^* ] x \leq m \]
Example (dL-based model-predictive control design)

\[ v^2 \leq 2b(m - x) \land v \geq 0 \land A \geq 0 \land b > 0 \rightarrow \]

\[
(\exists \begin{array}{l}
t := 0; \ x' = v, v' = A, t' = 1 \land v \geq 0 \land t \leq \varepsilon \\
a := A \end{array} \cup a := -b) ;
\]

\[ t := 0; \ \{ x' = v, v' = a, t' = 1 \land v \geq 0 \land t \leq \varepsilon \}^* \] \[ x \leq m \]
Example (dL-based model-predictive control design)

\[ v^2 \leq 2b(m-x) \land v \geq 0 \land A \geq 0 \land b > 0 \rightarrow \]

\[
\left( \left( \left( \left[ t := 0; x' = v, v' = A, t' = 1 \land v \geq 0 \land t \leq \varepsilon \right] \left[ x' = v, v' = -b \right] \right) x \leq m \right) \cup a := A \right) \cup a := -b; \\

\]

\[ t := 0; \left\{ x' = v, v' = a, t' = 1 \land v \geq 0 \land t \leq \varepsilon \right\}^{*} \right] x \leq m \]
Example (dL-based model-predictive control design)

\[
\begin{align*}
\nu^2 &\leq 2b(m - x) \land \nu \geq 0 \land A \geq 0 \land b > 0 \rightarrow \\
&\begin{cases}
\left(\left(2b(m - x) \geq \nu^2 + (A + b)(A\epsilon^2 + 2\epsilon\nu)\right); \\
a := A) \land \left(\cup a := -b\right);
\end{cases}
\end{align*}
\]

\[
t := 0; \{x' = \nu, \nu' = a, t' = 1 \land \nu \geq 0 \land t \leq \epsilon\}^* \]

\[
\begin{cases}
x \leq m
\end{cases}
\]
Runaround Robot with Dubins Paths

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 28 / 66
Runaround Robot with Dubins Paths

Example (Runaround Robot)

\[
((\omega := -1 \cup \omega := 1 \cup \omega := 0); \\
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*
\]
Example (Runaround Robot)

\[(x, y) \neq o \rightarrow \left[\left( (\omega := -1 \cup \omega := 1 \cup \omega := 0); \right.\right.\]
\[
\left.\left. \left\{ x' = v, y' = w, v' = \omega w, w' = -\omega v \right\} \right]^{*} \right) (x, y) \neq o \]
Example (Runaround Robot)

\[(x, y) \neq o \rightarrow \left[ ((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^* \right] (x, y) \neq o\]
Outline (Proving CPS)

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
Differential Dynamic Logic: Axiomatization

\[ x := e \] \( P(x) \leftrightarrow P(e) \)

\[ ? \] \( ? Q P \leftrightarrow (Q \rightarrow P) \)

\[ ' \] \( x' = f(x) \) \( P \leftrightarrow \forall t \geq 0 [x := y(t)] P \) \( (y'(t) = f(y)) \)

\[ \cup \] \( [\alpha \cup \beta] P \leftrightarrow [\alpha] P \land [\beta] P \)

\[ ; \] \( [\alpha; \beta] P \leftrightarrow [\alpha][\beta] P \)

\[ * \] \( [\alpha^*] P \leftrightarrow P \land [\alpha][\alpha^*] P \)

\( K \) \( [\alpha](P \rightarrow Q) \rightarrow ([\alpha]P \rightarrow [\alpha]Q) \)

\( I \) \( [\alpha^*] P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P) \)

\( C \) \( [\alpha^*]\forall v > 0 (P(v) \rightarrow \langle\alpha\rangle P(v-1)) \rightarrow \forall v (P(v) \rightarrow \langle\alpha^*\rangle \exists v \leq 0 P(v)) \)

LICS'12, JAR'17

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 29 / 66
Dynamic Axioms for Dynamical Systems

\[ := \quad [x := e] p(x) \leftrightarrow \]

\[ x := e \quad p(x) \quad \nu \]

\[ \omega \]

\[ x := e \quad p(x) \]

\[ \forall t \geq 0 \quad [x := y(t)] \quad p(x) \quad \nu \]

\[ x := y(t) \quad \omega \quad \nu \]

\[ \forall t \geq 0 \quad (x := y(t)) \quad p(x) \]

\[ Q \quad P \leftrightarrow (Q \to P) \]

\[ Q \quad \omega \quad \nu \]

\[ Q \quad \text{if } \omega \in [Q] \]

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL'19 30 / 66
Dynamic Axioms for Dynamical Systems

\[\begin{align*}
\text{[:=]} \quad [x := e]p(x) & \iff p(e) \\
\end{align*}\]
Dynamic Axioms for Dynamical Systems

[\[=\]] \ [x := e] p(x) \leftrightarrow p(e)

[\['\] \ [x' = f(x)] p(x) \leftrightarrow
Dynamic Axioms for Dynamical Systems

\[ \text{[:=]} \ \ [x := e]p(x) \leftrightarrow p(e) \]

\[ \text{[']} \ \ [x' = f(x)]p(x) \leftrightarrow [x := y(t)]p(x) \]
Dynamic Axioms for Dynamical Systems

[\text{:=}] \quad [x:=e]p(x) \leftrightarrow p(e)

[\text{'}] \quad [x'=f(x)]p(x) \leftrightarrow \forall t \geq 0 [x:=y(t)]p(x)
Dynamic Axioms for Dynamical Systems

\[ [\text{:=}] \ [x := e] p(x) \leftrightarrow p(e) \]

\[ ' [x' = f(x)] p(x) \leftrightarrow \forall t \geq 0 [x := y(t)] p(x) \]

\[ ' [x' = f(x) \& q(x)] p(x) \leftrightarrow \forall t \geq 0 ([x := y(t)] p(x)) \]
Dynamic Axioms for Dynamical Systems

\[ [:=] \; \chi[e]p(x) \leftrightarrow p(e) \]

\[ ['] \; [x' = f(x)]p(x) \leftrightarrow \forall t \geq 0 \; [x := y(t)]p(x) \]

\[ ['] \; [x' = f(x) \& q(x)]p(x) \leftrightarrow \forall t \geq 0 \; (\forall 0 \leq s \leq t \; q(y(s)) \rightarrow [x := y(t)]p(x)) \]
Dynamic Axioms for Dynamical Systems

\[ := \quad [x := e]p(x) \leftrightarrow p(e) \]

\['\] \[ x' = f(x) \]p(x) \leftrightarrow \forall t \geq 0 [x := y(t)]p(x) \]

\['\] \[ x' = f(x) \& q(x) \]p(x) \leftrightarrow \forall t \geq 0 (\forall 0 \leq s \leq t q(y(s)) \rightarrow [x := y(t)]p(x)) \]

\[ ? \] \[ ?Q]P \leftrightarrow \quad \text{if } \omega \in \llbracket Q \rrbracket \]
Dynamic Axioms for Dynamical Systems

\[ [\cdot] \quad [x := e]p(x) \leftrightarrow p(e) \]

\[ [\cdot'] \quad [x' = f(x)]p(x) \leftrightarrow \forall t \geq 0 [x := y(t)]p(x) \]

\[ [\cdot'] \quad [x' = f(x) \& q(x)]p(x) \leftrightarrow \forall t \geq 0 (\forall 0 \leq s \leq t q(y(s)) \rightarrow [x := y(t)]p(x)) \]

\[ [\cdot] \quad [?Q]P \leftrightarrow (Q \rightarrow P) \]
compositional semantics $\Rightarrow$ compositional proofs
\[\bigcup \quad [\alpha \cup \beta] P \iff\]

\[
\begin{array}{c}
\omega \\
\alpha \quad \alpha \cup \beta \\
\beta \quad P
\end{array}
\]

\[
\begin{array}{c}
\nu_1 \\
P
\end{array}
\]

\[
\begin{array}{c}
\nu_2 \\
P
\end{array}
\]
\[ \bigcup \ [\alpha \cup \beta] P \leftrightarrow [\alpha] P \land [\beta] P \]
Dynamic Axioms for Dynamical Systems

\[ \left[ \bigcup \right] [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ \left[ ; \right] [\alpha; \beta]P \leftrightarrow \]

\[ \omega \rightarrow \nu_1 \]
\[ \beta \]
\[ \alpha \cup \beta \]
\[ \nu_1 \rightarrow P \]

\[ \omega \rightarrow \nu_2 \]
\[ \beta \]
\[ \alpha; \beta \]
\[ \nu_2 \rightarrow P \]

\[ \omega \rightarrow \mu \]
\[ \alpha \]
\[ \mu \rightarrow \nu \]
\[ \beta \]
\[ \nu \rightarrow P \]
[∪] \([α ∪ β]P \leftrightarrow [α]P \land [β]P\)

[;] \([α; β]P \leftrightarrow [α][β]P\)
Dynamic Axioms for Dynamical Systems

\[ \cup ] \quad [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ ; ] \quad [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \]

\[ * ] \quad [\alpha^*]P \leftrightarrow \]
Dynamic Axioms for Dynamical Systems

\[ \bigcup \] \[ [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ ; \] \[ [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \]

\[ * \] \[ [\alpha^*]P \leftrightarrow P \land [\alpha][\alpha^*]P \]
Dynamic Axioms for Dynamical Systems

\[ \bigcup \ [\alpha \cup \beta]P \iff [\alpha]P \land [\beta]P \]

\[ ; \ [\alpha; \beta]P \iff [\alpha][\beta]P \]

\[ ! [\alpha^*]P \iff P \land \]

André Platzer (CMU)
Dynamic Axioms for Dynamical Systems

\[ \bigcup \] \[ \alpha \cup \beta \] P \leftrightarrow [\alpha]P \land [\beta]P

\[ ; \] \[ \alpha; \beta \] P \leftrightarrow [\alpha][\beta]P

\[ \mid \] \[ \alpha^* \] P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P)
Dynamic Axioms for Dynamical Systems

\[ \left[ \bigcup \right] \ [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ \left[ ; \right] \ [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \]

\[ \left[ \alpha^* \right]P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P) \]
[∪]  \([α ∪ β]P ⇔ [α]P ∧ [β]P\)

[;]  \([α; β]P ⇔ [α][β]P\)

[ ]  \([α^*]P ⇔ P ∧ [α^*](P → [α]P)\)
\[
\begin{align*}
\mathbf{\cup} & \quad [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \\
\mathbf{;} & \quad [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \\
\mathbf{!} & \quad [\alpha^*]P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P)
\end{align*}
\]
Dynamic Axioms for Dynamical Systems

\[ \biguplus \ [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ ; \ [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \]

\[ \mathbf{I} \ [\alpha^*]P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P) \]
Dynamic Axioms for Dynamical Systems

$$[\bigcup] \ [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P$$

$$[;] \ [\alpha ; \beta]P \leftrightarrow [\alpha][\beta]P$$

$$| \ [\alpha^*]P \leftrightarrow P \land [\alpha^*](P \to [\alpha]P)$$
[∪] \([\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P\)

