Verifying Switched System Stability With Logic

Yong Kiam Tan  Stefan Mitsch  André Platzer

Computer Science Department, Carnegie Mellon University

HSCC, Online, 5 May 2022
Outline

1. Switched Systems and Stability
2. Switched Systems as Hybrid Programs
3. Loop Invariants for Stability
4. Implementation & Case Studies
5. Conclusion
Outline

1. Switched Systems and Stability
2. Switched Systems as Hybrid Programs
3. Loop Invariants for Stability
4. Implementation & Case Studies
5. Conclusion
Hybrid Systems & Stability

Many real world systems feature **hybrid** (discrete + continuous) dynamics:
Hybrid Systems & Stability

Many real world systems feature **hybrid** (discrete + continuous) dynamics:

Various controllers driving a car near cruising velocity $V_c$:

- ✓ Cruise control
- ✗ Converge to $V_c$
- ✗ Stay close to $V_c$

Stability is a key correctness criterion for control systems deserving proofs. **Prior work**: Stability verification for ordinary diff. eqs. [TACAS’21].
Fact: Hybrid **switching** control can be used to achieve control objectives that cannot otherwise be achieved by purely continuous means.

Example: Discontinuity in controller, e.g., with switching, is needed to invert pendulum globally, from all initial states.

Others: Adaptive control, bang-bang control, gain scheduling, ...
**Fact:** Hybrid **switching** control can be used to achieve control objectives that cannot otherwise be achieved by purely continuous means.

**Fact:** Discrete switching between stable continuous ODEs can be unstable.
Switched Systems & Stability

**Fact:** Hybrid *switching* control can be used to achieve control objectives that cannot otherwise be achieved by purely continuous means.

**Fact:** Discrete switching between stable continuous ODEs can be unstable.

**Challenge:** Need adequate stability justification for switching designs, e.g., state-dependent [1, 5], time-dependent [8], automata-based [3, 4], . . .

**This work:** Trustworthy, uniform stability verification framework for switching designs by combining ideas from controls & verification.
Verifying Switched System Stability With Logic

Switched System Stability Verification

Controls
- Switching Mechanisms
- Uniform Global Pre-Asymptotic Stability
- Common/Multiple Lyapunov Functions
- Examples & Case Studies

Verification
- Hybrid Programs
- Differential Dynamic Logic (dL)
- Loop Invariants & Compositional Proofs
- KeYmaera X

Switched System Stability Verification
Outline

1. Switched Systems and Stability

2. Switched Systems as Hybrid Programs

3. Loop Invariants for Stability

4. Implementation & Case Studies

5. Conclusion
Switched Systems

Switched systems consist of a family of continuous ODEs and a discrete switching signal choosing between those ODEs.

Switched system:

\[ x' = f_{\sigma(t)}(x) \]

- \( x' = f_p(x), p \in P \), finite family of autonomous ODEs
- \( \sigma(t) \), switching signal chooses ODE to follow at time \( t \)

\[ \sigma(t) \]

\[ \sigma = 1 \]

\[ \sigma = 2 \]

\[ \sigma = 3 \]

\[ x'(t) \]

\[ x_1' = f_1(x) \]

\[ x_2' = f_2(x) \]

\[ x_3' = f_3(x) \]

\[ t \]

Controller 1

Controller 2

Controller 3

\[ \ldots \]

Switching Logic

Controller Switching

Plant
Hybrid Programs

Differential dynamic logic (dL) uses the **hybrid programs** language to model hybrid systems.

Hybrid programs:

\[
\alpha, \beta ::= x' = f(x) \land Q \mid x := e \mid ?Q \mid \alpha; \beta \mid \alpha \cup \beta \mid \alpha^*
\]

- **ODE**
- **Discrete Assign.**
- **Test**
- **Seq. Compose**
- **Nondet. Choice**
- **Nondet. Loop**

Properties of hybrid program \(\alpha\) are specified in dL's formula language.
Hybrid Programs

Differential dynamic logic (dL) uses the **hybrid programs** language to model hybrid systems.

### Hybrid programs:

\[
\alpha, \beta ::= x' = f(x) \& Q \mid x := e \mid ?Q \mid \alpha; \beta \mid \alpha \cup \beta \mid \alpha^*
\]

|-----|------------------|------|--------------|----------------|--------------|

Properties of hybrid program \(\alpha\) are specified in dL’s formula language.

