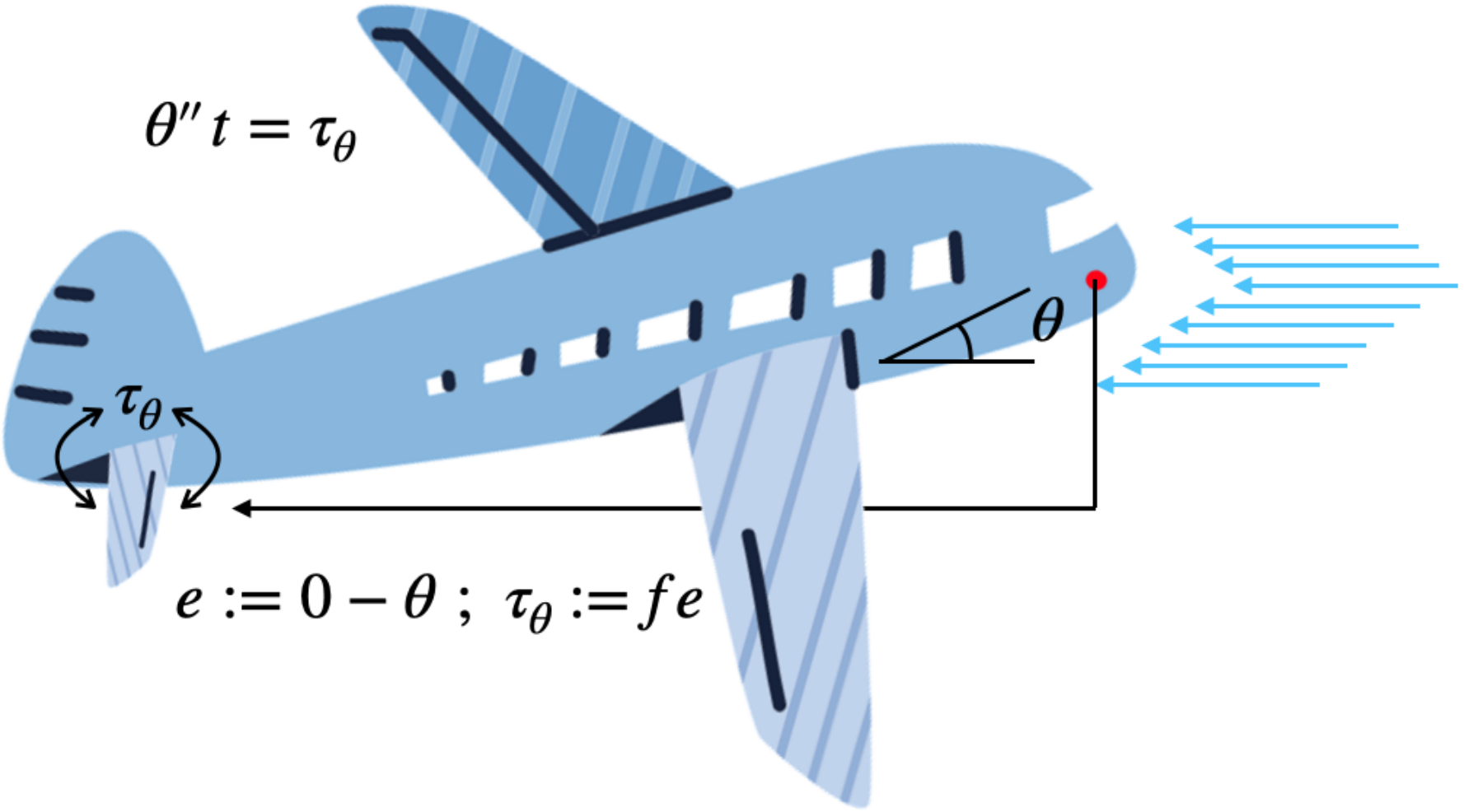
Algebraic Verification of Hybrid Systems in Isabelle/HOL

