Stefan Mitsch    André Platzer

Computer Science Department
Carnegie Mellon University, Pittsburgh, PA

Simplex for Hybrid System Models (FMSD’16)
Outline

1. Motivation
2. Learning Objectives
3. ModelPlex Runtime
   - ModelPlex Runtime Monitors
   - ModelPlex Compliance
4. ModelPlex
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Example: Water Tank
   - Controller Monitors
   - Prediction Monitors
5. Evaluation
6. Summary
Correctness Questions in Complex System Design

**Safety**  The system must be safe under all circumstances

**Liveness**  The system must reach a given goal

How do we make cyber-physical systems safe?

- Extensive testing?
- Code reviews?

When are we done? How many test cases are enough? Did we cover all relevant tests?
## Benefits of Logical Foundations for CPS V & V

### Proofs

<table>
<thead>
<tr>
<th>Benefits</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safety</td>
<td>Formalize system properties: What is “Safe”? “Reach goal”?</td>
</tr>
<tr>
<td>Models</td>
<td>Formalize system models, clarify behavior</td>
</tr>
<tr>
<td>Assumptions</td>
<td>Make assumptions explicit rather than silently</td>
</tr>
<tr>
<td>Predictions</td>
<td>Safety analysis predicts behavior for infinitely many cases</td>
</tr>
<tr>
<td>Constraints</td>
<td>Reveal invariants, switching conditions, operating conditions</td>
</tr>
<tr>
<td>Design</td>
<td>Invariants/proofs guide safe controller design</td>
</tr>
</tbody>
</table>

### Byproducts

<table>
<thead>
<tr>
<th>Benefits</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Determine design trade-offs &amp; feasibility early</td>
</tr>
<tr>
<td>Synthesis</td>
<td>Turn models into code &amp; safety monitors</td>
</tr>
<tr>
<td>Certificate</td>
<td>Proofs as evidence for certification</td>
</tr>
</tbody>
</table>

### Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeYmaera X</td>
<td>aXiomatic Tactical Theorem Prover for CPS</td>
</tr>
</tbody>
</table>

---

*Andre Platzer (CMU)*

*FCPS / 14: Verified Models & Verified Runtime Validation*
An aXiomatic Tactical Theorem Prover for CPS

http://keymaeraX.org/

KeyMaera X  Dashboard  Models  Proofs 2  Help

Hybrid Car  Auto  Normalize  Step back

Propositional  Quantifiers  Hybrid Programs  Differential Equations  Closing

implies(R(1)) & loop("v >= 0") & on("Induction Step", composeb(1) & choiceb(1) & assignb(1, 0::Nil) & choiceb(1, 1::Nil) & assignb(1, 1::Nil)), "Base Case", QE, "Use Ce

Base Case 4

Use Case 5

Induction Step 11

Proof Step

implies(A) [x:=a] p(x) <-> p(c)

G[]

Δ

Andre Platzer (CMU)  FCPS / 14: Verified Models & Verified Runtime Validation  FCPS 5 / 27
## An aXiomatic Tactical Theorem Prover for CPS


<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Small Core</strong></td>
<td>Increases trust, modularity, enables experimentation</td>
</tr>
<tr>
<td><strong>Tactics</strong></td>
<td>Bridging between small core and powerful reasoning steps</td>
</tr>
<tr>
<td><strong>Separation</strong></td>
<td>Tactics can make courageous inferences</td>
</tr>
<tr>
<td><strong>Search&amp;Do</strong></td>
<td>Search-based tactics that follow proof search strategies</td>
</tr>
<tr>
<td><strong>Interaction</strong></td>
<td>Interactive proofs mixed with tactical proofs and proof search</td>
</tr>
<tr>
<td><strong>Extensible</strong></td>
<td>Flexible for new algorithms, new tactics, new logics, new proof rules, new axioms, . . .</td>
</tr>
<tr>
<td><strong>Customize</strong></td>
<td>Modular user interface, API</td>
</tr>
</tbody>
</table>

(1677) (Hilbert) (Sequent++)

