Cyber-Physical Systems Verification with KeYmaera X

André Platzer

Carnegie Mellon University
Outline

1. Cyber-Physical Systems
2. Foundation: Differential Dynamic Logic
3. ModelPlex: Model Safety Transfer
4. VeriPhy: Executable Proof Transfer
5. Applications
   - Airborne Collision Avoidance System
   - Safe Learning in CPS
6. Summary
1 Cyber-Physical Systems

2 Foundation: Differential Dynamic Logic

3 ModelPlex: Model Safety Transfer

4 VeriPhy: Executable Proof Transfer

5 Applications
   • Airborne Collision Avoidance System
   • Safe Learning in CPS

6 Summary
Cyber-Physical Systems Safety

Prospects: Safety & Efficiency

| (Autonomous) cars | Pilot support | Robots near humans |

Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.

André Platzer (CMU)
## Cyber-Physical Systems Safety

### Prospects: Safety & Efficiency

| (Autonomous) cars | Pilot support | Robots near humans |

---

## Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.
Cyber-Physical Systems Safety

Prospects: Safety & Efficiency

(Autonomous) cars  Pilot support  Robots near humans

Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.
Cyber-Physical Systems Safety

Prospects: Safety & Efficiency

(Autonomous) cars | Pilot support | Robots near humans

Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.
Cyber-Physical Systems Safety

Prospects: Safety & Efficiency

(Autonomous) cars  Pilot support  Robots near humans

Cyber-Physical Systems

CPSs combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.
1. Cyber-Physical Systems

2. Foundation: Differential Dynamic Logic

3. ModelPlex: Model Safety Transfer

4. VeriPhy: Executable Proof Transfer

5. Applications
   - Airborne Collision Avoidance System
   - Safe Learning in CPS

6. Summary
Approach: Proofs for Cyber-Physical Systems

KeYmaera X generates proofs

actions: \{acc, brake\}
motion: \(x'' = a\)

ModelPlex proof synthesizes

Monitor transfers safety

Model Safety

Compliance Monitor

Proof and invariant search

André Platzer (CMU)  
Cyber-Physical Systems Verification with KeYmaera X  
LFCS'20 3 / 24
CPS Analysis

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

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

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

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]

\[ x \neq m \]
CPS Analysis

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

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

\[ x \neq m \] for all runs

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

\[ \text{init} \rightarrow \text{post} \]
CPS Analysis

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

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

\[ x \neq m \wedge b > 0 \rightarrow [((\text{init}) SB(x, m)) \ a := -b ; \ x' = v, v' = a]^* x \neq m \]

all runs
Definition (Hybrid program)
\[ \alpha, \beta ::= x := e \mid ?Q \mid x' = f(x) & Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^* \]

Definition (Differential dynamic logic)
\[ P, Q ::= e \geq \tilde{e} \mid \neg P \mid P \land Q \mid P \lor Q \mid P \rightarrow Q \mid \forall x P \mid \exists x P \mid [\alpha]P \mid \langle \alpha \rangle P \]
Differential Dynamic Logic dL: Axiomatization

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

\[ \text{[?]} \ [?Q]P \leftrightarrow (Q \to P) \]

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

\[ \text{[\cup]} \ [\alpha \cup \beta]P \leftrightarrow [\alpha]P \land [\beta]P \]

\[ \text{[;]} \ [\alpha; \beta]P \leftrightarrow [\alpha][\beta]P \]

\[ \text{[*]} \ [\alpha^*]P \leftrightarrow P \land [\alpha][\alpha^*]P \]

\[ \text{K} \ [\alpha](P \to Q) \to ([\alpha]P \to [\alpha]Q) \]

\[ \text{I} \ [\alpha^*]P \leftrightarrow P \land [\alpha^*](P \to [\alpha]P) \]

\[ \text{C} \ [\alpha^*]\forall v>0 (P(v) \to \langle\alpha\rangle P(v-1)) \to \forall v (P(v) \to \langle\alpha^*\rangle \exists v \leq 0 P(v)) \]
**Theorem (Algebraic Completeness) (LICS’18,JACM’20)**

\( \text{dL calculus is a sound & complete axiomatization of algebraic invariants of polynomial differential equations. They are decidable by DI,DC,DG in dL.} \)