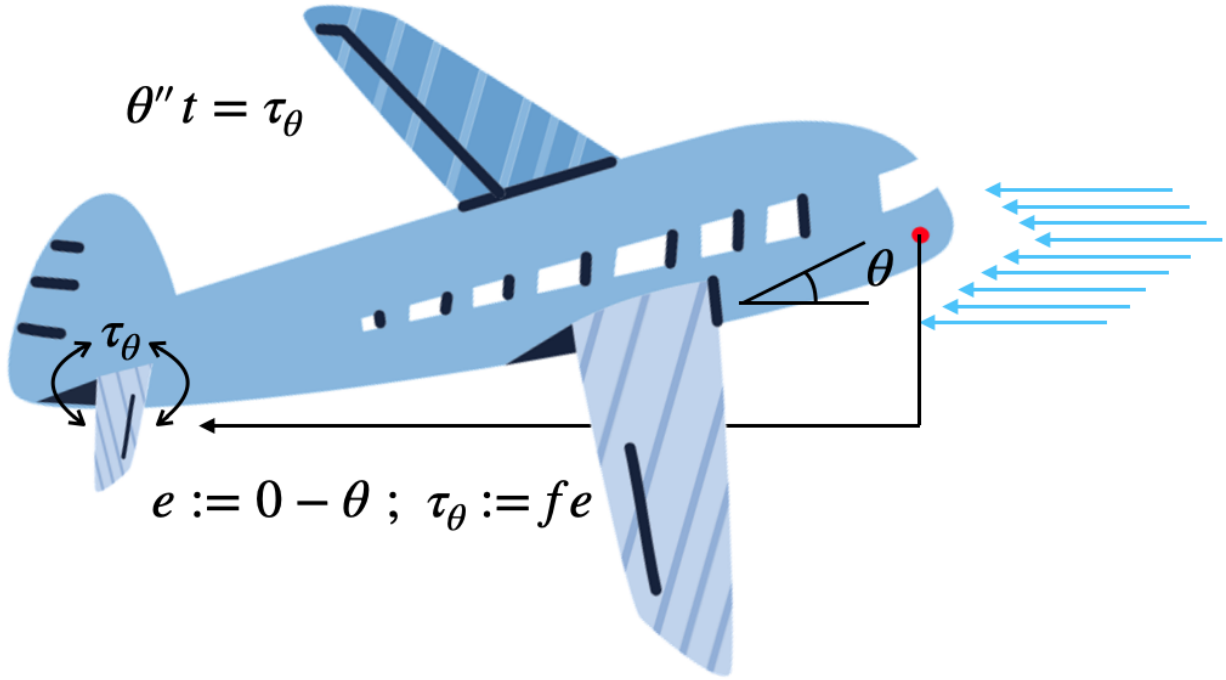
Jonathan Julián Huerta y Munive

University of Sheffield, UK

Hybrid Systems

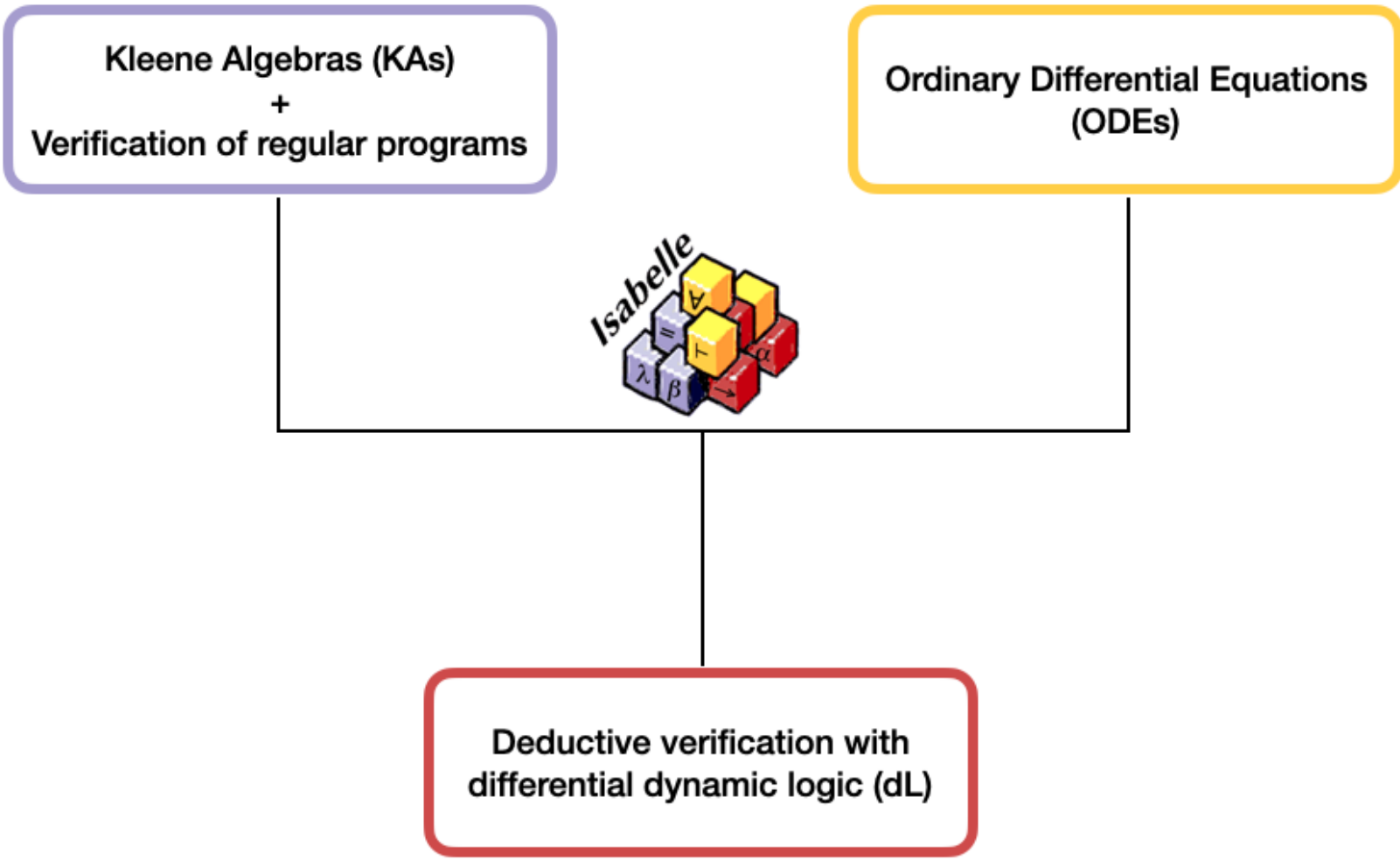


Verification of Hybrid Systems



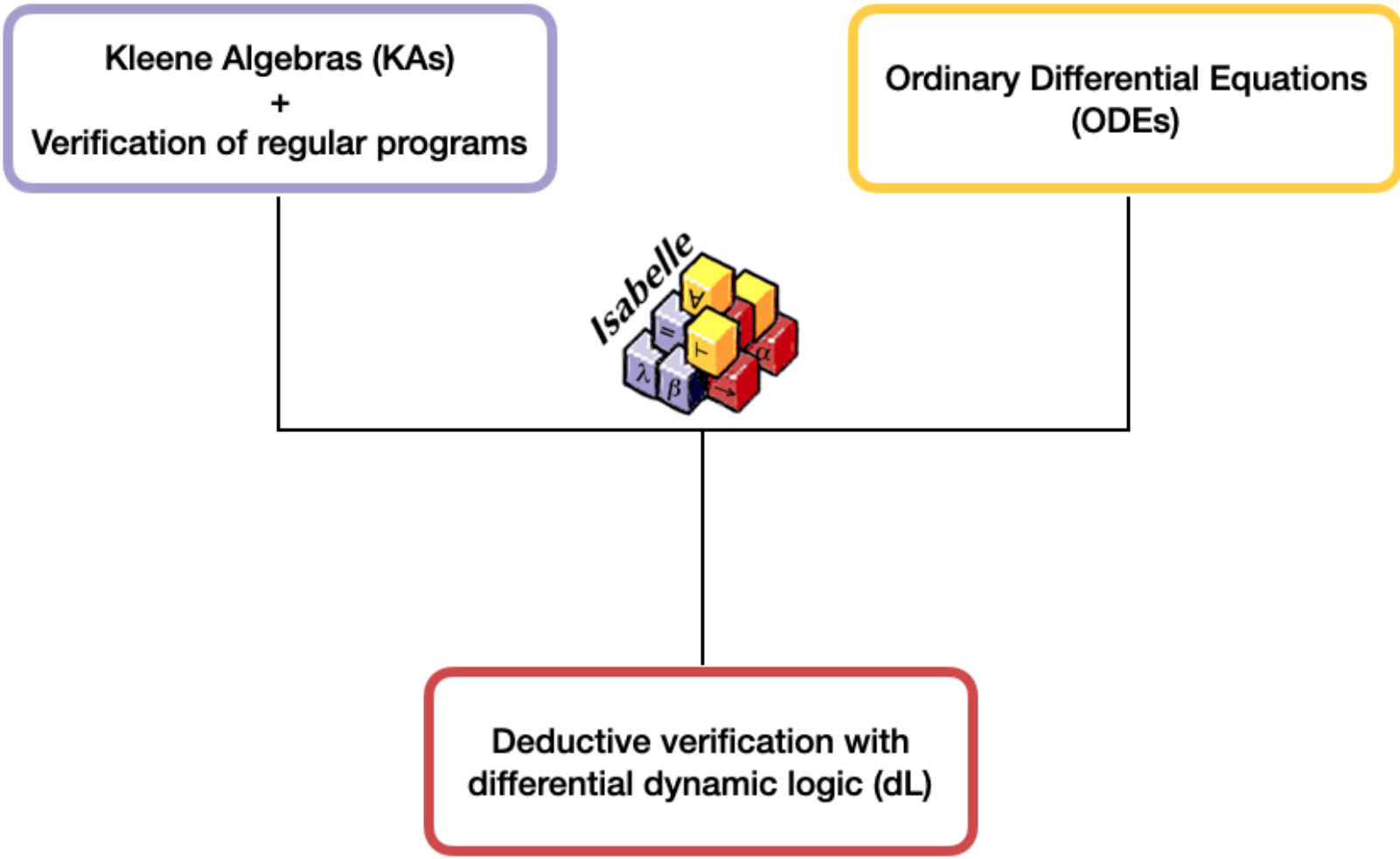
$$
\begin{aligned}
\text{dynamics} &= \theta'' \, t = \tau_\theta \\
\text{pre} &= \theta_m \leq \theta \, t \leq \theta_M \\
\text{pos} &= \theta_m \leq \theta \, t \leq \theta_M \\
\text{control} &= e := -\theta \, ; \tau_\theta := f \, e \\
\text{aircraft} &= (\text{control} \, ; \text{dynamics})^* \qquad \text{hybrid program} \\
\{\text{pre}\} \; \{\text{aircraft}\} \; \{\text{pos}\} & \qquad \text{correctness spec}
\end{aligned}
$$

Context



$$\{p\}\, x\, \{q\} \leftrightarrow p \rightarrow |x]\, q$$

Research Question



Can we integrate KAs and ODEs for d$\mathcal{L}$-style verification of HSs?

Main Contribution

Semantic framework:

