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
   - 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

1 Motivation

2 Learning Objectives

3 ModelPlex Runtime
   - ModelPlex Runtime
   - ModelPlex Compliance

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

5 Evaluation

6 Summary
Formal Verification in CPS Development

Real CPS

Proof

Reachability Analysis

Verification Results

safe

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

Real CPS

Model $\alpha^*$

Control $\alpha_{ctrl}$

$sense \quad v := v + 1 \quad act$

Plant $\alpha_{plant}$

$x' = v$

Proof

Reachability Analysis

Verification Results

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

$\Rightarrow$ Verifiably correct runtime model validation

André Platzer (CMU)

FCPS / 14: Verified Models & Verified Runtime Validation
Verification results about models
only apply if CPS fits to the model
Formal Verification in CPS Development

Real CPS

Model

\[ v := v + 1 \]

Plant \( \alpha_{\text{plant}} \)

\[ x' = v \]

\[ \text{sense act} \]

\[ \text{Reachability Analysis} \ldots \]

\[ \text{Verification Results} \]

Challenge

Verification results about models

**only apply if CPS fits to the model**

\[ \rightsquigarrow \text{Verifiably correct runtime model validation} \]
1 Motivation

2 **Learning Objectives**

3 ModelPlex Runtime
   - ModelPlex Runtime
   - 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

CT
M&C
CPS

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

1 Motivation

2 Learning Objectives

3 ModelPlex Runtime
   • ModelPlex Runtime
   • 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”
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

Challenge

Model describes behavior, but at runtime we get sampled observations

\[ \sim \rightarrow \text{Transform model into observation-monitor} \]

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

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

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

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

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

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

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

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

**Lemma**

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

- **Semantical:** $(\omega, \nu) \in \llbracket \alpha \rrbracket$
  - Lemma: $(\omega, \nu) \models \langle \alpha \rangle (x = x^+)$
  - $d\mathcal{L}$ proof

- **Logical $d\mathcal{L}$:** $(\omega, \nu) \models F(x, x^+)$
  - check at runtime (efficient)

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

- **Offline**

- **Model $\alpha$**
  - a prior state characterized by $x$
  - a posterior state characterized by $x^+$
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:**

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

**Logical $d\mathcal{L}$:**

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

**Arithmetical:**

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

- **Lemma:**
  - exists a run of $\alpha$ to a state where $x = x^+$

- **$d\mathcal{L}$ proof:**

- **check at runtime (efficient)**

Andrzej Platzer (CMU)
Logic reduces CPS safety to runtime monitor with offline proof

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

Offline

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

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

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

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

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

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

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

- **dL proof**: \( A \rightarrow [\alpha]S \)
- **Semantical**: \((\omega, \nu) \in [\alpha]\)
- **Logical dL**: \((\omega, \nu) \models \langle \alpha \rangle (x = x^+)\)
- **Arithmetical**: \((\omega, \nu) \models F(x, x^+)\)

\(\omega\) to \(\nu\) with\( \cap \)

- **Offline**: \(\text{Init } \omega \models A\) \(\quad \text{Safe } \nu \models S\)
- **Lemma**: \((\omega, \nu) \models \langle \alpha \rangle (x = x^+)\)
- **dL proof**: check at runtime (efficient)
Logic reduces CPS safety to runtime monitor with offline proof

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

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

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

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

**Semantical:**
\[
(\omega, \nu) \in [\alpha] \\
\uparrow \text{ Lemma}
\]

**Logical dL:**
\[
(\omega, \nu) \models \langle \alpha \rangle (x = x^+) \\
\uparrow \text{ dL proof}
\]

**Arithmetical:**
\[
(\omega, \nu) \models F(x, x^+) \\
\rightarrow \text{ check at runtime (efficient)}
\]
Logic reduces CPS safety to runtime monitor with offline proof.

