# Modelling Events for CPS

Enguerrand Prebet

3rd February 2025

Karlsruhe Institute of Technology

# Motivation

Safe CPS aren’t blind.

”In this situation, what decisions should I take?”

# Motivation

Safe CPS aren't blind.

"In this situation, what decisions should I take?"

Today: "In which situation, can/should I take a decision?"

$\rightarrow$ How does it affect the first question?

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0\};$$
$$\texttt{if}(x = 0)\, v := -cv)^*](0 \leq x \leq 5)$$

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0\};$$
$$\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv)^*](0 \leq x \leq 5)$$

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5\};$$
$$\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv)^*](0 \leq x \leq 5)$$

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5\};$$
$$\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv)^*](0 \leq x \leq 5)$$

The law of physics are non-negotiable!

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5\} \cup \{x' = v, v' = -g \,\&\, x \geq 5\};$$
$$\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv)^*](0 \leq x \leq 5)$$

The law of physics are non-negotiable!

# Event-triggered systems

Optimistic setting: continuous sensing

Define precisely when the controller need to be ran.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\{x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5\} \cup \{x' = v, v' = -g \,\&\, x \geq 5\};$$
$$\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5 \land \underbrace{v > 0}_{\text{only tap down}})\, v := -fv)^*](0 \leq x \leq 5)$$

The law of physics are non-negotiable!

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\texttt{if}(x = 0)\, v := -cv;$$
$$\{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0\})^*](0 \leq x \leq 5)$$

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\texttt{if}(x = 0)\, v := -cv;$$
$$t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\})^*](0 \leq x \leq 5)$$

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv;$$
$$t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\})^*](0 \leq x \leq 5)$$

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$[(\texttt{if}(x = 0)\, v := -cv; \texttt{if}(4 \leq x \leq 5)\, v := -fv;$$
$$t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\})^*](0 \leq x \leq 5)$$

Now need to take delay into account

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$\big[\big(\texttt{if}(x = 0)\, v := -cv; \texttt{if}(x + v - \tfrac{g}{2} > 5)\, v := -fv;$$
$$t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\}\big)^*\big](0 \leq x \leq 5)$$

Now need to take delay into account

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$
\begin{array}{l}
0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow \\
\big[\big(\texttt{if}(x = 0)\, v := -cv; \texttt{if}(x + v - \frac{g}{2} > 5 \lor \ldots)\, v := -fv; \\
t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\}\big)^*\big](0 \leq x \leq 5)
\end{array}
$$

Now need to take delay into account

The safety condition needs to be always true, not only at the end.

$\rightarrow$ Tricky to get right

# Time-triggered system

Software are discrete not continuous.

Only time causes the controller to act.

$$0 \leq x \leq 5 \land v \leq 0 \land 0 < g \land 0 \leq c \leq 1 \land 0 \leq f \rightarrow$$
$$\Big[\Big(\texttt{if}(x = 0)\, v := -cv; \texttt{if}(x + v - \tfrac{g}{2} > 5 \lor (x + \tfrac{v^2}{2g} > 5 \land v - g < 0))\, v := -fv;$$
$$t := 0; \{x' = v, v' = -g, t' = 1 \,\&\, x \geq 0 \land t \leq 1\}\Big)^*\Big](0 \leq x \leq 5)$$

Now need to take delay into account

The safety condition needs to be always true, not only at the end.

$\rightarrow$ Tricky to get right

# Summary

| | Event-triggered | Time-triggered |
|---|---|---|
| Sensing | Continuous sensing | Discrete sensing |
| Easier to | Verify | Implement |
| More info | Chapter 08[3] | Chapter 09[3] |

Physics matters!

# Summary

| | Event-triggered | Time-triggered |
|---|---|---|
| Sensing | Continuous sensing | Discrete sensing |
| Easier to | Verify | Implement |
| More info | Chapter 08[3] | Chapter 09[3] |

Physics matters!