[;] \([\alpha; \beta]P \leftrightarrow [\alpha][\beta]P\)

[\(\alpha^*\)]P \leftrightarrow P \land [\alpha^*](P \rightarrow [\alpha]P)
Lemma (Loop invariant rule is derived)

\[
\begin{array}{c}
\Gamma \vdash J, \Delta \\
J \vdash [\alpha] J \\
J \vdash P
\end{array} \quad \Rightarrow \\
\Gamma \vdash [\alpha^*] P, \Delta
\]

Finding invariant \( J \) can be a challenge. Misplaced \([\alpha^*]\) suggests that \( J \) needs to carry along info about \( \alpha^* \) history.
Lemma (Loop invariant rule is derived)

\[ \Gamma \vdash J, \Delta \quad J \vdash [\alpha]J \quad J \vdash P \]

\[ \Gamma \vdash [\alpha^*]P, \Delta \]

Proof (Derived rule).

\[ \Gamma \vdash J, \Delta \quad J \vdash [\alpha]J \quad J \vdash [\alpha^*]J \quad J \vdash P \]

\[ \Gamma \vdash [\alpha^*]P, \Delta \]
Proof Rule: Loop Invariants

\[
\begin{array}{c}
P \\
\hline
\alpha \vdash P \\
\end{array}
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ Quad
Example (Bouncing Ball)

$$v = 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ \{ x' = v, \ v' = -g \land x \geq 0 \}; \right.
\left. \text{if}(x = 0) \ v := -cv \right] * 0 \leq x \leq H$$
\[ A \vdash [(\text{grav}; (?x=0; v:=−cv \cup ?x≠0))^*]B(x,v) \]

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]

\[
B(x,v) \equiv 0 \leq x \land x \leq H
\]

\[
\text{grav} \equiv \{x' = v, v' = −g \land x \geq 0\}
\]
\[
A \vdash j(x, v)
\]

\[
\begin{array}{c}
  j(x, v) \vdash [\text{grav}; (?x=0; v:=-cv \cup ?x\neq 0)]j(x, v) \\
  A \vdash [(\text{grav}; (?x=0; v:=-cv \cup ?x\neq 0))^*]B(x, v)
\end{array}
\]

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]

\[
B(x, v) \equiv 0 \leq x \land x \leq H
\]

\[
\text{grav} \equiv \{x' = v, v' = -g \land x \geq 0\}
\]
\[ j(x, v) \vdash [\text{grav}]j(x, v) \]

\[ j(x, v) \vdash [?x=0; v:= -cv \cup ?x\neq 0]j(x, v) \]

\[ A \vdash j(x, v) \]

\[ A \vdash [\text{grav}; (?x=0; v:= -cv \cup ?x\neq 0)]j(x, v) \]

\[ j(x, v) \vdash [?x=0; v:= -cv \cup ?x\neq 0]j(x, v) \]

\[ A \vdash [(\text{grav}; (?x=0; v:= -cv \cup ?x\neq 0))^*]B(x, v) \]

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]

\[ B(x, v) \equiv 0 \leq x \land x \leq H \]

\[ \text{grav} \equiv \{ x' = v, v' = -g \land x \geq 0 \} \]
Proving Quantum the Acrophobic Bouncing Ball

\[
\begin{align*}
\text{MR} & \quad j(x, v), x=0 \vdash j(x, -cv) \\
\text{[?], } & \quad \text{[grav] } j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \\
\text{∧R} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v) \\
\text{[;]} & \quad j(x, v) \vdash [?x=0; \ v:= -cv] j(x, v) \land [?x \neq 0] j(x, v) \\
\text{[?]} & \quad j(x, v) \vdash [?x \neq 0] j(x, v)
\end{align*}
\]

\[
\begin{align*}
A \vdash j(x, v) \\
\text{loop} & \quad A \vdash [(\text{grav}; (\?x=0; \ v:= -cv \cup \?x \neq 0))^*] B(x, v)
\end{align*}
\]

\[
\begin{align*}
A & \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \\
B(x, v) & \equiv 0 \leq x \land x \leq H \\
\text{grav} & \equiv \{x' = v, \ v' = -g \land x \geq 0\}
\end{align*}
\]
A ⊢ j(x,v)

j(x,v) ⊢ [grav](j(x,v))

j(x,v), x=0 ⊢ j(x,(-cv))

j(x,v), x≠0 ⊢ j(x,v)

j(x,v) ⊢ B(x,v)

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]

\[
B(x,v) \equiv 0 \leq x \land x \leq H
\]

\[
\text{grav} \equiv \{x' = v, v' = -g \land x \geq 0 \}
\]
\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x,v)
\]

\[
j(x,v) \vdash [(x' = v, v' = -g \& x \geq 0)](j(x,v))
\]

\[
j(x,v), x=0 \vdash j(x,-cv)
\]

\[
j(x,v), x \neq 0 \vdash j(x,v)
\]

\[
j(x,v) \vdash 0 \leq x \land x \leq H
\]

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]

\[
B(x,v) \equiv 0 \leq x \land x \leq H
\]

\[
\text{grav} \equiv \{x' = v, v' = -g \& x \geq 0\}
\]
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \vdash j(x,v) \]

\[ j(x,v) \vdash [\{x' = v, v' = -g \wedge x \geq 0\}](j(x,v)) \]

\[ j(x,v), x = 0 \vdash j(x,(-c)v) \]

\[ j(x,v), x \neq 0 \vdash j(x,v) \]

\[ j(x,v) \vdash 0 \leq x \wedge x \leq H \]

\[ j(x,v) \equiv 0 \leq x \wedge x \leq H \]

\[ A \equiv 0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \]

\[ B(x,v) \equiv 0 \leq x \wedge x \leq H \]

\[ \text{grav} \equiv \{x' = v, v' = -g \wedge x \geq 0\} \]
0 ≤ x ∧ x = H ∧ v = 0 ∧ g > 0 ∧ 1 ≥ c ≥ 0 ⊢ j(x, ν)

\[ j(x, ν) \vdash [\{ x' = ν, \ ν' = -g \ \& \ x ≥ 0 \}](j(x, ν)) \]

\[ j(x, ν), x = 0 \vdash j(x, (-cv)) \]

\[ j(x, ν), x ≠ 0 \vdash j(x, ν) \]

\[ j(x, ν) \vdash 0 ≤ x ∧ x ≤ H \]

\[ j(x, ν) \equiv 0 ≤ x ∧ x ≤ H \]

weak: fails ODE if ν \gg 0

\[ A \equiv 0 ≤ x ∧ x = H ∧ v = 0 ∧ g > 0 ∧ 1 ≥ c ≥ 0 \]

\[ B(x, ν) \equiv 0 ≤ x ∧ x ≤ H \]

\[ \text{grav} \equiv \{ x' = ν, \ ν' = -g \ \& \ x ≥ 0 \} \]
\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x,v)
\]
\[
j(x,v) \vdash [\{x'=v, v'=-g \land x\geq0\}](j(x,v))
\]
\[
j(x,v), x=0 \vdash j(x,(cv))
\]
\[
j(x,v), x\neq0 \vdash j(x,v)
\]
\[
j(x,v) \vdash 0 \leq x \land x \leq H
\]

1. \[j(x,v) \equiv x \geq 0\]

2. \[j(x,v) \equiv 0 \leq x \land x \leq H\]  \hspace{1cm} weak: fails ODE if \( v \gg 0 \)

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]
\[
B(x,v) \equiv 0 \leq x \land x \leq H
\]
\[
grav \equiv \{x' = v, v' = -g \land x \geq 0\}
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x, v) \]

\[ j(x, v) \vdash [\{ x' = v, \ v' = -g \ \& \ x \geq 0 \}] (j(x, v)) \]

\[ j(x, v), x = 0 \vdash j(x, (-cv)) \]

\[ j(x, v), x \neq 0 \vdash j(x, v) \]

\[ j(x, v) \vdash 0 \leq x \land x \leq H \]

1. \[ j(x, v) \equiv x \geq 0 \]
   weaker: fails postcondition if \( x > H \)

2. \[ j(x, v) \equiv 0 \leq x \land x \leq H \]
   weak: fails ODE if \( v \gg 0 \)

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]

\[ B(x, v) \equiv 0 \leq x \land x \leq H \]

\[ \text{grav} \equiv \{ x' = v, \ v' = -g \ \& \ x \geq 0 \} \]
0 ≤ x ∧ x = H ∧ v = 0 ∧ g > 0 ∧ 1 ≥ c ≥ 0 ⊢ j(x,v)

j(x,v) ⊢ [(x′ = v, v′ = −g & x ≥ 0)](j(x,v))

j(x,v), x=0 ⊢ j(x,(−cv))

j(x,v), x≠0 ⊢ j(x,v)

j(x,v) ⊢ 0 ≤ x ∧ x ≤ H

1. j(x,v) ≡ x ≥ 0  weaker: fails postcondition if x > H
2. j(x,v) ≡ 0 ≤ x ∧ x ≤ H  weak: fails ODE if v ≫ 0
3. j(x,v) ≡ x = 0 ∧ v = 0

A ≡ 0 ≤ x ∧ x = H ∧ v = 0 ∧ g > 0 ∧ 1 ≥ c ≥ 0

B(x,v) ≡ 0 ≤ x ∧ x ≤ H

grav ≡ \{x′ = v, v′ = −g & x ≥ 0\}
0 ≤ x ∧ x = H ∧ v = 0 ∧ g > 0 ∧ 1 ≥ c ≥ 0 ⊢ j(x,v)

\[ j(x,v) \vdash \{ x' = v, v' = -g \land x \geq 0 \} (j(x,v)) \]

\[ j(x,v), x = 0 \vdash j(x,(-cv)) \]

\[ j(x,v), x \neq 0 \vdash j(x,v) \]

\[ j(x,v) \vdash 0 \leq x \land x \leq H \]

1. \( j(x,v) \equiv x \geq 0 \) weaker: fails postcondition if \( x > H \)
2. \( j(x,v) \equiv 0 \leq x \land x \leq H \) weak: fails ODE if \( v \gg 0 \)
3. \( j(x,v) \equiv x = 0 \land v = 0 \) strong: fails initial condition if \( x > 0 \)

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]

\[ B(x,v) \equiv 0 \leq x \land x \leq H \]

\[ \text{grav} \equiv \{ x' = v, v' = -g \land x \geq 0 \} \]
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x,v) \]

\[ j(x,v) \vdash \{ x' = v, v' = -g & x \geq 0 \} (j(x,v)) \]

\[ j(x,v), \ x = 0 \vdash j(x,(-cv)) \]

\[ j(x,v), \ x \neq 0 \vdash j(x,v) \]

\[ j(x,v) \vdash 0 \leq x \land x \leq H \]

