Outline

1. Learning Objectives
2. Fundamental Challenges with Inevitable Models
3. Runtime Monitors
4. Model Compliance
5. Provably Correct Monitor Synthesis
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Controller Monitors
   - Prediction Monitors
6. Summary
Outline

1. Learning Objectives
2. Fundamental Challenges with Inevitable Models
3. Runtime Monitors
4. Model Compliance
5. Provably Correct Monitor Synthesis
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Controller Monitors
   - Prediction Monitors
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
runtime validation
online monitor
prediction vs. run
Learning Objectives

Fundamental Challenges with Inevitable Models

Runtime Monitors

Model Compliance

Provably Correct Monitor Synthesis
- Logical State Relations
- Model Monitors
- Correct-by-Construction Synthesis
- Controller Monitors
- Prediction Monitors

Summary
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*] S \]
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Wrong?
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Wrong?

Right answer to wrong question.
Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Wrong?

\( S \) Right answer to wrong question.
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Wrong?

- **S** Right answer to wrong question.
- **A** Proof, so can’t forget condition. Except too picky to turn on.
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Wrong?

\[ S \] Right answer to wrong question.

\[ A \] Proof, so can’t forget condition.

Except too picky to turn on.
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; \text{plant})^*]S \]

Wrong?

\( S \)
Right answer to wrong question.

\( A \)
Proof, so can’t forget condition.
Except too picky to turn on.

\( ctrl \)
Control model vs. controller implementation

Wrong?

All models are wrong but some are useful. G. Box
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Wrong?

\( S \) Right answer to wrong question.

\( A \) Proof, so can’t forget condition.
   Except too picky to turn on.

\( \text{ctrl} \) Control model vs. controller implementation
   Abstraction helps scale!
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Wrong?

\( S \)  Right answer to wrong question.

\( A \)  Proof, so can’t forget condition.
   Except too picky to turn on.

\( ctrl \)  Control model vs.
   controller implementation
   Abstraction helps scale!
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*] S \]

Wrong?

\[ S \] Right answer to wrong question.

\[ A \] Proof, so can’t forget condition.
   Except too picky to turn on.

\[ ctrl \] Control model vs.
   controller implementation
   Abstraction helps scale!

\[ plant \] Plant model vs.
   real physics
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

All models are wrong but some are useful. G. Box

\( S \) Right answer to wrong question.

\( A \) Proof, so can’t forget condition.
   Except too picky to turn on.

\( ctrl \) Control model vs.
   controller implementation
   Abstraction helps scale!

\( plant \) Plant model vs.
   real physics
   Models are inevitable!
What Else Could Possibly Go Wrong?

**Proposition (System Proved Safe)**

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

All models are wrong but some are useful. G. Box

**Models**

- **S** Right answer to wrong question.
- **A** Proof, so can’t forget condition.
  Except too picky to turn on.
- **ctrl** Control model vs.
  controller implementation
  Abstraction helps scale!
- **plant** Plant model vs.
  real physics
  Models are inevitable!

**Predictions need models!**
What Else Could Possibly Go Wrong?

**Proposition (System Proved Safe)**

\[ A \rightarrow [(ctrl; plant)^*] S \]

**All models are wrong but some are useful. G. Box**

**Models**  
Predictions need models!

**S**  
Right answer to wrong question

**A**  
Proof, so can't forget condition.

**ctrl**  
Controller implementation.

Abstraction helps scale!

**plant**  
Plant model vs.

real physics

Models are inevitable!

**Challenge**

Verification results about models only apply if CPS fits to the model
What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

All models are wrong but some are useful. G. Box

Models

Predictions need models!

S

Right answer to wrong question

A

Proof, so can't forget condition.

Except too picky to turn on.

ctrl

Control model vs. controller implementation.

Abstraction helps scale!

plant

Plant model vs. real physics

Models are inevitable!

Challenge

Verification results about models

only apply if CPS fits to the model

\[ \iff \text{Verifiably correct runtime model validation} \]
Outline

1. Learning Objectives
2. Fundamental Challenges with Inevitable Models
3. Runtime Monitors
4. Model Compliance
5. Provably Correct Monitor Synthesis
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Controller Monitors
   - Prediction Monitors
6. Summary
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]
Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Monitor
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Monitor

Monitor easy if measurable.
Veto turns CPS off.
Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

Monitor easy if measurable. Veto turns CPS off.
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

\textbf{A} Monitor easy if measurable. Veto turns CPS off.