Use verified event-triggered model as basis for time-triggered model

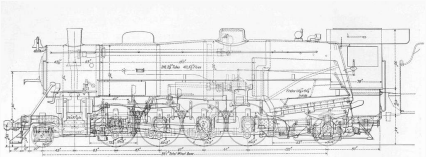
# Relating CPS via Differential Refinement Logic

Enguerrand Prebet

3rd February 2025

Karlsruhe Institute of Technology

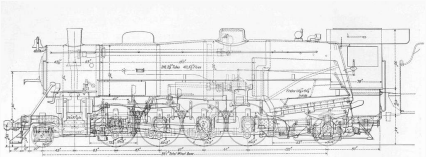
# Cyber-Physical Systems



⇕

[train]safe

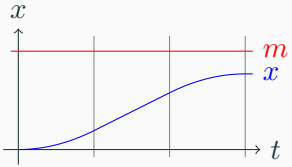
# Cyber-Physical Systems



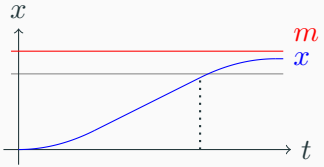
⇕

[train]safe

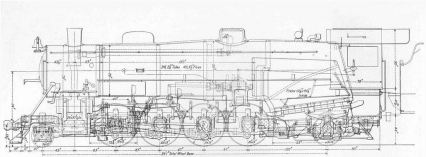
Time-triggered




Event-triggered



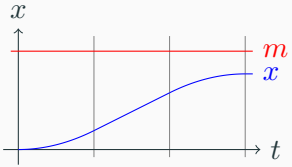
# Cyber-Physical Systems



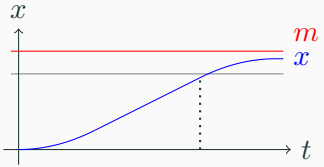
⇕

[train]safe

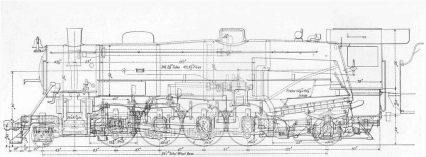
Time-triggered
Implementable




Event-triggered



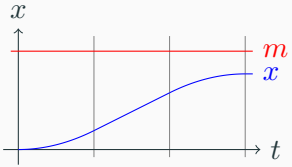
# Cyber-Physical Systems



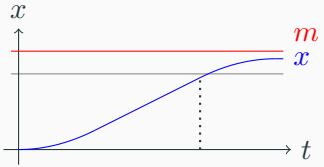
⇕

[train]safe

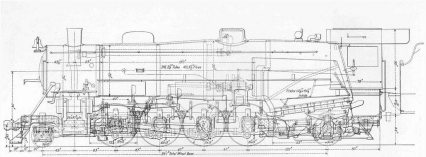
Time-triggered
Implementable




Event-triggered
Verifiable



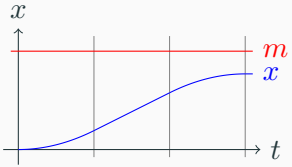
# Cyber-Physical Systems



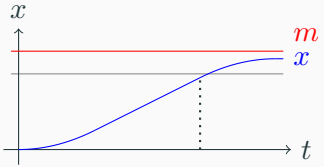
⇕

[train]safe

Time-triggered
Implementable



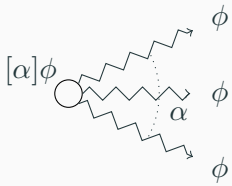

Event-triggered
Verifiable

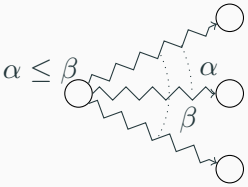



[time]safe ← [event]safe ?

# Refinements

[time]safe ← [event]safe ?