1. \( j(x,v) \equiv x \geq 0 \)
   weaker: fails postcondition if \( x > H \)

2. \( j(x,v) \equiv 0 \leq x \land x \leq H \)
   weak: fails ODE if \( v \gg 0 \)

3. \( j(x,v) \equiv x = 0 \land v = 0 \)
   strong: fails initial condition if \( x > 0 \)

4. \( j(x,v) \equiv x = 0 \lor x = H \land v = 0 \)

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]

\[ B(x,v) \equiv 0 \leq x \land x \leq H \]

\[ \text{grav} \equiv \{ x' = v, v' = -g & x \geq 0 \} \]
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x, v) \]
\[ j(x, v) \vdash [\{x' = v, v' = -g \land x \geq 0\}](j(x, v)) \]
\[ j(x, v), x = 0 \vdash j(x, (-cv)) \]
\[ j(x, v), x \neq 0 \vdash j(x, v) \]
\[ j(x, v) \vdash 0 \leq x \land x \leq H \]

1. \( j(x, v) \equiv x \geq 0 \) weaker: fails postcondition if \( x > H \)
2. \( j(x, v) \equiv 0 \leq x \land x \leq H \) weak: fails ODE if \( v \gg 0 \)
3. \( j(x, v) \equiv x = 0 \land v = 0 \) strong: fails initial condition if \( x > 0 \)
4. \( j(x, v) \equiv x = 0 \lor x = H \land v = 0 \) no space for intermediate states

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]
\[ B(x, v) \equiv 0 \leq x \land x \leq H \]
\[ \text{grav} \equiv \{x' = v, v' = -g \land x \geq 0\} \]
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x, v) \]

\[ j(x, v) \vdash \{ x' = v, v' = -g \land x \geq 0 \} (j(x, v)) \]

\[ j(x, v), x = 0 \vdash j(x, (-cv)) \]

\[ j(x, v), x \neq 0 \vdash j(x, v) \]

\[ j(x, v) \vdash 0 \leq x \land x \leq H \]

1. \( j(x, v) \equiv x \geq 0 \) weaker: fails postcondition if \( x > H \)

2. \( j(x, v) \equiv 0 \leq x \land x \leq H \) weak: fails ODE if \( v \gg 0 \)

3. \( j(x, v) \equiv x = 0 \land v = 0 \) strong: fails initial condition if \( x > 0 \)

4. \( j(x, v) \equiv x = 0 \lor x = H \land v = 0 \) no space for intermediate states

5. \( j(x, v) \equiv 2gx = 2gH - v^2 \land x \geq 0 \)

\[ A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \]

\[ B(x, v) \equiv 0 \leq x \land x \leq H \]

\[ \text{grav} \equiv \{ x' = v, v' = -g \land x \geq 0 \} \]
Proving Quantum the Acrophobic Bouncing Ball

\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash j(x, v)
\]

\[
j(x, v) \vdash \{x' = v, v' = -g \land x \geq 0\}(j(x, v))
\]

\[
j(x, v), x = 0 \vdash j(x, (-cv))
\]

\[
j(x, v), x \neq 0 \vdash j(x, v)
\]

\[
j(x, v) \vdash 0 \leq x \land x \leq H
\]

1. \(j(x, v) \equiv x \geq 0\) weaker: fails postcondition if \(x > H\)
2. \(j(x, v) \equiv 0 \leq x \land x \leq H\) weak: fails ODE if \(v \gg 0\)
3. \(j(x, v) \equiv x = 0 \land v = 0\) strong: fails initial condition if \(x > 0\)
4. \(j(x, v) \equiv x = 0 \lor x = H \land v = 0\) no space for intermediate states
5. \(j(x, v) \equiv 2gx = 2gH - v^2 \land x \geq 0\) works: implicitly links \(v\) and \(x\)

\[
A \equiv 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0
\]

\[
B(x, v) \equiv 0 \leq x \land x \leq H
\]

\[
\text{grav} \equiv \{x' = v, v' = -g \land x \geq 0\}
\]
\[0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]
\[2gx = 2gH - v^2 \land x \geq 0 \vdash \{x' = v, v' = -g & x \geq 0\}(2gx = 2gH - v^2 \land x \geq 0)\]
\[2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (-cv)^2 \land x \geq 0\]
\[2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]
\[2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H\]

1. \(j(x,v) \equiv x \geq 0\) weaker: fails postcondition if \(x > H\)
2. \(j(x,v) \equiv 0 \leq x \land x \leq H\) weak: fails ODE if \(v \gg 0\)
3. \(j(x,v) \equiv x = 0 \land v = 0\) strong: fails initial condition if \(x > 0\)
4. \(j(x,v) \equiv x = 0 \lor x = H \land v = 0\) no space for intermediate states
5. \(j(x,v) \equiv 2gx = 2gH - v^2 \land x \geq 0\) works: implicitly links \(v\) and \(x\)
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx=2gH−v^2 \land x \geq 0 \]
\[ 2gx=2gH−v^2 \land x \geq 0 \vdash \{x' = v, v' = −g \land x \geq 0\}(2gx=2gH−v^2 \land x \geq 0) \]
\[ 2gx=2gH−v^2 \land x \geq 0, x = 0 \vdash 2gx=2gH−(−cv)^2 \land x \geq 0 \]
\[ 2gx=2gH−v^2 \land x \geq 0, x \neq 0 \vdash 2gx=2gH−v^2 \land x \geq 0 \]
\[ 2gx=2gH−v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \]

1. \( j(x, v) \equiv x \geq 0 \)
   weaker: fails postcondition if \( x > H \)

2. \( j(x, v) \equiv 0 \leq x \land x \leq H \)
   weak: fails ODE if \( v \gg 0 \)

3. \( j(x, v) \equiv x = 0 \land v = 0 \)
   strong: fails initial condition if \( x > 0 \)

4. \( j(x, v) \equiv x = 0 \lor x = H \land v = 0 \)
   no space for intermediate states

5. \( j(x, v) \equiv 2gx=2gH−v^2 \land x \geq 0 \)
   works: implicitly links \( v \) and \( x \)
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0 \]
\[ 2gx = 2gH - v^2 \land x \geq 0 \vdash [\{x' = v, v' = -g \land x \geq 0\}](2gx = 2gH - v^2 \land x \geq 0) \]
\[ \checkmark 2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (-cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots \]
\[ 2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0 \]
\[ 2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \]

1. \( j(x, v) \equiv x \geq 0 \) weaker: fails postcondition if \( x > H \)
2. \( j(x, v) \equiv 0 \leq x \land x \leq H \) weak: fails ODE if \( v \gg 0 \)
3. \( j(x, v) \equiv x = 0 \land v = 0 \) strong: fails initial condition if \( x > 0 \)
4. \( j(x, v) \equiv x = 0 \lor x = H \land v = 0 \) no space for intermediate states
5. \( j(x, v) \equiv 2gx = 2gH - v^2 \land x \geq 0 \) works: implicitly links \( v \) and \( x \)
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx=2gH−v^2 \land x\geq0 \]
\[ 2gx=2gH−v^2 \land x\geq0 \vdash \{x′=v, v′=−g \& x\geq0\} \]
\[ (2gx=2gH−v^2 \land x\geq0) \]
\[ 2gx=2gH−v^2 \land x\geq0, x=0 \vdash 2gx=2gH−(−cv)^2 \land x\geq0 \quad \text{if } c = 1 \ldots \]
\[ 2gx=2gH−v^2 \land x\geq0, x\neq0 \vdash 2gx=2gH−v^2 \land x\geq0 \]
\[ 2gx=2gH−v^2 \land x\geq0 \vdash 0 \leq x \land x \leq H \]

1. \( j(x,v) \equiv x \geq 0 \)
   weaker: fails postcondition if \( x > H \)

2. \( j(x,v) \equiv 0 \leq x \land x \leq H \)
   weak: fails ODE if \( v \gg 0 \)

3. \( j(x,v) \equiv x = 0 \land v = 0 \)
   strong: fails initial condition if \( x > 0 \)

4. \( j(x,v) \equiv x = 0 \lor x = H \land v = 0 \)
   no space for intermediate states

5. \( j(x,v) \equiv 2gx=2gH−v^2 \land x\geq0 \)
   works: implicitly links \( v \) and \( x \)
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx=2gH−v^2 \land x \geq 0 \]
\[ 2gx=2gH−v^2 \land x \geq 0 \vdash [\{x'=v, v'=−g & x \geq 0\}](2gx=2gH−v^2 \land x \geq 0) \]
✓ \[ 2gx=2gH−v^2 \land x \geq 0, x=0 \vdash 2gx=2gH−(−cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots \]
✓ \[ 2gx=2gH−v^2 \land x \geq 0, x \neq 0 \vdash 2gx=2gH−v^2 \land x \geq 0 \]
\[ 2gx=2gH−v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \]

1. \( j(x,v) \equiv x \geq 0 \)
   weaker: fails postcondition if \( x > H \)

2. \( j(x,v) \equiv 0 \leq x \land x \leq H \)
   weak: fails ODE if \( v \gg 0 \)

3. \( j(x,v) \equiv x = 0 \land v = 0 \)
   strong: fails initial condition if \( x > 0 \)

4. \( j(x,v) \equiv x = 0 \lor x = H \land v = 0 \)
   no space for intermediate states

5. \( j(x,v) \equiv 2gx=2gH−v^2 \land x \geq 0 \)
   works: implicitly links \( v \) and \( x \)

André Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL'19 35 / 66
Proving Quantum the Acrophobic Bouncing Ball

\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx=2gH-v^2 \land x \geq 0
\]
\[
2gx=2gH-v^2 \land x \geq 0 \vdash \{x'=v, v'=-g \land x \geq 0\}(2gx=2gH-v^2 \land x \geq 0)
\]
\[
2gx=2gH-v^2 \land x \geq 0, x=0 \vdash 2gx=2gH-(-cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots
\]
\[
2gx=2gH-v^2 \land x \geq 0, x \neq 0 \vdash 2gx=2gH-v^2 \land x \geq 0
\]
\[
2gx=2gH-v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H
\]

1. \(j(x,v) \equiv x \geq 0\) weaker: fails postcondition if \(x > H\)
2. \(j(x,v) \equiv 0 \leq x \land x \leq H\) weak: fails ODE if \(v \gg 0\)
3. \(j(x,v) \equiv x = 0 \land v = 0\) strong: fails initial condition if \(x > 0\)
4. \(j(x,v) \equiv x = 0 \lor x = H \land v = 0\) no space for intermediate states
5. \(j(x,v) \equiv 2gx=2gH-v^2 \land x \geq 0\) works: implicitly links \(v\) and \(x\)
\[0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]

\[2gx = 2gH - v^2 \land x \geq 0 \vdash \{x' = v, v' = -g \land x \geq 0\}\]

\[2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (-cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots\]

\[2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]

\[2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \quad \text{because } g > 0\]

