Saturation-based Scaling Techniques for Symbolic Verification of Hybrid Systems

André Platzer  Edmund M. Clarke

Carnegie Mellon University, Computer Science Department, Pittsburgh, PA

Outline

1. Motivation

2. Compositional Verification Logic $\mathcal{dL}$

3. Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4. Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5. Case Studies & Experimental Results

6. Conclusions & Future Work
Hybrid Systems

continuous evolution along differential equations + discrete change
Hybrid Systems

continuous evolution along differential equations + discrete change
Hybrid Systems

continuous evolution along differential equations + discrete change

\[
\begin{aligned}
\dot{x}_1 &= -v_1 + v_2 \cos \vartheta + \omega x_2 \\
\dot{x}_2 &= v_2 \sin \vartheta - \omega x_1 \\
\dot{\vartheta} &= \varrho - \omega
\end{aligned}
\]
Example ("Solving" differential equations)

\[
x_1(t) = \frac{1}{\omega} \left( x_1 \omega \rho \cos t \omega - v_2 \omega \cos t \omega \sin \vartheta + v_2 \omega \cos t \omega \cos t \rho \sin \vartheta - v_1 \rho \sin t \omega \\
+ x_2 \omega \rho \sin t \omega - v_2 \omega \cos \vartheta \cos t \rho \sin t \omega - v_2 \omega \sqrt{1 - \sin^2 \vartheta} \sin t \omega \\
+ v_2 \omega \cos \vartheta \cos t \omega \sin t \rho + v_2 \omega \sin \vartheta \sin t \omega \sin t \rho \right) \ldots
\]
Example ("Solving" differential equations)

\[ \forall t \geq 0 \quad \frac{1}{\omega} \left( x_1 \omega \cos \omega + v_2 \omega \cos \omega \sin \theta + v_2 \omega \cos \omega \cos \theta \sin \theta - v_1 \omega \sin \theta - x_2 \omega \sin \omega + v_2 \omega \cos \theta \cos \omega \sin \omega - v_2 \omega \sqrt{1 - \sin^2 \theta} \sin \omega + v_2 \omega \cos \theta \cos \omega \sin \omega + v_2 \omega \sin \theta \sin \omega \sin \omega \sin \omega \right) \ldots \]
Symbolic Verification

- constant/nilpotent dynamics
- otherwise “no” solutions
- sound

Numerical Verification

- challenging dynamics
- approximation errors
- unsound, ... see [PC07]

\[
\begin{bmatrix}
 x_1' \\
 x_2' \\
 \vartheta'
\end{bmatrix} = \begin{bmatrix}
 -v_1 + v_2 \cos \vartheta + \omega x_2 \\
 v_2 \sin \vartheta - \omega x_1 \\
 \varrho - \omega
\end{bmatrix}
\]
Verification of Hybrid Systems & Air Traffic Control

\[ \begin{align*}
    x_1' &= -v_1 + v_2 \cos \theta + \omega x_2 \\
    x_2' &= v_2 \sin \theta - \omega x_1 \\
    \theta' &= \phi - \omega
\end{align*} \]

How To Get What We Really Need?

✓ challenging dynamics, e.g., curved flight
✓ automatic verification
✓ sound
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)
“Property that remains true in the direction of the dynamics”
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find all at once? 10000-dim
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find local diff. invariants?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find local diff. invariants?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find local diff. invariants?
- How to put local differential invariants together?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)
“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find local diff. invariants?
- How to put local differential invariants together?
- How do discrete transitions fit?
Idea: Exploit Vector Field of Differential Equations

“Definition” (Differential Invariant)

“Property that remains true in the direction of the dynamics”

- How to find diff. invariants?
- How do diff. invariants fit together?
- Find local diff. invariants?
- How to put local differential invariants together?
- How do discrete transitions fit?
- What does “fit” really mean?
Outline

1 Motivation

2 Compositional Verification Logic dL

3 Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4 Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5 Case Studies & Experimental Results

6 Conclusions & Future Work
Outline

1 Motivation

2 Compositional Verification Logic $\mathcal{dL}$

3 Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4 Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5 Case Studies & Experimental Results