\textbf{S} Too late to monitor. CPS already unsafe!
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Monitor

- Monitor easy if measurable. Veto turns CPS off.
- Too late to monitor. CPS already unsafe!
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

- **A** Monitor easy if measurable.
  - Veto turns CPS off.

- **S** Too late to monitor.
  - CPS already unsafe!

- **ctrl** Monitor each control decision.
  - Veto overrides decision.
Proposition (System Proved Safe)

\[ A \rightarrow [(\text{ctrl}; \text{plant})^*]S \]

\( A \) Monitor easy if measurable. Veto turns CPS off.

\( S \) Too late to monitor. CPS already unsafe!

\( \text{ctrl} \) Monitor each control decision. Veto overrides decision.

Controller

ModelPlex

Compliance Monitor

Sensors

Actuators

André Platzer (CMU)
Proposition (System Proved Safe)

\[ \text{A} \rightarrow [(\text{ctrl}; \text{plant})^*] \text{S} \]

**A** Monitor easy if measurable.
Veto turns CPS off.

**S** Too late to monitor.
CPS already unsafe!

**ctrl** Monitor each control decision.
Veto overrides decision.

**plant** No source code for physics.
Observe and compare.
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

**A** Monitor easy if measurable.
Veto turns CPS off.

**S** Too late to monitor.
CPS already unsafe!

**ctrl** Monitor each control decision.
Veto overrides decision.

**plant** No source code for physics.
Observe and compare.
Veto triggers best fallback.
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

**A** Monitor easy if measurable.
Veto turns CPS off.

**S** Too late to monitor.
CPS already unsafe!

**ctrl** Monitor each control decision.
Veto overrides decision.

**plant** No source code for physics.
Observe and compare.
Veto triggers best fallback.
Proposition (System Proved Safe)

\[ A \rightarrow [(ctrl; plant)^*]S \]

Monitor Verified runtime validation!

- **A** Monitor easy if measurable.
  Veto turns CPS off.

- **S** Too late to monitor.
  CPS already unsafe!

- **ctrl** Monitor each control decision.
  Veto overrides decision.

- **plant** No source code for physics.
  Observe and compare.
  Veto triggers best fallback.

Monitors must be correct
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 CPS when validating model compliance.
- Compliance with model is characterizable in logic.
- Compliance formula transformed by proof to monitor.
- Correct-by-construction verified runtime model validation.
Outline

1. Learning Objectives
2. Fundamental Challenges with Inevitable Models
3. Runtime Monitors
4. Model Compliance
   - Provably Correct Monitor Synthesis
     - Logical State Relations
     - Model Monitors
     - Correct-by-Construction Synthesis
     - Controller Monitors
     - Prediction Monitors
5. Summary
Model Monitor

Model $\alpha$

$i - 1 \rightarrow i \rightarrow i + 1$

turn

predict

ctrl

plant
Model Compliance

Is present CPS behavior included in the behavior of the model?

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

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

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

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

Is present CPS behavior included in the behavior of the model?

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

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
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ ([\{x' = v, v' = -g \& x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H) \]
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ [(\{x' = v, v' = -g \land x \geq 0\}; (x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H) \]

Example (Controller Monitor)

control changes \((x, v)\) to \((x^+, v^+)\)
# Quantum’s Bouncing Ball Monitors

## Proposition (Quantum can bounce around safely)

\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \\
[(\{x' = v, v' = -g \land x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H)
\]

## Example (Controller Monitor)

\[
(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x
\]

**control changes** \((x, v)\) **to** \((x^+, v^+)\)
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ [\{ x' = v, v' = -g \land x \geq 0 \}; (\forall x = 0; v := -cv \lor ?x \neq 0) ]^*(0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\[(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\]
Proposition (Quantum can bounce around safely)

\( 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \\
[([x' = v, v' = -g \land x \geq 0]; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H) \)

Example (Controller Monitor)

\( (x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x \)

Example (Plant Monitor)
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

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

\[(\{x' = v, v' = -g \land x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*] (0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\[(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\]

Example (Plant Monitor)

\[(v^+ = v - gt \land x^+ = x + vt - \frac{g}{2} t^2)\]
Proposition (Quantum can bounce around safely)

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

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

Example (Controller Monitor)

\[ (x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x \]

Example (Plant Monitor)

\[ 2g(x^+ - x) = v^2 - (v^+)^2 \]

from invariant

\[ 2gx = 2gH - v^2 \]
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \\
[(\{ x' = v, v' = -g & x \geq 0 \}; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H)
\]

Example (Controller Monitor)

\[(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\]

Example (Plant Monitor)

\[2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v\]

directionality: always falling
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ \left[ \{ x' = v, v' = -g \land x \geq 0 \}; (?x = 0; v := -cv \cup ?x \neq 0) \right]^* (0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\[ (x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x \]

Example (Plant Monitor)

\[ 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0 \]

Takeaway

Monitors are subtle, in desperate need of correctness proof.