André Platzer (CMU)
# KeYmaera X Microkernel for Soundness

<table>
<thead>
<tr>
<th>Tool</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeYmaera X</td>
<td>1652</td>
</tr>
<tr>
<td>KeYmaera</td>
<td>65989</td>
</tr>
<tr>
<td>KeY</td>
<td>51328</td>
</tr>
<tr>
<td>Nuprl</td>
<td>15000 + 50000</td>
</tr>
<tr>
<td>MetaPRL</td>
<td>8196</td>
</tr>
<tr>
<td>Isabelle/Pure</td>
<td>8913</td>
</tr>
<tr>
<td>Coq</td>
<td>16538</td>
</tr>
<tr>
<td>HOL Light</td>
<td>396</td>
</tr>
<tr>
<td>PHAVer</td>
<td>30000</td>
</tr>
<tr>
<td>HSolver</td>
<td>20000</td>
</tr>
<tr>
<td>SpaceEx</td>
<td>100000</td>
</tr>
<tr>
<td>Flow*</td>
<td>25000</td>
</tr>
<tr>
<td>dReal</td>
<td>50000 + millions</td>
</tr>
<tr>
<td>HyCreate2</td>
<td>6081 + user model analysis</td>
</tr>
</tbody>
</table>

Disclaimer: Self-reported estimates of the soundness-critical lines of code + rules
Formal Verification in CPS Development

Real CPS

Proof

Reachability Analysis

Verifiably correct runtime model validation

safe

Verification Results

Challenge

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

André Platzer (CMU)
Formal Verification in CPS Development

Real CPS

Model $\alpha^*$

Control $\alpha_{ctrl}$

Plant $\alpha_{plant}$

$\nu := \nu + 1$

$sense$

$\dot{\nu} = \nu$

abstract

Proof

Reachability Analysis

Verification Results

Safe

Verifiably correct runtime model validation

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

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

Model

\[ \alpha \]

\[ \alpha_{\text{ctrl}} \]

\[ \nu := \nu + 1 \]

\[ \text{sense} \]

\[ \text{act} \]

\[ \text{Plant } \alpha_{\text{plant}} \]

\[ x' = \nu \]

Verification results about models

\textbf{only apply if CPS fits to the model}

\[ \leadsto \]

\textbf{Verifiably correct runtime model validation}

Reachability Analysis

…”

Verification Results
Outline

1 Motivation

2 Learning Objectives

3 ModelPlex Runtime
   - ModelPlex Runtime Monitors
   - ModelPlex Compliance

4 ModelPlex
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Example: Water Tank
   - Controller Monitors
   - Prediction Monitors

5 Evaluation

6 Summary
Learning Objectives
Verified Models & Verified Runtime Validation

proof in a model vs. truth in reality
tracing assumptions
turning provers upside down
correct-by-construction
dynamic contracts
proofs for CPS implementations

models vs. reality
inevitable differences
model compliance
architectural design
tame CPS complexity
prediction vs. run
runtime validation
online monitor

André Platzer (CMU)
Contributions

ModelPlex ensures that verification results about models apply to CPS implementations
Outline

1 Motivation
2 Learning Objectives
3 ModelPlex Runtime
   - ModelPlex Runtime Monitors
   - ModelPlex Compliance
4 ModelPlex
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Example: Water Tank
   - Controller Monitors
   - Prediction Monitors
5 Evaluation
6 Summary
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.

- model adequate?
- control safe?
- until next cycle?
ModelPlex at Runtime

“Simplex for Models”

Controller

Sensors

Actuators

André Platzer (CMU)
Compliance Monitor Checks CPS for compliance with model at runtime
- Model Monitor: model adequate?
- Controller Monitor: control safe?
- Prediction Monitor: until next cycle?

Fallback Safe action, executed when monitor is not satisfied (veto)

Challenge What conditions do the monitors need to check to be safe?
Is current CPS behavior included in the behavior of the model?

- CPS observed through sensors
- Model describes behavior of CPS between states

Detect non-compliance ASAP to initiate fallback actions while still safe
Is current CPS behavior included in the behavior of the model?

- CPS observed through sensors
- Model describes behavior of CPS between states

Detect non-compliance ASAP to initiate fallback actions while still safe
Is current CPS behavior included in the behavior of the model?