**Theorem (Semialgebraic Completeness) (LICS’18,JACM’20)**

\( \text{dL calculus with RI is a sound & complete axiomatization of semialgebraic invariants of differential equations. They are decidable in dL.} \)
**Theorem (Algebraic Completeness)** (LICS’18, JACM’20)

\[ dL \text{ calculus is a sound & complete axiomatization of algebraic invariants of polynomial differential equations. They are decidable} \]

\[ \text{DRI } [x' = f(x) \& Q] e = 0 \iff (Q \rightarrow e'^* = 0) \quad (Q \text{ open}) \]

**Theorem (Semialgebraic Completeness)** (LICS’18, JACM’20)

\[ dL \text{ calculus with RI is a sound & complete axiomatization of semialgebraic invariants of differential equations. They are decidable} \]

\[ \text{SAI } \forall x (P \rightarrow [x' = f(x)]P) \iff \forall x (P \rightarrow P'^*) \land \forall x (\neg P \rightarrow (\neg P)'^*) \]

Definable \( e'^* \) is short for *all/significant* Lie derivative w.r.t. ODE

Definable \( e'^* \) is w.r.t. backwards ODE \( x' = -f(x) \). Also for \( P \).

\[

e'^* = 0 \equiv e=0 \land (e')'^* = 0 \\
 e'^* \geq 0 \equiv e\geq0 \land (e=0 \rightarrow (e')'^* \geq 0) \\
 (P \land Q)'^* \equiv P'^* \land Q'^* \\
 (P \lor Q)'^* \equiv P'^* \lor Q'^*
\]
Differential Invariants for Differential Equations

**Differential Invariant**

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

**Differential Cut**

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

**Differential Ghost**

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

deductive power added \( DI \prec DI + DC \prec DI + DC + DG \)

\[
\omega[(e)'] = \sum_x \omega(x') \frac{\partial \llbracket e \rrbracket}{\partial x} (\omega)
\]
Differential Invariants for Differential Equations

Differential Invariant

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

Differential Cut

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

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 \( g(x, y) = a(x)y + b(x) \), so has long solution!

Springer’10, LMCS’12, LICS’12, JAR’17, LICS’18, JACM’20
Ex: Runaround Robot

\[(x, y) \neq o \rightarrow \left\{ \begin{array}{l}
\omega := -1 \\
\omega := 1 \\
\omega := 0
\end{array} \right. \}

\[x' = v, \quad y' = w, \quad v' = \omega w, \quad w' = -\omega v\]
Example (Runaround Robot)

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

Example (Runaround Robot)

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

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

1 Cyber-Physical Systems

2 Foundation: Differential Dynamic Logic

3 ModelPlex: Model Safety Transfer

4 VeriPhy: Executable Proof Transfer

5 Applications
   • Airborne Collision Avoidance System
   • Safe Learning in CPS

6 Summary
Approach: Proofs for Cyber-Physical Systems

CPS

Monitor transfers safety

ModelPlex proof synthesizes

KeYmaera X

generates proofs

Proof and invariant search

Compliance Monitor

Model Safety

actions: \{acc, brake\}
motion: \(x'' = a\)
Formal Verification in CPS Development

Real CPS

Verification Results

Proof

Reachability Analysis

safe
Formal Verification in CPS Development

Real CPS

Model $\alpha^*$

Control $\alpha_{\text{ctrl}}$

$\nu := \nu + 1$

sense

Plant $\alpha_{\text{plant}}$

$x' = \nu$

act

abstract

Proof

reachability analysis

Verification Results

Verifiably correct runtime model validation

Verification results about models only apply if CPS fits to the model
Challenge

Verification results about models
only apply if CPS fits to the model

Verifiably correct runtime model validation
ModelPlex ensures that verification results about models apply to CPS implementations.
ModelPlex ensures that verification results about models apply to CPS implementations.

**Insights**

- Verification results about models transfer to the CPS when validating model compliance.
- Compliance with model is characterizable in logic $dL$.
- Compliance formula transformed by $dL$ proof to monitor.
- Correct-by-construction provably correct model validation at runtime.

(model adequate? control safe? until next cycle?)
Characterizing State Relations in Logic

When are two states linked through a run of model $\alpha$?

- A prior state characterized by $x^-$
- A posterior state characterized by $x^+$

Semantical: $(\omega, \nu) \in [\alpha]$ (reachability relation of $\alpha$)

Model $\alpha$
When are two states linked through a run of model $\alpha$?

- A prior state characterized by $x^-$
- A posterior state characterized by $x^+$

**Model $\alpha$**

Semantical:

$$(\omega, \nu) \in \llbracket \alpha \rrbracket$$

Logical dL:

$$(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$$

**Lemma**

exists a run of $\alpha$ to a state where $x = x^+$
Characterizing State Relations in Logic

When are two states linked through a run of model $\alpha$?