6 Conclusions & Future Work
Example

\( \text{safe} \land \text{far} \rightarrow \left[ \text{entry} \right] (\text{safe} \land \text{tangential}) \)

where \( \text{safe} \equiv (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \)
Example

\[
\begin{align*}
\text{safe} \land \text{far} & \quad \rightarrow \quad [\text{entry}] (\text{safe} \land \text{tangential}) \\
\text{safe} \land \text{tangential} & \quad \rightarrow \quad [\text{other subsystem}] \text{safe} \\
\text{where safe} & \quad \equiv \quad (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2
\end{align*}
\]
Example

\[
\begin{align*}
\text{safe} \land \text{far} & \quad \rightarrow \quad [\text{entry}](\text{safe} \land \text{tangential}) \\
\text{safe} \land \text{tangential} & \quad \rightarrow \quad [\text{other subsystem}]\text{safe} \\
\text{where safe} & \equiv (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2
\end{align*}
\]
### Definition (dL Formula $\phi$)

\[
\theta_1 \geq \theta_2 \mid \neg \phi \mid \phi \land \psi \mid \phi \lor \psi \mid \phi \rightarrow \psi \mid \forall x \phi \mid \exists x \phi \mid [\alpha]\phi
\]

with terms $\theta_1, \theta_2$ of nonlinear real arithmetic ($+, \cdot$)

### Definition (Hybrid program $\alpha$)

- $x' = f(x) \land H$ (continuous evolution)
- $x := f(x)$ (discrete jump)
- $?H$ (conditional execution)
- $\alpha; \beta$ (seq. composition)
- $\alpha \cup \beta$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)

}\{ jump & test

}\{ Kleene algebra


Outline

1 Motivation

2 Compositional Verification Logic dL

3 Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4 Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5 Case Studies & Experimental Results

6 Conclusions & Future Work
Verification by Symbolic Decomposition

\[ [\alpha] G \land [\beta] G \]

\[ [\alpha \cup \beta] G \]

\[ \alpha \cup \beta \]

\[ \beta \]

\[ v \]

\[ w_1 \]

\[ w_2 \]

\[ G \]

\[ f(x) \]

\[ x := f(x) \]

\[ x := f(x) \]
Verification by Symbolic Decomposition

\[ [\alpha]G \land [\beta]G \]
\[ [\alpha \cup \beta]G \]
\[ [\alpha; \beta]G \]

\[ [\alpha][\beta]G \]
\[ [\alpha; \beta]G \]
Verification by Symbolic Decomposition

\[ [\alpha] G \land [\beta] G \]

\[ [\alpha \cup \beta] G \]

\[ \alpha \cup \beta \]

\[ [\alpha][\beta] G \]

\[ [\alpha; \beta] G \]

\[ \alpha; \beta \]

\[ G^f_x \]

\[ x := f(x) \]

\[ [x := f(x)] G \]

\[ \alpha \]

\[ \beta \]
Verification by Discrete and Differential Induction

Definition (Discrete Invariant $F$)

$$
\forall_{\text{cl}}(F \rightarrow G) \\
\forall_{\text{cl}}(F \rightarrow [\alpha]F)
$$

Definition (Differential Invariant $F$)

$$
\forall_{\text{cl}}(F \rightarrow G) \\
\forall_{\text{cl}}(\nabla_x x' = f(x) \rightarrow F)
$$
Definition (Discrete Invariant $F$)

\[
\begin{align*}
F & \\
\forall_{cl}(F \rightarrow G) & \\
\forall_{cl}(F \rightarrow [\alpha]F) & \\
\end{align*}
\]

\[
\begin{align*}
[\alpha^*]G & \\
\end{align*}
\]

Definition (Differential Invariant $F$)

\[
\begin{align*}
\nabla_{x'} = f(x) F & \\
\forall_{cl}(\nabla_{x'} = f(x) F) & \\
\forall_{cl}(\nabla_{x'} = f(x) F) & \\
\end{align*}
\]

\[
\begin{align*}
[x' = f(x)]G & \\
\end{align*}
\]
Verification by Discrete and Differential Induction

\[ \nabla x' = f_1(x) \land \ldots \land x'_n = f_n(x) \quad \text{F is} \quad \bigwedge_{(b \geq c) \in F} \left( \sum_{i=1}^{n} \frac{\partial b}{\partial x_i} f_i(x) \geq \sum_{i=1}^{n} \frac{\partial c}{\partial x_i} f_i(x) \right) \]