What proof implies a safe system if the monitors pass?

André Platzer (CMU)

LFCPS/19: Verified Models & Verified Runtime Validation
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ ((\{x' = v, v' = -g \land x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*)(0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\[(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\]

Example (Plant Monitor)

\[2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0\]

Example (Model Monitor)

Takeaway

Monitors are subtle, in desperate need of correctness proof.

What proof implies a safe system if the monitors pass?
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = \text{H} \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ [(\{ x' = v, v' = -g \land x \geq 0 \}; (\exists x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\[ (x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x \]

Example (Plant Monitor)

\[ 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0 \]

Example (Model Monitor)

\[ x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \]
\[ \lor x^+ = 0 \land c^22g(x^+ - x) = c^2v^2 - (v^+)^2 \land v^+ \geq -cv \land x \geq 0 \]

Takeaway

Monitors are subtle, in desperate need of correctness proof. What proof implies a safe system if the monitors pass?
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ [(\{x' = v, v' = -g \land x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H) \]

Example (Controller Monitor)
\[(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\]

Example (Plant Monitor)
\[2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0\]

Example (Model Monitor)
\[x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0\]
\[\lor x^+ = 0 \land c^2 2g(x^+ - x) = c^2 v^2 - (v^+)^2 \land v^+ \geq -cv \land x \geq 0\]
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[
[([x' = v, v' = -g \land x \geq 0]; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H)
\]

Example (Controller Monitor)

\[
(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x
\]

Example (Plant Monitor)

\[
2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0
\]

Example (Model Monitor)

\[
x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0
\]
\[
\lor x^+ = 0 \land c^2 2g(x^+ - x) = c^2 v^2 - (v^+)^2 \land v^+ \geq -cv \land x \geq 0
\]

Takeaway

Monitors are subtle, in desperate need of correctness proof.

What proof implies a safe system if the monitors pass?

André Platzer (CMU)

LFCPS/19: Verified Models & Verified Runtime Validation
# Quantum’s Bouncing Ball Monitors

## Proposition
(Quantum can bounce around safely)

\[
0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \\

[([x' = v, v' = -g \land x \geq 0]; (?x = 0; v := -cv \cup ?x \neq 0))^\ast] (0 \leq x \land x \leq H)
\]

## Example (Controller Monitor)

\[
(x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x
\]

## Example (Plant Monitor)

\[
2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0
\]

## Example (Model Monitor)

\[
x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \\
\lor x^+ = 0 \land c^2 2g(x^+ - x) = c^2 v^2 - (v^+)^2 \land v^+ \geq -cv \land x \geq 0
\]
Quantum’s Bouncing Ball Monitors

Proposition (Quantum can bounce around safely)

\[ 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow \]
\[ [(\{ x' = v, v' = -g & x \geq 0 \}; (?x = 0; v := -cv \cup ?x \neq 0) )^* ](0 \leq x \land x \leq H) \]

Example (Controller Monitor)

\((x = 0 \land v^+ = -cv \lor x > 0 \land v^+ = v) \land x^+ = x\)

Example (Plant Monitor)

\(2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land x^+ \geq 0\)

Example (Model Monitor)

\(x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0 \land v^+ \geq -cv \land x \geq 0\)
### Proposition (Quantum can bounce around safely)

$$0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \rightarrow$$

$$[(\{x' = v, v' = -g \land x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*](0 \leq x \land x \leq H)$$

### Example (Controller Monitor)

$$x = 0 \land v + = -cv \lor x > 0 \land v + = v \land x + = x$$

### Example (Model Monitor)

$$x^+ > 0 \land 2g(x^+ - x) = v^2 - (v^+)^2 \land v^+ \leq v \land x \geq 0$$

$$\forall x^+ = 0 \land c^2 2g(x^+ - x) = c^2 v^2 - (v^+)^2 \land v^+ \geq -cv \land x \geq 0$$