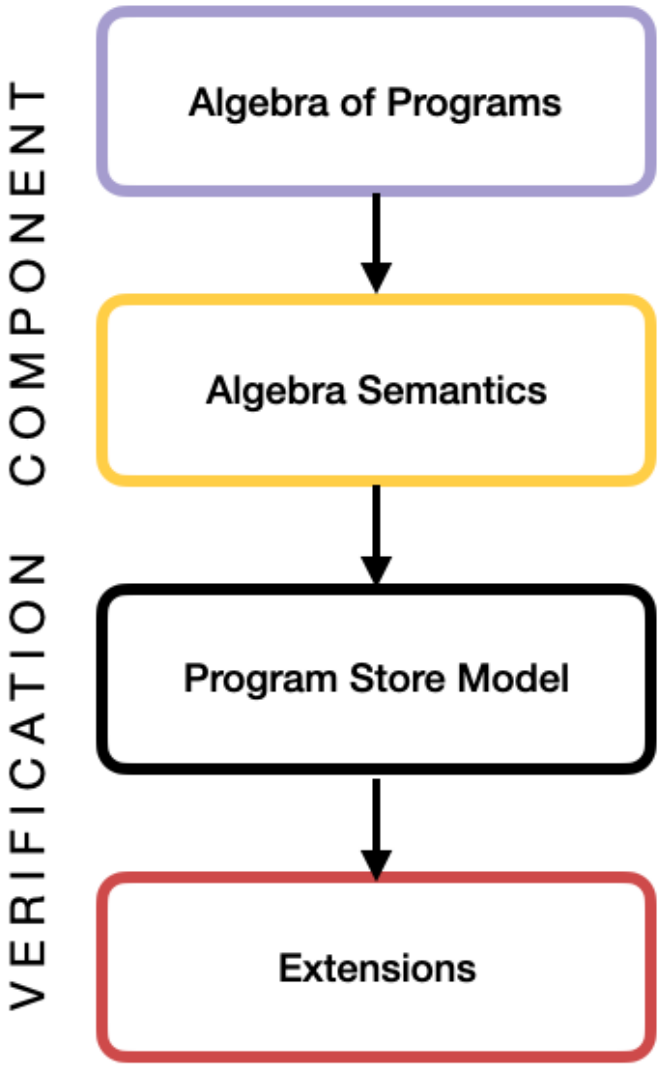
- Modular and open
- For (various syles of) hybrid program verification
- With a general-purpose proof assistant
- Formalised in Isabelle/HOL

Technical Contributions

1. Mathematical
   - 1.1 Hybrid weakest liberal precondition calculus
   - 1.2 Minimalistic calculi for verification of hybrid programs $\mathrm{d}\mathcal{H}$
   - 1.3 Related calculi for construction of hybrid programs $\mathrm{d}\mathcal{R}$
   - 1.4 Derivation and soundness of the inference rules in the thesis
   - 1.5 Generalisation of orbits and invariants
2. Formalisations
   - 2.1 Formalisation of state transformer model of KAs and variants
   - 2.2 Reformulation of Picard-Lindelöf's Theorem
   - 2.3 Tactic for automatic certification of derivatives
   - 2.4 Formalisation of results about linear systems of ODEs
3. Various verification components in Isabelle/HOL
4. Formal verifications of several hybrid systems

https://github.com/yonoteam/CPSVerification

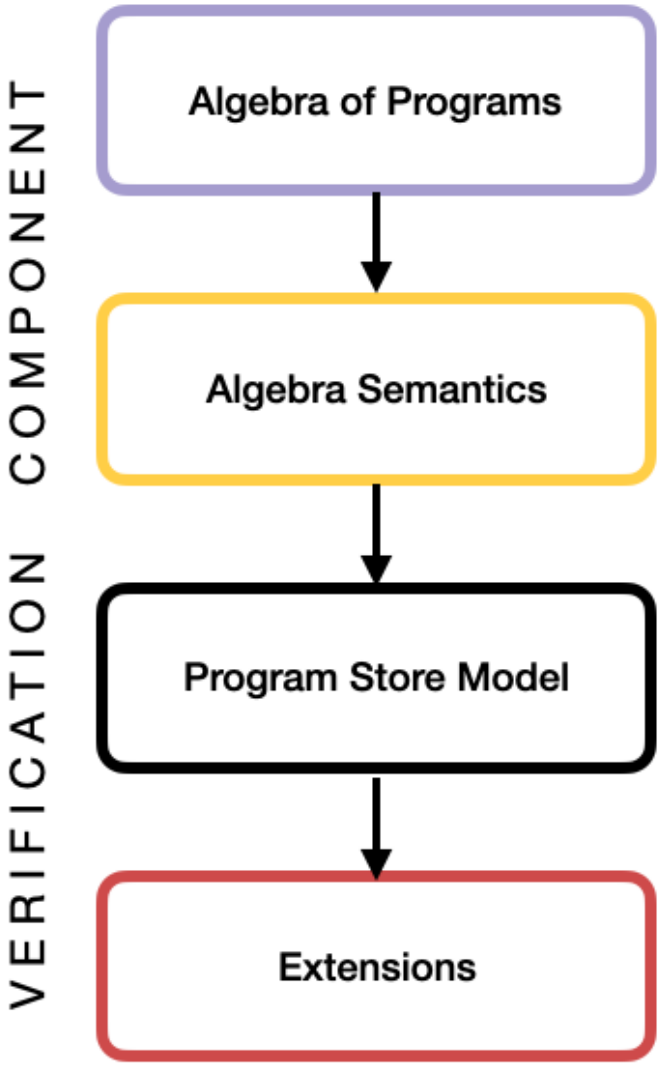
Seed work



- Modal Kleene Algebra
  $(K, +, \cdot, 0, 1, ^*, ad, ar)$
  if p **then** x **else** $y = p \cdot x + (ad\ p) \cdot y$
- Relational $R \subseteq S \times S$
  $(x := e) = \{(s, s[x \mapsto e\ s]) \mid s \in S\}$
  $(x' = f\ \&\ G) = \{(s, \varphi\ t\ s) \mid s \in S\}$
  where φ solves the ODE
- Program states are functions
  $s : string \to \mathbb{R}$
- Rules of $\mathrm{d}\mathcal{L}$
- Hoare and $\mathrm{d}\mathcal{L}$-style verification

proof of concept for integration

Modularity work



- Modal Kleene Algebra
- Predicate transformers $|-] : S^{\mathbb{B}} \to S^{\mathbb{B}}$
- Relational $R \subseteq S \times S$
- State transformer $f : S \to \mathcal{P}\, S$
- Program states are tuples $s \in \mathbb{R}^n$
- Annotate ODEs $x' = f \,\&\, G$, certify flows φ
- Semantic invariant deduction I'
- Annotate flows φ

