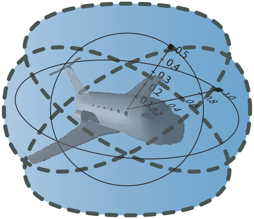
15-819/18-879: Hybrid Systems Analysis & Theorem Proving

07: Dynamic Logic for Hybrid Systems

André Platzer

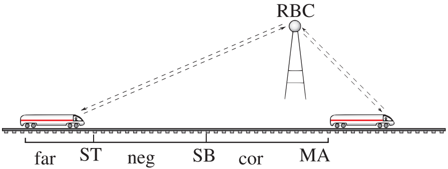
aplatzer@cs.cmu.edu
Carnegie Mellon University, Pittsburgh, PA



Outline

Outline

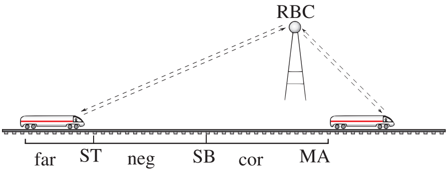
1. Motivation
2. Hybrid Programs
   - Design Motives
   - Syntax
   - Semantics
   - Train Control Examples
3. Hybrid Programs vs. Hybrid Automata
4. Differential Dynamic Logic $d\mathcal{L}$
   - Syntax
   - Semantics



ETCS objectives:

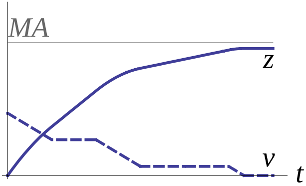
1. Collision free
2. Maximise throughput & velocity (300 km/h)
3. $2.1 * 10^6$ passengers/day

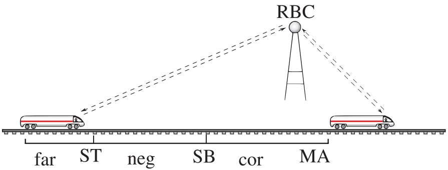
Verifying Parametric Hybrid Systems



Parametric Hybrid Systems

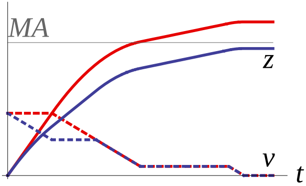
continuous evolution along differential equations + discrete change

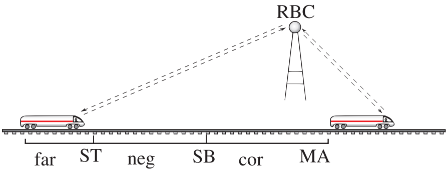




Parametric Hybrid Systems

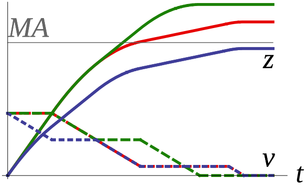
continuous evolution along differential equations + discrete change



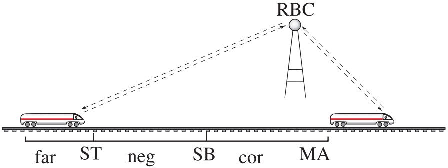


Parametric Hybrid Systems

continuous evolution along differential equations + discrete change



Verifying Parametric Hybrid Systems

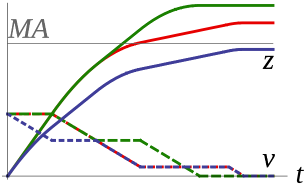


Parametric Hybrid Systems

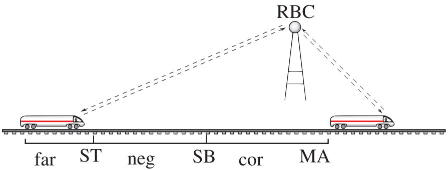
continuous evolution along differential equations + discrete change

- Parameters have nonlinear influence
- Handle *SB* as free symbolic parameter?
- Challenge: verification (falsifying is "easy")
- Which constraints for *SB*?

$\forall MA\, \exists SB$ "train always safe"

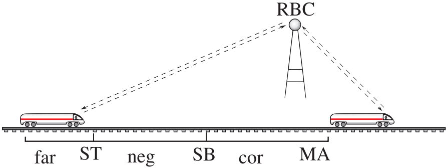


Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓

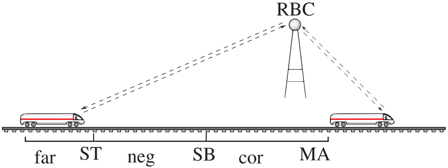
Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓

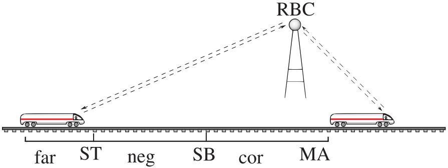
- × no finite-state bisimulation for HS
- × no general handling of free parameters
- × with parameters, everything gets nonlinear!

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\text{Ax}(ETCS) \to z < MA)$	TL-calculus	×	×	✓	..	×

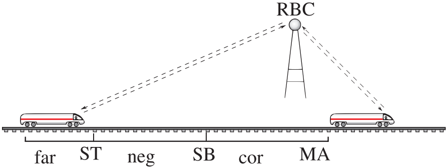
Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

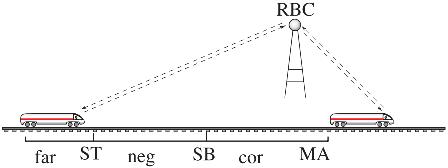
- × declaratively axiomatise operational model
- × expressiveness for characterisation?
- × automation

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

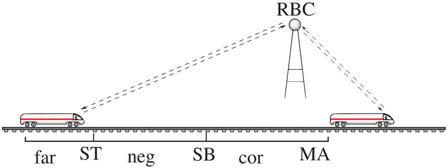
Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\text{Ax}(ETCS) \to z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

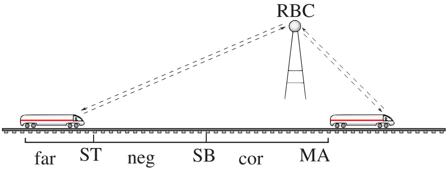
- ✓ $[RBC]\text{partitioned} \rightarrow \exists SB\, \langle \text{Train} \rangle [RBC]\text{safe}$
- × intermediate states
- × automation

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	?

differential dynamic logic

$$d\mathcal{L} = \mathrm{DL} + \mathrm{HP}$$

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (\text{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (\mathrm{Ax}(ETCS) \to z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

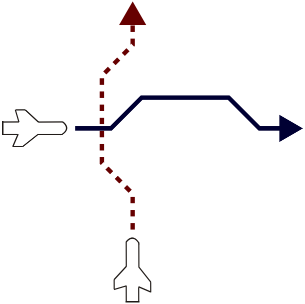
$$\phi ::= P \mid \phi \land \psi \mid \phi \lor \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (\mathrm{Ax}(ETCS) \to z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

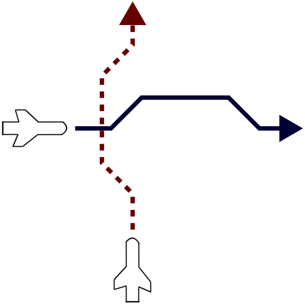
$$\phi ::= P \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$



problem	technique	Op	Par	T	CI	Aut
$\models (\mathrm{Ax}(ETCS) \to z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \land \psi \mid \phi \lor \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$

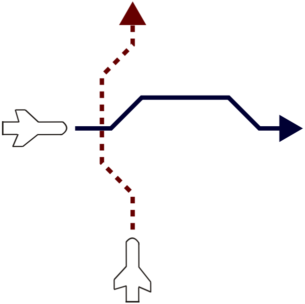


- (left ∨ straight ∨ right)$\mathcal{U}$cruise

problem	technique	Op	Par	T	CI	Aut
$\models (Ax(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$



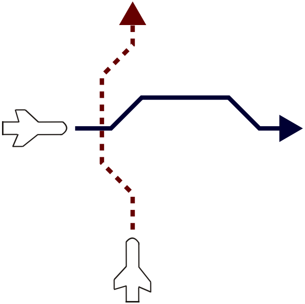
- $(\text{left} \vee \text{straight} \vee \text{right})\mathcal{U}\text{cruise}$
- $\Box(\text{cruise} \rightarrow \text{cruise}\mathcal{U}(\text{left} \vee \text{right}))$

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (Ax(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$



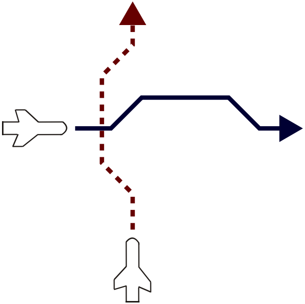
- $(\text{left} \vee \text{straight} \vee \text{right})\mathcal{U}\text{cruise}$
- $\Box(\text{cruise} \rightarrow \text{cruise}\mathcal{U}(\text{left} \vee \text{right}))$
- $\Box(\text{straight} \rightarrow \circ(\text{left} \vee \text{right}))$

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (Ax(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \land \psi \mid \phi \lor \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$