### Specifications:

\[
\phi, \psi ::= e \sim \tilde{e} \mid \phi \land \psi \mid \cdots \mid \forall x \phi \mid \exists x \phi \mid [\alpha]\phi \mid \langle \alpha \rangle\phi
\]

<table>
<thead>
<tr>
<th>Compare</th>
<th>And, Or, etc.</th>
<th>For all, Exists</th>
<th>(\phi) true after all (\alpha) runs</th>
<th>(\phi) true after some (\alpha) run</th>
</tr>
</thead>
</table>

Red boxes are key for switched system stability specifications.
Switched Systems as Hybrid Programs [ADHS’21]

Hybrid program:

\[ \alpha_{arb} \equiv \left( \bigcup_{p \in \mathcal{P}} x' = f_p(x) \right) \]
\[ \bigcup_{p \in \mathcal{P}} \alpha_p \equiv \alpha_1 \cup \alpha_2 \cup \ldots \cup \alpha_m \]

Repeat switching in a loop

Each iteration nondet. picks an ODE to run
Switched Systems as Hybrid Programs [ADHS’21]

Hybrid program:
\[ \alpha_{arb} \equiv \left( \bigcup_{p \in P} x' = f_p(x) \right)^* \]
\[ \bigcup_{p \in P} \alpha_p \equiv \alpha_1 \cup \alpha_2 \cup \ldots \cup \alpha_m \]

Repeat switching in a loop

Each iteration nondet. picks an ODE to run

Hybrid program (simplified):
\[ \alpha_{ctrl} \equiv \left( u := ctrl(x); x' = f_u(x) \right)^* \]

Switching uses specification and reasoning for loops in dL.
Uniform Global Pre-Asymptotic Stability [Goebel et al.]

Switched system model $\alpha$ & specification

Switched system is $\text{UGpAS}$ iff:

- **Unif. Stable:** for all $\epsilon > 0$, there exists $\delta > 0$, all switching solutions $\varphi$ from $\|\varphi(0)\| < \delta$ satisfy $\|\varphi(t)\| < \epsilon$ for all times.

- **Unif. Pre-Attractive:** for all $\epsilon > 0, \delta > 0$, there exists $T \geq 0$, all switching solutions $\varphi$ from $\|\varphi(0)\| < \delta$ satisfy $\|\varphi(t)\| < \epsilon$ for all times $T \leq t$.

**dL UGpAS specification:**

- **Unif. Stable:**
  \[
  \forall \epsilon > 0 \exists \delta > 0 \forall x (\|x\| < \delta \rightarrow [\alpha] \|x\| < \epsilon)
  \]

- **Unif. Pre-Attractive:**
  \[
  \forall \epsilon > 0 \forall \delta > 0 \exists T \geq 0 \forall x (\|x\| < \delta \rightarrow [t := 0; \alpha, t' = 1] (T \leq t \rightarrow \|x\| < \epsilon))
  \]
Switched system model $\alpha$ & specification

Switched system is **UGpAS** iff:
- **Unif. Stable**: for all $\varepsilon > 0$, there exists $\delta > 0$, all switching solutions $\varphi$ from $\|\varphi(0)\| < \delta$ satisfy $\|\varphi(t)\| < \varepsilon$ for all times.

**dL UGpAS specification**:
- **Unif. Stable**:
  $$\forall \varepsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \rightarrow [\alpha] \|x\| < \varepsilon \right)$$

**This talk**: Focuses on deductive dL proofs of (uniform) stability for switched systems, i.e., “if system starts close to origin, it stays close”.

![Diagram showing a switched system and its stability criteria](image-url)
Outline

1. Switched Systems and Stability
2. Switched Systems as Hybrid Programs
3. Loop Invariants for Stability
4. Implementation & Case Studies
5. Conclusion
Stability Under Arbitrary Switching

Hybrid program & Stability:

\[ \alpha_{arb} \equiv \left( \bigcup_{p \in P} x' = f_p(x) \right)^* \]

\[ \forall \varepsilon > 0 \ \exists \delta > 0 \ \forall x \ (\|x\| < \delta \rightarrow [\alpha_{arb}] \|x\| < \varepsilon) \]
Stability Under Arbitrary Switching

Hybrid program & Stability:
\[ \alpha_{arb} \equiv \left( \bigcup_{p \in \mathcal{P}} x' = f_p(x) \right)^* \]
\[ \forall \varepsilon > 0 \ \exists \delta > 0 \ \forall x \ (\|x\| < \delta \rightarrow [\alpha_{arb}] \|x\| < \varepsilon) \]

Arithmetic conditions on common Lyapunov function \( V \) for all modes:

- \( V(0) = 0 \) and \( V(x) > 0 \) for all \( \|x\| > 0 \);
- for each ODE \( x' = f_p(x), p \in \mathcal{P} \), the Lie derivative \( \mathcal{L}_{f_p}(V) \) satisfies \( \mathcal{L}_{f_p}(V) \leq 0 \).
Hybrid program & Stability:
\[ \alpha_{arb} \equiv \left( \bigcup_{p \in P} x' = f_p(x) \right)^* \]

\[ \forall \varepsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \rightarrow \left[ \alpha_{arb} \right] \|x\| < \varepsilon \right) \]