Definition (Differential Invariant F)

\[ F \quad \forall_{cl}(F \rightarrow G) \quad \forall_{cl}(\nabla x' = f(x) F) \]

\[ [x' = f(x)] G \]

\[ \nabla x' = f(x) \quad f \quad x' = f(x) \]

\[ F \quad \neg F \quad F \neg F \]
\[ x'_1 = d_1 \land d'_1 = -\omega d_2 \land x'_2 = d_2 \land d'_2 = \omega d_1 \ldots \] 
\[ (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \]
Differential Induction for Roundabout Mode

\[
\frac{\partial\|x-y\|^2}{\partial x_1} x_1' + \frac{\partial\|x-y\|^2}{\partial y_1} y_1' + \frac{\partial\|x-y\|^2}{\partial x_2} x_2' + \frac{\partial\|x-y\|^2}{\partial y_2} y_2' \geq \frac{\partial p^2}{\partial x_1} x_1' \ldots
\]

\[x_1' = d_1 \wedge d_1' = -\omega d_2 \wedge x_2' = d_2 \wedge d_2' = \omega d_1 \ldots \] \((x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2\)
Differential Induction for Roundabout Mode

\[ \frac{\partial \|x-y\|^2}{\partial x_1} x'_1 + \frac{\partial \|x-y\|^2}{\partial y_1} y'_1 + \frac{\partial \|x-y\|^2}{\partial x_2} x'_2 + \frac{\partial \|x-y\|^2}{\partial y_2} y'_2 \geq \frac{\partial p^2}{\partial x_1} x'_1 \ldots \]

\[ [x'_1 = d_1 \land d'_1 = -\omega d_2 \land x'_2 = d_2 \land d'_2 = \omega d_1 \ldots] (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \]
Differential Induction for Roundabout Mode

\[ \frac{\partial \|x-y\|^2}{\partial x_1} d_1 + \frac{\partial \|x-y\|^2}{\partial y_1} e_1 + \frac{\partial \|x-y\|^2}{\partial x_2} d_2 + \frac{\partial \|x-y\|^2}{\partial y_2} e_2 \geq \frac{\partial p^2}{\partial x_1} d_1 \ldots \]

\[ x'_1 = d_1 \land d'_1 = -\omega d_2 \land x'_2 = d_2 \land d'_2 = \omega d_1 \ldots (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \]
\[ 2(x_1 - y_1)(d_1 - e_1) + 2(x_2 - y_2)(d_2 - e_2) \geq 0 \]

\[ \frac{\partial \|x-y\|^2}{\partial x_1} d_1 + \frac{\partial \|x-y\|^2}{\partial y_1} e_1 + \frac{\partial \|x-y\|^2}{\partial x_2} d_2 + \frac{\partial \|x-y\|^2}{\partial y_2} e_2 \geq \frac{\partial p^2}{\partial x_1} d_1 \ldots \]

\[ [x'_1 = d_1 \land d'_1 = -\omega d_2 \land x'_2 = d_2 \land d'_2 = \omega d_1 \ldots ](x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \]
Differential Induction for Roundabout Mode

\[2(x_1 - y_1)(d_1 - e_1) + 2(x_2 - y_2)(d_2 - e_2) \geq 0\]

\[\frac{\partial \|x-y\|^2}{\partial x_1} d_1 + \frac{\partial \|x-y\|^2}{\partial y_1} e_1 + \frac{\partial \|x-y\|^2}{\partial x_2} d_2 + \frac{\partial \|x-y\|^2}{\partial y_2} e_2 \geq \frac{\partial p^2}{\partial x_1} d_1 \ldots\]

\[\begin{aligned}
x_1' &= d_1 \land d_1' = -\omega d_2 \land x_2' &= d_2 \land d_2' = \omega d_1 \ldots
\end{aligned}\]

\[(x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2\]

\[\begin{aligned}
d_1' &= -\omega d_2 \land e_1' &= -\omega e_2 \land x_2' &= d_2 \land d_2' = \omega d_1 \ldots
\end{aligned}\]

\[d_1 - e_1 = -\omega(x_2 - y_2)\]
\[
2(x_1 - y_1)(d_1 - e_1) + 2(x_2 - y_2)(d_2 - e_2) \geq 0 \\
\frac{\partial}{\partial x_1} ||x-y||^2 d_1 + \frac{\partial}{\partial y_1} ||x-y||^2 e_1 + \frac{\partial}{\partial x_2} ||x-y||^2 d_2 + \frac{\partial}{\partial y_2} ||x-y||^2 e_2 \geq \frac{\partial p^2}{\partial x_1} d_1 \ldots \\
(x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2 \\
\]