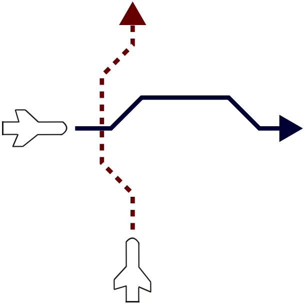
- $(\text{left} \lor \text{straight} \lor \text{right})\mathcal{U}\text{cruise}$
- $\Box\big(\text{cruise} \rightarrow \text{cruise}\mathcal{U}(\text{left} \lor \text{right})\big)$
- $\Box(\text{straight} \rightarrow \circ(\text{left} \lor \text{right}))$
- $\Box\big(x = x_0 \rightarrow (\exists\lambda{\geq}0\, x = \lambda x_0)\mathcal{U}(\text{left} \lor \text{right})\big)$

Temporal Logics for Hybrid Systems

problem	technique	Op	Par	T	CI	Aut
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$

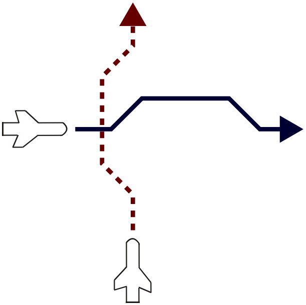


- $(\mathrm{left} \vee \mathrm{straight} \vee \mathrm{right})\mathcal{U}\mathrm{cruise}$
- $\Box\big(\mathrm{cruise} \rightarrow \mathrm{cruise}\mathcal{U}(\mathrm{left} \vee \mathrm{right})\big)$
- $\Box(\mathrm{straight} \rightarrow \circ(\mathrm{left} \vee \mathrm{right}))$
- $\Box\big(x = x_0 \rightarrow (\exists\lambda{\geq}0\, x = \lambda x_0)\mathcal{U}(\mathrm{left} \vee \mathrm{right})\big)$
- How far do two aircraft fly in the same time?

problem	technique	Op	Par	T	CI	Aut
$\models (Ax(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

Definition (Linear Temporal Logic, LTL)

$$\phi ::= P \mid \phi \wedge \psi \mid \phi \vee \psi \mid \neg\phi \mid \Box\phi \mid \Diamond\phi \mid \phi\mathcal{U}\psi \mid \text{“}\circ\phi\text{”}$$



- $(\text{left} \vee \text{straight} \vee \text{right})\mathcal{U}\text{cruise}$
- $\Box(\text{cruise} \rightarrow \text{cruise}\mathcal{U}(\text{left} \vee \text{right}))$
- $\Box(\text{straight} \rightarrow \circ(\text{left} \vee \text{right}))$
- $\Box(x = x_0 \rightarrow (\exists\lambda{\geq}0\, x = \lambda x_0)\mathcal{U}(\text{left} \vee \text{right}))$
- How far do two aircraft fly in the same time?
- How to describe curved flight?

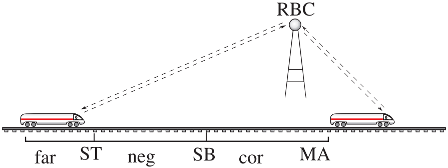
Outline

Outline

dℒ Motives: The Logic of Hybrid Systems

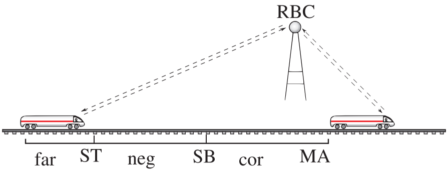
differential dynamic logic

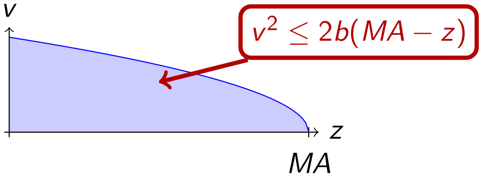
$d\mathcal{L} = \quad DL + HP$



differential dynamic logic

$\mathsf{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}}$

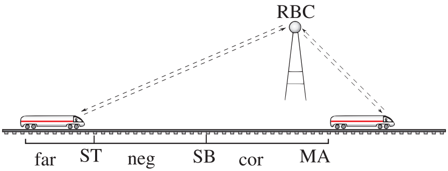




d$\mathcal{L}$ Motives: Regions in First-order Logic

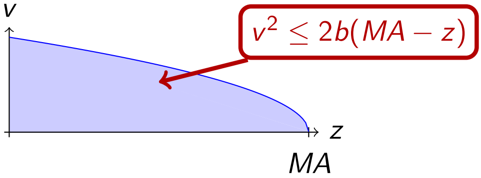
differential dynamic logic

$\text{d}\mathcal{L} = \text{FOL}_{\mathbb{R}}$



$\forall MA \, \exists SB \, \ldots$

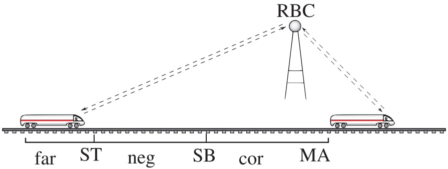
$\forall t \geq 0 \, \ldots$



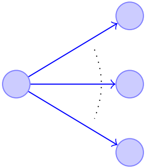
dℒ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$d\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} +$



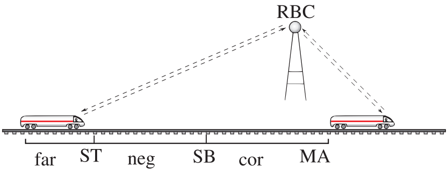
$v^2 \leq 2b$

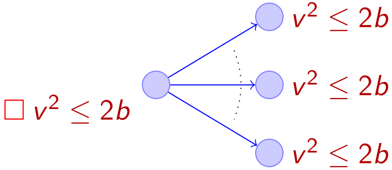


dℒ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$d\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} + \mathrm{ML}$

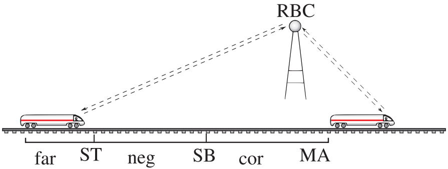


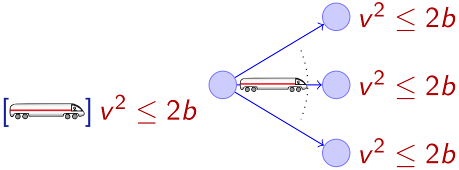


d$\mathcal{L}$ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL$

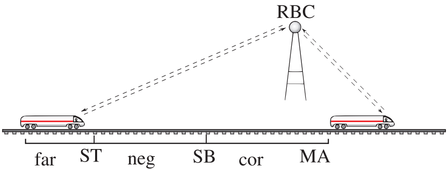




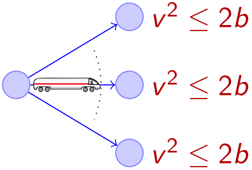
dℒ Motives: Hybrid Programs as Uniform Model

differential dynamic logic

$d\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} + \mathrm{DL} + \mathrm{HP}$



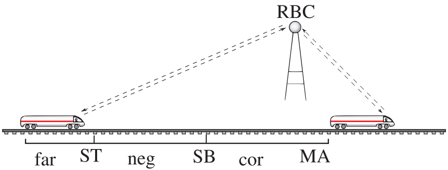
$[z'' = a]\, v^2 \leq 2b$



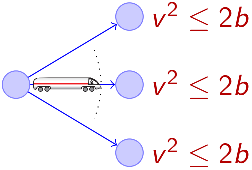
d$\mathcal{L}$ Motives: Hybrid Programs as Uniform Model

differential dynamic logic

$$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$$

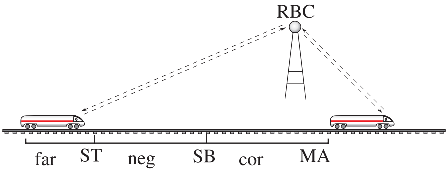


$$[\texttt{if}(z > SB)\, a := -b;\; z'' = a]\, v^2 \leq 2b$$

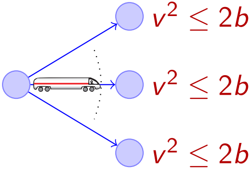


differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$

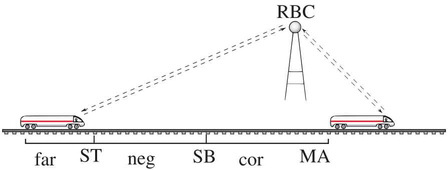


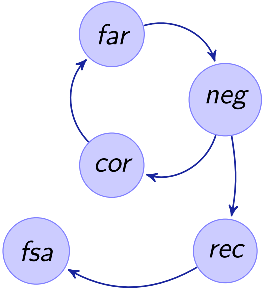
$$[\underbrace{\mathtt{if}(z > SB)\, a := -b;\ z'' = a}_{\text{hybrid program}}]\, v^2 \le 2b$$



differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$



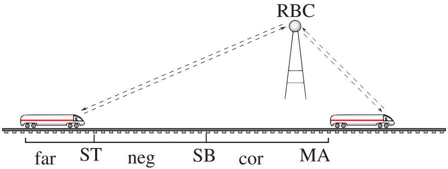


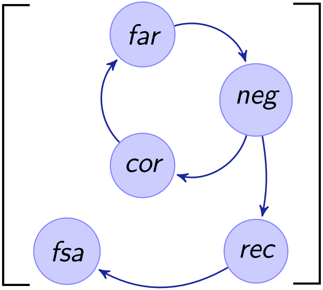
How about hybrid automata?