- CPS observed through sensors
- Model describes behavior of CPS between states

Detect non-compliance ASAP to initiate fallback actions while still safe

Challenge

Model describes behavior, but at runtime we get sampled observations

Transform model into observation-monitor

Detect non-compliance ASAP to initiate fallback actions while still safe
Outline

1 Motivation
2 Learning Objectives
3 ModelPlex Runtime
   • ModelPlex Runtime Monitors
   • ModelPlex Compliance
4 ModelPlex
   • Logical State Relations
   • Model Monitors
   • Correct-by-Construction Synthesis
   • Example: Water Tank
   • Controller Monitors
   • Prediction Monitors
5 Evaluation
6 Summary
Outline

Model $\alpha$

Model Monitor

André Platzer (CMU)
When are two states linked through a run of model $\alpha$?
When are two states linked through a run of model $\alpha$?

- Initial state.
- Model repeats.

$\omega \subseteq \text{offline}\left(\omega, \nu\right) \in \left[\alpha\right]$ 

Semantical: reachability relation of $\alpha$ 

$\Leftrightarrow \text{Lemma }\left(\omega, \nu\right) |\Leftarrow \langle \alpha \rangle \left(x = x^+\right)$

Logical $\exists$ a run of $\alpha$ to a state where $x = x^+$ 

Arithmetical $\exists L$ 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 $x^+$

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

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

A prior state characterized by $x$

Model $\alpha$

A posterior state characterized by $x^+$

Offline

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

\[\iff \text{Lemma} \]

Logical $d\mathcal{L}$: $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$

exists a run of $\alpha$ to a state where $x = x^+$
When are two states linked through a run of model $\alpha$?

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

- **Logical $d\mathcal{L}$**: $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$
  - $\implies$ $d\mathcal{L}$ proof

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

**Offline**

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

---

André Platzer (CMU)
When are two states linked through a run of model $\alpha$?

- **Semantical:** $(\omega, \nu) \in \llbracket \alpha \rrbracket$

- **Logical $d\mathcal{L}$:** $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$

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

There exists a run of $\alpha$ to a state where $x = x^+$, check at runtime (efficient).
Logical Reductions for Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

Semantical:

\( (\omega, \nu) \in \{\alpha\} \)

\( \uparrow \) 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)
Logic reduces CPS safety to runtime monitor with offline proof

\[ dL \text{ proof} \quad A \rightarrow [\alpha]S \]

Offline

Init \( \omega \models A \)

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)
Logic reduces CPS safety to runtime monitor with offline proof

\[ dL \text{ proof} \quad A \rightarrow [\alpha]S \]

**Offline**

\[ \text{Init} \quad \omega \models A \quad \text{Safe} \quad \nu \models S \]

**Semantical:**

\[ (\omega, \nu) \in \llbracket \alpha \rrbracket \]

\[ \Leftrightarrow \text{Lemma} \]

**Logical dL:**

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

\[ \Leftrightarrow \text{dL proof} \]

**Arithmetical:**

\[ (\omega, \nu) \models F(x, x^+) \]

\[ \text{check at runtime (efficient)} \]
Logic reduces CPS safety to runtime monitor with offline proof

Semantical: \((\omega, \nu) \in \llbracket \alpha \rrbracket\)

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

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

\[\begin{align*}
&\text{dL proof} \quad A \rightarrow [\alpha]S \\
&\text{Offline} \quad \text{Init} \quad \omega \models A \quad \text{Safe} \quad \nu \models S \\
&\text{Semantical:} \quad (\omega, \nu) \in \llbracket \alpha \rrbracket \\
&\text{Logical dL:} \quad (\omega, \nu) \models \langle \alpha \rangle (x = x^+) \\
&\text{Arithmetical:} \quad (\omega, \nu) \models F(x, x^+) \\
&\text{Lemma} \\
&\text{dL proof} \\
&\text{check at runtime (efficient)}
\end{align*}\]
Logic reduces CPS safety to runtime monitor with offline proof

\[ A \rightarrow [\alpha]S \]

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

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

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

check at runtime (efficient)

Semantical: \( (\omega, \nu) \in \llbracket \alpha \rrbracket \)

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

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

\( \models \) Lemma

\( \models \) \( dL \) proof

Init: \( \omega \models A \)

Safe: \( \nu \models S \)
Logics reduces CPS safety to **runtime** monitor with offline proof.

