KeYmaera X: Theorem Proving for Hybrid Systems

Nathan Fulton
Carnegie Mellon University

May 6, 2016
Milieu

Safety-critical control software is pervasive and increasingly complicated.
Safety-critical control software is pervasive and increasingly complicated.
**KeYmaera X**

**Small Core** Increases trust, enables experimentation

<table>
<thead>
<tr>
<th>System</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeYmaera X</td>
<td>1,682</td>
</tr>
<tr>
<td>Isabelle/Pure</td>
<td>8,113</td>
</tr>
<tr>
<td>Coq</td>
<td>20,000</td>
</tr>
<tr>
<td>dReal</td>
<td>50,000</td>
</tr>
<tr>
<td>SpaceEx</td>
<td>100,000</td>
</tr>
</tbody>
</table>
KeYmaera X

**Small Core** Increases trust, enables experimentation

<table>
<thead>
<tr>
<th>System</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeYmaera X</td>
<td>1682</td>
</tr>
<tr>
<td>Isabelle/Pure</td>
<td>8113</td>
</tr>
<tr>
<td>Coq</td>
<td>20000</td>
</tr>
<tr>
<td>dReal</td>
<td>50000</td>
</tr>
<tr>
<td>SpaceEx</td>
<td>100000</td>
</tr>
</tbody>
</table>

**Tactics** Maintainable and readable proof automation
KeYmaera X

**Small Core** Increases trust, enables experimentation

**System** LOC

KeYmaera X 1682

Isabelle/Pure 8113

Coq 20000

dReal 50000

SpaceEx 100000

**Tactics** Maintainable and readable proof automation

**GUI** A point-and-prove interface for interacting with deeply nested formulas

---

![Diagram of KeYmaera X user interface](image)
Assign  \( x := \theta \)

Sequence  \( \alpha; \beta \)

Iteration  \( \alpha^* \)

Choice  \( \alpha \cup \beta \)

Test  \( ?\varphi \)

ODEs  \( \{ x'_1 = \theta_1, \ldots, x'_n = \theta_n & P \} \)
A Hybrid System Specification

\[ \text{vel} \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{\text{acc} := A \cup \text{acc} := -B\}; \{\text{pos'} = \text{vel}, \text{vel'} = \text{acc} \land \text{vel} \geq 0\}]^* \text{vel} \geq 0 \]
A Hybrid System Specification

\[ vel \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{ \text{acc} := A \cup \text{acc} := -B \}; \{ \text{pos}' = vel, \text{vel}' = \text{acc} \land \text{vel} \geq 0 \}]^* \text{vel} \geq 0 \]
Core: Uniform Substitution

\[ \text{Uniform Substitution} \]

\[
\begin{align*}
\phi & \\
\sigma(\phi) & 
\end{align*}
\]

Where \( \sigma \) performs admissible substitutions on functions, predicates, and program constants.
Axiom "K modal modus ponens".
\[
[a;](p(?) \rightarrow q(?)) \rightarrow (([a;]p(?)) \rightarrow ([a;]q(?)))
\]
End.

Axiom "DC differential cut".
\[
([c \& H(?);]p(?)) \leftrightarrow ([c \& (H(?) \& r(?));]p(?)) \leftarrow [c \& H(?);]r(?)
\]
End.

Axiom "[++] choice".
\[
[a ++ b]p(?) \leftrightarrow ([a;]p(?)) \& [b;]p(?))
\]
End.
Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]

\[
[\{\{ a := A \cup a := -B\}; \{x' = \nu, \nu' = a&\nu \geq 0\}\}^*] \nu \geq 0
\]
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [{\{ a := A \cup a := -B \}; \{ x' = v, v' = a \land v \geq 0 \}}^*] v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [{\{ \text{ctrl}; \text{plant} \}}^*] \psi \) Model:

1. Propositional Reasoning
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [{\{a := A \cup a := -B\}; \{x' = v, v' = a \& v \geq 0\}}^*]v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [{\{\text{ctrl; plant}\}}^*]\psi \) Model:

1. Propositional Reasoning
2. Identify System Loop Invariant
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{ a := A \cup a := -B \}; \{ x' = v, v' = a \& v \geq 0 \}]^* v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow \{ \text{ctrl; plant}\}^* \psi \) Model:

1. Propositional Reasoning
2. Identify System Loop Invariant
3. Symbolically Execute Control Program
\textbf{Tactics: Sketching and Searching} 

\textbf{Theorem} 
\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \] 
\[ [\{a := A \cup a := -B\}; \{x' = \nu, \nu' = a \& \nu \geq 0\}^*] \nu \geq 0 \]

A Prototypical Proof Outline for a $\varphi \rightarrow [\{\text{ctrl}; \text{plant}\}^*] \psi$ Model:
\begin{enumerate}
\item Propositional Reasoning
\item Identify System Loop Invariant
\item Symbolically Execute Control Program
\item Solve ODE or identify Differential Invariant(s)
\end{enumerate}
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]

\[ [{\{a := A \cup a := -B\}; \{x' = v, v' = a \land v \geq 0\}}^*}] v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [{\{ctrl; plant\}}^*] \psi \) Model:

1. Propositional Reasoning
2. Identify System Loop Invariant
3. Symbolically Execute Control Program
4. Solve ODE or identify Differential Invariant(s)
5. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]

\[ [[\{a := A \cup a := -B\}; \{x' = \nu, v' = a \land \nu \geq 0\}]^*] \nu \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [{\text{ctrl; plant}}^*] \psi \) Model:

1. Propositional Reasoning
2. Identify System Loop Invariant
3. Symbolically Execute Control Program
4. Solve ODE or identify Differential Invariant(s)
5. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem
\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ \left[ \left\{ \{ a := A \cup a := -B \}; \{ x' = v, v' = a \land v \geq 0 \} \right\} \right]^* v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [\{\text{ctrl}; \text{plant}\}^*] \psi \) Model:

1. ImplyR &
2. Identify System Loop Invariant
3. Symbolically Execute Control Program
4. Solve ODE or identify Differential Invariant(s)
5. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{ a := A \cup a := -B \}; \{ x' = \nu, \nu' = a \land \nu \geq 0 \}]^* \nu \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [\{ \text{ctrl; plant} \}^*] \psi \) Model:

1. ImplyR &
2. Loop("\( \nu \geq 0 \)")\( <(\text{QE,QE,} \)
3. Symbolically Execute Control Program
4. Solve ODE or identify Differential Invariant(s)
5. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ \{\{a := A \cup a := -B\}; \{x' = v, v' = a \land v \geq 0\}\}^*]v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow \{\{ctrl; plant\}\}^*\psi \)

Model:

- ImplyR &
- Loop("v \geq 0")<(QE,QE,
  Seq & Choice & BoxAssign &
4. Solve ODE or identify Differential Invariant(s)
5. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ \left[ \left\{ \{ a := A \cup a := -B \}; \{ x' = v, v' = a \& v \geq 0 \} \right\}^* \right] v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow \left[ \{ \text{ctrl; plant} \}^* \right] \psi \) Model:

1. \text{ImplyR} \&
2. \text{Loop("}v \geq 0\)\text{")} \& (QE, QE, Seq \& Choice \& BoxAssign \&
3. \text{DiffInv("}v \geq 0\)\text{")} \&
4. Appeal to Decision Procedure for Real Arithmetic
Tactics: Sketching and Searching

Theorem

\[ v \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ \left[ \begin{array}{l}
\{ a := A \cup a := -B \}; \\
\{ x' = v, v' = a \land v \geq 0 \} \end{array} \right]^* \]
\[ v \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow \left[ \{ \text{ctrl}; \text{plant} \}^* \right] \psi \) Model:

ImplyR &
Loop("v \geq 0")<(QE,QE,
Seq & Choice & BoxAssign &
DiffInv("v \geq 0") &
Arithmetic & Close)
Tactics: Sketching and Searching

Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{a := A \cup a := -B\}; \{x' = \nu, \nu' = a & \nu \geq 0\}]^* \nu \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [\{\text{ctrl; plant}\}]^* \psi \) Model:

- Prop &
- Loop("\( \nu \geq 0 \)"")<(QE,QE,
- SymbolicExecution &
- DiffInv("\( \nu \geq 0 \)") &
- Arithmetic & Close)
Tactics: Sketching and Searching

Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{ a := A \cup a := -B \}; \{ x' = \nu, \nu' = a \land \nu \geq 0 \}]^* \nu \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [\{ \text{ctrl}; \text{plant} \}]^* \psi \)

Model:

Prop &

Loop("\nu \geq 0") < (QE, QE, SymbolicExecution &

DiffInv(DIGen) &

Arithmetic & Close)
Theorem

$$v \geq 0 \land A > 0 \land B > 0 \rightarrow$$

$$[\{\{a := A \cup a := -B\}; \{x' = v, v' = a \land v \geq 0\}\}^*]v \geq 0$$

A Prototypical Proof Outline for a $$\varphi \rightarrow [\{\text{ctrl; plant}\}^*]$$\psi

Model:

Prop &
Loop(LoopInvGen)<(QE, QE &
SymbolicExecution &
DiffInv(DIGen) &
Arithmetic & Close)
Tactics: Sketching and Searching

Theorem

\[\nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[\left\{\left\{a := A \cup a := -B\right\}; \left\{x' = \nu, \nu' = a \& \nu \geq 0\right\}\right\}^{*} \nu \geq 0\]

A Prototypical Proof Outline for a \(\varphi \rightarrow \left\{\text{ctrl; plant}\right\}^{*} \psi\) Model:

\[\text{Prop \&}
\text{Loop(LoopInvGen)}\prec (\text{QE, QE \& SymbolicExecution \& DiffInv(DIGen) \& Arithmetic \& Close})\]
\[\Leftarrow\]
Theorem

\[ \nu \geq 0 \land A > 0 \land B > 0 \rightarrow \]
\[ [\{a := A \cup a := -B\}; \{x' = \nu, v' = a \& v \geq 0\}]^* \nu \geq 0 \]

A Prototypical Proof Outline for a \( \varphi \rightarrow [\{\text{ctrl}; \text{plant}\}^*] \psi \) Model:

- Prop &
- Loop(LoopInvGen) < (QE, QE &
  SymbolicExecution &
  DiffInv(DIGen) &
  Arithmetic & Close)
Applications and Uses

- Education: Foundations of CPS Course at CMU
- ACAS X
- ModelPlex
Challenge 1: Steep learning curve

- Commonplace mathematical objects are not primitives
  \[ e, \pi, \sin(x), \cos(x), \ldots \]
Challenge 1: Steep learning curve

- Commonplace mathematical objects are not primitives
  \[ e, \pi, \sin(x), \cos(x), \ldots \]
- Subtle modeling mistakes are easy
  Vacuous models: \[ [?H]P, [x' = \theta \land H]P, \ldots \]
  Non-implementable models
  \[ \ldots \]
Challenge 1: Steep learning curve

- Commonplace mathematical objects are not primitives
  \( e, \pi, \sin(x), \cos(x), \ldots \)

- Subtle modeling mistakes are easy
  Vacuous models: \([?H]P, [x' = \theta \land H]P, \ldots\)
  Non-implementable models
  \ldots

- Abrupt transitions as models become more difficult
  - From automated proving to interactive proving
  - From web UI to custom tactics
Challenge 2: Large Proofs are Difficult and Fragile

- Existing implementation: MDP $\Rightarrow$ large lookup table.
- Idea: Verify model, compare to outputs.
- **Possible!** But painful.
Conclusion

Small Core Increases trust, enables experimentation
Tactics Prover automation and proof reuse
Extensible New logics, proof rules, axioms
GUI point-and-click proofs for deeply nested formulas.

Developers: André Platzer, Stefan Mitsch, Nathan Fulton, Jan-David Quesel, Marcus Völп, Brandon Bohrer
Thanks: Ran Ji, Jean-Baptiste Jeannin, Sarah Loos, João Martins, Khalil Ghorbal

Download: http://keymaeraX.org