$d\mathcal{L}$ Motives: What about Hybrid Automata?

differential dynamic logic

$d\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} + \mathrm{DL} + \mathrm{HP}$

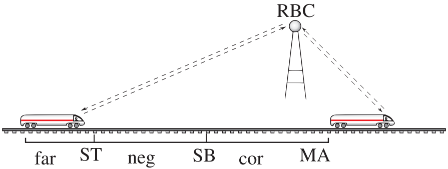


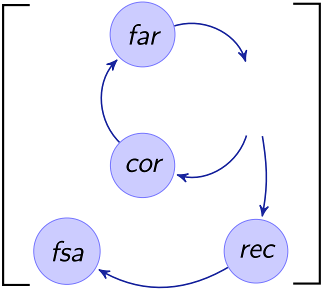


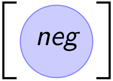
$v^2 \leq 2b\,..$

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$



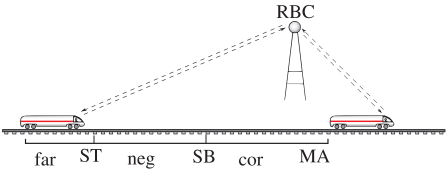


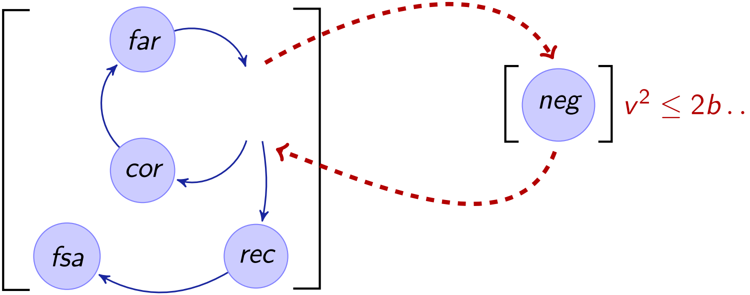


$v^2 \leq 2b\,..$

differential dynamic logic

$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$

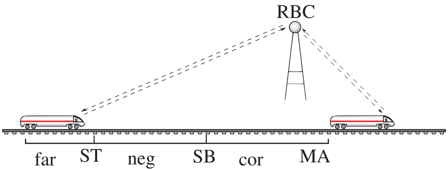


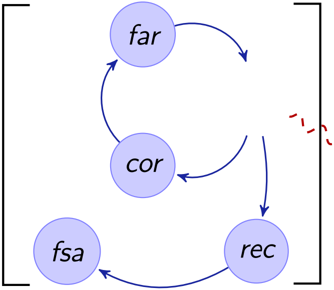


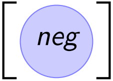
dℒ Motives: What about Hybrid Automata?

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$







$v^2 \leq 2b ..$

not compositional

Hybrid Programs: Syntax

Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$?\chi$	(conditional execution)	
$\alpha; \beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	
α^*	(nondet. repetition)	

Hybrid Programs: Syntax

Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$?\chi$	(conditional execution)	
$\alpha; \beta$	(seq. composition)	
$\alpha \cup \beta$	(nondet. choice)	Kleene algebra
α^*	(nondet. repetition)	

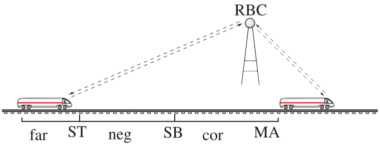
$$ETCS \equiv (ctrl\,;\,drive)^*$$

$$ctrl \equiv (?MA - z \leq SB; a := -b)$$

$$\cup\, (?MA - z \geq SB; a := \ldots)$$

$$drive \equiv z'' = a$$

$$\wedge\, v \geq 0 \wedge \tau \leq \varepsilon$$

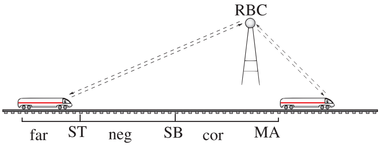


Hybrid Programs: Syntax

Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$?\chi$	(conditional execution)	
$\alpha; \beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	
α^*	(nondet. repetition)	

$$\begin{aligned} ETCS &\equiv (ctrl\,; drive)^* \\ ctrl &\equiv (?MA - z \leq SB; a := -b) \\ &\quad \cup (?MA - z \geq SB; a := \dots) \\ drive &\equiv \tau := 0; z' = v, v' = a, \tau' = 1 \\ &\quad \wedge v \geq 0 \wedge \tau \leq \varepsilon \end{aligned}$$

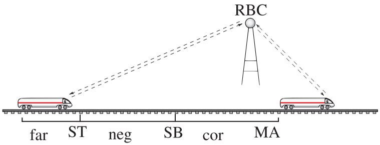


Hybrid Programs: Syntax

Definition (Hybrid program α)

$x' = f(x) \wedge \chi$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$?\chi$	(conditional execution)	
$\alpha; \beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	
α^*	(nondet. repetition)	

$$
\begin{aligned}
ETCS &\equiv (ctrl\,; drive)^* \\
ctrl &\equiv (?MA - z \leq SB; a := -b) \\
&\quad \cup (?MA - z \geq SB; a := \dots) \\
drive &\equiv \tau := 0; z' = v, v' = a, \tau' = 1 \\
&\quad \wedge v \geq 0 \wedge \tau \leq \varepsilon
\end{aligned}
$$

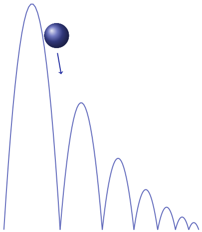


Hybrid Program Example: Controlled Moving Point

```
(
    if (x>0) then
        a := −4                  /* move left */
    else
        a := 4                   /* move right */
    fi;
    t := 0;                      /* reset clock variable t */
    {x'=a, t'=1, t ≤ c}          /* continuous evolution */
)*                               /* repeat these transitions */
```

Hybrid Program Example: Bouncing Ball

```
(
    {h'=v, v'=-g, t'=1,  h ≥ 0};  /* falling/jumping */
    if (t>0 ∧ h=0) then          /* if on ground */
        v := -c*v;               /* bounce back */
        t := 0
    fi.
)*                               /* repeat these transitio
```



States in Hybrid Systems

What is a state of a hybrid program?

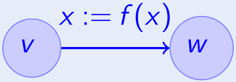
States in Hybrid Systems

What is a state of a hybrid program?

Definition (Kripke state)

$v : V \to \mathbb{R}$ with set of variables V

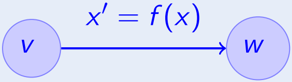
Definition (Hybrid programs α: transition semantics)

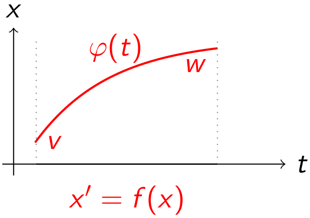


$$w(x) \doteq [\![f(x)]\!]_v$$

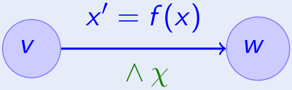
Hybrid Programs: Transition Semantics

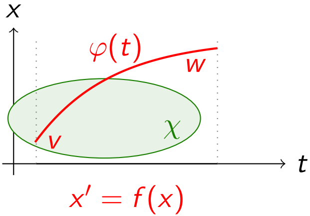
Definition (Hybrid programs α: transition semantics)





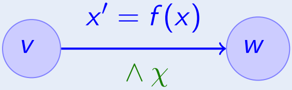
Definition (Hybrid programs α: transition semantics)

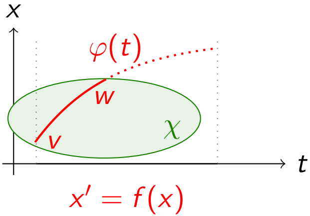




Hybrid Programs: Transition Semantics

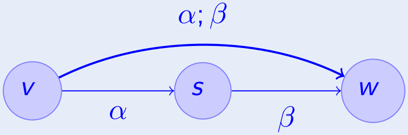
Definition (Hybrid programs α: transition semantics)





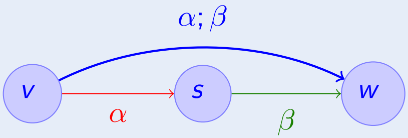
Hybrid Programs: Transition Semantics

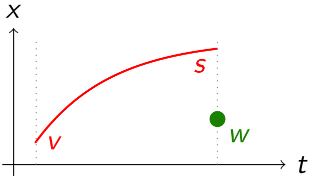
Definition (Hybrid programs α: transition semantics)