---

**Takeaway**

Monitors are subtle, in desperate need of correctness proof.
What proof implies a safe system if the monitors pass?
Outline

1 Learning Objectives

2 Fundamental Challenges with Inevitable Models

3 Runtime Monitors

4 Model Compliance

5 Provably Correct Monitor Synthesis
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Controller Monitors
   - Prediction Monitors

6 Summary
When are two states linked through a run of model $\alpha$?
Characterizing State Relations in Logic

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

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

reachability relation of $\alpha$

André Platzer (CMU)
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:**

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

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

**Logical dL:**

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

**Arithmetical:**

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

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

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

- **Lemma:** exists a run of $\alpha$ to a state where $x = x^+$
- **dL proof:** check at runtime (efficient)
Characterizing State Relations in Logic

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

$\Leftrightarrow$ Lemma

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

$\uparrow$ $dL$ proof

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

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

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

**Offline**

- **Semantical:** \((\omega, \nu) \in [\alpha]\)
- **Lemma:** \((\omega, \nu) \models \langle \alpha \rangle(x = x^+)\)
- **Logical dL:** \((\omega, \nu) \models F(x, x^+)\)
- **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.

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

Offline

Init \( \omega \in [A] \)

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

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

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

\( \Leftarrow \) Lemma

\( \Uparrow \) dL proof

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

Logic reduces CPS safety to runtime monitor with offline proof

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

Offline

Init \( \omega \in [A] \)
Safe \( \nu \in [S] \)

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.

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

\( \omega \) is the initial state, characterized by \( x \).
\( \nu \) is the posterior state, also characterized by \( x \).

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

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

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

The check at runtime is efficient.
Logic reduces CPS safety to runtime monitor with offline proof

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

\(\omega\) is the initial state, \(\nu\) is the posterior state.

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

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

\( \omega \) \quad \text{Model } \alpha \quad \nu

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

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

\text{Arithmetical: } (\omega, \nu) \models F(x, x^+) \quad \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:** \((\omega, \nu) \models F(x, x^+)\)

**Lemma:** \(\uparrow \) dL proof

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

Logic reduces CPS safety to runtime monitor with offline proof

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

\[ \omega \rightarrow \in \rightarrow v \]

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

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

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

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

Logic reduces CPS safety to runtime monitor with offline proof

Not initial state. Model repeats...

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

Model \( \alpha \) \[ \omega \rightarrow v \]

\( \omega \in [A] \) Init

\( v \in [S] \) Safe

Semantical:

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

Logical dL:

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

Arithmetical:

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

check at runtime (efficient)

André Platzer (CMU) LFCPS/19: Verified Models & Verified Runtime Validation
Logical Reductions for $\alpha^*$ Model Safety Transfer

Logic reduces CPS safety to runtime monitor with offline proof

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

Offline

$\omega \in [A]$

Semantical:

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

$\uparrow$

Lemma

Logical $\text{dL}$:

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

$\uparrow$

$dL$ proof

Arithmetical:

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

$\Rightarrow$ check at runtime (efficient)
ModelPlex Model Monitor Correctness

Theorem (Model Monitor Correctness) (FMSD'16)

System safe as long as monitor satisfied.
ModelPlex Model Monitor Correctness

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

0 \quad \text{Model } \alpha \quad i-1 \quad \text{Model } \alpha \quad i \quad \text{Model } \alpha \quad i+1

System safe as long as monitor satisfied.
dL proof $A \rightarrow [\alpha^*] S$

Init $0 \models A$

Model $\alpha$

$0 \rightarrow i-1 \rightarrow i \rightarrow i+1$

System safe as long as monitor satisfied.
$dL$ proof $A \rightarrow [\alpha^*]S$

$0 \models A$

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

System safe as long as monitor satisfied.

\[ \text{dL proof } A \rightarrow [\alpha^*] S \]
Theorem (Model Monitor Correctness)  
(FMSD’16)

System safe as long as monitor satisfied.
Correct-by-Construction Synthesis

- dL proof calculus executes models symbolically

Prior state $x \rightarrow i-1$ Model $\alpha$ $\rightarrow i$ Posterior state $x^+$

Proof attempt

$\langle \alpha(x) \rangle (x = x^+)$
dL proof calculus executes models symbolically