Loop invariant \( Inv \) is preserved across all loop iterations for \( \alpha_{arb} \):
Stability Under Arbitrary Switching

Hybrid program & Stability:

$$\alpha_{arb} \equiv \left( \bigcup_{p \in P} x' = f_p(x) \right)^*$$

$$\forall \varepsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \to [\alpha_{arb}] \|x\| < \varepsilon \right)$$

Formal proof syntactically deduces sound arithmetic conditions on $V$:

\[ \Gamma \vdash Inv \]

\[ \vdash V(0) = 0 \quad \|x\| > 0 \vdash V(x) > 0 \quad \vdash L_{f_p}(V)(x) \leq 0 \]

\[ \vdash \bigcup_{p \in P} x' = f_p(x) \]

\[ \vdash L_{f_p}(V)(x) \leq 0 \]

\[ \Gamma \vdash \|x\| < \varepsilon \]

\[ \vdash \forall \varepsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \to [\alpha_{arb}] \|x\| < \varepsilon \right) \]
Stability Under Arbitrary Switching

Hybrid program & Stability:

\[ \alpha_{arb} \equiv \left( \bigcup_{p \in P} x' = f_p(x) \right)^* \]

\[ \forall \varepsilon > 0 \, \exists \delta > 0 \, \forall x \left( \|x\| < \delta \rightarrow [\alpha_{arb}] \|x\| < \varepsilon \right) \]

Formal proof syntactically deduces **sound arithmetic conditions** on \( V \):

Summarized as a derived dL proof rule and implemented in KeYmaera X:

\[
\frac{\vdash V(0) = 0 \quad \|x\| > 0 \vdash V(x) > 0 \quad \vdash \mathcal{L}_{f_p}(V)(x) \leq 0}{\vdash \forall \varepsilon > 0 \, \exists \delta > 0 \, \forall x \left( \|x\| < \delta \rightarrow [\alpha_{arb}] \|x\| < \varepsilon \right)}
\]
Hybrid program & Stability:
\[ \alpha_{ctrl} \equiv \left( u := \text{ctrl}(x); x' = f_u(x) \right)^* \]

\[ \forall \epsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \rightarrow \right. \]

\[ \left. \left[ \alpha_{ctrl} \right] \|x\| < \epsilon \right) \]

Compositional proof yields **correct-by-construction** conditions on \( V_p \):

**Deduction**

\[
\begin{array}{c}
\vdash Inv \equiv \|x\| < \epsilon \land \bigvee_{p \in P} \left( u = p \land V_p < W \right) \\
\end{array}
\]

\[
\begin{array}{c}
\Gamma \vdash Inv \\
\vdash Inv \vdash \left[ u := \text{ctrl}(x) \right] \quad Inv \vdash \left[ x' = f_u(x) \right] \\
\vdash \left[ u := \text{ctrl}(x); x' = f_u(x) \right] Inv \\
\vdash Inv \vdash \|x\| < \epsilon \\
\vdash \forall \epsilon > 0 \exists \delta > 0 \forall x \left( \|x\| < \delta \rightarrow \left[ \alpha_{ctrl} \right] \|x\| < \epsilon \right) \\
\end{array}
\]
Outline

1 Switched Systems and Stability
2 Switched Systems as Hybrid Programs
3 Loop Invariants for Stability
4 Implementation & Case Studies
5 Conclusion
The implementation adds switched system support to KeYmaera X’s IDE and fully automates arguments for standard switching designs.

Users switched systems in graph-based language.

Automatically generated, user-customizable dL models and specifications for stability/attractivity/etc.
The implementation adds switched system support to KeYmaera X’s IDE and fully automates arguments for standard switching designs.

Users can input Lyapunov function(s) or generate candidates automatically with sum-of-squares techniques.

KeYmaera X automates stability reasoning for standard classes of switching mechanisms using the input Lyapunov function(s).
Case Studies (see paper)

Semi-automated verification of **non-standard** switching design/arguments:

**Nonholonomic Integrator:**
\[
x' = u, \quad y' = v, \quad z' = xv - yu
\]

**Canonical Max System:**
\[
x' = y, \quad y' = -ax - by + \max(fx + gy + \gamma, 0)
\]

**Automatic cruise controller:**

\[
\forall \text{ eps ( eps > 0 -> ... // Abridged stability specification}
\]