More variations and extensions

Using (refinement) Kleene algebra with tests provides

- minimalistic differential Hoare logic $\mathsf{d}\mathcal{H}$
- (Morgan-style) differential refinement calculus $\mathsf{d}\mathcal{R}$

Extending linear algebra and ODEs in Isabelle/HOL provides

- matrix support
- increased proof automation
- removes certification procedures

Stand-alone version in Isabelle/HOL

- does not have algebras
- easy to set up

components correct by construction

Formal Verifications and Future Work

Friendly competition of the 7th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH2020)

- Solved 53 of 61 attempted benchmark problems
- Discovered that Isabelle/HOL needs more proof-support to reason about expressions with real-arithmetic
- Need more methods to prove invariant properties about ODEs
- Benefit from integration with computer algebra system

Case studies

- Formalised invariants about PID-controller of quadcopter flight
- Analysing an autonomous marine vehicle with the University of York

Papers and Isabelle Components

- Huerta y Munive and Struth, "Verifying Hybrid Systems with Modal Kleene Algebra". RAMiCS 2018
- Huerta y Munive and Struth, "Predicate Transformer Semantics for Hybrid Systems: Verification Components for Isabelle/HOL", ArXiV 2019
- Foster, Huerta y Munive, and Struth, "Differential Hoare Logics and Refinement Calculi for Hybrid Systems with Isabelle/HOL", RAMiCS 2020
- Huerta y Munive, "Affine Systems of ODEs in Isabelle/HOL for Hybrid-Program Verification", SEFM 2020
- Mitsch et al., "ARCH-COMP20 Category Report: Hybrid Systems Theorem Proving", ARCH 2020
- Huerta y Munive, "Verification Components for Hybrid Systems", Archive of Formal Proofs, 2019
- Huerta y Munive, "Matrices for ODEs", Archive of Formal Proofs, 2020

Conclusions

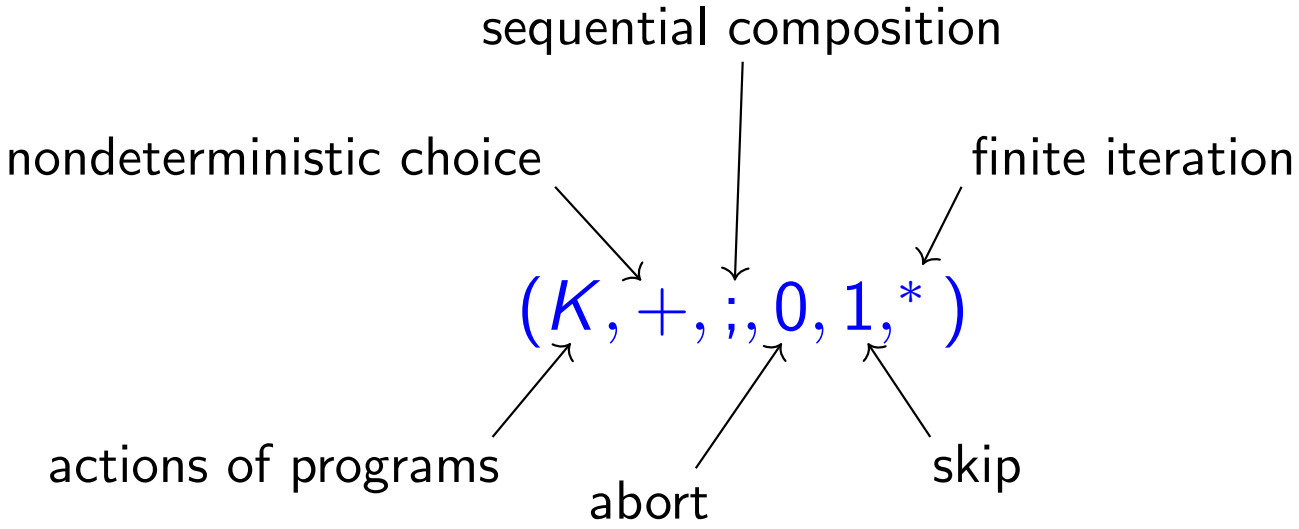
Expressive, modular and open semantic framework for deductively verifying hybrid systems in a general-purpose proof assistant with support for various verification styles

Implemented in Isabelle/HOL

END

Kleene Algebras

Kleene Algebra



Programs and partial correctness specifications

- $(B, +, ;, 0, 1, \neg)$ is a boolean algebra
- use $\alpha, \beta \in K$ and $p, q \in B$ where $B \subseteq K$
- **if** p **then** α **else** $\beta = p\,;\alpha + \neg p\,;\beta$
- **while** p **do** $\alpha = (p\,;\alpha)^*\,;\neg p$
- $\{p\}\,\alpha\,\{q\} \leftrightarrow p\,;\alpha \leq \beta\,;q$

State Transformer Model

Programs are functions $S \to \mathcal{P}\, S$:

$$(\alpha + \beta)\, s = \alpha\, s \cup \beta\, s$$

$$(\alpha\,;\beta)\, s = (\alpha \circ_K \beta)\, s = \bigcup \{\beta\, s' \mid s' \in \alpha\, s\}$$

$$0\, s = \emptyset$$

$$1\, s = \{s\}$$

$$\alpha^*\, s = \bigcup_{n \geq 0} \alpha^n\, s \text{ where } \alpha^0\, s = 1\, s \text{ and } \alpha^{n+1} = \alpha^n \circ_K \alpha$$

$$(\neg p)\, s = \begin{cases} \{s\}, & \text{if } p\, s = \emptyset \\ \emptyset, & \text{otherwise} \end{cases}$$

$$\{p\}\, \alpha\, \{q\} \leftrightarrow (\forall s_1.\ p\, s_1 \to (\forall s_2.\ s_2 \in \alpha\, s_1 \to q\, s_2))$$

What about Assignments?