Semantical:

\[(\omega, \nu) \in \semantics{\alpha}\]

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

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

Arithmetical:

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

\(d\mathcal{L}\) proof

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

Init: \[\omega \models A\]

Safe: \[\nu \models S\]

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

\[
\begin{align*}
\text{Model } \alpha & \quad \Downarrow \quad \text{Lemma} \\
(\omega, \nu) & \models \langle \alpha \rangle (x = x^+) \\
\Downarrow & \quad \text{dL proof} \\
(\omega, \nu) & \models F(x, x^+)
\end{align*}
\]

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

\[
\begin{align*}
\text{dL proof} & \quad A \rightarrow [\alpha]S \\
\text{Offline} & \quad \text{Init } \omega \models A \quad \text{Safe } \nu \models S \\
\text{Semantical:} & \quad (\omega, \nu) \in [\alpha] \\
\text{Logical dL:} & \quad (\omega, \nu) \models \langle \alpha \rangle(x = x^+) \\
\text{Arithmetical:} & \quad (\omega, \nu) \models F(x, x^+) \\
\end{align*}
\]

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

Not initial state. Model repeats...

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

Offline

Init \( \omega \models A \)

Safe \( \nu \models S \)

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

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

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

Check at runtime (efficient)

Andre Platzer (CMU)
Logical Reductions for $\alpha^*$ Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

$dL$ proof

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

Offline

Init $\omega \models A$

Safe $\nu \models S$

Semantical:

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

Logical $dL$:

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

Arithmetical:

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

check at runtime (efficient)
Theorem (Model Monitor Correctness)

"System safe as long as monitor satisfied." FMSD'16

André Platzer (CMU)
\[ \mathcal{L} \text{ proof } A \rightarrow [\alpha^*] S \]
Proof: Let $\alpha$ be a model. We prove $A \rightarrow [\alpha^*]S$ using the following diagram:

\[
\begin{align*}
0 & \xrightarrow{\text{Model } \alpha} i-1 & \xrightarrow{\text{Model } \alpha} i & \xrightarrow{\text{Model } \alpha} i+1 \\
\text{Init } 0 \models A
\end{align*}
\]

This diagram shows the progression from $0$ to $i+1$, maintaining the model $\alpha$ at each step, and verifying that $A$ holds at each state.

Theorem (Model Monitor Correctness): "System safe as long as monitor satisfied." FMSD'16

André Platzer (CMU)
ModelPlex Model Monitor Correctness

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

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

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

"System safe as long as monitor satisfied." FMSD’16

Andrè Platzer (CMU)
Theorem (Model Monitor Correctness)

"System safe as long as monitor satisfied."

FMSD’16
Proof calculus of \( d\mathcal{L} \) executes models symbolically

\[
\begin{align*}
\text{prior state } x & \rightarrow \text{Model } \alpha \rightarrow \text{posterior state } x^+ \\
\text{proof attempt} & \quad \langle \alpha(x) \rangle (x = x^+) 
\end{align*}
\]
Proof calculus of $\mathcal{dL}$ executes models symbolically.

- Proof attempt:
  \[ \langle \text{climb} \cup \text{descend} \rangle (x = x^+) \]
  \[ \langle \text{climb} \cup \text{descend} \rangle P \iff \langle \text{climb} \rangle P \lor \langle \text{descend} \rangle P \]

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

Proof attempt

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

Language Monitor

The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model ⇝ close at runtime

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

Proof attempt

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

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

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

$F_1(x, x^+)$

$F_2(x, x^+)$
Proof calculus of $d\mathcal{L}$ executes models symbolically

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

Monitor: $F_1(x, x^+) \lor F_2(x, x^+)$
• Proof calculus of $\mathcal{dL}$ executes models symbolically

![Diagram showing state transitions and proof attempt]

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

Model $\alpha$

prior state $x_i-1$  $\xrightarrow{\text{climb}}$  $\xleftarrow{\text{descend}}$  $x_i$  posterior state $x_i^+$