1. \(j(x,v) \equiv x \geq 0\)  
   weaker: fails postcondition if \(x > H\)

2. \(j(x,v) \equiv 0 \leq x \land x \leq H\)  
   weak: fails ODE if \(v \gg 0\)

3. \(j(x,v) \equiv x = 0 \land v = 0\)  
   strong: fails initial condition if \(x > 0\)

4. \(j(x,v) \equiv x = 0 \lor x = H \land v = 0\)  
   no space for intermediate states

5. \(j(x,v) \equiv 2gx = 2gH - v^2 \land x \geq 0\)  
   works: implicitly links \(v\) and \(x\)
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0 \]

\[ 2gx = 2gH - v^2 \land x \geq 0 \vdash [\{x' = v, v' = -g & x \geq 0\}](2gx = 2gH - v^2 \land x \geq 0) \]

\checkmark \quad 2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots

\checkmark \quad 2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0

\checkmark \quad 2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \quad \text{because } g > 0

1. \quad j(x, v) \equiv x \geq 0 \quad \text{weaker: fails postcondition if } x > H

2. \quad j(x, v) \equiv 0 \leq x \land x \leq H \quad \text{weak: fails ODE if } v \gg 0

3. \quad j(x, v) \equiv x = 0 \land v = 0 \quad \text{strong: fails initial condition if } x > 0

4. \quad j(x, v) \equiv x = 0 \lor x = H \land v = 0 \quad \text{no space for intermediate states}

5. \quad j(x, v) \equiv 2gx = 2gH - v^2 \land x \geq 0 \quad \text{works: implicitly links } v \text{ and } x
Proving Quantum the Acrophobic Bouncing Ball

\[0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]

\[2gx = 2gH - v^2 \land x \geq 0 \vdash \{x' = v, v' = -g \& x \geq 0\}\] \[2gx = 2gH - v^2 \land x \geq 0\]

\[2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (-cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots\]

\[2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0\]

\[2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \quad \text{because } g > 0\]

1. \(j(x,v) \equiv x \geq 0\)\hspace{1cm}\text{weaker: fails postcondition if } x > H
2. \(j(x,v) \equiv 0 \leq x \land x \leq H\)\hspace{1cm}\text{weak: fails ODE if } v \gg 0
3. \(j(x,v) \equiv x = 0 \land v = 0\)\hspace{1cm}\text{strong: fails initial condition if } x > 0
4. \(j(x,v) \equiv x = 0 \lor x = H \land v = 0\)\hspace{1cm}\text{no space for intermediate states}
5. \(j(x,v) \equiv 2gx = 2gH - v^2 \land x \geq 0\)\hspace{1cm}\text{works: implicitly links } v \text{ and } x

\[x(t) = H - \frac{g}{2} t^2\]

\[v(t) = -gt\]
Proving Quantum the Acrophobic Bouncing Ball

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0 \]

\[ 2gx = 2gH - v^2 \land x \geq 0 \vdash \{ x' = v, v' = -g \land x \geq 0 \} (2gx = 2gH - v^2 \land x \geq 0) \]

\[ 2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (cv)^2 \land x \geq 0 \quad \text{if } c = 1 \ldots \]

\[ 2gx = 2gH - v^2 \land x \geq 0, x \neq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0 \]

\[ 2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H \quad \text{because } g > 0 \]

1. \( j(x,v) \equiv x \geq 0 \)
   weaker: fails postcondition if \( x > H \)

2. \( j(x,v) \equiv 0 \leq x \land x \leq H \)
   weak: fails ODE if \( v \gg 0 \)

3. \( j(x,v) \equiv x = 0 \land v = 0 \)
   strong: fails initial condition if \( x > 0 \)

4. \( j(x,v) \equiv x = 0 \lor x = H \land v = 0 \)
   no space for intermediate states

5. \( j(x,v) \equiv 2gx = 2gH - v^2 \land x \geq 0 \)
   works: implicitly links \( v \) and \( x \)

\[ x(t) = H - \frac{g}{2} t^2 \sim 2gx(t) = 2gH - g^2 t^2 \quad v(t)^2 = g^2 t^2 \quad \text{weaknesses: } v(t) = -gt \]
Example (Bouncing Ball)

\[ v = 0 \land 1 \geq c \geq 0 \land H = x \geq 0 \land g > 0 \rightarrow \left[ (\{x' = v, v' = -g \land x \geq 0\}; \right. \]

\[ \text{if}(x = 0) v := -cv)^* \left. \right] 0 \leq x \leq H \]
The lion’s share of understanding comes from understanding what does change (variants/progress measures) and what doesn’t change (invariants).

Invariants are a fundamental force of CS

Variants are another fundamental force of CS
Example (Runaround Robot)

\[(x, y) \neq o \rightarrow [((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0));
\{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*] (x, y) \neq o\]
Runaround Robot with Dubins Paths

\[ Q_\omega \equiv (x + \frac{w}{\omega} - o_x)^2 + (y - \frac{v}{\omega} - o_y)^2 \neq (\frac{v}{\omega})^2 + (\frac{w}{\omega})^2 \]

\[ Q_0 \equiv (x - o_x)w \neq (y - o_y)v \]

1. Obstacle not on tangential circle
2. Obstacle not on ray \((x, y) + \mathbb{R}(v, w)\)

Example (Runaround Robot)

\[(x, y) \neq o \rightarrow [((?Q_{-1}; \omega := -1 \cup ?Q_1; \omega := 1 \cup ?Q_0; \omega := 0); \{x' = v, y' = w, v' = \omega w, w' = -\omega v\})^*] (x, y) \neq o\]
KeYmaera X aXiomatic Tactical Theorem Prover for CPS

Proof search

Model

ctrl: $a := -b$;
plant: $x'' = a$

generates proofs

ModelPlex proof

KeYmaera X

Model Safety

Compliance Monitor

Trustworthy

Uniform substitution
Sound & complete
Small core: 1700 LOC

Flexible

Proof automation
Interactive UI
Programmable

Customizable

Scala+Java API
Command line
REST API

André Platzer (CMU)
Programming Cyber-Physical Systems With Logic
POPL'19 39 / 66
KeYmaera X 1,652 LOC

KeYmaera 1,652 LOC

KeY 0 LOC

Nuprl 25,000 LOC

MetaPRL 50,000 LOC

Isabelle/Pure 75,000 LOC

Coq 100,000 LOC

HOL Light

PHAVer

HSolver

SpaceEx

Cora

Flow*

dReal

HyCreate2

Disclaimer: Self-reported estimates of the soundness-critical lines of code + rules
Uniform Substitution

**Theorem (Soundness)**

\[
\text{US} \quad \frac{\phi}{\sigma(\phi)}
\]

provided \(\text{FV}(\sigma|_{\Sigma(\theta)}) \cap \text{BV}(\otimes(\cdot)) = \emptyset\) for each operation \(\otimes(\theta)\) in \(\phi\)

i.e. bound variables \(U = \text{BV}(\otimes(\cdot))\) of operator \(\otimes\)
are not free in the substitution on its argument \(\theta\) \((U\text{-admissible})\)

\[
\begin{align*}
\text{US} & \quad [a \cup b]p(\bar{x}) \iff [a]p(\bar{x}) \land [b]p(\bar{x}) \\
[x := x + 1 \cup x' = 1]x \geq 0 & \iff [x := x + 1]x \geq 0 \land [x' = 1]x \geq 0
\end{align*}
\]
Uniform Substitution

**Theorem (Soundness)** replace all occurrences of $p(\cdot)$

\[
\text{US} \quad \frac{\phi}{\sigma(\phi)}
\]

provided $\text{FV}(\sigma\mid\Sigma(\theta)) \cap \text{BV}(\otimes(\cdot)) = \emptyset$ for each operation $\otimes(\theta)$ in $\phi$

i.e. bound variables $U = \text{BV}(\otimes(\cdot))$ of operator $\otimes$
are not free in the substitution on its argument $\theta$ (U-admissible)

Uniform substitution $\sigma$ replaces all occurrences of $p(\theta)$ for any $\theta$ by $\psi(\theta)$
function $f(\theta)$ for any $\theta$ by $\eta(\theta)$
quantifier $C(\phi)$ for any $\phi$ by $\psi(\theta)$
program const. $a$ by $\alpha$

\[
\text{US} \quad \frac{[a \cup b]p(\vec{x}) \iff [a]p(\vec{x}) \land [b]p(\vec{x})}{[x := x + 1 \cup x' = 1]x \geq 0 \iff [x := x + 1]x \geq 0 \land [x' = 1]x \geq 0}
\]
Uniform Substitution

Theorem (Soundness) replace all occurrences of \( p(\cdot) \)

\[
\begin{align*}
US & \quad \frac{\phi}{\sigma(\phi)} \\
\end{align*}
\]

provided \( FV(\sigma | \Sigma(\theta)) \cap BV(\otimes(\cdot)) = \emptyset \) for each operation \( \otimes(\theta) \) in \( \phi \)

i.e. bound variables \( U = BV(\otimes(\cdot)) \) of operator \( \otimes \)
are not free in the substitution on its argument \( \theta \) (\( U \)-admissible)

Uniform substitution \( \sigma \) replaces all occurrences of \( p(\theta) \) for any \( \theta \) by \( \psi(\theta) \)
function \( f(\theta) \) for any \( \theta \) by \( \eta(\theta) \)
quantifier \( C(\phi) \) for any \( \phi \) by \( \psi(\theta) \)
program const. \( a \) by \( \alpha \)

\[
[v := f]p(v) \leftrightarrow p(f) \\
[v := -x][x' = v]x \geq 0 \leftrightarrow [x' = -x]x \geq 0
\]
Uniform Substitution

**Theorem (Soundness)**

replace all occurrences of \( p(\cdot) \)

\[
\frac{\phi}{\sigma(\phi)}
\]

provided \( \text{FV}(\sigma|_{\Sigma(\theta)}) \cap \text{BV}(\otimes(\cdot)) = \emptyset \) for each operation \( \otimes(\theta) \) in \( \phi \)

i.e. bound variables \( U = \BV(\otimes(\cdot)) \) of operator \( \otimes \)

are not free in the substitution on its argument \( \theta \) \hfill (\( U \)-admissible)

Uniform substitution \( \sigma \) replaces all occurrences of \( p(\theta) \) for any \( \theta \) by \( \psi(\theta) \)

function \( f(\theta) \) for any \( \theta \) by \( \eta(\theta) \)

quantifier \( C(\phi) \) for any \( \phi \) by \( \psi(\theta) \)

program const. \( a \) by \( \alpha \)

\[ [v := f]p(v) \leftrightarrow p(f) \]

\[ [v := -x][x' = v]x \geq 0 \leftrightarrow [x' = -x]x \geq 0 \]
Outline (Proving ODEs)