Lenses

- Variables are lenses $x = (A, S, get_x, put_x)$ denoted $x : A \Longrightarrow S$ where

$$get_x : S \to A \text{ and } put_x : S \to A \to S$$

- A is a variable type while S is the source
- They satisfy the axioms

$$get_x\ (put_x\ s\ v) = v$$
$$put_x\ (put_x\ s\ u)\ v = put_x\ s\ v$$
$$put_x\ s\ (get_x\ s) = s$$

Semantics $S \to \mathcal{P}\, S$ for assignments is

$$(x := e)\, s = \{put_x\, s\, (e\, s)\}$$

Verification Rules

- Traditional Hoare logic:

$$
\begin{aligned}
p_1 \leq p_2 \wedge \{p_2\}\,\alpha\,\{q_2\} \wedge q_2 \leq q_1 &\to \{p_1\}\,\alpha\,\{q_1\} \\
\{p\}\,\alpha\,\{r\} \wedge \{r\}\,\beta\,\{q\} &\to \{p\}\,\alpha\,;\beta\,\{q\} \\
\{r\,;p\}\,\alpha\,\{q\} \wedge \{\neg r\,;p\}\,\beta\,\{q\} &\to \{p\}\,\textbf{if } r \textbf{ then } \alpha \textbf{ else } \beta\,\{q\} \\
\{r\,;p\}\,\alpha\,\{p\} &\to \{p\}\,\textbf{while } r \textbf{ do } \alpha\,\{\neg r\,;p\} \\
&\to \{\lambda s.\ q\,(put_x\,s\,(e\,s))\}\,x := e\,\{q\}
\end{aligned}
$$

- Extended with d$\mathcal{L}$'s hybrid programs

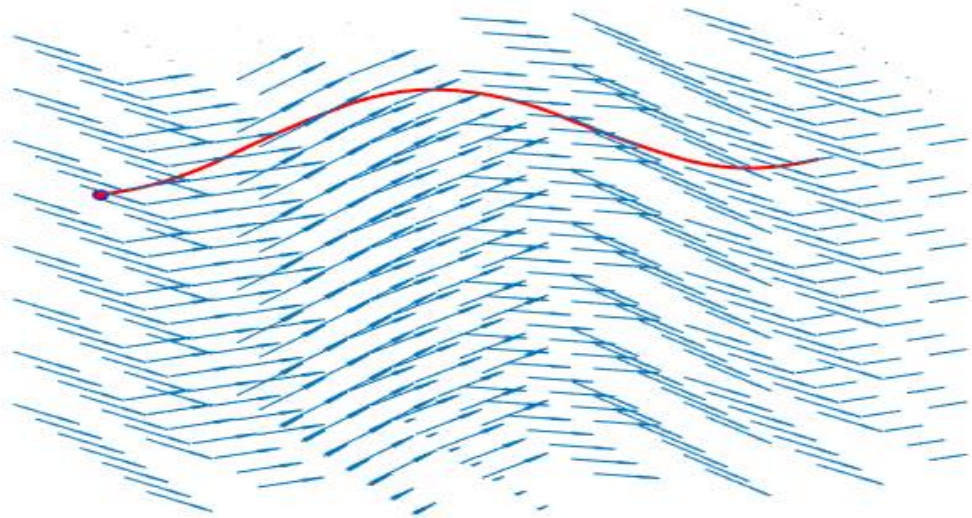
$$
\begin{aligned}
&\to \{p\}\,\textbf{skip}\,\{p\} \\
&\to \{p\}\,\textbf{abort}\,\{q\} \\
\{p\}\,\alpha\,\{q\} \wedge \{p\}\,\beta\,\{q\} &\to \{p\}\,\alpha + \beta\,\{q\} \\
\{p\}\,\alpha\,\{p\} &\to \{p\}\,\textbf{loop } \alpha\,\{p\}
\end{aligned}
$$

where **loop** $\alpha = \alpha^*$, **skip** $= 1$, and **abort** $= 0$

What about ODEs?

Vector Field

$$X'\ t = f\ t\,(X\ t)$$



where

$$X : T \subseteq \mathbb{R} \to S \qquad f : T \to S \to S \qquad X\,0 = s$$

$$\text{orbit} : s \mapsto \{X\ t \mid t \in T\}$$

Semantics for ODEs

- Solutions to initial value problems (IVPs)

$$\mathsf{Sols}\, f\ T\ s = \{X : T \to S \mid (\forall t \in T.\ X'\, t = f\, t\, (X\, t) \wedge X\, 0 = s\}$$

- Guarded orbit

$$\mathsf{orbit}_G^X\, s = \{X\, t \mid t \in T \wedge (\forall \tau \in [0, t].\ G\, (X\, \tau))\}$$

- Semantics $S \to \mathcal{P}\, S$ for assignments are

$$(x' = f\ \&\ G)\, s = \bigcup \{\mathsf{orbit}_G^X\, s \mid X \in \mathsf{Sols}\, f\ T\ s\}$$