Hybrid Programs: Transition Semantics

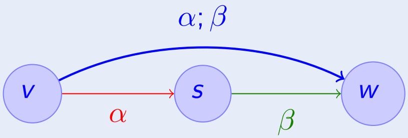
Definition (Hybrid programs α: transition semantics)

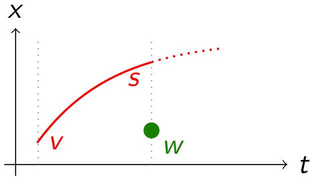




Hybrid Programs: Transition Semantics

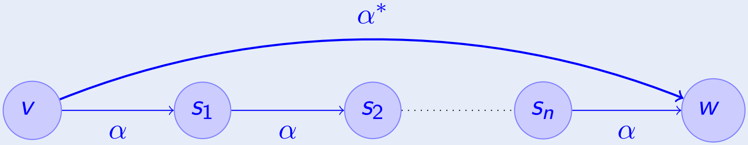
Definition (Hybrid programs α: transition semantics)





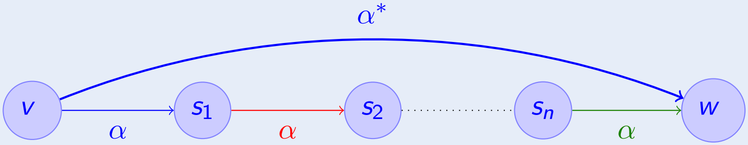
Hybrid Programs: Transition Semantics

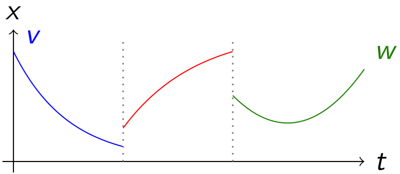
Definition (Hybrid programs α: transition semantics)



Hybrid Programs: Transition Semantics

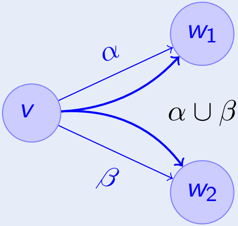
Definition (Hybrid programs α: transition semantics)





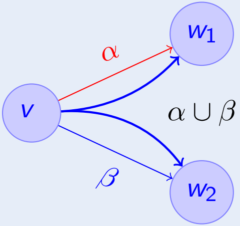
Hybrid Programs: Transition Semantics

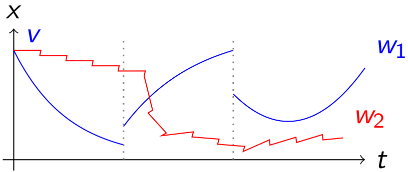
Definition (Hybrid programs α: transition semantics)



Hybrid Programs: Transition Semantics

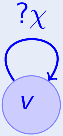
Definition (Hybrid programs α: transition semantics)





Hybrid Programs: Transition Semantics

Definition (Hybrid programs α: transition semantics)



if $v \models \chi$

Definition (Hybrid programs α: transition semantics)

if $v \not\models \chi$

Hybrid Programs: Formal Semantics

Definition (Hybrid programs α)

$$\rho(x' = f(x)) = \{(\varphi(0), \varphi(r)) : \varphi \models x' = f(x) \text{ for duration } r\}$$

$$(v, w) \in \rho(x := \theta) :\Longleftrightarrow w = v[x \mapsto [\![\theta]\!]_v]$$

$$\rho(?\chi) = \{(v, v) : v \models \chi\}$$

$$\rho(\alpha \cup \gamma) = \rho(\alpha) \cup \rho(\gamma)$$

$$\rho(\alpha; \gamma) = \rho(\alpha) \circ \rho(\gamma)$$

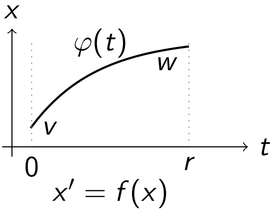
$$(v, w) \in \rho(\alpha^*) :\Longleftrightarrow \text{there is } v \xrightarrow{\rho(\alpha)} v_1 \xrightarrow{\rho(\alpha)} v_2 \cdots\cdots \xrightarrow{\rho(\alpha)} w$$

Definition (Hybrid programs α)

$$\rho(x' = f(x)) \quad = \quad \{(\varphi(0), \varphi(r)) \; : \; \varphi \models x' = f(x) \text{ for duration } r\}$$

with $[\![x']\!]_{\varphi(\zeta)} = \frac{d\varphi(t)(x)}{dt}(\zeta)$

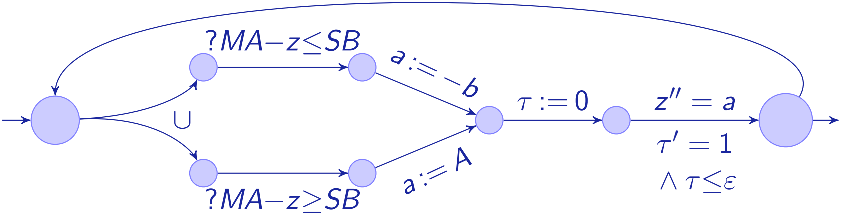
- there is $\varphi : [0, r] \to$ States "with $\varphi(0) = v, \varphi(r) = w$"
- $[\![x]\!]_{\varphi(\zeta)}$ is continuous in ζ on $[0, r]$
- $\frac{d\,[\![x]\!]_{\varphi(t)}}{dt}(\zeta) = [\![f(x)]\!]_{\varphi(\zeta)}$ for $\zeta \in (0, r)$
- $[\![y]\!]_{\varphi(\zeta)} = [\![y]\!]_v$ otherwise



Branching Executions in Hybrid Programs: ETCS

$$\begin{aligned}
\text{system} &\equiv (cor\,; drive)^* \\
cor &\equiv (?MA - z \leq SB; a := -b) \cup (?MA - z \geq SB; a := A) \\
drive &\equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \wedge v \geq 0 \wedge \tau \leq \varepsilon)
\end{aligned}$$

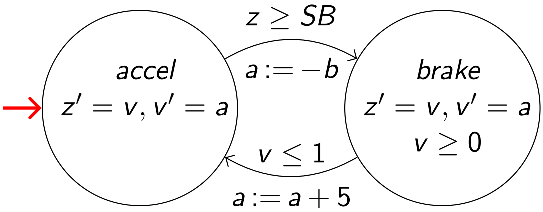
$$\text{system} \equiv (cor\,; drive)^*$$
$$cor \equiv (?MA - z \le SB; a := -b) \cup (?MA - z \ge SB; a := A)$$
$$drive \equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \land v \ge 0 \land \tau \le \varepsilon)$$



$$
\begin{array}{lcl}
\text{ETCS:} & & (\text{train} \cup \text{rbc})^* \\
\text{train} & : & \text{spd};\ \text{atp};\ \text{move} \\
\text{spd} & : & (?\tau.v \le \mathbf{m}.r;\ \tau.a := *;\ ?-b \le \tau.a \le A) \\
& & \cup(?\tau.v \ge \mathbf{m}.r;\ \tau.a := *;\ ?0 > \tau.a \ge -b) \\
\text{atp} & : & SB := \frac{\tau.v^2 - \mathbf{m}.d^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon\ \tau.v\right); \\
& & (?(\mathbf{m}.e - \tau.p \le SB \lor rbc.message = emergency);\ \tau.a := -b) \\
& & \cup(?\mathbf{m}.e - \tau.p \ge SB \land rbc.message \ne emergency) \\
\text{move} & : & t := 0;\ (\tau.p' = \tau.v, \tau.v' = \tau.a, t' = 1 \land \tau.v \ge 0 \land t \le \varepsilon) \\
\text{rbc} & : & (rbc.message := emergency) \\
& & \cup\ \big(\mathbf{m}_0 := \mathbf{m}; \mathbf{m} := *; \\
& & \quad ?\mathbf{m}.r \ge 0 \land \mathbf{m}.d \ge 0 \land \mathbf{m}_0.d^2 - \mathbf{m}.d^2 \le 2b(\mathbf{m}.e - \mathbf{m}_0.e)\big)
\end{array}
$$

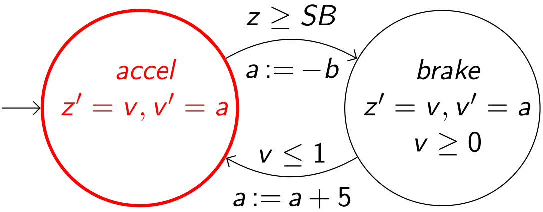
Outline

Embedding Hybrid Automata as Hybrid Programs



$$
\begin{aligned}
& q := accel; \\
& \big(\quad (?q = accel;\quad z' = v, v' = a) \\
& \cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0) \\
& \cup\ (?q = brake;\quad z' = v, v' = a \land v \geq 0) \\
& \cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel)\big)^*
\end{aligned}
$$