Relating the two systems... → refinement [2]
...in a trusted way. → uniform substitution

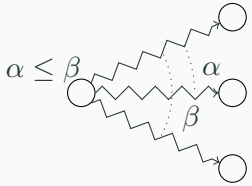
# Differential Refinement Logic (dRL)

dL $\subseteq$ dRL

| | |
|---|---|
| Term: | $\theta, \eta ::= x \mid \theta + \eta \mid \theta \cdot \eta \mid (\theta)'$ |
| Formula: | $\phi, \psi ::= \theta \leq \eta \mid \neg\phi \mid \phi \wedge \psi \mid \forall x\, \phi \mid [\alpha]\phi \mid \alpha \leq \beta \mid \alpha = \beta$ |
| Program: | $\alpha, \beta ::= ?\psi \mid x := \theta \mid x := * \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid x' = \theta \,\&\, \psi$ |

# Semantics

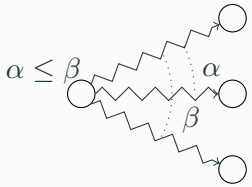
$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$




Quiz time:

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

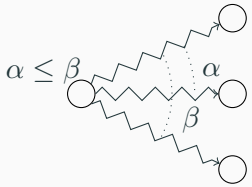



Quiz time:

$$?x = 0 \quad \leq \quad ?x \geq 0$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$



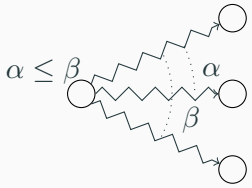

Quiz time:

$$?x = 0 \quad \leq \quad ?x \geq 0$$

✓

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

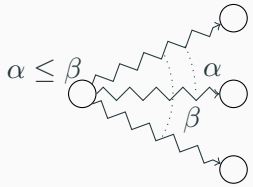



Quiz time:

$$
\begin{array}{lcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) &
\end{array}
$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

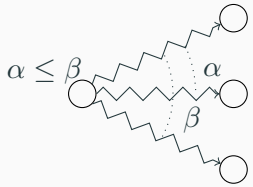



Quiz time:

$$
\begin{array}{lcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) & \checkmark
\end{array}
$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

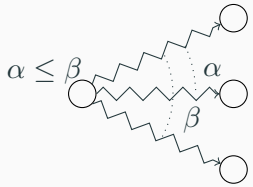



Quiz time:

$$
\begin{array}{rcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) & \checkmark \\
(?x \geq 0; ?x \leq 0 \cup x := 2x) & \leq & x := 2x &
\end{array}
$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

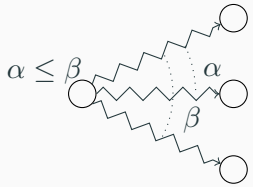



Quiz time:

$$
\begin{array}{rcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) & \checkmark \\
(?x \geq 0; ?x \leq 0 \cup x := 2x) & \leq & x := 2x & \checkmark!
\end{array}
$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$

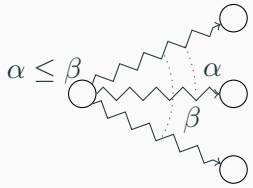



Quiz time:

$$
\begin{array}{rcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) & \checkmark \\
(?x \geq 0; ?x \leq 0 \cup x := 2x) & \leq & x := 2x & \checkmark \\
x := y & = & x := 0 &
\end{array}
$$

# Semantics

$\nu \in [\![\alpha \leq \beta]\!]$ iff $(\nu, \omega) \in [\![\beta]\!]$ for all $(\nu, \omega) \in [\![\alpha]\!]$




Quiz time:

$$
\begin{array}{rcll}
?x = 0 & \leq & ?x \geq 0 & \checkmark \\
x := 2x & \leq & (?x \geq 0; ?x \leq 0 \cup x := 2x) & \checkmark \\
(?x \geq 0; ?x \leq 0 \cup x := 2x) & \leq & x := 2x & \checkmark \\
x := y & = & x := 0 & \text{iff } y = 0
\end{array}
$$

# Using refinement

dL: reasoning according to the structure of the HP

dRL: allows changing the structure itself

$$\text{(Congr)}\quad \frac{\alpha = \beta}{C(\alpha) \leftrightarrow C(\beta)} \qquad\qquad \alpha \leq \beta \rightarrow ([\alpha]\phi \leftarrow [\beta]\phi)$$

# dRL Calculus and Uniform Substitution

# dRL Axioms: the very basics

All dL axioms...

