Logic and Hybrid Systems

Manasvi Saxena

Formal Systems Lab, UIUC

Hybrid Systems

- Dynamical Systems exhibiting both discrete (jump) and continuous (flow) behaviors.
- Serve as models of physical systems, from thermostats to trains.
- Continuous dynamics specified using Differential Equations.

Differential Dynamic Logic (dL)

- Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).

Differential Dynamic Logic (dL)

- Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).
- Practical deductive verification of hybrid systems.

Differential Dynamic Logic (dL)

- Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).
- Practical deductive verification of hybrid systems.
- Introduces Hybrid Program - program notation for hybrid systems.

Differential Dynamic Logic (dL)

- Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).
- Practical deductive verification of hybrid systems.
- Introduces Hybrid Program - program notation for hybrid systems.
- Dynamic Logic for Hybrid Programs, a generalization of Dynamic Logic.

Differential Dynamic Logic (dL)

- Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).
- Practical deductive verification of hybrid systems.
- Introduces Hybrid Program - program notation for hybrid systems.
- Dynamic Logic for Hybrid Programs, a generalization of Dynamic Logic.
- Suited for automation.

Hybrid Automata

- Commonly used to model Hybrid Systems, via Graphs.
- Nodes specify continuous dynamics. Edges describe discrete transitions.
- Intuitive, but not suitable for deductive verification.

Hybrid Automata

- Commonly used to model Hybrid Systems, via Graphs.
- Nodes specify continuous dynamics. Edges describe discrete transitions.
- Intuitive, but not suitable for deductive verification.

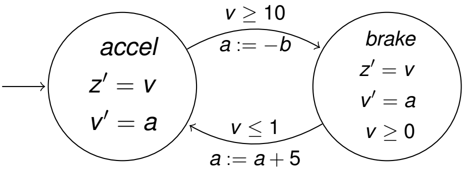

Figure: Hybrid Automata (simplified) of a Train Control System

- **First Order Logic** - No builtin means for referring to state transitions.
- **Temporal Logics** - Modal operators allow referring to state transitions. But valid formulas only express generic facts.

- **First Order Logic** - No builtin means for referring to state transitions.
- **Temporal Logics** - Modal operators allow referring to state transitions. But valid formulas only express generic facts.
- **Dynamic Logic (DL)** - Combines operational system models with operators for reasoning.
  - Provides parameterized modal operators, $[\alpha]$, $\langle\alpha\rangle$ that refer to states reachable by system α.
  - $[\alpha]\phi$ expresses all states reachable by α satisfy ϕ, allowing reasoning about discrete systems.
  - Say $(b > 0) \rightarrow [a := -b](a < 0)$ expresses a discrete transition. We can prove $(b > 0) \vdash (a < 0)[b/a]$ using DL's calculus.
  - No built in notion for describing or reasoning about continuous dynamics.

- Generalize DL so operational models α can be used in modal formulas like $[\alpha]\phi$. dL refers to generalized models as “Hybrid Programs”.
- A compositional calculus for verification. Decompose $[\alpha]\phi$ into an equivalent formula $[\alpha_1]\phi_1 \wedge [\alpha_2]\phi_2$.
- Prove subsystems and subproperties $[\alpha_i]\phi_i$ independently and combine results conjuntively.
- Complete relative to handling of differential equations.

dL formulas built over

- V, set of real-valued logical variables and signature Σ containing functions, predicate symbols over reals, like $0, 1, +, \geq$.
- Signature Σ containing functions and predicates, like $0, 1 \geq$. Σ also contains *System State Variables*. Unlike rigid symbols, like $1, 2$, their interpretation can change from state to state.
- Set $\mathrm{Trm}(\Sigma, V)$ of *terms* defined as classical FOL polynomial (or rational) expressions over V with additional skolem terms $s(X_1, \ldots, X_n)$, where $X_1, \ldots, X_n \in V$.

Hybrid Programs

Consider $x_i \in \Sigma$, $\theta_i, \vartheta_i \in \mathrm{Trm}(\Sigma, V)$ for $1 \leq i \leq n$, χ a (Σ, V) FOL-formula, $\alpha, \beta \in \mathrm{HP}(\Sigma, V)$ Set $\mathrm{HP}(\Sigma, V)$, is defined inductively as -

- $(x_1 := \theta_1, \ldots, x_n := \theta_n) \in \mathrm{HP}(\Sigma, V)$
- $(x'_1 = \vartheta_i, \ldots, x'_n = \vartheta_n) \,\&\, \chi \in \mathrm{HP}(\Sigma, V)$. $x'_i = \vartheta_i$ is a differential equation where x'_i is the first order time derivative of x_i.
- $(?\chi) \in \mathrm{HP}(\Sigma, V)$.
- $\alpha \cup \beta \in \mathrm{HP}(\Sigma, V)$.
- $\alpha; \beta \in \mathrm{HP}(\Sigma, V)$.
- $\alpha^* \in \mathrm{HP}(\Sigma, V)$.