Offline

1. **Init** $\omega \models A$
2. **Safe** $\nu \models S$

- **Semantical:** $(\omega, \nu) \in \llbracket \alpha \rrbracket$
  - $\models$ **Lemma**

- **Logical $\mathcal{L}$:** $(\omega, \nu) \models \langle \alpha \rangle(x = x^+)$
  - $\models$ **$\mathcal{L}$ proof**

- **Arithmetical:** $(\omega, \nu) \models F(x, x^+)$
  - check at runtime (efficient)
Logic reduces CPS safety to runtime monitor with offline proof

\[ A \rightarrow [\alpha]S \]

\[ \omega \quad \text{Model } \alpha \quad \nu \]

**Offline**

- **Init** \( \omega \models A \)
- **Safe** \( \nu \models S \)

**Semantical:**

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

\[ \uparrow \quad \text{Lemma} \]

**Logical \( d\mathcal{L} \):**

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

\[ \uparrow \quad \text{d\( \mathcal{L} \) proof} \]

**Arithmetical:**

\[ (\omega, \nu) \models F(x, x^+) \]

\[ \uparrow \quad \text{check at runtime (efficient)} \]
Logic reduces CPS safety to runtime monitor with offline proof

\[ A \rightarrow [\alpha]S \]

Offline

Init: \( \omega \models A \)
Safe: \( \nu \models S \)

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

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

Arithmetical:
\[ (\omega, \nu) \models F(x, x^+) \]

check at runtime (efficient)
Logic reduces CPS safety to runtime monitor with offline proof

\[ A \rightarrow [\alpha] S \]

**Not initial state. Model repeats...**

**d\(\mathcal{L}\) proof**

**Offline**

**Init** \( \omega \models A \)  
**Safe** \( \nu \models S \)

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

\[ \updownarrow \text{Lemma} \]

**Logical d\(\mathcal{L}\):**  
\( (\omega, \nu) \models \langle \alpha \rangle (x = x^+) \)

\[ \uparrow \text{d\(\mathcal{L}\) proof} \]

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

\[ \text{check at runtime (efficient)} \]
Logical Reductions for $\alpha^*$ Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

$A \rightarrow [\alpha^*]S$

Offline

Init $\omega \models A$

Safe $\nu \models S$

Semantical:

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

$\uparrow$ Lemma

Logical $d\mathcal{L}$:

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

$\uparrow$ $d\mathcal{L}$ proof

Arithmetical:

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

check at runtime (efficient)
Theorem (Model Monitor Correctness) (FMSD'16)

"System safe as long as monitor satisfied."

André Platzer (CMU)

FCPS 15 / 27
\( dL \) proof \( A \rightarrow [\alpha^*]S \)

Andrē Platzer (CMU)
ModelPlex Model Monitor Correctness

\[ \text{dL proof } A \rightarrow [\alpha^*] S \]

\[ \text{Init } 0 \models A \]

Andre Platzer (CMU)  
FCPS / 14: Verified Models & Verified Runtime Validation
0 \models A \\

\text{dL proof } A \rightarrow [\alpha^*]S \\

\text{Check } (i, i+1) \models \langle \alpha \rangle x=x^+
Theorem (Model Monitor Correctness) (FMSD'16)

"System safe as long as monitor satisfied."

\[ \text{proof } A \rightarrow [\alpha^*]S \]
Theorem (Model Monitor Correctness)  

"System safe as long as monitor satisfied."

\(d\mathcal{L} \text{ proof } A \rightarrow [\alpha^*]S\)

Init \(0 \models A\)

Check \((i, i+1) \models \langle \alpha \rangle x = x^+\)

Safe \(i+1 \models S\)
Proof calculus of $\mathcal{dL}$ executes models symbolically

Proof attempt

$\langle \alpha(x) \rangle (x = x^+)
Proof calculus of $d\mathcal{L}$ executes models symbolically

- Proof attempt:
  - $\langle\text{climb} \cup \text{descend}\rangle(x = x^+)$

$\langle\text{climb} \cup \text{descend}\rangle P \leftrightarrow \langle\text{climb}\rangle P \lor \langle\text{descend}\rangle P$
Provably Correct Synthesis of Monitors