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
Differential Equation Axiomatization

1. Classical approach: ① Given ODE ② Solve ODE ③ Analyze solution
2. Descriptive power of ODEs: ODE much easier than its solution
3. Analyzing ODEs via their solutions undoes their descriptive power!

\[ \text{describe ODE} \iff \text{analyze ODE} \quad \text{Poincaré 1881} \]

\[ \text{describe solution} \iff \text{analyze solution} \]

1. Logical foundations of differential equation invariants
2. Decide invariance by dL proof

\[ x'' = -x \quad \text{has} \quad x(t) = \sin(t) = t - \frac{t^3}{3!} + \frac{t^5}{5!} - \frac{t^7}{7!} + \frac{t^9}{9!} - \ldots \]

\[ x''(t) = e^{t^2} \quad \text{has no elementary closed-form solution} \]
Properties of Differential Equations

Concept (Differential Dynamic Logic) (JAR’08, LICS’12)

\[ u^2 \leq v^2 + \frac{9}{2} \rightarrow \left[ u' = -v + \frac{u}{4} (1 - u^2 - v^2), \ v' = u + \frac{v}{4} (1 - u^2 - v^2) \right] \]

\[ u^2 + v^2 = 1 \rightarrow \left[ u' = -v + \frac{u}{4} (1 - u^2 - v^2), \ v' = u + \frac{v}{4} (1 - u^2 - v^2) \right] \]

\[ u^2 + v^2 = 1 \rightarrow \left[ u' = -v + \frac{u}{4} (1 - u^2 - v^2), \ v' = u + \frac{v}{4} (1 - u^2 - v^2) \right] \]
dL calculus is a sound & complete axiomatization of arithmetic invariants of differential equations. They are decidable with a derived axiom.
Theorem (Invariant Completeness) (LICS’18)

dL calculus is a sound & complete axiomatization of arithmetic invariants of differential equations. They are decidable with a derived axiom.

Theorem (Sound & Complete) (JAR’08, LICS’12, JAR’17)

dL calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations or to discrete dynamics.
Differential Invariants for Differential Equations

Differential Invariant

Differential Cut

Differential Ghost

\[ x' = f(x) \]

\[ x' = f(x) \]

\[ x' = f(x) \]
Differential Invariants for Differential Equations

Differential Invariant

\[ x' = f(x) \& Q \]

Differential Cut

\[ x' = f(x) \& Q \]

Differential Ghost

\[ x' = f(x) \& Q \]
Differential Invariants for Differential Equations

**Differential Invariant**

\[ x' = f(x) \land Q \]

**Differential Cut**

\[ x' = f(x) \land Q \]

**Differential Ghost**

\[ x' = f(x) \land Q \]
Differential Invariants for Differential Equations

**Differential Invariant**

**Differential Cut**

**Differential Ghost**

\[
\begin{align*}
\dot{x} &= f(x) \\
\dot{y} &= g(x, y)
\end{align*}
\]

\[
\begin{align*}
x' &= f(x) & Q \\
x' &= f(x) & Q \\
x' &= f(x) & Q
\end{align*}
\]
Differential Invariants for Differential Equations

Differential Invariant

Differential Cut

Differential Ghost

\[ x' = f(x) \]

\[ x' = f(x) \]

\[ x' = f(x) \]
Differential Invariants for Differential Equations

Differential Invariant

\[ x' = f(x) \& Q \]

Differential Cut

\[ x' = f(x) \& Q \]

Differential Ghost

\[ x' = f(x) \& Q \]
Differential Invariants for Differential Equations

**Differential Invariant**

**Differential Cut**

**Differential Ghost**

\[
\begin{align*}
& \frac{dx}{dt} = f(x) \\
& \frac{dy}{dt} = g(x, y)
\end{align*}
\]
Differential Invariants for Differential Equations

Differential Invariant

Differential Cut

Differential Ghost

\[ x' = f(x) & Q \]

\[ y' = g(x, y) \]

\[ x' = f(x) \]

\[ 0 \rightarrow t \]

Andre Platzer (CMU)
Differential Invariants for Differential Equations

Differential Invariant

\[ x' = f(x) \& Q \]

Differential Cut

\[ x' = f(x) \& Q \]

Differential Ghost

\[ y' = g(x, y) \]

André Platzer (CMU)
Differential Invariants for Differential Equations

**Differential Invariant**

\[
Q \vdash [x' := f(x)](P)' \\
\therefore P \vdash [x' = f(x) \& Q]P
\]

**Differential Cut**

\[
P \vdash [x' = f(x) \& Q]C \\
P \vdash [x' = f(x) \& Q \land C]P \\
\therefore P \vdash [x' = f(x) \& Q]P
\]

**Differential Ghost**

\[
P \leftrightarrow \exists y G \\
G \vdash [x' = f(x), y' = g(x, y) \& Q]G \\
\therefore P \vdash [x' = f(x) \& Q]P
\]

deductive power adds DI $\prec$ DC $\prec$ DG

\[\text{\textcopyright JLogComput'10, LMCS'12, LICS'12, JAR'17, LICS'18}\]
Differential Invariants for Differential Equations

### Differential Invariant

\[
\frac{Q \vdash [x' := f(x)](P)'}{P \vdash [x' = f(x) \& Q]P}
\]

### Differential Cut

\[
\frac{P \vdash [x' = f(x) \& Q]C \quad P \vdash [x' = f(x) \& Q \land C]P}{P \vdash [x' = f(x) \& Q]P}
\]

### Differential Ghost

\[
P \leftrightarrow \exists y G \quad G \vdash [x' = f(x), y' = g(x, y) \& Q]G
\]

\[
P \vdash [x' = f(x) \& Q]P
\]

if new \( y' = g(x, y) \) has long enough solution

---

JLogComput’10, LMCS’12, LICS’12, JAR’17, LICS’18

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL’19 46 / 66
\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y & \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]
\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]
\( \omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 x' + 2yy' \leq 0 \)

\( \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \)

damped oscillator
\[
\omega \geq 0 \land d \geq 0 \vdash \frac{1}{2} \omega^2 x y + 2y(-\omega^2 x - 2d\omega y) \leq 0
\]

\[
\omega \geq 0 \land d \geq 0 \vdash [x':=y][y':=-\omega^2 x - 2d\omega y] 2\omega^2 x' + 2yy' \leq 0
\]

\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]
\[ \omega \geq 0 \land d \geq 0 \vdash 2\omega^2 xy + 2y(-\omega^2 x - 2d\omega y) \leq 0 \]

\[ \omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 x' + 2yy' \leq 0 \]

\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y & \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]

Differential Invariants for Differential Equations

\[ \omega \geq 0 \land d \geq 0 \vdash 2\omega^2 xy + 2y(-\omega^2 x - 2d\omega y) \leq 0 \]

\[ \omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 x' + 2yy' \leq 0 \]

\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y & \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]

Damped oscillator
\[ \omega \geq 0 \land d \geq 0 \vdash 2 \omega^2 x y + 2 y (-\omega^2 x - 2 d \omega y) \leq 0 \]

\[ \omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2 d \omega y] 2 \omega^2 x x' + 2 y y' \leq 0 \]

\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2 d \omega y \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]
$$\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2$$
\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]
\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \& \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]

increasingly damped oscillator
\( \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \)

\( \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \land \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \)

Increasingly damped oscillator
\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]

\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]

\[ \omega \geq 0 \vdash [d' := 7] d' \geq 0 \]

\[ d \geq 0 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0] d \geq 0 \]

increasingly damped oscillator
\[
\begin{align*}
\omega^2 x^2 + y^2 &\leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 &\omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 &\leq c^2 \\
\omega^2 x^2 + y^2 &\leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 &\omega \geq 0] \omega^2 x^2 + y^2 &\leq c^2
\end{align*}
\]

\[
\begin{align*}
\omega \geq 0 &\vdash 7 \geq 0 \\
\omega \geq 0 &\vdash [d':=7] d' \geq 0 \\
d \geq 0 &\vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 &\omega \geq 0] d \geq 0
\end{align*}
\]

increasingly damped oscillator
Differential Cuts for Differential Equations

\[
\begin{align*}
\omega^2 x^2 + y^2 &\leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 &\leq c^2 \\
\omega^2 x^2 + y^2 &\leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0] \omega^2 x^2 + y^2 &\leq c^2
\end{align*}
\]

\[\omega \geq 0 \vdash 7 \geq 0\]

\[\omega \geq 0 \vdash [d' := 7] d' \geq 0\]

\[d \geq 0 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 & \omega \geq 0] d \geq 0\]

increasingly damped oscillator
\[
\omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d \omega y] 2\omega^2 x x' + 2y y' \leq 0
\]

\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \land \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]

\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \land \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]

\[
\omega \geq 0 \vdash 7 \geq 0
\]

\[
\omega \geq 0 \vdash [d' := 7] d' \geq 0
\]

\[
d \geq 0 \vdash [x' = y, y' = -\omega^2 x - 2d \omega y, d' = 7 \land \omega \geq 0] d \geq 0
\]

increasingly damped oscillator
\[
\omega \geq 0 \land d \geq 0 \vdash 2\omega^2 xy + 2y(-\omega^2 x - 2d\omega y) \leq 0
\]

\[
\omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 xx' + 2yy' \leq 0
\]

\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0 \land d \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]

\[
\omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2
\]

* 

\[
\omega \geq 0 \vdash 7 \geq 0
\]

\[
\omega \geq 0 \vdash [d' := 7] d' \geq 0
\]

\[
d \geq 0 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] d \geq 0
\]

increasingly damped oscillator
\begin{align*}
\omega &\geq 0 \land d \geq 0 \Rightarrow 2\omega^2 xy + 2y(-\omega^2 x - 2d\omega y) \leq 0 \\
\omega &\geq 0 \land d \geq 0 \Rightarrow [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 xx' + 2yy' \leq 0 \\
\omega^2 x^2 + y^2 \leq c^2 &\Rightarrow [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \land \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \\
\omega^2 x^2 + y^2 \leq c^2 &\Rightarrow [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \land \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \\
\omega &\geq 0 \Rightarrow 7 \geq 0 \\
\omega &\geq 0 \Rightarrow [d' := 7] d' \geq 0 \\
d \geq 0 &\Rightarrow [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \land \omega \geq 0] d \geq 0
\end{align*}

increasingly damped oscillator
Differential Cuts for Differential Equations

\[ \omega \geq 0 \land d \geq 0 \vdash 2\omega^2 xy + 2y(-\omega^2 x - 2d\omega y) \leq 0 \]
\[ \omega \geq 0 \land d \geq 0 \vdash [x' := y][y' := -\omega^2 x - 2d\omega y] 2\omega^2 xx' + 2yy' \leq 0 \]
\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]
\[ \omega^2 x^2 + y^2 \leq c^2 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] \omega^2 x^2 + y^2 \leq c^2 \]

init

\[ \omega \geq 0 \vdash 7 \geq 0 \]
\[ \omega \geq 0 \vdash [d' := 7] d' \geq 0 \]
\[ d \geq 0 \vdash [x' = y, y' = -\omega^2 x - 2d\omega y, d' = 7 \& \omega \geq 0] d \geq 0 \]

