ModelPlex: Verified Runtime Monitors and Verified Test Oracles for Safety of Cyber-Physical Systems

Stefan Mitsch

Computer Science Department, Carnegie Mellon University

CPS V&V I&F Workshop 2017
May 12, 2017

joint work with André Platzer
Formal Verification of Cyber-Physical Systems

Analyze the physical effect of software 

Proof guarantees correct model
Monitor correctly checks deviation of model from reality

Proof Strategy
Hybrid System
Model
KeYmaera X
Counterexample

Monitor
Specification
Proof
Control
Sensors Actuators
Control
Monitor
Actuators

Discrete computation + continuous physics
Theorem proving ensures correct model

Proof guarantees correct model
Monitor correctly checks deviation of model from reality
Proof Strategy
Hybrid System
Model
KeYmaera X
Counterexample
Monitor
Specification
Proof
Control
Sensors Actuators Sensors
Control
Monitor
Actuators

Safety Proof
Never collide

Stefan Mitsch—ModelPlex
Runtime monitoring ensures model compliance

Monitor desired effect + safe environment

- Runtime: ensure safety and detect anomalies
- Testing: generate and analyze test cases
How to Achieve Safety Guarantees at Runtime?

Real CPS

Proof

Reachability Analysis

safe

Verification Results

Others may not satisfy the model assumptions
Non-verified implementation may have bugs

→

Verification results about models only apply if CPS fits to the model
How to Achieve Safety Guarantees at Runtime?

Real CPS

Model $\alpha^*$

Control $\alpha_{ctrl}$

$v := v + 1$

Plant $\alpha_{plant}$

$x' = v$

abstract

Proof

Reachability Analysis

safe

Verification Results

Others may not satisfy the model assumptions
Non-verified implementation may have bugs

$\Rightarrow$ Verification results about models only apply if CPS fits to the model
How to Achieve Safety Guarantees at Runtime?

Real CPS

Model $\alpha^*$

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

$\nu := \nu + 1$

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

$x' = \nu$

Abstract

Synthesize

Proof

Reachability Analysis

Verification Results

Others may not satisfy the model assumptions

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

Stefan Mitsch—ModelPlex
How to Achieve Safety Guarantees at Runtime?

$$\alpha^*$$

$$\begin{align*}
v & := K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt} \\
x' & \leq v, v' = \\
& \frac{T_{exg} \times_d n}{r_w} - \frac{1}{2} C_d \rho v^2
\end{align*}$$

Real CPS

Model $$\alpha^*$$

synthesize

abstract

Verification Results

Proof

Reachability Analysis

safe

 kazafu may not satisfy the model assumptions

Non-verified implementation may have bugs

$$\Rightarrow$$

Verification results about models only apply if CPS fits to the model
How to Achieve Safety Guarantees at Runtime?

Real CPS

Model

Challenge
- Others may not satisfy the model assumptions
- Non-verified implementation may have bugs

Verfication results about models only apply if CPS fits to the model
ModelPlex at Runtime

- Sensors
- Controller
- Actuators
Compliance Monitor  Checks CPS for compliance with model at runtime

Want: Monitor satisfied at runtime $\rightarrow$ Real state safe

ModelPlex  Which conditions guarantee safety? Derive monitoring conditions from model by proof

Fallback  Safe control, executed when monitor is not satisfied
Principle Behind a ModelPlex Monitor

Hard to execute, impossible to check

\[ F(p, v, \hat{v}, p^+, \hat{p}^+) \]

prior state

posterior state

Model

\[ \{ v := -v \cup \{ v = 0 \} \} \]

\[ p' = v \]

measure

measure

evolve, e.g., move

Stefan Mitsch—ModelPlex 5 of 9
Principle Behind a ModelPlex Monitor

Hard to execute, impossible to check

Model

\[ \{ v := -v \cup ?v = 0 \} \]

\[ p' = v \]

\[ v^+ \]

\[ p^+ \]

\[ \hat{v}^+ \]

\[ \hat{p}^+ \]

measure

prior state

evolve, e.g., move

posterior state

measure
**Monitor:** efficient arithmetic check $F(p, v, \hat{v}^{+}, \hat{p}^{+})$

Hard to execute, impossible to check

$\uparrow$ derive

Model

\[
\{ v := -v \\
\cup \ ?v = 0 \}
\]

$p' = v$

\[
\begin{align*}
\text{evolve,} \\
e.g., \\
\text{move}
\end{align*}
\]

\[
\begin{array}{cccccc}
\text{measure} & \quad & \quad & \quad & \quad & \quad \\
p & \quad & \quad & \quad & \quad & \quad \\
\vdots & \quad & \quad & \quad & \quad & \quad \\
v & \quad & \quad & \quad & \quad & \quad \\
\text{measure} & \quad & \quad & \quad & \quad & \quad \\
\hat{v}^{+} & \quad & \quad & \quad & \quad & \quad \\
\hat{p}^{+} & \quad & \quad & \quad & \quad & \quad
\end{array}
\]

prior state

\[
\begin{array}{cccccc}
p' & \quad & \quad & \quad & \quad & \quad \\
\text{prior state} & \quad & \quad & \quad & \quad & \quad \\
p' & \quad & \quad & \quad & \quad & \quad \\
\text{posterior state} & \quad & \quad & \quad & \quad & \quad \\
p^{+} & \quad & \quad & \quad & \quad & \quad
\end{array}
\]

Stefan Mitsch—ModelPlex 5 of 9
When are two states linked through a run of model $\alpha$?