- Proof calculus of $\mathcal{dL}$ executes models symbolically

Model $\alpha$

prior state $x_{i-1}$ \[ \text{climb} \] \[ \text{descend} \] \[ i \] \[ \text{posterior state } x_i \]

proof attempt

\[ \langle \text{climb} \cup \text{descend} \rangle (x = x^+) \]

\[ \langle \text{climb} \rangle (x = x^+) \] \[ \lor \] \[ \langle \text{descend} \rangle (x = x^+) \]
Proof calculus of $d\mathcal{L}$ executes models symbolically

Model $\alpha$

prior state $x$

$i-1$  

climb

descend

posterior state $x^+$

$i$

proof attempt

$\langle \text{climb} \cup \text{descend} \rangle (x = x^+)$

$\langle \text{climb} \rangle (x = x^+)$  

$\langle \text{descend} \rangle (x = x^+)$

$F_1(x, x^+)$  

$F_2(x, x^+)$

Monitor:
The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\Rightarrow$ close at runtime

Immediate detection of model violation $\Rightarrow$ Mitigates safety issues with safe fallback action
Proof calculus of $d\mathcal{L}$ executes models symbolically

\[ \langle \text{climb} \cup \text{descend} \rangle (x = x^+) \]

\[ \langle \text{climb} \rangle (x = x^+) \quad \langle \text{descend} \rangle (x = x^+) \]

\[ F_1(x, x^+) \quad F_2(x, x^+) \]

Monitor: \[ F_1(x, x^+) \lor F_2(x, x^+) \]
Proof calculus of $dL$ executes models symbolically

Proof attempt

$\langle \text{climb} \cup \text{descend} \rangle (x = x^+) \\
\langle \text{climb} \rangle (x = x^+) \lor \langle \text{descend} \rangle (x = x^+) \\
F_1(x, x^+) \lor F_2(x, x^+)$

Monitor: $F_1(x, x^+) \lor F_2(x, x^+)$

The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\rightsquigarrow$ close at runtime
Provably Correct Synthesis of Monitors

- Proof calculus of $d\mathcal{L}$ executes models symbolically

\[
\text{Model } \alpha \quad \text{prior state } x \quad i-1 \quad \text{climb} \quad \text{descend} \quad i \quad \text{posterior state } x^+
\]

Model Monitor

Immediate detection of model violation

$\leadsto$ Mitigates safety issues with safe fallback action

\[
F_1(x, x^+) \quad F_2(x, x^+)
\]

Monitor: $F_1(x, x^+) \lor F_2(x, x^+)$

- The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\leadsto$ close at runtime
Water Tank Example: Monitor Conjecture

Variables

- \(x\) current level
- \(m\) maximum level
- \(\varepsilon\) control cycle
- \(f\) flow

Model and Safety Property

\[
0 \leq x \leq m \land \varepsilon > 0 \rightarrow \left[ \left( f := *; \ ? (-1 \leq f \leq \frac{m - x}{\varepsilon}); \right) \right.
\]
\[
t := 0; \ (x' = f, \ t' = 1 \land x \geq 0 \land t \leq \varepsilon) \left. \right) \right)^* \]
\[
A \upharpoonright \text{const} \begin{cases} \gamma_{V_m}^+ \end{cases}
\]
\[
(0 \leq x \leq m) \]

Model Monitor Specification Conjecture

\[
\varepsilon > 0 \rightarrow \left\langle f := *; \ ? (-1 \leq f \leq \frac{m - x}{\varepsilon}); \ 
\right. \left. t := 0; \ (x' = f, \ t' = 1 \land x \geq 0 \land t \leq \varepsilon) \right\rangle \left( x = x^+ \land f = f^+ \land t = t^+ \right) \]

André Platzer (CMU)
Water Tank Example: Nondeterministic Assignment

Proof Rules

\[\langle * \rangle \frac{\Gamma \vdash \exists X \langle x := X \rangle P, \Delta}{\Gamma \vdash \langle x := * \rangle P, \Delta} \quad (X \text{ is a new logical variable})\]

\[\exists R \quad \frac{\exists \Gamma \vdash p(e), \exists x p(x), \Delta}{\exists \Gamma \vdash \exists x p(x), \Delta} \quad (e \text{ is any arbitrary term})\]

\[WR \quad \frac{\Gamma \vdash \Delta}{\Gamma \vdash P, \Delta}\]