Could repeatedly diffcut in formulas to help the proof
Assuming Differential Invariance

\[ \neg F \]

\[ \neg F \]

\[ Q \vdash [x' := f(x)](F)' \]

\[ \frac{F \vdash [x' = f(x) \& Q]F}{F \oplus Q \vdash [x' := f(x)](F)'} \]

\[ F \wedge Q \vdash [x' := f(x)](F)' \]

\[ \frac{F \vdash [x' = f(x) \& Q]F}{F \oplus Q \vdash [x' := f(x)](F)'} \]
Assuming Differential Invariance

\[ \neg F \rightarrow F \]

\[ \neg F \rightarrow F \]

\[ Q \vdash [x' := f(x)](F)' \]

\[ \frac{F \vdash [x' = f(x) \land Q]F}{F \vdash [x' = f(x) \land Q]F} \]

\[ F \land Q \vdash [x' := f(x)](F)' \]

\[ \frac{F \vdash [x' = f(x) \land Q]F}{F \vdash [x' = f(x) \land Q]F} \]

Example (Inductive hypothesis)

\[ v^2 - 2v + 1 = 0 \vdash [v' = w, w' = -v]v^2 - 2v + 1 = 0 \]
Assuming Differential Invariance

\[ Q \vdash [x' := f(x)](F)' \]
\[ F \vdash [x' = f(x) & Q]F \]

\[ F \land Q \vdash [x' := f(x)](F)' \]
\[ F \vdash [x' = f(x) & Q]F \]

---

Example (Inductive hypothesis)

\[ v^2 - 2v + 1 = 0 \vdash [v' := w][w' := -v]2v v' - 2v' = 0 \]
\[ v^2 - 2v + 1 = 0 \vdash [v' = w, w' = -v]v^2 - 2v + 1 = 0 \]
Example (Inductive hypothesis)

\[
\begin{align*}
\nu^2 - 2\nu + 1 &= 0 \vdash 2\nu w - 2w = 0 \\
\nu^2 - 2\nu + 1 &= 0 \vdash [v' := w][w' := -\nu]2\nu v' - 2v' = 0 \\
\nu^2 - 2\nu + 1 &= 0 \vdash [v' = w, w' = -\nu]\nu^2 - 2\nu + 1 = 0
\end{align*}
\]
Assuming Differential Invariance

\[ Q \vdash \left[ x' := f(x) \right] (F)' \]

\[ F \vdash \left[ x' = f(x) \& Q \right] F \]

Example (Inductive hypothesis is unsound!)

(unsound)

\[ \vdash v^2 - 2v + 1 = 0 \]

\[ \vdash 2vw - 2w = 0 \]

\[ \vdash [v' := w][w' := -v]2vv' - 2v' = 0 \]

\[ \vdash [v' = w, w' = -v]v^2 - 2v + 1 = 0 \]
Proofs use continuously changing basis to keep invariants at constant local coordinates.

Proving ODE invariance with logic

Local coordinates: \( \left( \frac{z_1}{z_4} \right) \)

Local coordinates: \( \left( \frac{z_6}{z_5} \right) \)

Proofs with higher Lie derivatives

\( p'' \) safe
\( p' \) inconclusive
\( p \) unsafe
Conservatively bounded next velocity above parachute's limit velocity.

Limit by differential ghost:

\[ y' = -p^2 (v - \sqrt{\frac{g}{p}}) y^2 (v + \sqrt{\frac{g}{p}}) > 0 \]

\( v \geq v_{old} - gt \) if closed.
Example (Parachute)

\[
((?Q \land r = a) \cup r:=p); t:=0;
\{x' = v, v' = -g + rv^2, t' = 1 \land t \leq T \land x \geq 0 \land v < 0\})^* 
\]
Example (Parachute)

\[
\rightarrow \left[ (??(Q \land r = a) \cup r := p); t := 0; \\
\{ x' = v, v' = -g + rv^2, t' = 1 \land t \leq T \land x \geq 0 \land v < 0 \}^* \right] \\
(x = 0 \rightarrow v \geq m)
\]
Example (Parachute)

\[
\rightarrow \left[ \left( \left( Q \land r = a \right) \lor r := p \right) ; t := 0; \right.
\left. \begin{array}{l}
    \{ x' = v, v' = -g + rv^2, t' = 1 \land t \leq T \land x \geq 0 \land v < 0 \}^* \\
    \end{array} \right] \\
\left( x = 0 \rightarrow v \geq m \right)
\]
**Ex: Parachute Open or Keep Closed**

\[ Q \equiv v - gT > -\sqrt{g/p} \]

Conservatively bounded next velocity above parachute’s limit velocity.

---

**Example (Parachute)**

\[
m < -\sqrt{g/p} \rightarrow \left[ \left( \left( ?(Q \wedge r = a) \cup r := p \right); t := 0; \right) \left\{ x' = v, v' = -g + rv^2, t' = 1 \wedge t \leq T \wedge x \geq 0 \wedge v < 0 \right\}^* \right] \]

\[
(x = 0 \rightarrow v \geq m)\]
Ex: Parachute Open or Keep Closed

\[ Q \equiv v - gT > -\sqrt{g/p} \]

Conservatively bounded next velocity above parachute’s limit velocity.

Limit by differential ghost:

\[ y' = -\frac{p}{2} (v - \sqrt{g/p}) \quad y^2(v + \sqrt{g/p}) > 0 \]

Example (Parachute)

\[ m < -\sqrt{g/p} \rightarrow \left[ ((?Q \land r = a) \cup r := p); t := 0; \right. \]
\[ \{ x' = v, v' = -g + rv^2, t' = 1 \land t \leq T \land x \geq 0 \land v < 0 \}^* \]

\[ (x = 0 \rightarrow v \geq m) \]
Ex: Parachute Open or Keep Closed

\[ Q \equiv v - gT > -\sqrt{g/p} \]

Conservatively bounded next velocity above parachute’s limit velocity. Limit by differential ghost:

\[ y' = -\frac{p}{2}(v - \sqrt{g/p}) \quad y^2(v + \sqrt{g/p}) = 1 \quad >0 \]

\[ v \geq \text{old}(v) - gt \text{ if closed} \]

Example (Parachute)

\[ m < -\sqrt{g/p} \rightarrow \left[ ((?\ (Q \land r = a) \cup r := p) ; t := 0 ; \right. \]

\[ \left. \{ x' = v, v' = -g + rv^2, t' = 1 \land t \leq T \land x \geq 0 \land v < 0 \}\}^* \]

\[ (x = 0 \rightarrow v \geq m) \]
### Outline (CPS Application Highlights)

1. **CPS are Multi-Dynamical Systems**
2. **CPS Programs**
   - Syntax
   - Semantics
   - Examples
3. **Differential Dynamic Logic**
   - Syntax
   - Semantics
   - Example: Car Control Design
4. **Dynamic Axioms for Dynamical Systems**
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. **Differential Invariants for Differential Equations**
6. **Applications**
7. **Verified Compilation of CPS Programs**
8. **Summary**
- Developed by the FAA to replace current TCAS in aircraft
- Approximately optimizes Markov Decision Process on a grid
- Advisory from lookup tables with numerous 5D interpolation regions

1. Identified safe region for each advisory symbolically
2. Proved safety for hybrid systems flight model in KeYmaera X
ACAS X table comparison shows safe advisory in 97.7% of the 648,591,384,375 states compared (15,160,434,734 counterexamples).

ACAS X issues DNC advisory, which induces collision unless corrected.
Airborne Collision Avoidance System ACAS X: Refine

- Conservative, so too many counterexamples
- Settle for: safe for a little while, with safe future advisory possibility
- Safeable advisory: a subsequent advisory can safely avoid collision

1. Identified safeable region for each advisory symbolically
2. Proved safety for hybrid systems flight model in KeYmaera X
ACAS X table comparison shows safeable advisory in more of the 648,591,384,375 states compared ($\approx 899 \times 10^6$ counterexamples).

ACAS X issues Maintain advisory instead of CL1500
ACAS X table comparison shows safeable advisory in more of the 648,591,384,375 states compared ($\approx 899 \times 10^6$ counterexamples).

ACAS X issues Maintain advisory instead of CL1500
Fundamental safety question for ground robot navigation

When will which control decision avoid obstacles?

Depends on safety objective, physical capabilities of robot + obstacle

1. Identified safe region for each safety notion symbolically
2. Proved safety for hybrid systems ground robot model in KeYmaera X
Ground Robot Obstacle Avoidance: Verify

- Fundamental safety question for ground robot navigation
- When will which control decision avoid obstacles?
- Depends on safety objective, physical capabilities of robot + obstacle

1. Identified safe region for each safety notion symbolically
2. Proved safety for hybrid systems ground robot model in KeYmaera X
Ground Robot Obstacle Avoidance: Verify

- Fundamental safety question for ground robot navigation
- When will which control decision avoid obstacles?
- Depends on safety objective, physical capabilities of robot + obstacle

---

1. Identified safe region for each safety notion symbolically
2. Proved safety for hybrid systems ground robot model in KeYmaera X

---

André Platzer (CMU) Programming Cyber-Physical Systems With Logic POPL’19 56 / 66
Ground Robot Obstacle Avoidance: Verify

- Fundamental safety question for ground robot navigation
- When will which control decision avoid obstacles?
- Depends on safety objective, physical capabilities of robot + obstacle

1. Identified safe region for each safety notion symbolically
2. Proved safety for hybrid systems ground robot model in KeYmaera X
Ground Robot Obstacle Avoidance: Verify

- Fundamental safety question for ground robot navigation
- When will which control decision avoid obstacles?
- Depends on safety objective, physical capabilities of robot + obstacle

1. Identified safe region for each safety notion symbolically
2. Proved safety for hybrid systems ground robot model in KeYmaera X
<table>
<thead>
<tr>
<th>Safety</th>
<th>Invariant</th>
<th>+ Safe Control</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>static</strong></td>
<td>$|p - o|_\infty &gt; \frac{s^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon s\right)$</td>
<td></td>
</tr>
<tr>
<td><strong>passive</strong></td>
<td>$s \neq 0 \rightarrow |p - o|_\infty &gt; \frac{s^2}{2b} + V\frac{s}{b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V)\right)$</td>
<td></td>
</tr>
<tr>
<td>+ sensor</td>
<td>$|\hat{p} - o|_\infty &gt; \frac{s^2}{2b} + V\frac{s}{b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V)\right) + \Delta_p$</td>
<td></td>
</tr>
<tr>
<td>+ disturb</td>
<td>$|p - o|_\infty &gt; \frac{s^2}{2b\Delta_a} + V\frac{s}{b\Delta_a} + \left(\frac{A}{b\Delta_a} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V)\right)$</td>
<td></td>
</tr>
<tr>
<td>+ failure</td>
<td>$|\hat{p} - o|_\infty &gt; \frac{s^2}{2b} + V\frac{s}{b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(v + V)\right) + \Delta_p + g\Delta$</td>
<td></td>
</tr>
<tr>
<td>friendly</td>
<td>$|p - o|_\infty &gt; \frac{s^2}{2b} + V\left(\frac{s}{b} + \tau\right) + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V)\right)$</td>
<td></td>
</tr>
</tbody>
</table>

\[\ldots\]