- A prior state characterized by $x^-$
- A posterior state characterized by $x^+$

**Offline**

- Semantical: $(ω, ν) ∈ [α]$
- Logical dL: $(ω, ν) ⊧ ⟨α⟩(x = x^+)$
- Arithmetical: $(ω, ν) ⊧ F(x^-, x^+)$

**Lemma**

- Exists a run of $\alpha$ to a state where $x = x^+$
- dL proof

**Check at runtime (efficient)**
When are two states linked through a run of model $\alpha$?

- **a prior state characterized by** $x^-$
- **a posterior state characterized by** $x^+$

**Offline**

- **Semantical:** $(\omega, \nu) \in [\alpha]

  \iff \text{Lemma}

- **Logical dL:** $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$

  \uparrow \text{dL proof}

- **Arithmetical:** $(\omega, \nu) \models F(x^-, x^+)$

  \checkmark \text{check at runtime (efficient)}

**FMSD’16**
Logical Reductions for Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

$dL$ proof $A \rightarrow [\alpha]S$

Offline

Init $\omega \in [A]$ Safe $\nu \in [S]$

Semantical: $(\omega, \nu) \in [\alpha]$

Logical $dL$: $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$

Arithmetical: $(\omega, \nu) \models F(x^-, x^+)$

check at runtime (efficient)
Logical Reductions for Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof.

Logic reduces

CPS safety

to runtime
monitor with offline proof

$dL$ proof

$A \rightarrow [\alpha] S$

Model $\alpha$

 Offline:

$\omega \in [A]$ $\leftarrow$ Init

Safety: $\nu \in [S]$ $\rightarrow$ Safe

Semantical:

$(\omega, \nu) \in [\alpha]$

$\Leftrightarrow$ Lemma

Logical $dL$:

$(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$

$\uparrow$ $dL$ proof

Arithmetical:

$(\omega, \nu) \models F(x^-, x^+)$

check at runtime (efficient)
Outline

1. Cyber-Physical Systems
2. Foundation: Differential Dynamic Logic
3. ModelPlex: Model Safety Transfer
4. VeriPhy: Executable Proof Transfer
5. Applications
   - Airborne Collision Avoidance System
   - Safe Learning in CPS
6. Summary
Approach: Proofs for Cyber-Physical Systems

KeYmaera X generates proofs

Monitor transfers safety

ModelPlex proof synthesizes

Compliance Monitor

André Platzer (CMU)
VeriPhy: Automatic, Verified EXEs from Controllers

Hybrid Systems Theorem Proving → Abstract Controllers and Monitors → Sound Discrete Arithmetic → Sound Monitor Compilation → Cyber Physical System
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
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

André Platzer (CMU)

Cyber-Physical Systems Verification with KeYmaera X
VeriPhy: Takeaway Metaphor

Your Model

Low-Level Proofs

Safe CPS
VeriPhy: Takeaway Metaphor

Your Model

Low-Level Proofs

VeriPhy Pipeline (VeriPhy.org)

Safe CPS
Outline

1. Cyber-Physical Systems
2. Foundation: Differential Dynamic Logic
3. ModelPlex: Model Safety Transfer
4. VeriPhy: Executable Proof Transfer
5. Applications
   - Airborne Collision Avoidance System
   - Safe Learning in CPS
6. Summary
Airborne Collision Avoidance System ACAS X: Verify

- 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

Identified safe region for each advisory symbolically
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
Reinforcement Learning learns from experience of trying actions
Learning to Act in a CPS

RL chooses an action, observes outcome, reinforces in policy if successful
ModelPlex monitor inspects each decision, vetoes if unsafe
ModelPlex monitor gives early feedback about possible future problems. No need to wait till disaster strikes and propagate back.
dL benefits from RL optimization.  

RL benefits from dL safety signal.
Outline

1. Cyber-Physical Systems
2. Foundation: Differential Dynamic Logic
3. ModelPlex: Model Safety Transfer
4. VeriPhy: Executable Proof Transfer
5. Applications
   - Airborne Collision Avoidance System
   - Safe Learning in CPS
6. Summary
Acknowledgments

Logical Systems Lab at Carnegie Mellon University, Computer Science
Yong Kiam Tan, Brandon Bohrer, Nathan Fulton, Sarah Loos, Katherine Cordwell
Stefan Mitsch, Khalil Ghorbal, Jean-Baptiste Jeannin, Andrew Sogokon
differential dynamic logic
\[ dL = DL + HP \]