**Proposition (Differential saturation)**

\[
F \text{ differential invariant of } [x' = \theta \land H]G, \text{ then} [x' = \theta \land H]G \iff [x' = \theta \land H \land F]G \\
\]

\[
[d_1' = -\omega d_2 \land e_1' = -\omega e_2 \land x_2' = d_2 \land d_2' = \omega d_1 \ldots]d_1 - e_1 = -\omega(x_2 - y_2) \\
\]
Differential Induction for Roundabout Mode

\[
\begin{align*}
2(x_1 - y_1)(-\omega(x_2 - y_2)) + 2(x_2 - y_2)\omega(x_1 - y_1) & \geq 0 \\
2(x_1 - y_1)(d_1 - e_1) + 2(x_2 - y_2)(d_2 - e_2) & \geq 0 \\
\frac{\partial}{\partial x_1} ||x-y||^2 d_1 + \frac{\partial}{\partial y_1} ||x-y||^2 e_1 + \frac{\partial}{\partial x_2} ||x-y||^2 d_2 + \frac{\partial}{\partial y_2} ||x-y||^2 e_2 & \geq \frac{\partial p^2}{\partial x_1} d_1 \ldots \\
[x_1' = d_1 \land d_1' = -\omega d_2 \land x_2' = d_2 \land d_2' = \omega d_1 \ldots] & (x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2
\end{align*}
\]

Proposition (Differential saturation)

\[
F \text{ differential invariant of } [x' = \theta \land H]G, \text{ then } [x' = \theta \land H]G \iff [x' = \theta \land H \land F]G
\]

\[
[d_1' = -\omega d_2 \land e_1' = -\omega e_2 \land x_2' = d_2 \land d_2' = \omega d_1 \ldots]d_1 - e_1 = -\omega(x_2 - y_2)
\]
Differential Invariants as Fixedpoints

$A \rightarrow [\alpha]G$

[Clarke’79]
Differential Invariants as Fixedpoints

\[
A_1 \rightarrow [\alpha_1]G_1 \\
A_2 \rightarrow [\alpha_2]G_2 \\
A \rightarrow [\alpha]G
\]

for \( \cup, ;, := \) do decompose
Differential Invariants as Fixedpoints

\[ A \rightarrow [\alpha]G \]

\[ A_1 \rightarrow [\alpha_1]G_1 \]
\[ A_2 \rightarrow [\alpha_2]G_2 \]
\[ A_3 \rightarrow [\alpha_3]G_3 \]
\[ A_4 \rightarrow [\alpha_4]G_4 \]

for \( \cup, ;, := \) do decompose
Differential Invariants as Fixedpoints

\[ A \rightarrow [\alpha]G \]

\[ A_1 \rightarrow [\alpha_1]G_1 \]

\[ A_2 \rightarrow [\alpha_2]G_2 \]

\[ A_3 \rightarrow [\alpha_3]G_3 \]

\[ A_4 \rightarrow [\alpha_4]G_4 \]

\[ \text{for } \cup, ;, := \text{ do decompose} \]

\[ \text{for } x' = f(x) \text{ do diffsat} \]
Differential Invariants as Fixedpoints

\[ A \rightarrow [\alpha] G \]

\[ A_1 \rightarrow [\alpha_1] G_1 \]
\[ A_2 \rightarrow [\alpha_2] G_2 \]
\[ A_3 \rightarrow [\alpha_3] G_3 \]
\[ A_4 \rightarrow [\alpha_4] G_4 \]

\[ \text{diffsat} \]

for \( \cup, ;, := \), do decompose
for \( x' = f(x) \), do diffsat
Differential Invariants as Fixedpoints

\[ A_1 \rightarrow [\alpha_1]G_1 \]
\[ A_2 \rightarrow [\alpha_2]G_2 \]
\[ A_3 \rightarrow [\alpha_3]G_3 \]
\[ A_4 \rightarrow [\alpha_4]G_4 \]

\[ A \rightarrow [\alpha]G \]

for \( \cup, ;, := \) do decompose
for \( x' = f(x) \) do diffsat
for \( \alpha^* \) do loopsat