[ ... // Initialize
\]

\[
\{ ... ++ // Transitions for other modes
\quad ?\text{mode} = \text{normalPI()};
\quad \{ ?13 \leq v \land v \leq 15 \land -500 \leq x \land x \leq 500; t := 0; } \quad \text{mode} := \text{sbrakeact()}; ++ ... \} \}
\]

\[
\{ ... ++ // Plant ODEs for other modes
\quad ?\text{mode} = \text{normalPI()};
\quad \{ v' = -0.001x - 0.052v, \quad x' = v, \quad t' = 0 \quad & ... \} \}
\]

\[
\}
\]

\[
\text{switching loop}
\]

\[
\text{v}^2 < \text{eps}^2
\]
Case Studies (see paper)

Semi-automated verification of **non-standard** switching design/arguments:

**Nonholonomic Integrator:**
\[ x' = u, y' = v, z' = xv - yu \]

Uses an initial event- or time-triggered control \( u, v \) to drive system out of **inapplicable region**.

**Canonical Max System:**
\[ x' = y, y' = -ax - by + \max(fx + gy + \gamma, 0) \]

**Automatic cruise controller:**
\[
\forall \text{eps} \quad (\text{eps} > 0 \rightarrow \ldots) \quad // \text{Abridged stability specification}
\]

\[
[ \ldots \quad // \text{Initialize}
\]
\[
\{ \{ \ldots \quad ++ \quad // \text{Transitions for other modes}
\quad ?\text{mode} = \text{normalPI}();
\quad \{ \{?13 \leq v \& v \leq 15 \& -500 \leq x \& x \leq 500; t := 0;\}
\quad \quad \text{mode} := \text{sbrakeact}(); ++ \ldots \} \}
\quad \{ \ldots ++ \quad // \text{Plant ODEs for other modes}
\quad ?\text{mode} = \text{normalPI}();
\quad \{ v' = -0.001*x - 0.052*v, x' = v, t' = 0 \& \ldots \} \}
\}\quad // \text{Switching loop}
\]
\[
\quad v^2 < \text{eps}^2
\]
Semi-automated verification of **non-standard** switching design/arguments:

**Nonholonomic Integrator:**

\[
\begin{align*}
x' &= u, \\
y' &= v, \\
z' &= xv - yu
\end{align*}
\]

Uses an initial event- or time-triggered control \(u, v\) to drive system out of **inapplicable region**.

**Canonical Max System:**

\[
x' = y, \\
y' = -ax - by + \max(fx + gy + \gamma, 0)
\]

Models longitudinal flight dynamics with elevator controller. Proof uses “non-customary” Lyapunov function.

Automatic cruise controller:

```latex
\forall \text{eps ( eps > 0 -> ... // Abridged stability specification} \\
[ ... // Initialize \\
\begin{cases} \{ \text{ ... ++ // Transitions for other modes} \\
\text{mode} = \text{normalPI()}; \\
\{ ?13 \leq v \land v \leq 15 \land -500 \leq x \land x \leq 500; t := 0; \} \\
\text{mode} := \text{brakeact()}; ++ ... \} \} \\
\text{... ++ // Plant ODEs for other modes} \\
\text{mode} = \text{normalPI()}; \\
\{ v' = -0.001*x - 0.052*v, \ x' = v, \ t' = 0 \ \& ... \} \} \\
\text{...} \} \\
\} \\
\text{)}* // Switching loop \\
\text{] v^2 < eps^2}
```

[Diagram of Nonholonomic Integrator and Canonical Max System]
Case Studies (see paper)

Semi-automated verification of **non-standard** switching design/arguments:

**Nonholonomic Integrator:**
\[ x' = u, \ y' = v, \ z' = xv - yu \]

Uses an initial event- or time-triggered control \( u, v \) to drive system out of **inapplicable region**.

**Canonical Max System:**
\[ x' = y, \ y' = -ax - by + \max(fx + gy + \gamma, 0) \]

Models longitudinal flight dynamics with elevator controller. Proof uses “non-customary” Lyapunov function.

**Automatic cruise controller:**
\[
\forall \epsilon > 0 \rightarrow \ldots \ // \ \text{Abridged stability specification}
[ \ldots \ // \ \text{Initialize}
\]

Hybrid automaton with 6 modes and 11 transitions: PI control, acceleration, service braking (2 modes), and emergency braking (2 modes).
Outline

1. Switched Systems and Stability
2. Switched Systems as Hybrid Programs
3. Loop Invariants for Stability
4. Implementation & Case Studies
5. Conclusion
Verifying Switched System Stability With Logic

Controls

Switching Mechanisms
Uniform Global Pre-Asymptotic Stability
Common/Multiple Lyapunov Functions
Examples & Case Studies

Verification

Hybrid Programs
Differential Dynamic Logic (dL)
Loop Invariants & Compositional Proofs
KeYmaera X

Switched System Stability Verification

This work: Automated support for modeling and trustworthy stability verification of various switching designs using dL and KeYmaera X.
References I