Sequent Deduction

\[\exists R, WR \quad \frac{A \vdash \langle f := F \rangle \langle ?-1 \leq f \leq \frac{m-x}{\varepsilon} \rangle \langle \text{plant} \rangle \gamma^+}{A \vdash \exists F \langle f := F \rangle \langle ?-1 \leq f \leq \frac{m-x}{\varepsilon} \rangle \langle \text{plant} \rangle \gamma^+} \quad \langle ?-1 \leq f \leq \frac{m-x}{\varepsilon} \rangle \langle \text{plant} \rangle \gamma^+\]

\[\langle * \rangle \quad \frac{A \vdash \langle f := * \rangle \langle ?-1 \leq f \leq \frac{m-x}{\varepsilon} \rangle \langle \text{plant} \rangle \gamma^+}{A \vdash \langle f := f^+ \rangle \langle \gamma^+ \rangle} \quad \exists R, WR \quad \ldots\]

with Opt. 1 (anticipate \( f = f^+ \) from \( \gamma^+ \))
Proof Rules

\[ \exists T \geq 0 \left( (\forall 0 \leq t \leq T \langle x := y(t) \rangle Q) \land \langle x := y(T) \rangle P \right) \]
\[ \implies \langle x' = f(x) \& Q \rangle P \]
(y(t) solution \( T, t \) new)

\[ \text{QE} \quad \frac{\text{QE}(P)}{P} \]
(iff \( P \leftrightarrow \text{QE}(P) \) in first-order real arithmetic)

Sequent Deduction

\[ A \vdash F = f^+ \land x^+ = x + Ft^+ \land t^+ \geq 0 \land x \geq 0 \land \varepsilon \geq t^+ \geq 0 \land Ft^+ + x \geq 0 \]
\[ \text{QE} \]
\[ A \vdash \forall 0 \leq \tilde{t} \leq T \left( x + f^+ \tilde{t} \geq 0 \land \tilde{t} \leq \varepsilon \right) \land F = f^+ \land x^+ = x + Ft^+ \land t^+ = t^+ \]
\[ \exists R, WR \]
\[ A \vdash \exists T \geq 0 \left( (\forall 0 \leq \tilde{t} \leq T \left( x + f^+ \tilde{t} \geq 0 \land \tilde{t} \leq \varepsilon \right)) \land F = f^+ \land (x^+ = x + FT \land t^+ = T) \right) \]
\[ \langle ' \rangle \quad A \vdash \langle f := F; t := 0 \rangle \langle \{ x' = f, t' = 1 \ & x \geq 0 \land t \leq \varepsilon \} \rangle \gamma^+ \]
Water Tank Example: Synthesized Model Monitor

Input: Model and Safety Property

\[ 0 \leq x \leq m \land \varepsilon > 0 \rightarrow \begin{cases} f := \ast; & ? \left( -1 \leq f \leq \frac{m-x}{\varepsilon} \right); \\ t := 0; & (x' = f, \ t' = 1 \land x \geq 0 \land t \leq \varepsilon) \end{cases} \] 

Output: Synthesized Model Monitor

\[ -1 \leq f^+ \leq \frac{m-x}{\varepsilon} \land x^+ = x + f^+ t^+ \land x \geq 0 \land x + f^+ t^+ \geq 0 \land \varepsilon \geq t^+ \geq 0 \]

Proof (Generated by ModelPlex tactic).

A proof of correctness of the synthesized model monitor.

Andre Platzer (CMU)
For typical models $\alpha;\text{plant}$ we can check earlier
Controller Monitor: Early Compliance Checks

Model Monitor

prior state $x_i$ Model $\alpha$ \rightarrow posterior state $x_{i+1}$

Theorem $\forall (i, \nu) \in [\text{ctrl}]$:

- Semantical: reachability relation of ctrl
  $\Leftrightarrow$ Theorem $(i, \nu) = \langle \text{ctrl} \rangle (x = x +)$

- Logical $d^L$: exists a run of ctrl to a state where $x = x +$
  $\Leftrightarrow$ Proof $(i, \nu) = F(x, x +)$

- Arithmetical: check at runtime (efficient)

Theorem (Controller Monitor Correctness) (FMSD'16)

"Controller safe & in plant bounds as long as monitor satisfied."