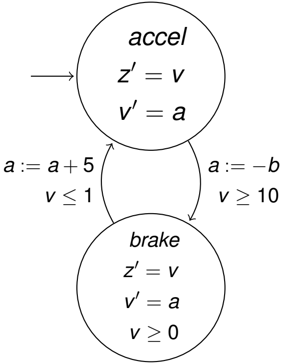

Figure: Hybrid Automata of Simple Train Control System

Differential Dynamic Logic

Hybrid Program Example

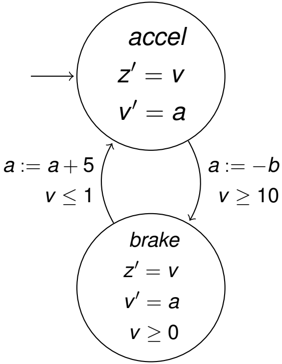

Figure: Hybrid Automata of Simple Train Control System

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

dL Formulas

Some Notation

- Interpretation *I* assigns functions and relations over Reals to rigid symbols in Σ.
- A state is a map $\nu : \Sigma_{fl} \rightarrow \mathbb{R}$.
- Assignment of logical variables is a map $\eta : V \rightarrow \mathbb{R}$.
- Note the difference between *Logical* and *State Variables*. The evaluation of a state variable “evolves” across states. Logical Variables can be quantified over, but not state variables.
- The models of dL are Kripke Structures, where nodes are hybrid system states.

dL Valuation of dL Terms

Say $val_{I,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation I, assignment η and state ν.

dL Valuation of dL Terms

Say $val_{I,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation I, assignment η and state ν.

- (State Variables) $val_{I,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$

dL Valuation of dL Terms

Say $val_{I,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation I, assignment η and state ν.

- (State Variables) $val_{I,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$
- (Logical Variables) $val_{I,\eta}(\nu, x) = \eta(x), x \in V$

dL Valuation of dL Terms

Say $val_{I,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation I, assignment η and state ν.

- (State Variables) $val_{I,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$
- (Logical Variables) $val_{I,\eta}(\nu, x) = \eta(x), x \in V$
- (Rigid Symbols)
  $val_{I,\eta}(\nu, f(\theta_1, \ldots, \theta_n)) = f_I(val_{I,\eta}(\nu, \theta_1), \ldots, val_{I,\eta}(\nu, \theta_n))$
  where f is n-ary rigid symbol in Σ

Valuation of dL Formulas

- $val_{I,\eta}(\nu, p(\theta_1, \ldots, \theta_n)) = p_I(val_{I,\eta}(\nu, \theta_1), \ldots, val_{I,\eta}(\nu, \theta_n))$
- $val_{I,\eta}(\nu, \varphi \wedge \psi) = \top$ iff $val_{I,\eta}(\nu, \varphi) = \top \wedge val_{I,\eta}(\nu, \psi) = \top$. Similarly for $\rightarrow, \neg, \vee$
- $val_{I,\eta}(\nu, \exists x.\, \varphi) = \top$ iff $val_{I,\eta[x \mapsto d]}(\nu, \varphi) = \top$ for some $d \in \mathbb{R}$

Valuation of dL Formulas

- $val_{I,\eta}(\nu, p(\theta_1, \ldots, \theta_n)) = p_I(val_{I,\eta}(\nu, \theta_1), \ldots, val_{I,\eta}(\nu, \theta_n))$
- $val_{I,\eta}(\nu, \varphi \wedge \psi) = \top$ iff $val_{I,\eta}(\nu, \varphi) = \top \wedge val_{I,\eta}(\nu, \psi) = \top$. Similarly for $\rightarrow, \neg, \vee$
- $val_{I,\eta}(\nu, \exists x.\, \varphi) = \top$ iff $val_{I,\eta[x\mapsto d]}(\nu, \varphi) = \top$ for some $d \in \mathbb{R}$
- $val_{I,\eta}(\nu, [\alpha]\varphi) = \top$ iff $val_{I,\eta}(\omega, \varphi) = \top$ for all states ω with $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$.
- $val_{I,\eta}(\nu, \langle\alpha\rangle\varphi) = \top$ iff $val_{I,\eta}(\omega, \varphi) = \top$ for some state ω with $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$.

Transition Semantics of Hybrid Programs

Evaluation $\rho_{I,\eta}(\alpha)$ of HP α. $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$ means state ω is reachable from ν by operations of α.

- $(\nu, \omega) \in \rho_{I,\eta}(x_1 := \theta_1, \ldots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto val_{I,\eta}(\nu, \theta_1), \ldots, x_n \mapsto val_{I,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \ldots, x_n\}). val_{I,\eta}(\nu, y) = val_{I,\eta}(\omega, y)$.

Transition Semantics of Hybrid Programs

Evaluation $\rho_{I,\eta}(\alpha)$ of HP α. $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$ means state ω is reachable from ν by operations of α.

- $(\nu, \omega) \in \rho_{I,\eta}(x_1 := \theta_1, \ldots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto val_{I,\eta}(\nu, \theta_1), \ldots, x_n \mapsto val_{I,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \ldots, x_n\}).val_{I,\eta}(\nu, y) = val_{I,\eta}(\omega, y)$.
- $\rho_{I,\eta}(\alpha \cup \beta) = \rho_{I,\eta}(\alpha) \cup \rho_{I,\eta}(\beta)$

Transition Semantics of Hybrid Programs

Evaluation $\rho_{I,\eta}(\alpha)$ of HP α. $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$ means state ω is reachable from ν by operations of α.