RSS’13, IJRR’17
<table>
<thead>
<tr>
<th>Safety</th>
<th>Invariant</th>
<th>+ Safe Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>static</td>
<td>( |p - o|_{\infty} &gt; \frac{s^2}{2b} )</td>
<td>+ ( (\frac{A}{b} + 1)\left(\frac{A}{2}\varepsilon^2 + \varepsilon s \right) )</td>
</tr>
<tr>
<td>passive</td>
<td>( s \neq 0 \rightarrow |p - o|_{\infty} &gt; \frac{s^2}{2b} )</td>
<td>+ ( V \frac{s}{b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V) \right) )</td>
</tr>
<tr>
<td>+ sensor</td>
<td>( |p - o|_{\infty} &gt; \frac{s^2}{2b} )</td>
<td>+ ( V \frac{s}{b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V) \right) )</td>
</tr>
<tr>
<td>+ disturb.</td>
<td>( |p - o|_{\infty} &gt; \frac{s^2}{2b} + V \frac{s}{b} )</td>
<td>+ ( \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(v + V) \right) ) + ( \Delta p )</td>
</tr>
<tr>
<td>+ failure</td>
<td>( |p - o|_{\infty} &gt; \frac{s^2}{2b} + V \frac{s}{b} )</td>
<td>+ ( \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(v + V) \right) ) + ( \Delta p + g\Delta )</td>
</tr>
<tr>
<td>friendly</td>
<td>( |p - o|_{\infty} &gt; \frac{s^2}{2b} + V \frac{V^2}{2b_o} + V\left(\frac{s}{b} + \tau \right) )</td>
<td>+ ( \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon(s + V) \right) )</td>
</tr>
</tbody>
</table>

Question: How to find and justify constraints? Proof!
Outline (CPS Executables)

1. CPS are Multi-Dynamical Systems

2. CPS Programs
   - Syntax
   - Semantics
   - Examples

3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design

4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X

5. Differential Invariants for Differential Equations

6. Applications

7. Verified Compilation of CPS Programs

8. Summary
VeriPhy: Automatic, Verified EXEs from Controllers

Hybrid Systems Theorem Proving → Abstract Controllers and Monitors → Sound Discrete Arithmetic → Sound Monitor Compilation → Cyber Physical System

- Hybrid Systems Theorem Proving
- Abstract Controllers and Monitors
- Sound Discrete Arithmetic
- Sound Monitor Compilation
- Cyber Physical System

POPL'19 58 / 66
VeriPhy: Automatic, Verified EXEs from Controllers

Programming Cyber-Physical Systems With Logic
André Platzer (CMU)

Hybrid Systems Theorem Proving → Abstract Controllers and Monitors → Sound Discrete Arithmetic → Sound Monitor Compilation → Cyber Physical System

Small Prover Core Proven Sound → Provably Correct Monitoring Conditions → Formalized Soundness Theorem → Verified Compiler → Verified Executable

PLDI'18
VeriPhy: Automatic, Verified EXEs from Controllers

Hybrid Systems Theorem Proving

Abstract Controllers and Monitors

Sound Discrete Arithmetic

Sound Monitor Compilation

Cyber Physical System

Small Prover Core Proven Sound

Provably Correct Monitoring Conditions

Formalized Soundness Theorem

Verified Compiler

Verified Executable

KeYmaera X

Isabelle/HOL

HOL4
VeriPhy: Automatic, Verified EXEs from Controllers

Hybrid Systems Theorem Proving

Abstract Controllers and Monitors

Sound Discrete Arithmetic

Sound Monitor Compilation

Cyber Physical System

Small Prover Core
Proven Sound

Provably Correct Monitoring Conditions

Formalized Soundness Theorem

Verified Compiler

Verified Executable

Invariant Generation

Keymaera X

ODE Solving

Isabelle/HOL

seL4 Libs

Multivariate Analysis

HOL4

Machine Models

CakeML Compiler

Sound Code Extraction

Keymaera X Web UI

Isabelle/HOL seL4 Libs

Multivariate Analysis

HOL4 CakeML Compiler

Sound Code Extraction

PLDI'18

André Platzer (CMU)

Programming Cyber-Physical Systems With Logic

POPL'19 58 / 66
VeriPhy: Automatic, Verified EXEs from Controllers

Hybrid Systems Theorem Proving

Abstract Controllers and Monitors

Sound Discrete Arithmetic

Sound Monitor Compilation

Cyber Physical System

Small Prover Core Proven Sound

Provably Correct Monitoring Conditions

Formalized Soundness Theorem

Verified Compiler

Verified Executable

Invariant Generation

KeYmaera X

Isabelle/HOL

HOL4

Verified Proof Transport!

seL4 Libs

Multivariate Analysis

(Ongoing Works)

CakeML Compiler

Sound Code Extraction

PLDI'18

André Platzer (CMU)
VeriPhy: Takeaway Metaphor

Your Model

Low-Level Proofs

Safe CPS
VeriPhy: Takeaway Metaphor

VeriPhy Pipeline (VeriPhy.org)
ModelPlex ensures that verification results about models apply to CPS implementations.
ModelPlex ensures that verification results about models apply to CPS implementations.

**Contributions**

- Verification results about models transfer to CPS when validating model compliance.
- Compliance with model is characterizable in logic.
- Compliance formula transformed by proof to executable monitor.
- Correct-by-construction provably correct runtime model validation.

\[ i - 1 \quad i \quad i + 1 \]

model adequate? control safe? until next cycle?

RV’14, FMSD’16
Sandboxed controller uses external controller when decision is safe, else uses verified fallback. Detects non-compliant plants.

\[ \phi \rightarrow [(ctrl; plant)^*] \psi \]

\[ \vec{x} := *; \]
\[ ?\phi; \]
\[ ( \vec{x}^+:= \text{extCtrl}; \]
\[ ( ?\text{ctrlMon}(\vec{x}, \vec{x}^+) \]
\[ \cup \text{fallback } ) ;\]
\[ \vec{x}:=\vec{x}^+; \]
\[ \vec{x}^+ := *; \]
\[ ?\text{plantMon}(\vec{x}, \vec{x}^+); \]
\[ \vec{x}:=\vec{x}^+)^* \]
Outline (Programming CPS with Logic)

1. CPS are Multi-Dynamical Systems
2. CPS Programs
   - Syntax
   - Semantics
   - Examples
3. Differential Dynamic Logic
   - Syntax
   - Semantics
   - Example: Car Control Design
4. Dynamic Axioms for Dynamical Systems
   - Axiomatics
   - Safe CPS Programming & Proving in KeYmaera X
5. Differential Invariants for Differential Equations
6. Applications
7. Verified Compilation of CPS Programs
8. Summary
Acknowledgments

Logical Systems Lab at Carnegie Mellon University
Yong Kiam Tan, Brandon Bohrer, Nathan Fulton, Sarah Loos
Stefan Mitsch, Khalil Ghorbal, Jean-Baptiste Jeannin, Andrew Sogokon
Programming Cyber-Physical Systems With Logic

Programming language principles affect CPSs

- Multi-dynamical systems
- Hybrid programs + dL logic
- Compositional proofs
- Logic impacts CPS

1. Analytic foundations
2. Practical proving
3. Significant applications
4. Bring sciences together

Programming CPS $\neq$ program cyber $\parallel$ program physics (mutual ignorance)
Further CPS Topics

Numerous wonders remain to be discovered

- Verified CPS implementations by ModelPlex [FMSD’16]
- Correct CPS execution [PLDI’18]
- CPS proof and tactic languages+libraries [ITP’17]
- Big CPS built from safe components [STTT’18]
- Stochastic hybrid systems [CADE’11]
- Invariant generation [FMSD’09 TACAS’14]
- Safe AI autonomy in CPS [AAAI’18]
- Correct model transformation [FM’14]
- Refinement + system property proofs [LICS’16]
- CPS information flow [LICS’18]
- Hybrid games [TOCL’15]

CPSs deserve proofs as safety evidence!
I Part: Elementary Cyber-Physical Systems
  2. Differential Equations & Domains
  3. Choice & Control
  4. Safety & Contracts
  5. Dynamical Systems & Dynamic Axioms
  6. Truth & Proof
  7. Control Loops & Invariants
  8. Events & Responses
  9. Reactions & Delays

II Part: Differential Equations Analysis
  10. Differential Equations & Differential Invariants
  11. Differential Equations & Proofs
  12. Ghosts & Differential Ghosts
  13. Differential Invariants & Proof Theory

III Part: Adversarial Cyber-Physical Systems
  14-17. Hybrid Systems & Hybrid Games

IV Part: Comprehensive CPS Correctness
Appendix

- Soundness and Completeness
- Differentials
- Differential Ghosts
- Differential Radical Invariants
**Theorem (Sound & Complete) (JAR’08, LICS’12, JAR’17)**

*dL calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations or to discrete dynamics.*

**Corollary (Complete Proof-theoretical Bridge)**

proving continuous = proving hybrid = proving discrete

\[ \models P \text{ iff } \text{FODE} \vdash_{dL} P \]
Theorem (Sound & Complete) (JAR’08, LICS’12, JAR’17)

dL calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations or to discrete dynamics.

Corollary (Complete Proof-theoretical Bridge)

proving continuous = proving hybrid = proving discrete
Theorem (Sound & Complete) (JAR’08, LICS’12, JAR’17)

dL calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations or to discrete dynamics.