**Model** $\alpha$

prior state $x$

$i - 1$ \(\xrightarrow{\text{climb}}\) \(\xrightarrow{\text{descend}}\) \(i\)

posterior state $x^+$

proof attempt

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

\[\langle \text{climb} \cup \text{descend} \rangle P \leftrightarrow \langle \text{climb} \rangle P \lor \langle \text{descend} \rangle P\]
Correct-by-Construction Synthesis

- dL proof calculus executes models symbolically

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

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

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

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

Immediate detection of model violation ⇝ Mitigates safety issues with safe fallback action

André Platzer (CMU)
dL proof calculus executes models symbolically

Model $\alpha$

Prior state $x_{i-1}$ $\xrightarrow{\text{climb}}$ Posterior state $x_i$

Proof attempt

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

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

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

Monitor:
The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\xrightarrow{\text{prove at runtime}}$

Immediate detection of model violation $\xrightarrow{\text{Mitigates safety issues with safe fallback action}}$
dL proof calculus executes models symbolically

\[ \text{prior state } x \xrightarrow{\text{climb}} i \xrightarrow{\text{descend}} \text{posterior state } x^+ \]

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

\[
\begin{align*}
\langle \text{climb} \rangle (x = x^+) & \lor \langle \text{descend} \rangle (x = x^+) \\
F_1(x, x^+) & \lor F_2(x, x^+) \\
\end{align*}
\]

Monitor: \[F_1(x, x^+) \lor F_2(x, x^+)\]
Correct-by-Construction Synthesis

- $dL$ proof calculus executes models symbolically

prior state $x ightarrow i - 1 \rightarrow i$ (climb, descend) posterior state $x^+$

\[
\begin{align*}
\langle \text{climb} \cup \text{descend} \rangle(x = x^+) \\
\langle \text{climb} \rangle(x = x^+) \lor \langle \text{descend} \rangle(x = x^+) \\
F_1(x, x^+) \lor F_2(x, x^+)
\end{align*}
\]

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 $\sim$ prove at runtime
**Correct-by-Construction Synthesis**

- $dL$ proof calculus executes models symbolically

$$\text{Model } \alpha \rightarrow \text{climb} \leftarrow \text{descend}$$

prior state $x^{i-1}$ \hspace{1cm} proof attempt

$$\langle \alpha(x) \rangle (x = x^i)$$

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

$$\langle \text{climb} \rangle P \lor \langle \text{descend} \rangle P$$

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

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

**Model Monitor**

Immediate detection of model violation

$\carat$ Mitigates safety issues with safe fallback action
Typical \((ctrl;\ plant)^*\) models can check earlier
Controller Monitor: Veto Early If Noncompliant

prior state $x \stackrel{i}{\to}$ Model $\alpha \stackrel{i+1}{\to}$ posterior state $x^+$

Model Monitor

Theorem (Controller Monitor Correctness) (FMSD'16)
Controller safe and in plant bounds as long as monitor satisfied.

Immediate detection of unsafe control before actuation
→ Safe execution of unverified implementations in perfect environments
Controller Monitor: Veto Early If Noncompliant

prior state $x$

Model $\alpha$

Controller Monitor before actuation

posterior state $x^+$

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

reachability relation of ctrl
Controller Monitor: Veto Early If Noncompliant

Model $\alpha$

prior state $x$

Controller Monitor before actuation

posterior state $x^+$

Controller Monitor Correctness (FMSD'16)

Controller safe and in plant bounds as long as monitor satisfied.

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

$\iff$ Theorem

Logical $\downarrow$:

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

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

André Platzer (CMU)

LFCPS/19: Verified Models & Verified Runtime Validation

LFCPS/19 13 / 16
Controller Monitor: Veto Early If Noncompliant

Controller Monitor before actuation

Semantical: \((\omega, v) \in [\text{ctrl}]\) 

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

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

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

check at runtime (efficient)
Controller Monitor: Veto Early If Noncompliant

**Model Monitor**

Prior state $x$ leads to posterior state $x^+$ through controller $\alpha$ and plant $\nu$.