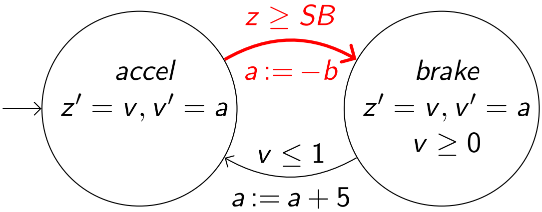
Embedding Hybrid Automata as Hybrid Programs



$\rightsquigarrow$

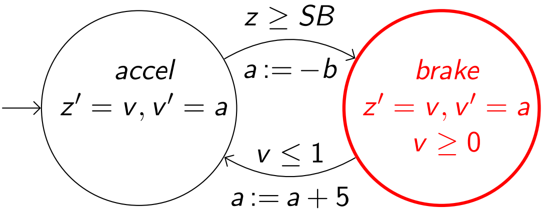
$$
\begin{aligned}
& q := accel; \\
& \big(\quad (?q = accel; \quad z' = v, v' = a) \\
& \cup \ (?q = accel \land z \geq SB; \quad a := -b; \quad q := brake; \quad ?v \geq 0) \\
& \cup \ (?q = brake; \quad z' = v, v' = a \land v \geq 0) \\
& \cup \ (?q = brake \land v \leq 1; \quad a := a + 5; \quad q := accel)\big)^*
\end{aligned}
$$

Embedding Hybrid Automata as Hybrid Programs



$\rightsquigarrow$

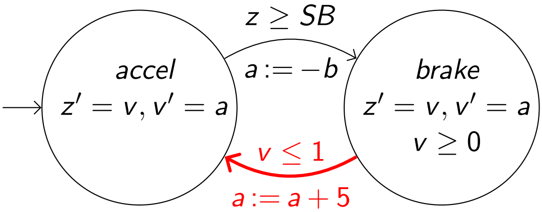
$$
\begin{aligned}
&q := accel;\\
&(\quad (?q = accel;\quad z' = v, v' = a)\\
&\cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
&\cup\ (?q = brake;\quad z' = v, v' = a \land v \geq 0)\\
&\cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel))^*
\end{aligned}
$$



$\rightsquigarrow$

$$
\begin{aligned}
& q := accel;\\
& (\quad (?q = accel;\quad z' = v, v' = a)\\
& \cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
& \cup\ (?q = brake;\quad z' = v, v' = a \land v \geq 0)\\
& \cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel)\big)^*
\end{aligned}
$$

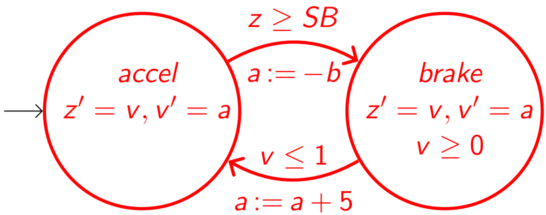
Embedding Hybrid Automata as Hybrid Programs



$\rightsquigarrow$

$$
\begin{aligned}
&q := accel;\\
&(\quad (?q = accel;\quad z' = v, v' = a)\\
&\cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
&\cup\ (?q = brake;\quad z' = v, v' = a \land v \geq 0)\\
&\cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel))^*
\end{aligned}
$$

Embedding Hybrid Automata as Hybrid Programs



$\rightsquigarrow$

$$
\begin{aligned}
&q := accel;\\
&\big(\quad (?q = accel;\quad z' = v, v' = a)\\
&\cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
&\cup\ (?q = brake;\quad z' = v, v' = a \land v \geq 0)\\
&\cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel)\big)^*
\end{aligned}
$$

Hybrid Automata

Definition (Hybrid Automata)

- Finite directed graph: vertices M (*modes*), edges E (*control switches*)
- continuous state space $\mathbb{R}^n$
- flow conditions $flow_v \subseteq \mathbb{R}^n \times \mathbb{R}^n$ determining the relationship of the continuous state $x \in \mathbb{R}^n$ and its time-derivative $x' \in \mathbb{R}^n$ during continuous evolution in mode $v \in M$;
- invariant conditions $inv_v \subseteq \mathbb{R}^n$ for $v \in M$
- jump relations $jump_e \subseteq \mathbb{R}^n \times \mathbb{R}^n$ for edges $e \in E$
  usually comprising guard on current state and reset relations

All relations definable in first-order real arithmetic

Hybrid Automata

Definition (Hybrid Automata)

- Finite directed graph: vertices M (*modes*), edges E (*control switches*)
- continuous state space $\mathbb{R}^n$
- flow conditions $flow_v \subseteq \mathbb{R}^n \times \mathbb{R}^n$ determining the relationship of the continuous state $x \in \mathbb{R}^n$ and its time-derivative $x' \in \mathbb{R}^n$ during continuous evolution in mode $v \in M$;
- invariant conditions $inv_v \subseteq \mathbb{R}^n$ for $v \in M$
- jump relations $jump_e \subseteq \mathbb{R}^n \times \mathbb{R}^n$ for edges $e \in E$
  usually comprising guard on current state and reset relations

All relations definable in first-order real arithmetic

Is this enough to do bounded model checking?

Hybrid System for Hybrid Automaton

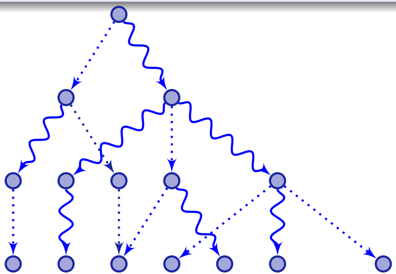
Definition (Hybrid Automata → Hybrid System)

- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.

Hybrid System for Hybrid Automaton

Definition (Hybrid Automata → Hybrid System)

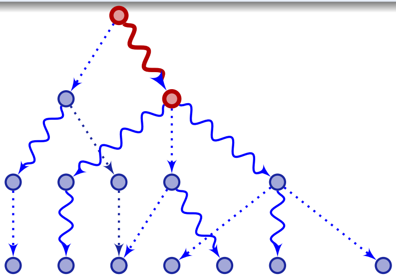
- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.



Hybrid System for Hybrid Automaton

Definition (Hybrid Automata → Hybrid System)

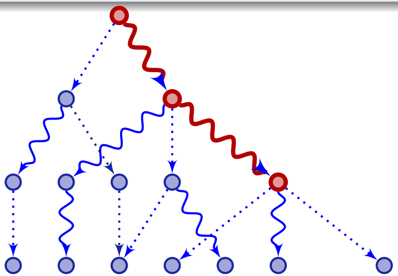
- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.



Hybrid System for Hybrid Automaton

Definition (Hybrid Automata → Hybrid System)

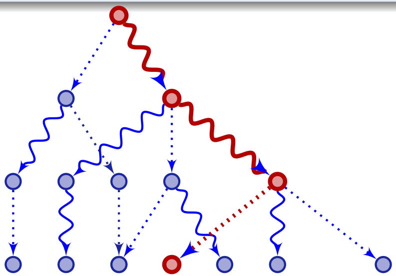
- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.



Hybrid System for Hybrid Automaton

Definition (Hybrid Automata → Hybrid System)

- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.



Hybrid System for Hybrid Automaton

Definition (Hybrid Automata → Hybrid System)

- $Q := (M \times \mathbb{R}^n) \cap \{(v, x) \ : \ x \in inv_v\}$
- Discrete transition $(v, x) \overset{a}{\curvearrowright} (v^+, x^+)$ iff there is an edge e from v to v^+ with input a such that $(x, x^+) \in jump_e$
- Continuous transition $(v, x) \overset{r}{\curvearrowright} (v, x^+)$ iff there is a differentiable function $f : [0, r] \to \mathbb{R}^n$ with $f(0) = x, f(r) = x^+$ and $(f(\zeta), f'(\zeta)) \in flow_q$ for $\zeta \in (0, r)$; and $f(\zeta) \in inv_q$ for each $\zeta \in [0, r]$.

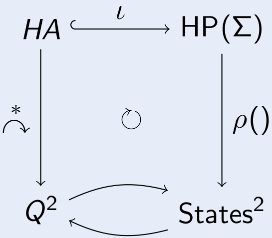
Definition (Reachability)

State $\sigma \in Q$ reachable from state $\sigma_0 \in Q$, denoted by $\sigma_0 \overset{*}{\curvearrowright} \sigma$ iff for some $n \in \mathbb{N}$, there is a sequence of states $\sigma_1, \sigma_2, \ldots, \sigma_n = \sigma \in Q$ such that $\sigma_{i-1} \curvearrowright \sigma_i$ for each $1 \leq i \leq n$. Where $\sigma_{i-1} \curvearrowright \sigma_i$ iff $\sigma_{i-1} \overset{a}{\curvearrowright} \sigma_i$ or $\sigma_{i-1} \overset{r}{\curvearrowright} \sigma_i$ for some $a \in A$ or $r \geq 0$, respectively.

Hybrid Automata Embedding Theorem

Proposition (Hybrid automata embedding)

There is an effective mapping ι such that the following diagram commutes:



Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \wedge inv_{v_1} \\ \quad \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \quad \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \wedge inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \ldots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \ldots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{r}{\curvearrowright} \sigma_n$ continuous in mode v_i for $r \geq 0$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x,x') \land inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x,x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{r}{\curvearrowright} \sigma_n$ continuous in mode v_i for $r \geq 0$.