Corollary (Complete Proof-theoretical Bridge)

proving continuous = proving hybrid = proving discrete
Differentials

Syntax

\[ e ::= x \mid x' \mid c \mid e + k \mid e \cdot k \mid (e)' \]

Semantics

\[ \omega[(e)'] = \sum_x \omega(x') \frac{\partial[e]}{\partial x}(\omega) \]

Axioms

\((e + k)' = (e)' + (k)'\)
\((e \cdot k)' = (e)' \cdot k + e \cdot (k)'
\((c())' = 0 \quad \text{for constants/numbers} \ c()\)
\((x)' = x' \quad \text{for variables} \ x \in \mathcal{V}\)

ODE

\[ [x' = f(x) \& Q] = \{(\varphi(0)|_{x'} \in \mathcal{C}, \varphi(r)) : \varphi \models x' = f(x) \land Q \}
\quad \text{for some} \ \varphi : [0, r] \rightarrow \mathcal{S}, \ \text{some} \ r \in \mathbb{R} \}
\]

\[ \varphi(z)(x') = \frac{d\varphi(t)(x)}{dt}(z) \quad \ldots \]
Differential Substitution Lemmas

**Lemma (Differential lemma)** (Differential value vs. Time-derivative)

If $\varphi \models x' = f(x) \land Q$ for duration $r > 0$, then for all $0 \leq z \leq r$, $\text{FV}(e) \subseteq \{x\}$:

$$\varphi(z) \llbracket (e)' \rrbracket = \frac{d\varphi(t)[e]}{dt}(z)$$

**Lemma (Differential assignment)** (Effect on Differentials)

If $\varphi \models x' = f(x) \land Q$ then $\varphi \models P \leftrightarrow [x' := f(x)]P$

**Lemma (Derivations)** (Equations of Differentials)

- $(e + k)' = (e)' + (k)'$
- $(e \cdot k)' = (e)' \cdot k + e \cdot (k)'$
- $(c())' = 0$ for constants/numbers $c()$
- $(x)' = x'$ for variables $x \in \mathcal{V}$
## Differential Substitution Lemmas $\mapsto$ Proofs

### Lemma (Differential lemma) (Differential value vs. Time-derivative)

If $\varphi \models x' = f(x) \land Q$ for duration $r > 0$, then for all $0 \leq z \leq r$, $FV(e) \subseteq \{x\}$:

<table>
<thead>
<tr>
<th>Syntactic</th>
<th>Analytic</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi(z)[(e)'] = \frac{d\varphi(t)[e]}{dt}(z)$</td>
<td></td>
</tr>
</tbody>
</table>

### Lemma (Differential assignment) (Effect on Differentials)

$DE \ [x' = f(x) \& Q]P \leftrightarrow [x' = f(x) \& Q][x' := f(x)]P$

### Lemma (Derivations) (Equations of Differentials)

| $+'$ | $(e + k)' = (e)' + (k)'$ |
| $\cdot'$ | $(e \cdot k)' = (e)' \cdot k + e \cdot (k)'$ |
| $c'$ | $(c())' = 0$ |
| $x'$ | $(x)' = x'$ |
Darboux inequalities are DG

\[ Q \vdash p' \geq gp \]
\[ p \succeq 0 \vdash [x' = f(x) \& Q]p \succeq 0 \]

\[
\begin{align*}
(1-u^2-v^2)' &\geq -\frac{1}{2}(u^2+v^2)(1-u^2-v^2) \\
\ldots \vdash & [u' = -v + \frac{u}{4}(1-u^2-v^2) \\
& v' = u + \frac{v}{4}(1-u^2-v^2) \\
& 1-u^2-v^2 > 0
\end{align*}
\]
ODE Axiomatization: Derived Darboux Rules

Darboux inequalities are DG

\[ Q \vdash p' \geq gp \]

\[ p \succ 0 \vdash [x' = f(x) & Q]p \succ 0 \]

\[ (g \in \mathbb{R}[x]) \]

\[ (1-u^2-v^2)' \geq -\frac{1}{2}(u^2+v^2)(1-u^2-v^2) \]

\[ \vdash \left[ u' = -v + \frac{u}{4}(1-u^2-v^2) \right. \]

\[ \left. v' = u + \frac{v}{4}(1-u^2-v^2) \right. \]

\[ y' = \frac{1}{2}(u^2+v^2)y \]

\[ 1-u^2-v^2 > 0 \]

\[ (1-u^2-v^2)y > 0 \]
Darboux Inequalities are Differential Ghosts: Details

\[ \begin{align*}
\mathbb{R} & \vdash (-gy)z^2 + y(2z(\frac{g}{2}z)) = 0 \\
\text{dl} & \quad yz^2 = 1 \vdash [x' = f(x), y' = -gy, z' = \frac{g}{2}z & Q]yz^2 = 1 \\
\text{M[\cdot], \forall R} & \quad y > 0 \vdash \exists z [x' = f(x), y' = -gy, z' = \frac{g}{2}z & Q] y > 0 \\
\text{DG} & \quad y > 0 \vdash [x' = f(x), y' = -gy & Q] y > 0
\end{align*} \]

\[ \begin{align*}
\mathbb{R} & \vdash p' \geq gp \\
\text{cut} & \quad Q, y > 0 \vdash p'y - gyp \geq 0 \\
\text{dl} & \quad p \geq 0, y > 0 \vdash [x' = f(x), y' = -gy & Q \land y > 0]py \geq 0 \\
\text{dC} & \quad p \geq 0, y > 0 \vdash [x' = f(x), y' = -gy & Q](y > 0 \land py \geq 0) \\
\text{M[\cdot], \forall R} & \quad p \geq 0 \vdash \exists y [x' = f(x), y' = -gy & Q] p \geq 0 \\
\text{DG} & \quad p \geq 0 \vdash [x' = f(x) & Q] p \geq 0
\end{align*} \]

P.S. \( z' = \frac{g}{2}z \) superfluous for open inequalities \( p > 0 \) and \( p \neq 0 \).
Differential Radical Invariants

Theorem (Differential radical invariant characterization)

\[
\begin{align*}
  h = 0 & \Rightarrow \bigwedge_{i=1}^{N-1} h_p^{(i)} = 0 \\
  h = 0 & \Rightarrow [x' = p] h = 0
\end{align*}
\]

characterizes all algebraic invariants, where \( N = \text{ord} \sqrt[h]{}(h) \), i.e.

\[
h_p^{(N)} = \sum_{i=0}^{N-1} g_i h_p^{(i)} \quad (g_i \in \mathbb{R}[x]) \quad h_p^{(i+1)} = [x' := p](h_p^{(i)})'
\]

Corollary (Algebraic Invariants Decidable)

Algebraic invariants of algebraic differential equations are decidable.

with Khalil Ghorbal TACAS’14
Example: Longitudinal Dynamics of an Airplane

Study (6th Order Longitudinal Flight Equations)

\[ u' = \frac{X}{m} - g \sin(\theta) - qw \quad \text{axial velocity} \]
\[ w' = \frac{Z}{m} + g \cos(\theta) + qu \quad \text{vertical velocity} \]
\[ x' = \cos(\theta)u + \sin(\theta)w \quad \text{range} \]
\[ z' = -\sin(\theta)u + \cos(\theta)w \quad \text{altitude} \]
\[ \theta' = q \quad \text{pitch angle} \]
\[ q' = \frac{M}{I_{yy}} \quad \text{pitch rate} \]

\[ X : \text{thrust along } u \quad Z : \text{thrust along } w \quad M : \text{thrust moment for } w \]
\[ g : \text{gravity} \quad m : \text{mass} \quad I_{yy} : \text{inertia second diagonal} \]

with Khalil Ghorbal TACAS’14
Result (DRI Automatically Generates Invariant Functions)

\[
\frac{M_z}{I_{yy}} + g\theta + \left(\frac{X}{m} - qw\right)\cos(\theta) + \left(\frac{Z}{m} + qu\right)\sin(\theta)
\]

\[
\frac{M_x}{I_{yy}} - \left(\frac{Z}{m} + qu\right)\cos(\theta) + \left(\frac{X}{m} - qw\right)\sin(\theta)
\]

\[- q^2 + \frac{2M\theta}{I_{yy}} \]

with Khalil Ghorbal TACAS’14
Example: Dubins Dynamics of 2 Airplanes

Result (DRI Automatically Generates Invariants)

\[
\omega_1 = 0 \land \omega_2 = 0 \rightarrow v_2 \sin \vartheta x = (v_2 \cos \vartheta - v_1)y > p(v_1 + v_2)
\]

\[
\omega_1 \neq 0 \lor \omega_2 \neq 0 \rightarrow -\omega_1 \omega_2 (x^2 + y^2) + 2v_2 \omega_1 \sin \vartheta x + 2(v_1 \omega_2 - v_2 \omega_1 \cos \vartheta)y + 2v_1 v_2 \cos \vartheta > 2v_1 v_2 + 2p(v_2 |\omega_1| + v_1 |\omega_2|) + p^2 |\omega_1 \omega_2|
\]
André Platzer.
*Logical Foundations of Cyber-Physical Systems.*
URL: http://www.springer.com/978-3-319-63587-3,
doi:10.1007/978-3-319-63588-0.

André Platzer.
Logic & proofs for cyber-physical systems.
doi:10.1007/978-3-319-40229-1_3.

André Platzer.
Logics of dynamical systems.
In *LICS* [22], pages 13–24.

André Platzer.
A complete uniform substitution calculus for differential dynamic logic.
André Platzer.
Differential dynamic logic for hybrid systems.

André Platzer.
Differential game logic.

André Platzer.
Differential hybrid games.

Stefan Mitsch, Khalil Ghorbal, David Vogelbacher, and André Platzer.
Formal verification of obstacle avoidance and navigation of ground robots.
André Platzer and Jan-David Quesel.
European Train Control System: A case study in formal verification.

Stefan Mitsch, Marco Gario, Christof J. Budnik, Michael Golm, and André Platzer.
Formal verification of train control with air pressure brakes.
doi:10.1007/978-3-319-68499-4_12.

Jean-Baptiste Jeannin, Khalil Ghorbal, Yanni Kouskoulas, Aurora Schmidt, Ryan Gardner, Stefan Mitsch, and André Platzer.
A formally verified hybrid system for safe advisories in the next-generation airborne collision avoidance system.
André Platzer.
The complete proof theory of hybrid systems.
In LICS [22], pages 541–550.
doi:10.1109/LICS.2012.64.

Nathan Fulton, Stefan Mitsch, Jan-David Quesel, Marcus Völ, and André Platzer.
KeYmaera X: An axiomatic tactical theorem prover for hybrid systems.
doi:10.1007/978-3-319-21401-6_36.

Stefan Mitsch and André Platzer.
ModelPlex: Verified runtime validation of verified cyber-physical system models.
Special issue of selected papers from RV’14.
André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.

André Platzer.
The structure of differential invariants and differential cut elimination.

Nathan Fulton, Stefan Mitsch, Brandon Bohrer, and André Platzer.
Bellerophon: Tactical theorem proving for hybrid systems.
doi:10.1007/978-3-319-66107-0_14.

André Platzer.
Stochastic differential dynamic logic for stochastic hybrid programs.
Khalil Ghorbal and André Platzer.
Characterizing algebraic invariants by differential radical invariants.
doi:10.1007/978-3-642-22438-6_34.

Thomas A. Henzinger.
The theory of hybrid automata.
doi:10.1109/LICS.1996.561342.

Jennifer M. Davoren and Anil Nerode.
Logics for hybrid systems.
doi:10.1109/5.871305.