Structural rewriting:

$$(\leq_{\text{refl}}) \quad \alpha \leq \alpha \qquad (\cup_{\text{id}}) \quad \alpha \cup ?\mathit{false} = \alpha$$

$$(\cup_{\text{idemp}}) \quad \alpha \cup \alpha = \alpha \qquad (\cup_{\text{comm}}) \quad \beta \cup \alpha = \alpha \cup \beta$$

$$(\cup_{\text{assoc}}) \quad (\alpha \cup \beta) \cup \gamma = \alpha \cup (\beta \cup \gamma) \qquad (;_{\text{assoc}}) \quad \alpha;(\beta;\gamma) = (\alpha;\beta);\gamma$$

$$(;_{\text{id-l}}) \quad ?\mathit{true};\alpha = \alpha \qquad (;_{\text{id-r}}) \quad \alpha;?\mathit{true} = \alpha$$

$$\text{(dist-l)} \quad (\alpha;\beta) \cup (\alpha;\gamma) = \alpha;(\beta \cup \gamma) \qquad \text{(dist-r)} \quad (\alpha;\gamma) \cup (\beta;\gamma) = (\alpha \cup \beta);\gamma$$

$$\text{(annih-l)} \quad ?\mathit{false};\alpha = ?\mathit{false} \qquad \text{(annih-r)} \quad \alpha;?\mathit{false} = ?\mathit{false}$$

$$(?_{\text{and}}) \quad ?\phi;?\psi = ?(\phi \wedge \psi) \qquad (?_{\text{or}}) \quad ?\phi \cup ?\psi = ?(\phi \vee \psi)$$

$$\text{(unfold-l)} \quad ?\mathit{true} \cup (\alpha;\alpha^*) = \alpha^* \qquad \text{(unfold-r)} \quad ?\mathit{true} \cup (\alpha^*;\alpha) = \alpha^*$$

Kleene Algebra (with tests) [1]

# (Some) Axioms of dRL

dRL-specific axioms:

$([\leq])$ $\alpha \leq \beta \to ([\alpha]\phi \leftarrow [\beta]\phi)$

$(;)$ $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$

$(\cup_l)$ $\alpha \cup \beta \leq \gamma \leftrightarrow \alpha \leq \gamma \wedge \beta \leq \gamma$

$(\cup_r)$ $\alpha \leq \beta \cup \gamma \leftarrow \alpha \leq \beta \vee \alpha \leq \gamma$

$(\leq_t)$ $\alpha \leq \beta \leftarrow (\alpha \leq \gamma \wedge \gamma \leq \beta)$

(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

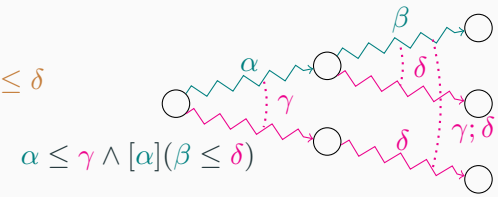
$(\text{loop}_r)$ $\alpha;\beta^* \leq \alpha \leftarrow \alpha;\beta \leq \alpha$

$(\text{loop}_l)$ $\alpha^*;\beta \leq \beta \leftarrow [\alpha^*]\alpha;\beta \leq \beta$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# Axioms soundness

(;) $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$




(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

$[\alpha^*](\alpha \leq \beta) \rightarrow (\underbrace{\alpha \leq \beta \wedge [\alpha]\alpha \leq \beta}_{\alpha;\alpha \leq \beta;\beta} \wedge [\alpha;\alpha]\alpha \leq \beta \wedge \cdots)$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# Axioms soundness

(;) $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$


(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

$[\alpha^*](\alpha \leq \beta) \rightarrow (\alpha \leq \beta \wedge [\alpha]\alpha \leq \beta \wedge [\alpha;\alpha]\alpha \leq \beta \wedge \cdots)$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# Axioms soundness

(;) $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$


(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

$$[\alpha^*](\alpha \leq \beta) \rightarrow (\underbrace{\alpha \leq \beta \wedge [\alpha]\alpha \leq \beta}_{\alpha;\alpha \leq \beta;\beta} \wedge [\alpha;\alpha]\alpha \leq \beta \wedge \cdots)$$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# Axioms soundness