$\Rightarrow$ There is $\varphi : [0, r] \to$ States with $\varphi(0) = \sigma_{n-1}, \varphi(r) = \sigma_n$, $\varphi \models flow_{v_i} \land inv_{v_i}$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{r}{\curvearrowright} \sigma_n$ continuous in mode v_i for $r \geq 0$.

$\Rightarrow$ There is $\varphi : [0, r] \to$ States with $\varphi(0) = \sigma_{n-1}, \varphi(r) = \sigma_n$, $\varphi \models flow_{v_i} \land inv_{v_i}$.

- α can copy the transition as $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$ using choice $?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_i}$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{r}{\curvearrowright} \sigma_n$ continuous in mode v_i for $r \geq 0$.

$\Rightarrow$ There is $\varphi : [0, r] \to$ States with $\varphi(0) = \sigma_{n-1}, \varphi(r) = \sigma_n$, $\varphi \models flow_{v_i} \land inv_{v_i}$.

- α can copy the transition as $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$ using choice $?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_i}$.
- Test succeeds, because $\Phi(\sigma_{n-1})(q) = v_i$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \left\{ \begin{array}{l} (?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \ldots\ )^* \end{array} \right.$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \ldots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{a}{\curvearrowright} \sigma_n$ discrete from mode v_i to v_j along edge e

$\square$

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ \mathit{flow}_{v_i}(x, x') \land \mathit{inv}_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?\mathit{jump}_e(x, x^+);\ x := x^+);\ ?\mathit{inv}_{v_j};\ q := v_j \\ \ \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{a}{\curvearrowright} \sigma_n$ discrete from mode v_i to v_j along edge e
- Then $(\sigma_{n-1}, \sigma) \in \mathit{jump}_e$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \wedge inv_{v_1} \\ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

- If $\sigma_{n-1} \overset{a}{\curvearrowright} \sigma_n$ discrete from mode v_i to v_j along edge e
- Then $(\sigma_{n-1}, \sigma) \in jump_e$.
- Thus, by choosing the values of σ_n for x^+, we have that $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$ by the choice $?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \cup\ \dots\ )^* \end{cases}$$

"$\subseteq$" Let $\sigma_0 \overset{*}{\curvearrowright} \sigma$, i.e., $\sigma_0 \curvearrowright \dots \sigma_{n-1} \curvearrowright \sigma_n = \sigma \in Q$.

IA $n = 0$ then $(\sigma_0, \sigma) \in \rho(\alpha^*)$ using zero repetitions.

IS By IH, $(\sigma_0, \sigma_{n-1}) \in \rho(\alpha^*)$. Show $(\sigma_{n-1}, \sigma_n) \in \rho(\alpha)$, thus $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$.

□

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \wedge inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ )^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \dots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \leq i \leq n$.

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ \mathit{flow}_{v_i}(x, x') \land \mathit{inv}_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?\mathit{jump}_e(x, x^+);\ x := x^+);\ ?\mathit{inv}_{v_j};\ q := v_j \\ \ \cup\ \ldots\ \big)^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \ldots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \leq i \leq n$.

IA For $n = 0$, there is nothing to show.

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \ldots\ \big)^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \ldots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \leq i \leq n$.

IA For $n = 0$, there is nothing to show.

IS By IH, $\sigma_{i-1} \curvearrowright \sigma_i$ for all $1 \leq i < n$. Show $\sigma_{n-1} \curvearrowright \sigma_n$, thus $\sigma_0 \overset{*}{\curvearrowright} \sigma_n$.

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x,x') \land inv_{v_1} \\ \ \cup\, ?q = v_i;\ (x^+ := *;\ ?jump_e(x,x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \dots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \leq i \leq n$.

IA For $n = 0$, there is nothing to show.

IS By IH, $\sigma_{i-1} \curvearrowright \sigma_i$ for all $1 \leq i < n$. Show $\sigma_{n-1} \curvearrowright \sigma_n$, thus $\sigma_0 \overset{*}{\curvearrowright} \sigma_n$.

- If $v_i := \sigma_{n-1}(q) = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ flow_{v_i}(x,x') \land inv_{v_i})$ by the structure of α.

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} (?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \cup\ \dots\ )^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \dots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \le i \le n$.

IA For $n = 0$, there is nothing to show.

IS By IH, $\sigma_{i-1} \curvearrowright \sigma_i$ for all $1 \le i < n$. Show $\sigma_{n-1} \curvearrowright \sigma_n$, thus $\sigma_0 \overset{*}{\curvearrowright} \sigma_n$.

- If $v_i := \sigma_{n-1}(q) = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_i})$ by the structure of α.
- Thus, $\sigma_{n-1} \overset{v_i}{\curvearrowright} \sigma_n$ by a continuous transition.

Hybrid Automata Embedding Theorem

Proof.

$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x, x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \dots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \le i \le n$.

IA For $n = 0$, there is nothing to show.

IS By IH, $\sigma_{i-1} \curvearrowright \sigma_i$ for all $1 \le i < n$. Show $\sigma_{n-1} \curvearrowright \sigma_n$, thus $\sigma_0 \overset{*}{\curvearrowright} \sigma_n$.

- If $v_i := \sigma_{n-1}(q) = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ flow_{v_i}(x, x') \land inv_{v_i})$ by the structure of α.
- If, otherwise, $v_i := \sigma_{n-1}(q) \ne v_j = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ (x^+ := *; ?jump_e(x, x^+); x := x^+); ?inv_{v_j}; q := v_j)$ according to a line of α that originates from some edge e from v_i to v_j.

Hybrid Automata Embedding Theorem

Proof.

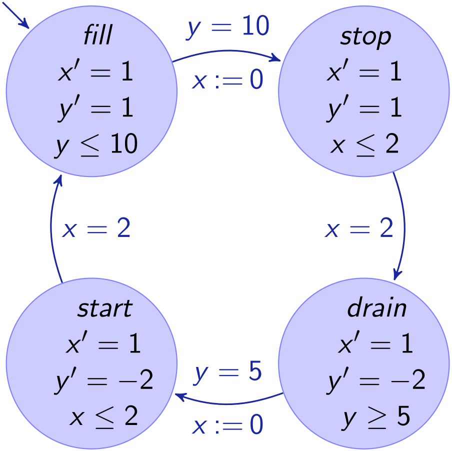
$$\alpha^* \begin{cases} \big(?q = v_i;\ flow_{v_i}(x,x') \land inv_{v_1} \\ \ \cup\ ?q = v_i;\ (x^+ := *;\ ?jump_e(x,x^+);\ x := x^+);\ ?inv_{v_j};\ q := v_j \\ \ \cup\ \dots\ \big)^* \end{cases}$$

"$\supseteq$" Let $(\sigma_0, \sigma_n) \in \rho(\alpha^*)$ along $\sigma_1, \dots, \sigma_{n-1} \in$ States with $(\sigma_{i-1}, \sigma_i) \in \rho(\alpha)$ for $1 \leq i \leq n$.

IA For $n = 0$, there is nothing to show.

IS By IH, $\sigma_{i-1} \curvearrowright \sigma_i$ for all $1 \leq i < n$. Show $\sigma_{n-1} \curvearrowright \sigma_n$, thus $\sigma_0 \overset{*}{\curvearrowright} \sigma_n$.

- If $v_i := \sigma_{n-1}(q) = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ flow_{v_i}(x,x') \land inv_{v_i})$ by the structure of α.
- If, otherwise, $v_i := \sigma_{n-1}(q) \neq v_j = \sigma_n(q)$, then $(\sigma_{n-1}, \sigma_n) \in \rho(?q = v_i;\ (x^+ := *; ?jump_e(x,x^+); x := x^+); ?inv_{v_j}; q := v_j)$ according to a line of α that originates from some edge e from v_i to v_j.
- Thus, $(\sigma_{n-1}, \sigma_n) \in jump_e$ and $\sigma_n \models inv_{v_j}$, hence, $\sigma_{n-1} \overset{e}{\curvearrowright} \sigma_n$ by a discrete transition.



$$
\begin{aligned}
q = \mathit{fill} \to [(\\
&(?q = \mathit{fill};\ x' = 1, y' = 1 \land y \leq 10)\\
\cup\, &(?q = \mathit{fill} \land y = 10;\ x := 0;\ q := \mathit{stop})\\
\cup\, &(?q = \mathit{stop};\ x' = 1, y' = 1 \land x \leq 2)\\
\cup\, &(?q = \mathit{stop} \land x = 2;\ q := \mathit{drain})\\
\cup\, &(?q = \mathit{drain};\ x' = 1, y' = -2 \land y \geq 5)\\
\cup\, &(?q = \mathit{drain} \land y = 5; x := 0; q := \mathit{start})\\
\cup\, &(?q = \mathit{start};\ x' = 1, y' = -2 \land x \leq 2)\\
\cup\, &(?q = \mathit{start} \land x = 2;\ q := \mathit{fill})\\
)^*]\, &(1 \leq y \land y \leq 12)
\end{aligned}
$$