Immediate detection of unsafe control before actuation $\Rightarrow$ Safe execution of unverified implementations in perfect environments

André Platzer (CMU)
Controller Monitor: Early Compliance Checks

Prior state $x \xrightarrow{} \omega \xrightarrow{} \nu \xrightarrow{} i+1$

Model $\alpha$

Controller Monitor before actuation

Posterior state $x^+$

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket \xrightarrow{} \text{reachability relation of ctrl}$

Controller Monitor before actuation

Controller Monitor: Immediate detection of unsafe control before actuation

Safe execution of unverified implementations in perfect environments

Theorem (Controller Monitor Correctness) (FMSD'16)

"Controller safe & in plant bounds as long as monitor satisfied."

Andrée Platzer (CMU)

FCPS / 14: Verified Models & Verified Runtime Validation

FCPS 22 / 27
Controller Monitor: Early Compliance Checks

Model $\alpha$

prior state $x$

Controller Monitor before actuation

posterior state $x^+$

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket$

Logical $dL$: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

Controller Monitor before actuation

Controller Monitor before actuation

Offline

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket$

Logical $dL$: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

exists a run of ctrl to a state where $x = x^+$
Controller Monitor: Early Compliance Checks

Model $\alpha$

prior state $x$

Controller Monitor before actuation
posterior state $x^+$

Controller Monitor before actuation

Offline

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket$

Logical $\mathcal{L}$: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

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

exists a run of $\text{ctrl}$ to a state where $x = x^+$

d$\mathcal{L}$ proof

check at runtime (efficient)
Controller Monitor: Early Compliance Checks

prior state $x$ \quad $\omega$ \quad Model $\alpha$ \quad ctrl \quad plant \quad $\nu$ \quad $i+1$

Controller Monitor before actuation
posterior state $x^+$

Offline

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket$

Logical $\mathcal{L}$:
$(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

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

Semantical: $(\omega, \nu) \in \llbracket \text{ctrl} \rrbracket$

Logical $\mathcal{L}$:
$(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

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

Theorem (Controller Monitor Correctness) (FMSD’16)

“Controller safe & in plant bounds as long as monitor satisfied.”

Theorem (Controller Monitor Correctness) (FMSD’16)

“Controller safe & in plant bounds as long as monitor satisfied.”

check at runtime (efficient)

exists a run of ctrl to a state where $x = x^+$
Controller Monitor: Early Compliance Checks

Controller Monitor before actuation
posterior state $x^+$

Immediate detection of unsafe control before actuation
\[ \leadsto \text{Safe execution of unverified implementations in perfect environments} \]

Logical $\mathcal{L}$:
\[(\omega, \nu) \models (\text{ctrl})(x = x^+)\]
\[\uparrow \text{dL proof}\]

Arithmetical:
\[(\omega, \nu) \models F(x, x^+)\]
\[\text{check at runtime (efficient)}\]

Theorem (Controller Monitor Correctness) (FMSD'16)

“Controller safe & in plant bounds as long as monitor satisfied.”

André Platzer (CMU)
Safe despite evolution with disturbance?

Prediction is very difficult, especially if it's about the future. 

[Nils Bohr]
Safe despite evolution with disturbance?

“Prediction is very difficult, especially if it’s about the future.” [Nils Bohr]
Prediction Monitor: Compliance with Disturbance

Online $(i, μ) \mid = (ctrl) (x = x^+ \land [plant] ϕ)$

Invariant $ϕ$ implies safety $S$ (known from safety proof)

Logical $⇑L$ proof $(i, μ) \mid = F(x, x^+)$

Arithmetical:

Prediction Monitor with Disturbance

Proactive detection of unsafe control before actuation $\Rightarrow$ Safety in realistic environments

André Platzer (CMU)
plant of the form \( x' = f(x) \& Q \)

Model \( \alpha \)

Prior state \( x \)

\( \omega \rightarrow \nu \rightarrow i+1 \)