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

Immediate detection of model violation

$\leadsto$ Mitigates safety issues with safe fallback action

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

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

Model Monitor

Andrée Platzer (CMU)
Water Tank Example: Monitor Conjecture

Variables

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

Model and Safety Property

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

Model Monitor Specification Conjecture

\[
\varepsilon > 0 \rightarrow \left\langle \begin{array}{l}
f := \ast; \ ? \ (-1 \leq f \leq \frac{m-x}{\varepsilon}); \\
t := 0; \ (x' = f, \ t' = 1 \land x \geq 0 \land t \leq \varepsilon)
\end{array} \right\rangle \left( x = x^+ \land f = f^+ \land t = t^+ \right)
\]
## Water Tank Example: Nondeterministic Assignment

### Proof Rules

<table>
<thead>
<tr>
<th>Rule</th>
<th>Premise</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>⟨∗⟩</td>
<td>( \Gamma \vdash \exists X \langle x := X \rangle P, \Delta )</td>
<td>( \Gamma \vdash \langle x := \ast \rangle P, \Delta ) ( (X ) is a new logical variable)</td>
</tr>
<tr>
<td>( \exists R )</td>
<td>( \Gamma \vdash p(e), \exists x p(x), \Delta )</td>
<td>( \Gamma \vdash \exists x p(x), \Delta ) ( (e ) is any arbitrary term)</td>
</tr>
<tr>
<td>WR</td>
<td>( \Gamma \vdash \Delta )</td>
<td>( \Gamma \vdash \phi, \Delta )</td>
</tr>
</tbody>
</table>

### Sequent Deduction

\[ A \vdash \langle f := F \rangle \langle ?-1 \leq f \leq \frac{m-x}{\epsilon} \rangle \langle \text{plant} \rangle \gamma^+ \]

\( A \vdash \exists F \langle f := F \rangle \langle ?-1 \leq f \leq \frac{m-x}{\epsilon} \rangle \langle \text{plant} \rangle \gamma^+ \)

\[ A \vdash \langle f := f^+ \rangle \langle ?-1 \leq f \leq \frac{m-x}{\epsilon} \rangle \langle \text{plant} \rangle \gamma^+ \]

With Opt. 1 (anticipate \( f = f^+ \) from \( \gamma^+ \))
Water Tank Example: Differential Equations

Proof Rules

\[
\exists T \geq 0 ((\forall 0 \leq t \leq T \langle x := y(t) \rangle Q) \land \langle x := y(T) \rangle P) \quad (y(t) \text{ solution } T, t \text{ new})
\]

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

QE

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

Sequent Deduction

\[
A \vdash F = f^+ \land x^+ = x + Ft^+ \land t^+ \geq 0 \land x \geq 0 \land \varepsilon \geq t^+ \geq 0 \land Ft^+ + x \geq 0
\]

\[
\frac{\text{QE}}{A \vdash \forall 0 \leq \tilde{t} \leq T (x + f^+\tilde{t} \geq 0 \land \tilde{t} \leq \varepsilon) \land F = f^+ \land x^+ = x + Ft^+ \land t^+ = t^+}
\]

\[
\frac{\exists R, \text{WR}}{A \vdash \exists T \geq 0 (\forall 0 \leq \tilde{t} \leq T (x + f^+\tilde{t} \geq 0 \land \tilde{t} \leq \varepsilon)) \land F = f^+ \land (x^+ = x + FT \land t^+ = T)}
\]

\[
\langle f := F; t := 0 \rangle \langle \{x' = f, t' = 1 \& x \geq 0 \land t \leq \varepsilon\} \rangle \Upsilon^+
\]
Input: Model and Safety Property

\[
0 \leq x \leq m \land \varepsilon > 0 \rightarrow \left( f := *; ? ( -1 \leq f \leq \frac{m-x}{\varepsilon} );
\right.
\]

\[
t := 0; (x' = f, t' = 1 \land x \geq 0 \land t \leq \varepsilon )^* \left( \underbrace{0 \leq x \leq m}_{A} \right)
\]

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.
For typical models $\text{ctrl} \; ; \; \text{plant}$ we can check earlier
Controller Monitor: Early Compliance Checks

Prior state $x_i \xrightarrow{\text{Model } \alpha} x_{i+1}$

Model Monitor

Controller Monitor before actuation

Posterior state $x_i + 1$

André Platzer (CMU)
Controller Monitor: Early Compliance Checks

Controller Monitor before actuation

prior state $x^-$ \(\omega\) \(\rightarrow\) \(\nu\) \(\rightarrow\) \(i+1\)