- The corresponding rule of inference is

$$\{\lambda s.\ \forall t \in T.\ (\forall \tau \in [0, t].\ G\, (X\, t)) \to Q\, (X\, t)\}\ (x'{=}f\ \&\ G)\ \{Q\}$$

easy to obtain if there is a unique solution $X : T \to S$ to the IVPs associated to each s and the vector field f (Picard-Lindelöf's Theorem)

Invariants in d$\mathcal{H}$

- I is an invariant for f iff $\{I\}\, x'{=}f \,\&\, G\, \{I\}$, or equivalently

$$\bigcup(\mathcal{P}\,(x'{=}f \,\&\, G)\, I) \subseteq I$$

- We obtain the following rules

$$\begin{aligned}
p \leq i \land \{i\}\,\alpha\,\{i\} \land i \leq q &\to \{p\}\,\alpha\ \textbf{inv}\ i\,\{q\}\\
\{i\}\,\alpha\,\{i\} \land \{j\}\,\alpha\,\{j\} &\to \{i\,;j\}\,\alpha\,\{i\,;j\}\\
\{i\}\,\alpha\,\{i\} \land \{j\}\,\alpha\,\{j\} &\to \{i+j\}\,\alpha\,\{i+j\}\\
p \leq i \land \{i\,;t\}\,\alpha\,\{i\} \land \neg r\,;i \leq q &\to \{p\}\,\textbf{while}\ r\ \textbf{do}\ \alpha\ \textbf{inv}\ i\,\{q\}\\
p \leq i \land \{i\}\,\alpha\,\{i\} \land i \leq q &\to \{p\}\,\textbf{loop}\ \alpha\ \textbf{inv}\ i\,\{q\}\\
p \leq i \land i \text{ is inv. for } f \land (G\,;i) \leq q &\to \{p\}\,x' = f \,\&\, G\ \textbf{inv}\ i\,\{q\}
\end{aligned}$$

where operationally $\alpha\ \textbf{inv}\ i = \alpha$

Differential Refinement Calculus $\mathrm{d}\mathcal{R}$

Extend KAT with refinement operation $[-,-] : B \times B \to K$ such that

$$\{p\}\,\alpha\,\{q\} \leftrightarrow \alpha \leq [p,q]$$

Obtain traditional Morgan Style Refinement laws:

$$\begin{aligned}
\textbf{skip} &\leq [p,p] \\
\textbf{abort} &\leq [p,q], \\
[p',q'] &\leq [p,q] \qquad \text{if } p \leq p' \text{ and } q' \leq q \\
[p,r]\,;[r,q] &\leq [p,q] \\
[p,q] + [p,q] &\leq [p,q] \\
\textbf{if } t \textbf{ then } [t\,;p,q] \textbf{ else } [\neg t\,;p,q] &\leq [p,q] \\
\textbf{while } t \textbf{ do } [t\,;p,p] &\leq [p,\neg t\,;p] \\
\textbf{loop } [p,p] &\leq [p,p]
\end{aligned}$$

More Refinement Laws

- Laws for assignments $(x := e) \leq [\lambda s.\ Q\,(put_x\ s\,(e\,s)), Q]$
- Laws for evolution commands where $X'\,t = f\,(X\,t)$ and $X\,0 = s$

$$(x' = f\ \&\ G) \leq [\lambda s \in S. \forall t \in T.\ (\forall \tau \in [0, t].\ G\,(X\,\tau)) \rightarrow Q\,(X\,t), Q]$$

- Monotonoic laws and laws with invariants

$$\begin{array}{rll}
\textbf{if } t \textbf{ then } \alpha_1 \textbf{ else } \beta_1 \leq \textbf{if } t \textbf{ then } \alpha_2 \textbf{ else } \beta_2 & \text{if } \alpha_1 \leq \alpha_2 \text{ and } \beta_1 \leq \beta_2 \\
\textbf{while } t \textbf{ do } \alpha_1 \leq \textbf{while } t \textbf{ do } \alpha_2 & \text{if } \alpha_1 \leq \alpha_2 \\
\textbf{loop } \alpha_1 \leq \textbf{loop } \alpha_2 & \text{if } \alpha_1 \leq \alpha_2 \\
\textbf{while } t \textbf{ do } \alpha \textbf{ inv } i \leq [p, q] & \text{if } p \leq i\ ;\ t \text{ and } \alpha \leq [i, i] \text{ and } \neg t\ ;\ i \leq q \\
\textbf{loop } \alpha \textbf{ inv } i \leq [p, q] & \text{if } p \leq i \text{ and } \alpha \leq [i, i] \text{ and } i \leq q
\end{array}$$

Refinement of the Thermostat

abbreviation *dyn* T_m T_M *a* T_u $\tau \equiv$
IF $(\vartheta = 0)$ *THEN* $x' = f\ a\ 0\ \&\ G\ T_m\ T_M\ a\ 0$ *on* $\{0..\tau\}$ *UNIV @ 0*
ELSE $x' = f\ a\ T_u\ \&\ G\ T_m\ T_M\ a\ T_u$ *on* $\{0..\tau\}$ *UNIV @ 0*