(;) $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$




(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

$[\alpha^*](\alpha \leq \beta) \rightarrow (\underbrace{\alpha \leq \beta \wedge [\alpha]\alpha \leq \beta}_{\alpha;\alpha \leq \beta;\beta} \wedge [\alpha;\alpha]\alpha \leq \beta \wedge \cdots)$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# Relating event and time

$$((\overbrace{a := c \cup a := *; ?\mathsf{Safe}_E}^{\mathsf{ctrl}_E}); t := 0; (\{x' = f(x), t' = 1 \,\&\, E(x) \geq 0\} \cup \{x' = f(x), t' = 1 \,\&\, E(x) \leq 0\}))^*$$

$$\vee\mathsf{I}$$

$$((\underbrace{a := c \cup a := *; ?\mathsf{Safe}_T(T)}_{\mathsf{ctrl}_T}); t := 0; \{\underbrace{x' = f(x), t' = 1}_{\mathsf{ode}} \,\&\, t \leq T\})^*$$

# Relating event and time

$$((\overbrace{a := c \cup a := *; ?\mathsf{Safe}_E}^{\mathsf{ctrl}_E}); t := 0; (\{x' = f(x), t' = 1 \,\&\, E(x) \geq 0\} \cup \{x' = f(x), t' = 1 \,\&\, E(x) \leq 0\}))^*$$

$$\vee \mathsf{I}$$

$$((\underbrace{a := c \cup a := *; ?\mathsf{Safe}_T(T)}_{\mathsf{ctrl}_T}); t := 0; \{\underbrace{x' = f(x), t' = 1}_{\mathsf{ode}} \,\&\, t \leq T\})^*$$

General proof tree structure on board

# Uniform Substitution

Axioms are formulas, instantiations are recovered by uniform substitution.

| | | |
|---|---|---|
| Term: | $\theta, \eta ::= \ldots \mid f(\theta)$ | $[\![\cdot]\!] : \mathcal{I} \to \mathcal{S} \to \mathbb{R}$ |
| Formula: | $\phi, \psi ::= \ldots \mid p(\theta)$ | $[\![\cdot]\!] : \mathcal{I} \to \mathcal{P}(\mathcal{S})$ |
| Program: | $\alpha, \beta ::= \ldots \mid a$ | $[\![\cdot]\!] : \mathcal{I} \to \mathcal{P}(\mathcal{S} \times \mathcal{S})$ |

# (Some) Axioms of dRL with constants

dRL-specific axioms:

$([\leq])$ $\alpha \leq \beta \to ([\alpha]\phi \leftarrow [\beta]\phi)$

$(;)$ $\alpha;\beta \leq \gamma;\delta \leftarrow \alpha \leq \gamma \wedge [\alpha]\beta \leq \delta$

$(\cup_l)$ $\alpha \cup \beta \leq \gamma \leftrightarrow \alpha \leq \gamma \wedge \beta \leq \gamma$

$(\cup_r)$ $\alpha \leq \beta \cup \gamma \leftarrow \alpha \leq \beta \vee \alpha \leq \gamma$

$(\leq_t)$ $\alpha \leq \beta \leftarrow (\alpha \leq \gamma \wedge \gamma \leq \beta)$

(unloop) $\alpha^* \leq \beta^* \leftarrow [\alpha^*](\alpha \leq \beta)$

$(\text{loop}_r)$ $\alpha;\beta^* \leq \alpha \leftarrow \alpha;\beta \leq \alpha$

$(\text{loop}_l)$ $\alpha^*;\beta \leq \beta \leftarrow [\alpha^*]\alpha;\beta \leq \beta$

(ODE) $x' = \theta \,\&\, \phi \leq x' = \eta \,\&\, \psi \leftrightarrow [x' = \theta \,\&\, \phi](x' = \eta \wedge \psi)$

# (Some) Axioms of dRL with constants

dRL-specific axioms:

$([\leq])$ $a \leq b \to ([a]\phi \leftarrow [b]\phi)$

$(;)$ $a;b \leq c;d \leftarrow a \leq c \wedge [a]b \leq d$

$(\cup_l)$ $a \cup b \leq c \leftrightarrow a \leq c \wedge b \leq c$

$(\cup_r)$ $a \leq b \cup c \leftarrow a \leq b \vee a \leq c$

$(\leq_t)$ $a \leq b \leftarrow (a \leq c \wedge c \leq b)$

(unloop) $a^* \leq b^* \leftarrow [a^*](a \leq b)$

$(\text{loop}_r)$ $a;b^* \leq a \leftarrow a;b \leq a$

$(\text{loop}_l)$ $a^*;b \leq b \leftarrow [a^*]a;b \leq b$

(ODE) $x' = f(x)\,\&\,p(x) \leq x' = g(x)\,\&\,q(x) \leftrightarrow [x' = f(x)\,\&\,p(x)](x' = g(x) \wedge q(x))$

# Soundness of Uniform Substitution

$$() \quad \textbf{??} \quad () \text{ if } \sigma(\phi) \text{ defined}$$

"Do not introduce new free variables in a context where they are bound."

Additional case: $\sigma(\alpha \leq \beta) = \sigma(\alpha) \leq \sigma(\beta)$

And... that's it?

# Static semantics

The free variables of a refinement is not defined.

$$\mathsf{FV}(\alpha \leq \beta) = \mathsf{FV}(\alpha) \cup \mathsf{FV}(\beta) \ ?$$

# Static semantics

The free variables of a refinement is not defined.

$$\mathsf{FV}(\alpha \leq \beta) = \mathsf{FV}(\alpha) \cup \mathsf{FV}(\beta) \ ?$$

$$(?y = 1 \leq x := 0) \leftrightarrow (y = 1 \rightarrow x = 0)$$

# Static semantics

The free variables of a refinement is not defined.

$$\mathsf{FV}(\alpha \leq \beta) = \mathsf{FV}(\alpha) \cup \mathsf{FV}(\beta) \cup ((\mathsf{BV}(\alpha) \cup \mathsf{BV}(\beta)) \setminus (\mathsf{MBV}(\alpha) \cap \mathsf{MBV}(\beta)))$$

$$(?y = 1 \leq x := 0) \leftrightarrow (y = 1 \rightarrow x = 0)$$

# Summary

- Hybrid system refinement
  $\rightarrow$ CPS proofs no longer bound by the HP shape
  $\rightarrow$ transfer event-triggered systems' safety to time-triggered systems
- Uniform substitution-style calculus
  $\rightarrow$ easily implemented in KeYmaera X

# References



Dexter Kozen.
**Kleene algebra with tests.**
*ACM Trans. Program. Lang. Syst.*, 19(3):427–443, 1997.


Sarah M. Loos and André Platzer.
**Differential refinement logic.**
In *LICS*, pages 505–514, 2016.


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


André Platzer and Yong Kiam Tan.
**Differential equation invariance axiomatization.**
*J. ACM*, 67(1):6:1–6:66, 2020.


Enguerrand Prebet and André Platzer.
**Uniform substitution for differential refinement logic.**
In Chris Benzmüller, Marijn Heule, and Renate Schmidt, editors, *IJCAR*, LNCS. Springer, 2024.

# Decidability

# Subset of interest

**Theorem**

$(ctrl_1; plant_1)^* \leq (ctrl_2; plant_2)^*$ *is decidable.*

**Example of a decidable property**

$$\Big(((?\mathsf{check}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\Big)^*$$

$$\vee\mathsf{I}$$

$$\Big(((?\mathsf{checks}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\Big)^*$$

# Subset of interest

**Theorem**

$(ctrl_1; plant_1)^* \leq (ctrl_2; plant_2)^*$ *is decidable.*

- No symbol
- $ctrl_i$ are loop-free
- $ctrl_2$ is idempotent: $ctrl_2; ctrl_2 = ctrl_2$
- $plant_i$ are "nice" enough (e.g. time is explicit)

**Example of a decidable property**

$$\Big(((?\text{check}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\Big)^*$$

$$\vee\mid$$

$$\Big(((?\text{checks}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\Big)^*$$