Model $\alpha$

Controller Monitor before actuation

posterior state $x^+$

Semantical: \((\omega, \nu) \in \llbracket \text{ctrl} \rrbracket\) reachability relation of ctrl
Controller Monitor: Early Compliance Checks

Prior state \( x \)

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

Controller Monitor before actuation

Posterior state \( x^+ \)

Model \( \alpha \)

Controller Monitor: Offline

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

\( \iff \) Theorem

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

Exists a run of ctrl to a state where \( x = x^+ \)

André Platzer (CMU)

FCPS / 14: Verified Models & Verified Runtime Validation 17 / 22
Controller Monitor: Early Compliance Checks

### Model $\alpha$

- **Offline**: prior state $\omega$ to $\nu$ to $i+1$

### Controller Monitor before actuation

- **Controller Monitor**
- **Posterior state** $x^{+}$

### Semantical:

$(\omega, \nu) \in [[\text{ctrl}]]$

### Logical $dL$

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

### Arithmetical:

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

- **Theorem**: exists a run of ctrl to a state where $x = x^{+}$
- **$dL$ proof**: check at runtime (efficient)
Controller Monitor: Early Compliance Checks

Model $\alpha$

prior state $x$  $\omega$  $\nu$  $i+1$

Controller Monitor before actuation
posterior state $x^+$

Offline

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

Logical $d\mathcal{L}$: $(\omega, \nu) \models \left< \text{ctrl} \right>(x = x^+)$

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

Theorem (Controller Monitor Correctness)

"Controller safe & in plant bounds as long as monitor satisfied."  FMSD'16

 theorem
Controller Monitor: Early Compliance Checks

Model $\alpha$

Controller Monitor before actuation

prior state $x$

offline

Controller Monitor

Immedite detection of unsafe control before actuation

$\leadsto$ Safe execution of unverified implementations in perfect environments

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

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

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

check at runtime (efficient)

Theorem (Controller Monitor Correctness)

“Controller safe & in plant bounds as long as monitor satisfied.” FMSD’16

André Platzer (CMU)
Safe despite evolution with disturbance?

André Platzer (CMU)
Safe despite evolution with disturbance?

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

Model $\alpha$

prior state $x$

posterior state $x^+$

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

prior state \( x \)  

Model \( \alpha \)  

ctrl  

plant  

posterior state \( x^+ \)  

Prediction Monitor before actuation
Prediction Monitor: Compliance with Disturbance

Prediction Monitor before actuation

prior state $x$

Model $\alpha$

ctrl

plant

posterior state $x^+$

states reachable within $\varepsilon$ time

Prediction Monitor with Disturbance

Proactive detection of unsafe control before actuation despite disturbance

$\Rightarrow$ Safety in realistic environments

André Platzer (CMU)

FCPS / 14: Verified Models & Verified Runtime Validation
disturbance $t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 & Q \land t \leq \varepsilon \right)$
disturbance $t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 & Q \land t \leq \varepsilon \right)$

prior state $x$

Model $\alpha$

ctrl

plant

$i+1$

Prediction Monitor before actuation

posterior state $x^+$

states reachable within $\varepsilon$ time

Offline

Logical $\mathcal{L}$:

$(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \land [\text{plant}] \varphi)$

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

Arithmetical:

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

Invariant $\varphi$ implies safety $S$

(known from safety proof)
disturbance $t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 & Q \land t \leq \varepsilon \right)$

Prior state $x$

Model $\alpha$

$\omega \rightarrow ctrl \rightarrow plant \rightarrow i + 1$

Prediction Monitor with Disturbance

Proactive detection of unsafe control before actuation despite disturbance

$\leadsto$ Safety in realistic environments

Offline

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

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

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
   - ModelPlex Compliance

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

5 Evaluation

6 Summary
Evaluation

- Evaluated 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
   - 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 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
Proof

Model

safe!

Validated by ModelPlex

safe!

Model

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

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 dynamic logic for hybrid systems.

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