Prediction Monitor before actuation

Prediction Monitor

Posterior state \( x^+ \)
Prediction Monitor: Compliance with Disturbance

Prior state $x$: $\omega$  

Model $\alpha$: $\nu$  

Posterior state $x^+$: $i+1$  

Prediction Monitor before actuation

states reachable within $\varepsilon$ time

Time bound $t := 0; (x' = f(x), \ t' = 1 & Q \wedge t \leq \varepsilon)$
disturbance \( t := 0; (f(x) - \delta \leq x' \leq f(x) + \delta, t' = 1 \& Q \land t \leq \varepsilon) \)

Prior state \( x \) \( \xrightarrow{\omega} \nu \)

Model \( \alpha \)

Plant \( \xrightarrow{\text{ctrl}} \cdot \xrightarrow{\text{plant}} i+1 \)

Prediction Monitor before actuation

Posterior state \( x^+ \)

States reachable within \( \varepsilon \) time
disturbance \( t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 & Q \land t \leq \varepsilon \right) \)

Prediction Monitor: Compliance with Disturbance

Prior state \( x \)

Model \( \alpha \)

Plant

Posterior state \( x^+ \)

Prediction Monitor before actuation

States reachable within \( \varepsilon \) time

Offline

Logical \( d\mathcal{L} \): \( (\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \land [\text{plant}] \varphi) \)

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

Invariant \( \varphi \) implies safety \( S \)

(known from safety proof)
Prediction Monitor: Compliance with Disturbance

\[
\text{disturbance } t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 \land Q \land t \leq \varepsilon \right)
\]

Proactive detection of unsafe control before actuation despite disturbance

\[ \rightsquigarrow \text{Safety in realistic environments} \]

Logical \( d\mathcal{L} \): \((\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \land [\text{plant}]\varphi) \)

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

Invariant \(\varphi\) implies safety \(S\) (known from safety proof)
Outline

1 Motivation

2 Learning Objectives

3 ModelPlex Runtime
   • ModelPlex Runtime Monitors
   • ModelPlex Compliance

4 ModelPlex
   • Logical State Relations
   • Model Monitors
   • Correct-by-Construction Synthesis
   • Example: Water Tank
   • Controller Monitors
   • Prediction Monitors

5 Evaluation

6 Summary
Evaluating on hybrid system case studies:

- Water tank
- Cruise control
- Traffic control
- Ground robots
- Train control

Model sizes: 5–16 variables
Monitor sizes: 20–150 operations
Synthesis duration: 0.3–23 seconds (axiomatic) 6.2–211 (sequent)
ModelPlex tactic produces correct-by-construction monitor in KeYmaera X

Theorem: ModelPlex is decidable and monitor synthesis fully automated for controller monitor synthesis and for important classes.
Outline

1. Motivation
2. Learning Objectives
3. ModelPlex Runtime
   - ModelPlex Runtime Monitors
   - ModelPlex Compliance
4. ModelPlex
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Example: Water Tank
   - Controller Monitors
   - Prediction Monitors
5. Evaluation
6. Summary

André Platzer (CMU)
ModelPlex ensures that proofs apply to real CPS

- Validate model compliance
- Characterize compliance with model in logic
- Prover transforms compliance formula to executable monitor
- Provably correct runtime model validation

![Diagram of ModelPlex system]

- Model Monitor: model adequate?
- Controller Monitor: control safe?
- Prediction Monitor: until next cycle?
Proof Model safe!

Validated by ModelPlex

Model safe!
Stefan Mitsch and André Platzer.
ModelPlex: Verified runtime validation of verified cyber-physical system models.
Special issue of selected papers from RV’14.

Stefan Mitsch and André Platzer.
ModelPlex: Verified runtime validation of verified cyber-physical system models.
doi:10.1007/978-3-319-11164-3_17.

André Platzer.
Logics of dynamical systems.

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


Nathan Fulton and André Platzer.
A logic of proofs for differential dynamic logic: Toward independently checkable proof certificates for dynamic logics.

André Platzer.
Differential dynamic logic for hybrid systems.

André Platzer.
A uniform substitution calculus for differential dynamic logic.
doi:10.1007/978-3-319-21401-6_32.