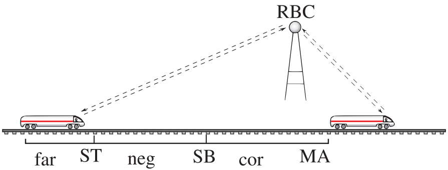
Outline

1. Motivation
2. Hybrid Programs
   - Design Motives
   - Syntax
   - Semantics
   - Train Control Examples
3. Hybrid Programs vs. Hybrid Automata
4. Differential Dynamic Logic $d\mathcal{L}$
   - Syntax
   - Semantics

$\mathcal{R}$ d$\mathcal{L}$ Motives: The Logic of Hybrid Systems

differential dynamic logic

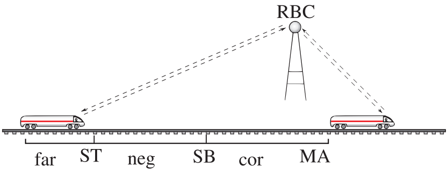
$$d\mathcal{L} = \qquad \text{DL} + \text{HP}$$

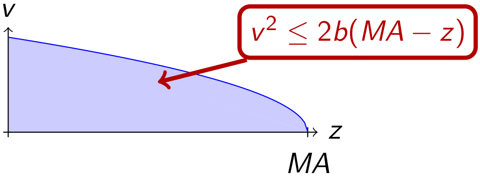


$\mathcal{R}$ d$\mathcal{L}$ Motives: Regions in First-order Logic

differential dynamic logic

$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}}$

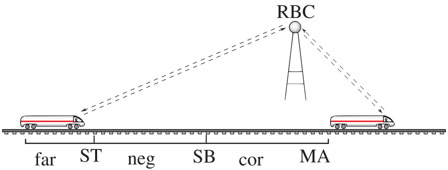




dℒ Motives: Regions in First-order Logic

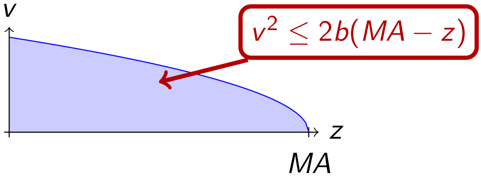
differential dynamic logic

$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}}$



$\forall MA\, \exists SB\ \ldots$

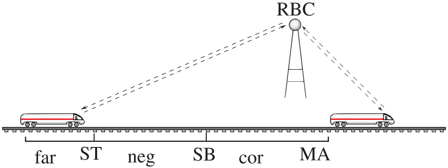
$\forall t {\geq} 0\ \ldots$



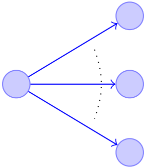
$\mathcal{A}$ d$\mathcal{L}$ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} +$



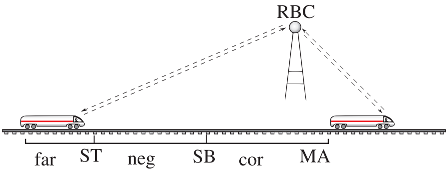
$v^2 \leq 2b$



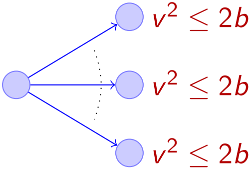
d$\mathcal{L}$ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + ML$



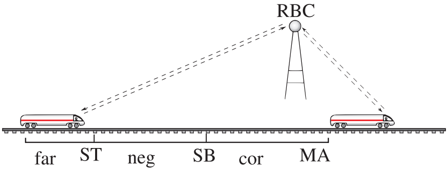
$\Box\, v^2 \leq 2b$

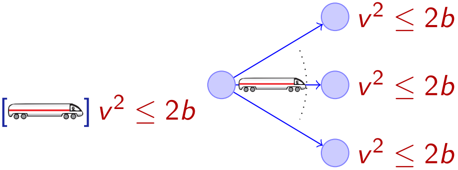


$\mathcal{A}$ d$\mathcal{L}$ Motives: State Transitions in Dynamic Logic

differential dynamic logic

$$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} + \mathrm{DL}$$

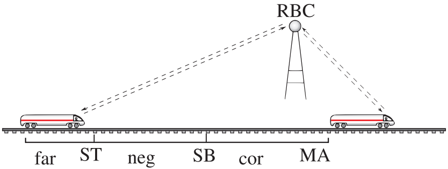




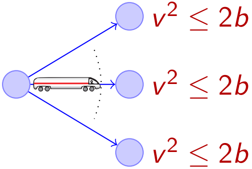
dℒ Motives: Hybrid Programs as Uniform Model

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$



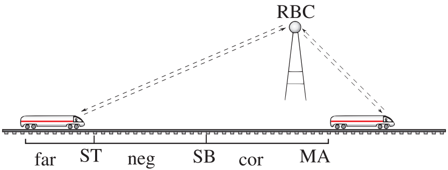
$[z'' = a]\, v^2 \leq 2b$



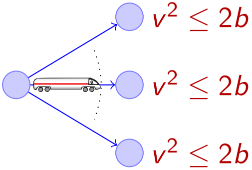
dℒ Motives: Hybrid Programs as Uniform Model

differential dynamic logic

$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$



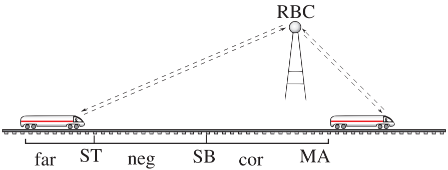
$[\mathtt{if}(z > SB)\, a := -b;\ z'' = a]\, v^2 \leq 2b$



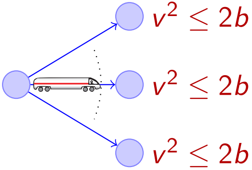
dℒ Motives: Hybrid Programs as Uniform Model

differential dynamic logic

$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$



$$[\underbrace{\text{if}(z > SB)\, a := -b;\ z'' = a}_{\text{hybrid program}}]\, v^2 \le 2b$$



Differential Dynamic Logic d$\mathcal{L}$: Syntax

Definition (d$\mathcal{L}$ Signature Σ)

Countable set of predicate or function symbols along with natural numbers as arities containing $0, 1, +, \cdot, /, =, \leq, >, \geq, <$ for reals

Differential Dynamic Logic d$\mathcal{L}$: Syntax

Definition (d$\mathcal{L}$ Signature Σ)

Countable set of predicate or function symbols along with natural numbers as arities containing $0, 1, +, \cdot, /, =, \leq, >, \geq, <$ for reals

Definition (d$\mathcal{L}$ Term t)

$t ::=$

x	for variable $x \in V$
$f(t_1, \dots, t_n)$	for function $f/n \in \Sigma$ of arity $n \geq 0$

Differential Dynamic Logic d$\mathcal{L}$: Syntax

Definition (d$\mathcal{L}$ Signature Σ)

Countable set of predicate or function symbols along with natural numbers as arities containing $0, 1, +, \cdot, /, =, \leq, >, \geq, <$ for reals

Definition (d$\mathcal{L}$ Formula ϕ, ψ)

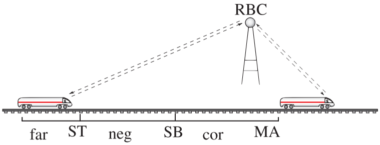
$\phi ::=$	
$[\alpha]\phi$	"all α reachables"
$\langle\alpha\rangle\phi$	"some α reachable"
$p(t_1, \dots, t_n)$	for predicate $p/n \in \Sigma$ of arity $n \geq 0$
$\neg\phi$	"not"
$(\phi \wedge \psi)$	"and"
$(\phi \vee \psi)$	"or"
$(\phi \rightarrow \psi)$	"implies"
$\forall x\, \phi$	"universal quantifier/forall" for $x \in V$
$\exists x\, \phi$	"existential quantifier/exists" for $x \in V$

Definition ($d\mathcal{L}$ Formulas ϕ)

$\neg, \wedge, \vee, \rightarrow, \ \forall x, \exists x, \ =, \leq, \ +, \cdot$ ($\mathbb{R}$-first-order part)

$[\alpha]\phi, \quad \langle\alpha\rangle\phi$ (dynamic part)

$$SB \geq \ldots \ \rightarrow \ [(ctrl; drive)^*]\, z \leq MA$$



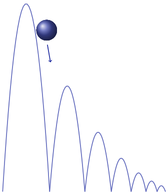
All trains respect MA
RBC partitions MA
$\Rightarrow$ system collision free

Differential Dynamic Logic d$\mathcal{L}$: Ex. Moving Point

```
/* initial state characterization */
      x^2 < (4*c)^2 →
[(
    if (x>0) then
        a := −4                /* move left */
    else
        a := 4                 /* move right */
    fi;
    t := 0;                    /* reset clock variable t */
    {x'=a, t'=1, t ≤ c}        /* continuous evolution */
)*                             /* repeat these transitions */
] (x^2 ≤ (4*c)^2)              /* safety / postcondition */
```