# Subset of interest

**Theorem**

$(ctrl_1; plant_1)^* \leq (ctrl_2; plant_2)^*$ *is decidable.*

- No symbol
- $ctrl_i$ are loop-free
- $ctrl_2$ is idempotent: $ctrl_2; ctrl_2 = ctrl_2$
- $plant_i$ are "nice" enough (e.g. time is explicit)

**Example of a decidable property**

$$\Big(((?\text{check}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\Big)^*$$

$$\vee\mid$$

$$\Big(((?\text{check}(x, m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 0.5\Big)^*$$

# Subset of interest

**Theorem**

$(ctrl_1; plant_1)^* \leq (ctrl_2; plant_2)^*$ *is decidable.*

- No symbol
- $ctrl_i$ are loop-free
- $ctrl_2$ is idempotent: $ctrl_2; ctrl_2 = ctrl_2$
- $plant_i$ are "nice" enough (e.g. time is explicit)

**Example of a decidable property**

$$\big(((?\text{check}(x,m); a := A) \cup a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\big)^*$$

$$\vee\mid$$

$$\big((\text{if check}(x,m) \text{ then } a := A \text{ else } a := -B); t := 0; x' = v, v' = a, t' = 1 \,\&\, t \leq 1\big)^*$$

# Proof outline

**Handling discrete programs:** $ctrl_1 \leq ctrl_2$

Normalise $ctrl_i$ to $x := *; ?\phi_i(x, x^+)$, then $ctrl_1 \leq ctrl_2$ iff $\phi_1(x, x^+) \rightarrow \phi_2(x, x^+)$

**Handling continuous programs:** $[ctrl_1](\underbrace{x' = \theta \,\&\, \phi}_{plant_1} \leq \underbrace{x' = \eta \,\&\, \psi}_{plant_2})$

(ODE) $x' = f(x) \,\&\, p(x) \leq x' = g(x) \,\&\, q(x) \leftrightarrow [x' = f(x) \,\&\, p(x)](x' = g(x) \wedge q(x))$

Remaining goal: $[ctrl_1][plant_1](\theta = \eta \wedge \psi)$

Decidable when:

- $x' = \theta$ solvable
- or $\psi$ of the form $\bigwedge_i \bigvee_j Q_{ij}(x) = 0$ [4]

# Proof outline

**Handling discrete programs:** $ctrl_1 \leq ctrl_2$

Normalise $ctrl_i$ to $x := *; ?\phi_i(x, x^+)$, then $ctrl_1 \leq ctrl_2$ iff $\phi_1(x, x^+) \rightarrow \phi_2(x, x^+)$

**Handling continuous programs:** $[ctrl_1](\underbrace{x' = \theta \,\&\, \phi}_{plant_1} \leq \underbrace{x' = \eta \,\&\, \psi}_{plant_2})$

(ODE) $x' = f(x) \,\&\, p(x) \leq x' = g(x) \,\&\, q(x) \leftrightarrow [x' = f(x) \,\&\, p(x)](x' = g(x) \wedge q(x))$

Remaining goal: $[ctrl_1][plant_1](\theta = \eta \wedge \psi)$

Decidable when:

- $x' = \theta$ solvable
- or $\psi$ of the form $\bigwedge_i \bigvee_j Q_{ij}(x) = 0$ [4]

# Proof outline

**Handling discrete programs:** $ctrl_1 \leq ctrl_2$

Normalise $ctrl_i$ to $x := *; ?\phi_i(x, x^+)$, then $ctrl_1 \leq ctrl_2$ iff $\phi_1(x, x^+) \to \phi_2(x, x^+)$

**Handling continuous programs:** $[ctrl_1](\underbrace{x' = \theta \,\&\, \phi}_{plant_1} \leq \underbrace{x' = \eta \,\&\, \psi}_{plant_2})$

(ODE) $x' = f(x) \,\&\, p(x) \leq x' = g(x) \,\&\, q(x) \leftrightarrow [x' = f(x) \,\&\, p(x)](x' = g(x) \wedge q(x))$

Remaining goal: $[ctrl_1][plant_1](\theta = \eta \wedge \psi)$

Decidable when:

- $x' = \theta$ solvable
- or $\psi$ of the form $\bigwedge_i \bigvee_j Q_{ij}(x) = 0$ [4]