Andrè Platzer, Edmund M. Clarke (CMU)
Scaling Symbolic Verification of Hybrid Systems
Berkeley 9 / 14
Differential Invariants as Fixedpoints

\[ A \rightarrow [\alpha]G \]

\[ A_1 \rightarrow [\alpha_1]G_1 \]
\[ A_2 \rightarrow [\alpha_2]G_2 \]
\[ A_3 \rightarrow [\alpha_3]G_3 \]
\[ A_4 \rightarrow [\alpha_4]G_4 \]

for \( \cup, ;, := \) do decompose
for \( x' = f(x) \) do diffsat
for \( \alpha^* \) do loopsat
\[ \left\{ \begin{array}{c} \text{repeat until fixedpoint} \end{array} \right. \]
Example (a L formula of verification subgoal)
Example (dL formula of verification subgoal)

\[ \text{safe} \land \text{far} \rightarrow [\text{agree}](\text{safe} \land \text{far} \land \text{compatible}) \]
Example (dL formula of verification subgoal)

\[ \text{safe} \land \text{far} \land \text{compatible} \rightarrow [\text{entry}] (\text{safe} \land \text{tangential}) \]
Example (dL formula of verification subgoal)

\[ \text{safe} \land \text{tangential} \rightarrow [\text{circ}](\text{safe} \land \text{tangential}) \]
Example (dŁ formula of verification subgoal)

\[ \text{safe} \land \text{tangential} \rightarrow [\text{exit}] (\text{safe} \land \text{far}) \]
Fixedpoint Iterations for Air Traffic Control

Example (dL formula of verification subgoal)

\[ \text{safe} \land \text{far} \rightarrow \lbrack \text{free} \rbrack (\text{safe} \land \text{far}) \]
Outline

1 Motivation

2 Compositional Verification Logic dL

3 Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4 Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5 Case Studies & Experimental Results

6 Conclusions & Future Work
## Experimental Results

<table>
<thead>
<tr>
<th>Case Study</th>
<th>Time(s)</th>
<th>Mem(Mb)</th>
<th>Proof Steps</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roundabout (2)</td>
<td>14</td>
<td>8</td>
<td>117</td>
<td>13</td>
</tr>
<tr>
<td>Roundabout (3)</td>
<td>387</td>
<td>42</td>
<td>182</td>
<td>18</td>
</tr>
<tr>
<td>Roundabout (4)</td>
<td>730</td>
<td>39</td>
<td>234</td>
<td>23</td>
</tr>
<tr>
<td>Roundabout (5)</td>
<td>1964</td>
<td>88</td>
<td>317</td>
<td>28</td>
</tr>
<tr>
<td>bounded speed entry</td>
<td>20</td>
<td>34</td>
<td>28</td>
<td>12</td>
</tr>
<tr>
<td>flyable entry (simplif.)</td>
<td>6</td>
<td>10</td>
<td>98</td>
<td>8</td>
</tr>
<tr>
<td>ETCS-kernel</td>
<td>27</td>
<td>28</td>
<td>53</td>
<td>9</td>
</tr>
<tr>
<td>ETCS-safety</td>
<td>183</td>
<td>87</td>
<td>169</td>
<td>15</td>
</tr>
<tr>
<td>ETCS binary</td>
<td>56</td>
<td>27</td>
<td>147</td>
<td>15</td>
</tr>
<tr>
<td>ETCS controllability</td>
<td>1</td>
<td>6</td>
<td>17</td>
<td>5</td>
</tr>
<tr>
<td>RBC controllability</td>
<td>1</td>
<td>7</td>
<td>45</td>
<td>16</td>
</tr>
</tbody>
</table>
Outline

1 Motivation

2 Compositional Verification Logic dL

3 Decompositional Inductive Verification of Hybrid Systems
   - Verification by Symbolic Decomposition
   - Discrete Induction
   - Differential Induction

4 Computing Differential Invariants by Combining Local Fixedpoints
   - Local Fixedpoints & Differential Saturation
   - Global Fixedpoints & Interplay

5 Case Studies & Experimental Results

6 Conclusions & Future Work
Conclusions

Verifying hybrid systems with challenging dynamics:

- Verification by decomposition: differential dynamic logic $dL$
- Differential invariants instead of reachability along solutions
- Computing differential invariants as fixedpoints
- Differential saturation procedure
- Exploit locality in system designs
- Verify challenging aircraft control
- Sound “by construction”
Future Work

- Compare differential invariants with classical state reachability?
  - Particularly good for hybrid systems with parameterized dynamics
  - Single initial state ⇒ simulation more appropriate

- Case studies
  - Successful for roundabout and train control
  - Performance for other case studies? STARMAC platform?

- Probabilistic model classes