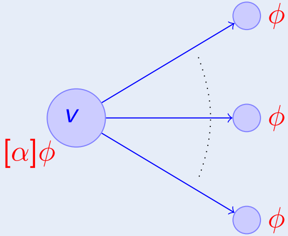
Differential Dynamic Logic d$\mathcal{L}$: Ex. Bouncing Ball

```
/* initial state characterization */
      g>0 ∧ h ≥ 0∧t ≥ 0 ∧ v^2 ≤ 2*g*(H−h) ∧ H ≥ 0 →
[(
    {h'=v, v'=−g, t'=1, h ≥ 0};  /* falling/jumping */
    if (t>0 ∧ h=0) then          /* if on ground */
        v := −c*v;               /* bounce back */
        t := 0
    fi
)*                               /* repeat these transitio
] (0 ≤ h ∧ h ≤ H)                /* safety / postcondition
```



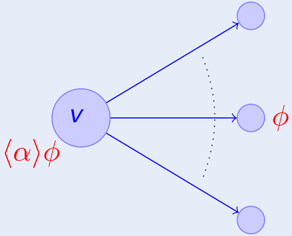
$\mathcal{A}$ Differential Dynamic Logic d$\mathcal{L}$: Semantics

Definition (Formulas ϕ)



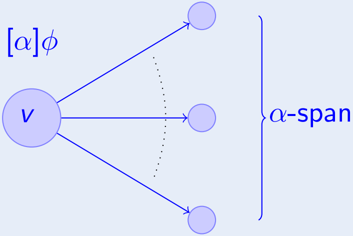
Differential Dynamic Logic d$\mathcal{L}$: Semantics

Definition (Formulas ϕ)



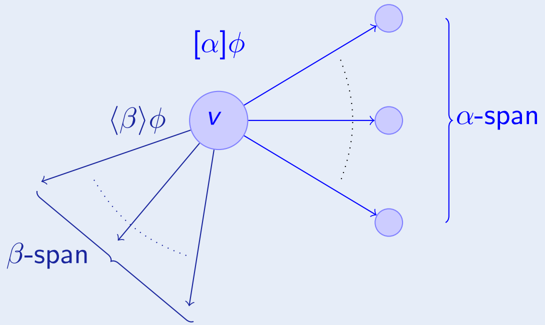
$\mathcal{A}$ Differential Dynamic Logic d$\mathcal{L}$: Semantics

Definition (Formulas ϕ)



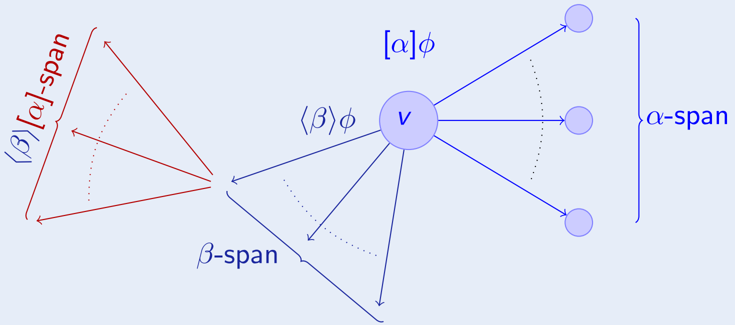
Differential Dynamic Logic d$\mathcal{L}$: Semantics

Definition (Formulas ϕ)

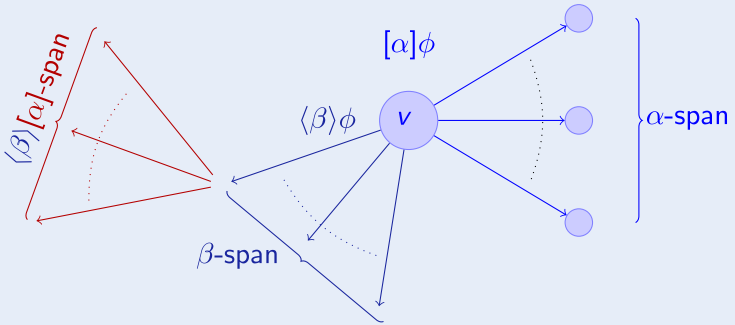


Differential Dynamic Logic d$\mathcal{L}$: Semantics

Definition (Formulas ϕ)



Definition (Formulas ϕ)



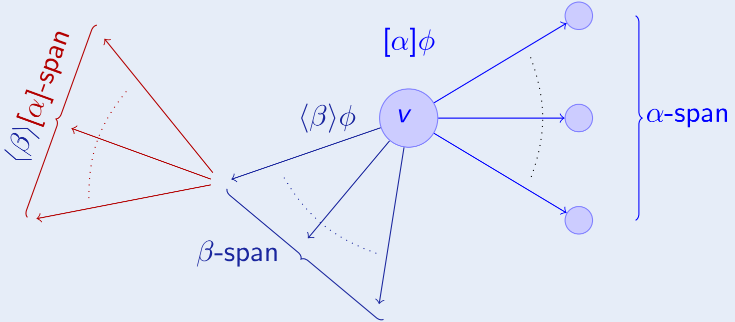
compositional semantics!

Differential Dynamic Logic d$\mathcal{L}$: Formal Semantics

Definition (Formulas ϕ)

$$
\begin{array}{lll}
v \models \theta_1 \geq \theta_2 & :\Longleftrightarrow & [\![\theta_1]\!]_v \geq [\![\theta_2]\!]_v \\
v \models \phi \land \psi & :\Longleftrightarrow & v \models \phi \text{ and } v \models \psi \\
v \models \neg\phi & :\Longleftrightarrow & v \models \phi \text{ does not hold} \\
v \models \forall x\, \phi & :\Longleftrightarrow & w \models \phi \text{ for all } w \text{ that agree with } v \\
 & & \quad \text{except for the value of } x \\
v \models \exists x\, \phi & :\Longleftrightarrow & w \models \phi \text{ for some } w \text{ that agrees with } v \\
 & & \quad \text{except for the value of } x \\
v \models [\alpha]\phi & :\Longleftrightarrow & w \models \phi \quad \text{for all } w \text{ with } (v, w) \in \rho(\alpha) \\
v \models \langle\alpha\rangle\phi & :\Longleftrightarrow & w \models \phi \quad \text{for some } w \text{ with } (v, w) \in \rho(\alpha)
\end{array}
$$

Definition (Formulas ϕ)



Nested Modalities

- $[RBC]\text{partitioned} \;\rightarrow\; \exists SB\, \langle \text{Train} \rangle [RBC]\text{safe}$

Nested Modalities

- $[RBC]$partitioned $\rightarrow$ $\exists SB\, \langle \text{Train} \rangle [RBC]$safe
- $([\text{Train}]\text{safe}) \leftrightarrow \frac{v^2}{2b} \leq m - z \ldots$

Nested Modalities

- $[RBC]\text{partitioned} \;\rightarrow\; \exists SB\,\langle\text{Train}\rangle[RBC]\text{safe}$
- $([\text{Train}]\text{safe}) \leftrightarrow \frac{v^2}{2b} \leq m - z \dots$
- $[\text{rbc}](M \rightarrow [\text{spd}]\langle SB := *\rangle[\text{atp}; \text{drive}]\text{safe})$

Nested Modalities

- $[RBC]\text{partitioned} \;\rightarrow\; \exists SB\,\langle\text{Train}\rangle[RBC]\text{safe}$
- $([\text{Train}]\text{safe}) \leftrightarrow \frac{v^2}{2b} \leq m - z \ldots$
- $[\text{rbc}](M \rightarrow [\text{spd}]\langle SB := *\rangle[\text{atp}; \text{drive}]\text{safe})$
- $[\text{aircraft}_1]\langle\text{aircraft}_2\rangle\text{separate}$

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?
- Division is definable anyhow by $z = y/x \equiv xz = y \land x \neq 0$

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?
- Division is definable anyhow by $z = y/x \equiv xz = y \wedge x \neq 0$
- But what about differential equations $x' = y/x$?

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?
- Division is definable anyhow by $z = y/x \equiv xz = y \land x \neq 0$
- But what about differential equations $x' = y/x$?
- $xx' = y \land x \neq 0$ doesn't exactly look like an ODE ...

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?
- Division is definable anyhow by $z = y/x \equiv xz = y \land x \neq 0$
- But what about differential equations $x' = y/x$?
- $xx' = y \land x \neq 0$ doesn't exactly look like an ODE ...
- They are restricted to explicit form!

Coming Back to Signatures

- Does it make a difference if $/ \in \Sigma$?
- Division is definable anyhow by $z = y/x \equiv xz = y \land x \neq 0$
- But what about differential equations $x' = y/x$?
- $xx' = y \land x \neq 0$ doesn't exactly look like an ODE ...
- They are restricted to explicit form!
- What about divisions by zero anyhow?

A. Platzer.
Differential dynamic logic for verifying parametric hybrid systems.
In N. Olivetti, editor, *TABLEAUX*, volume 4548 of *LNCS*, pages 216–232. Springer, 2007.