... $i-2$ $\rightarrow$ $i-1$ $\rightarrow$ $i$

Model $\alpha$
How to Construct Monitor $F(x, x^+)$

When are two states linked through a run of model $\alpha$?
How to Construct Monitor $F(x, x^+)$

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

Semantical: $(\omega, \nu) \in \rho(\alpha)$

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

Semantical: $(\omega, \nu) \in \rho(\alpha)$

$\Updownarrow$ Lemma

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

exists a run of $\alpha$ to a state where $x = x^+$?
How to Construct Monitor $F(x, x^+)$

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

- **Semantical:** $(\omega, \nu) \in \rho(\alpha)$
- **Logic ($d\mathcal{L}$):** $(\omega, \nu) \models \langle \alpha(\pi) \rangle (x = x^+)$
- **Real arithmetic:** $(\omega, \nu) \models F(x, x^+)$

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

**Check at runtime (efficient):**

Stefan Mitsch—ModelPlex 6 of 9
How to Construct Monitor $F(x, x^+)$

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

Offline

Semantical: $(\omega, \nu) \in \rho(\alpha)$

Lemma

Logic (d$L$): $(\omega, \nu) \models \langle \alpha(x) \rangle (x = x^+)$

d$L$ proof

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

check at runtime (efficient)

exists a run of $\alpha$ to a state where $x = x^+$?
How to Construct Monitor $F(x, x^+)$

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

Semantical: $(\omega, \nu) \in \rho(\text{if } (z > 7) \ y := -y \text{ else } z' = y)$
How to Construct Monitor $F(x, x^+)$

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

semantical: $(\omega, \nu) \in \rho(\text{if } (z>7) y := -y \text{ else } z' = y)$

logic ($d\mathcal{L}$): $(\omega, \nu) \models \langle \text{if } (z>7) y := -y \text{ else } z' = y \rangle (y = y^+ \land z = z^+)$
How to Construct Monitor $F(x, x^+)$

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

Offline

Semantical: $(\omega, \nu) \in \rho(\text{if } (z>7) \ y := -y \ \text{else } z'=y)$

Logic (d$\mathcal{L}$): $(\omega, \nu) \models \langle \text{if } (z>7) \ y := -y \ \text{else } z'=y \rangle \ (y=y^+ \ \land \ z=z^+)$

Real arithmetic: $(\omega, \nu) \models z>7 \ \land \ -y = y^+ \ \lor \ (z\leq 7 \ \land \ z + y\Delta t = z^+)$
Logic reduces CPS safety to runtime monitor with offline proof.

**Offline**

- **Semantical**: $\omega, \nu \in \rho(\alpha)$
  \[ \uparrow \text{Lemma} \]
- **Logic (dL)**: $\omega, \nu \models \langle \alpha(x) \rangle (x = x^+) $
  \[ \uparrow \text{dL proof} \]
- **Real arithmetic**: $\omega, \nu \models F(x, x^+) $
Logic reduces **CPS safety** to runtime monitor with offline proof.

**Offline**

- Semantical: \((\omega, \nu) \in \rho(\alpha)\)
  - \(\upuparrows\) Lemma
- Logic (dL): \((\omega, \nu) \models \langle \alpha(x) \rangle (x = x^+)\)
  - \(\upuparrows\) dL proof
- Real arithmetic: \((\omega, \nu) \models F(x, x^+)\)

**Safe** \(\nu \in [S]\)
Logical Reductions for Model Safety Transfer

Logic reduces CPS safety to **runtime monitor** with offline proof

Offline

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

Semantical: $(\omega, \nu) \in \rho(\alpha)$

$\uparrow$ Lemma

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

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

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

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

Logic reduces CPS safety to runtime monitor with offline proof

\[ \omega \rightarrow \alpha \rightarrow [\alpha]S \]

Offline

Semantical: \((\omega, \nu) \in \rho(\alpha)\)

\[ \Omega \rightarrow \text{Lemma} \]

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

Real arithmetic: \((\omega, \nu) \models F(x, x^+)\)
Logical Reductions for Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

Conclusion

Runtime validation is required to guarantee safety

Offline

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

Logic \((d\mathcal{L})\): \((\omega, \nu) \models \langle \alpha(x) \rangle (x = x^+)\)
\[\uparrow d\mathcal{L} \text{ proof}\]

Real arithmetic: \((\omega, \nu) \models F(x, x^+)\)
Measure Distance to Safety Boundary

Related to Robustness in (Metric/Signal) Temporal Logic ModelPlex

synthesis pre-processes \( dL \) to predicates over real arithmetic

\[ \Rightarrow \] easy metric definition

**Proof** ModelPlex synthesis, normal form transformation, and metric derivation by proof

Terms, formulas e.g., \( d(t \geq s) = t - s \), \( d(p \land q) = \min(d(p), d(q)) \)

Safety monitor \( p \leq S \)

Safety monitor \( v \geq 0 \land p \leq S \)
Test Case Analysis and Synthesis

Test Analysis  Run monitor on input/expected outcome
Generate Tests  Pick input and synthesize expected values

Safe

Boundary

Unsafe

Acceleration
Velocity
Position
Safety Margin
Boundary

Stefan Mitsch—ModelPlex
Summary

Proof guarantees correct model
Monitor correctly checks deviation of model from reality

Proof Strategy
Hybrid System
Model
KeYmaera X
Counterexample
Monitor
Specification
Proof
Control
Sensors Actuators
Control
Monitor
Actuators

Dynamics  Analyze software for **physical effects**
Validation  Offline proofs hold at **system runtime**
Tool  ModelPlex implemented as tactic in KeYmaera X

Stefan Mitsch
smitsch@cs.cmu.edu
www.keymaeraX.org