- Compositional formal verification
- Logic & proofs for CPS
- Small soundness core
- Proof by pointing
- Interactive proof clicking
- Tactical proof programming
- Proof search automation
- Flexible + modular API

KeYmaera X

http://keymaeraX.org/
Further CPS Topics

- Verified CPS systems by ModelPlex  
  FMSD’16
- Verified CPS execution by VeriPhy  
  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  
  FM’19
- Safe AI autonomy in CPS  
  AAAI’18 TACAS’19
- Correct model transformation  
  FM’14
- Refinement + system property proofs  
  LICS’16
- Automatic ODE proofs  
  LICS’18
- CPS information flow  
  LICS’18
- Hybrid games  
  TOCL’15

CPSs deserve proofs as safety evidence!
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

Part: Differential Equations Analysis
10. Differential Equations & Differential Invariants
11. Differential Equations & Proofs
12. Ghosts & Differential Ghosts
13. Differential Invariants & Proof Theory

Part: Adversarial Cyber-Physical Systems
14-17. Hybrid Systems & Hybrid Games

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

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

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

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 \textbf{no} operator \( \otimes \)
are free in the substitution on its argument \( \theta \)

\( (U\text{-admissible}) \)

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

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

$$US \quad \frac{\phi}{\sigma(\phi)}$$

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 no operator $\otimes$
are free in the substitution on its argument $\theta$ (U-admissible)

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

$$[v := -x][x' = v] x \geq 0 \leftrightarrow [x' = -x] x \geq 0$$
Uniform Substitution

### Theorem (Soundness)

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

<table>
<thead>
<tr>
<th>Modular interface: Prover vs. Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>$US \frac{\phi}{\sigma(\phi)}$</td>
</tr>
</tbody>
</table>

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 no operator $\otimes$ are free in the substitution on its argument $\theta$  

(U-admissible)

If you bind a free variable, you go to logic jail!

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

\[
[v := -x][x' = v] x \geq 0 \leftrightarrow [x' = -x] x \geq 0
\]

Clash
Differential Dynamic Logic dL: Semantics

**Definition (Hybrid program semantics)**

\[ ([·] : HP \rightarrow \mathcal{P}(S \times S)) \]

\[ [x := e] = \{ (\omega, \nu) : \nu = \omega \text{ except } \nu[x] = \omega[e] \} \]

\[ [? Q] = \{ (\omega, \omega) : \omega \in [Q] \} \]

\[ [x' = f(x)] = \{ (\varphi(0), \varphi(r)) : \varphi \models x' = f(x) \text{ for some duration } r \} \]

\[ [\alpha \cup \beta] = [\alpha] \cup [\beta] \]

\[ [\alpha; \beta] = [\alpha] \circ [\beta] \]

\[ [\alpha^*] = [\alpha]^* = \bigcup_{n \in \mathbb{N}} [\alpha^n] \]

**Definition (dL semantics)**

\[ ([·] : Fml \rightarrow \mathcal{P}(S)) \]

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

\[ \lnot P = [P]^c \]

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

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

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

\[ \exists x P = \{ \omega : \omega^r_x \in [P] \text{ for some } r \in \mathbb{R} \} \]
André Platzer.
Logical Foundations of Cyber-Physical Systems.
Springer, Cham, 2018.
doi:10.1007/978-3-319-63588-0.

André Platzer.
Differential dynamic logic for hybrid systems.

André Platzer.
Logics of dynamical systems.
In LICS [20], pages 13–24.

André Platzer.
A complete uniform substitution calculus for differential dynamic logic.

André Platzer.
The complete proof theory of hybrid systems.
In LICS [20], pages 541–550.
doi:10.1109/LICS.2012.64.

André Platzer and Yong Kiam Tan.
Differential equation axiomatization: The impressive power of differential ghosts.

André Platzer and Yong Kiam Tan.
Differential equation invariance axiomatization.

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.

André Platzer and Edmund M. Clarke.

André Platzer.

André Platzer.

Stefan Mitsch and André Platzer.
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. 

Nathan Fulton and André Platzer.
Safe reinforcement learning via formal methods: Toward safe control through proof and learning.

Nathan Fulton and André Platzer.
Verifiably safe off-model reinforcement learning.

André Platzer.
Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics.
doi:10.1007/978-3-642-14509-4.

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.
doi:10.1007/978-3-642-22438-6_34.

André Platzer.
Differential game logic.
André Platzer (CMU)

Cyber-Physical Systems Verification with KeYmaera X