lemma *R-therm-down*:
assumes $a > 0$ **and** $0 \leq \tau$ **and** $0 < T_m$ **and** $T_M < T_u$
shows $[\vartheta = 0 \wedge I\ T_m\ T_M \wedge t = 0 \wedge T_0 = T, I\ T_m\ T_M] \geq$
$(x' = f\ a\ 0\ \&\ G\ T_m\ T_M\ a\ 0$ *on* $\{0..\tau\}$ *UNIV @ 0*$)$
apply(*rule local-flow.R-g-ode-ivl*[*OF local-flow-therm*])
using *therm-dyn-down*[*OF assms*(1,3), *of* - T_M] *assms* **by** *rel-auto'*

lemma *R-therm-up*:
assumes $a > 0$ **and** $0 \leq \tau$ **and** $0 < T_m$ **and** $T_M < T_u$
shows $[\neg\ \vartheta = 0 \wedge I\ T_m\ T_M \wedge t = 0 \wedge T_0 = T, I\ T_m\ T_M] \geq$
$(x' = f\ a\ T_u\ \&\ G\ T_m\ T_M\ a\ T_u$ *on* $\{0..\tau\}$ *UNIV @ 0*$)$
apply(*rule local-flow.R-g-ode-ivl*[*OF local-flow-therm*])
using *therm-dyn-up*[*OF assms*(1) - - *assms*(4), *of* T_m] *assms* **by** *rel-auto'*

Refinement of the Thermostat

abbreviation *ctrl* T_m $T_M \equiv$
$(t ::= 0)$; $(T_0 ::= T)$;
$(IF\ (\vartheta = 0 \wedge T_0 \leq T_m + 1)\ THEN\ (\vartheta ::= 1)\ ELSE$
$IF\ (\vartheta = 1 \wedge T_0 \geq T_h - 1)\ THEN\ (\vartheta ::= 0)\ ELSE\ skip)$

lemma *R-therm-time*: $[I\ T_m\ T_M,\ I\ T_m\ T_M \wedge t = 0] \geq (t ::= 0)$
by (*rule R-assign-law*, *pred-simp*)

lemma *R-therm-temp*:
$[I\ T_m\ T_M \wedge t = 0,\ I\ T_m\ T_M \wedge t = 0 \wedge T_0 = T] \geq (T_0 ::= T)$
by (*rule R-assign-law*, *pred-simp*)

lemma *R-thermostat-flow*:
assumes $a > 0$ **and** $0 \leq \tau$ **and** $0 < T_m$ **and** $T_M < T_u$
shows $[I\ T_m\ T_M,\ I\ T_m\ T_M] \geq therm\ T_m\ T_M\ a\ T_u\ \tau$
by (*refinement*;(*rule R-therm-time*)?,(*rule R-therm-temp*)?,
(*rule R-assign-law*)?, (*rule R-therm-up*[*OF assms*])?,
(*rule R-therm-down*[*OF assms*])?) *rel-auto'*

Verification Rules of $\mathrm{d}\mathcal{H}$

$$\frac{}{\{q\}\,1\,\{q\}} \qquad \frac{}{\{\lambda s.\ q\,(put_x\,s\,(e\,s))\}\,x := e\,\{q\}} \qquad \frac{\{p\}\,\alpha\,\{r\} \quad \{r\}\,\beta\,\{q\}}{\{p\}\,\alpha;\beta\,\{q\}}$$

$$\frac{\{r \wedge p\}\,\alpha\,\{q\} \quad \{\neg r \wedge p\}\,\beta\,\{q\}}{\{p\}\,\textbf{if}\ r\ \textbf{then}\ \alpha\ \textbf{else}\ \beta\,\{q\}} \qquad \frac{\{p \wedge r\}\,\alpha\,\{q\}}{\{p\}\,\textbf{while}\ r\ \textbf{do}\ \alpha\,\{\neg r \wedge q\}}$$

$$\frac{p \to r_1 \quad \{r_1\}\,\alpha\,\{r_2\} \quad r_2 \to q}{\{p\}\,\alpha\,\{q\}}$$

$$\frac{\forall s \in S.\exists! X : T \to S.\ X\,0 = s \wedge (\forall t \in T.\ X'\,t = f\,t\,(X\,t))}{\{\lambda s.\ s \in S \to \forall t \in T.\ (\forall \tau \in [0,t].\ G\,(X\,t)) \to q\,(X\,t)\}\,x' = f\ \&\ G\,\{q\}}$$

Extended verification Rules of $\mathrm{d}\mathcal{H}$

$$\frac{}{\{q\}\,1\,\{q\}} \qquad \frac{}{\{\lambda s.\ q\,(put_x\,s\,(e\,s))\}\,x := e\,\{q\}} \qquad \frac{}{\{p\}\,0\,\{q\}}$$

$$\frac{\{p\}\,\alpha\,\{q\} \quad \{p\}\,\beta\,\{q\}}{\{p\}\,\alpha + \beta\,\{q\}} \qquad \frac{\{p\}\,\alpha\,\{r\} \quad \{r\}\,\beta\,\{q\}}{\{p\}\,\alpha;\beta\,\{q\}}$$

$$\frac{p \to r_1 \quad \{r_1\}\,\alpha\,\{r_2\} \quad r_2 \to q}{\{p\}\,\alpha\,\{q\}} \qquad \frac{\{p\}\,\alpha\,\{p\}}{\{p\}\,\alpha^*\,\{p\}}$$

$$\frac{\forall s \in S.\exists! X : T \to S.\ X\,0 = s \wedge (\forall t \in T.\ X'\,t = f\,t\,(X\,t))}{\{\lambda s.\ s \in S \to \forall t \in T.\ (\forall \tau \in [0,t].\ G\,(X\,t)) \to q\,(X\,t)\}\,x' = f\ \&\ G\,\{q\}}$$