- $(\nu, \omega) \in \rho_{I,\eta}(x_1 := \theta_1, \ldots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto val_{I,\eta}(\nu, \theta_1), \ldots, x_n \mapsto val_{I,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \ldots, x_n\}).val_{I,\eta}(\nu, y) = val_{I,\eta}(\omega, y)$.
- $\rho_{I,\eta}(\alpha \cup \beta) = \rho_{I,\eta}(\alpha) \cup \rho_{I,\eta}(\beta)$
- $\rho_{I,\eta}(?\chi) = \{(\nu, \nu) : val_{I,\eta}(\nu, \chi) = \top\}$

Transition Semantics of Hybrid Programs

Evaluation $\rho_{I,\eta}(\alpha)$ of HP α. $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$ means state ω is reachable from ν by operations of α.

- $(\nu, \omega) \in \rho_{I,\eta}(x_1 := \theta_1, \ldots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto val_{I,\eta}(\nu, \theta_1), \ldots, x_n \mapsto val_{I,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \ldots, x_n\}). val_{I,\eta}(\nu, y) = val_{I,\eta}(\omega, y)$.
- $\rho_{I,\eta}(\alpha \cup \beta) = \rho_{I,\eta}(\alpha) \cup \rho_{I,\eta}(\beta)$
- $\rho_{I,\eta}(?\chi) = \{(\nu, \nu) : val_{I,\eta}(\nu, \chi) = \top\}$
- $\rho_{I,\eta}(\alpha; \beta) = \{(\nu, \omega) : (\nu, z) \in \rho_{I,\eta}(\alpha) \wedge (z, \omega) \in \rho_{I,\eta}(\beta)$ for some state $z\}$

Transition Semantics of Hybrid Programs

Evaluation $\rho_{I,\eta}(\alpha)$ of HP α. $(\nu, \omega) \in \rho_{I,\eta}(\alpha)$ means state ω is reachable from ν by operations of α.

- $(\nu, \omega) \in \rho_{I,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto val_{I,\eta}(\nu, \theta_1), \dots, x_n \mapsto val_{I,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \dots, x_n\}). val_{I,\eta}(\nu, y) = val_{I,\eta}(\omega, y)$.
- $\rho_{I,\eta}(\alpha \cup \beta) = \rho_{I,\eta}(\alpha) \cup \rho_{I,\eta}(\beta)$
- $\rho_{I,\eta}(?\chi) = \{(\nu, \nu) : val_{I,\eta}(\nu, \chi) = \top\}$
- $\rho_{I,\eta}(\alpha; \beta) = \{(\nu, \omega) : (\nu, z) \in \rho_{I,\eta}(\alpha) \wedge (z, \omega) \in \rho_{I,\eta}(\beta)$ for some state $z\}$
- $(\nu, \omega) \in \rho_{I,\eta}(\alpha^*)$ iff for $n \in \mathbb{N}$, there are states $\nu = \nu_0, \dots, \nu_n = \omega$ s.t. $(\nu_i, \nu_{i+1}) \in \rho_{I,\eta}(\alpha)$ for $0 \leq i < n$.

Transition Semantics of Hybrid Programs

$(\nu, \omega) \in \rho_{I,\eta}((x_1' = \vartheta_1 \ldots x_n' = \vartheta_n) \& \chi)$ iff there is a flow of some duration $r \geq 0$, from ν to ω respecting the differential equations and evolution domain. Formally, there is a function $f : [0, r] \to \mathrm{Sta}(\Sigma)$ such that -

- $f(0) = \nu$ and $f(r) = \omega$.
- $\forall \delta : [0, r].val_{I,\eta}(f(\delta), x_i)$ is continuous and $val_{I,\eta}(f(\delta), \chi) = \top$.
- $\forall \epsilon : (0, r).val_{I,\eta}(f(\epsilon), \vartheta_i) = \dot{f}(\epsilon)$.
- For any $z \notin \{x_1, x_2, \ldots, x_n\}$, $val_{I,\eta}(f(\zeta), z)$ remains constant for $\zeta \in [0, r]$. In other words, all other state variables remain unchanged.

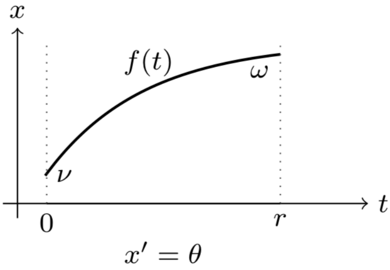

Figure: Unbounded Evolution

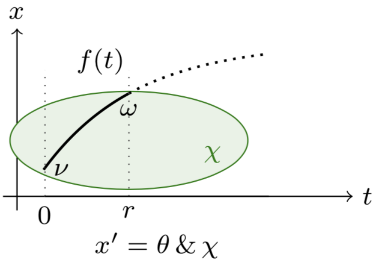

Figure: Evolution bound by χ