**Controller Monitor before actuation**

- **Semantical:** $(\omega, \nu) \in \lfloor \text{ctrl} \rfloor$
  - Theorem
- **Logical dL:** $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$
  - dL proof
- **Arithmetical:** $(\omega, \nu) \models F(x, x^+)$
  - check at runtime (efficient)

**Theorem (Controller Monitor Correctness) (FMSD’16)**

Controller safe and in plant bounds as long as monitor satisfied.
Controller Monitor: Veto Early If Noncompliant

Model $\alpha$


 prior state $x$

Controller Monitor before actuation

posterior state $x^+$

Controller Monitor Correctness (FMSD’16)

Controller safe and in plant bounds as long as monitor satisfied.

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

check at runtime (efficient)

Logical dL:

exists a run of ctrl to a state where $x = x^+$

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

Semantical: reachability relation of ctrl

Theorem (Controller Monitor Correctness) (FMSD’16)

Controller safe and in plant bounds as long as monitor satisfied.

André Platzer (CMU)
Safe despite evolution with disturbance?

Model $\alpha$ 

Turn 

Prediction Monitor
Safe despite evolution with disturbance?

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

\[ \text{Model } \alpha \]

prior state \( x \)

\[ i \xrightarrow{\text{ctrl}} \text{plant} \xrightarrow{\text{J}} i+1 \]

posterior state \( x^+ \)

\[ \text{Invariant } J \text{ implies safety } S \text{ (known from safety proof)} \]

\[ \text{Logical dL: } \uparrow \operatorname{dL proof} (i, \nu) | = F (x, x^+) \]

\[ \text{Arithmetical: } \]

\[ \text{Prediction Monitor with Disturbance} \]

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

André Platzer (CMU)

LFCPS/19: Verified Models & Verified Runtime Validation
\begin{align*}
\text{prior state } x &\quad \xrightarrow{\omega} \quad \text{ctrl} \\
\text{plant} &\quad \xrightarrow{\text{Model } \alpha} \\
\text{posterior state } x^+ &\quad \xrightarrow{i+1}
\end{align*}

\text{plant of the form } \left( x' = f(x) \& Q \right)

\text{Prediction Monitor before actuation despite disturbance}

\text{Safety in realistic environments}

André Platzer (CMU)

LFCPS/19: Verified Models & Verified Runtime Validation
Prediction Monitor: Compliance with Disturbance

\[
\text{time bound } t := 0; \left( x' = f(x), \; t' = 1 & Q \land t \leq \varepsilon \right)
\]

prior state \( x \)

Prediction Monitor before actuation

posterior state \( x^+ \)

states reachable within time \( \varepsilon \)
disturbance $t := 0; \left( f(x) - \delta \leq x' \leq f(x) + \delta, \ t' = 1 \& Q \land t \leq \epsilon \right)$
Prediction Monitor: Compliance with Disturbance

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

Prior state \( x \) \hspace{1cm} Model \( \alpha \) \hspace{1cm} Posterior state \( x^+ \)

Prediction Monitor before actuation

\( \omega \) \hspace{1cm} \( \nu \) \hspace{1cm} \( i + 1 \)

Offline

Logical dL: \((\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \land [\text{plant}] J)\)

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

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

Invariant \( J \) implies safety \( S \) (known from safety proof)
Prediction Monitor: Compliance with Disturbance

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

Model \( \alpha \)

Prior state \( x \)

\( \omega \rightarrow \text{ctrl} \rightarrow \text{plant} \rightarrow i+1 \)

Detection unsafe control before actuation despite disturbance

\( \leadsto \text{Safety in realistic environments} \)

Logical dL:

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

\( \uparrow \text{dL proof} \)

Arithmetical:

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

Invariant \( J \) implies safety \( S \)

(known from safety proof)
1. Learning Objectives

2. Fundamental Challenges with Inevitable Models

3. Runtime Monitors

4. Model Compliance

5. Provably Correct Monitor Synthesis
   - Logical State Relations
   - Model Monitors
   - Correct-by-Construction Synthesis
   - Controller Monitors
   - Prediction Monitors

6. Summary
ModelPlex ensures that proofs transfer to real CPS

- Validate model compliance
- Characterize compliance with model in logic
- Prover transforms compliance formula to executable monitor
- Provably correct runtime model validation by offline + online proof
André Platzer.

*Logical Foundations of Cyber-Physical Systems.*
Springer, Cham, 2018.

URL: [http://www.springer.com/978-3-319-63587-3](http://www.springer.com/978-3-319-63587-3),
doi:10.1007/978-3-319-63588-0.

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.
A complete uniform substitution calculus for differential dynamic logic.  