

# COUNTEREXAMPLES IN CPS THEOREM PROVING - FALSIFICATION OF DISCRETE-CONTINUOUS PROGRAMS

Tessa Hall, Stefan Mitsch

School of Computing, DePaul University, Chicago IL 60604, USA

{thall42, smitsch}@depaul.edu

## Motivation

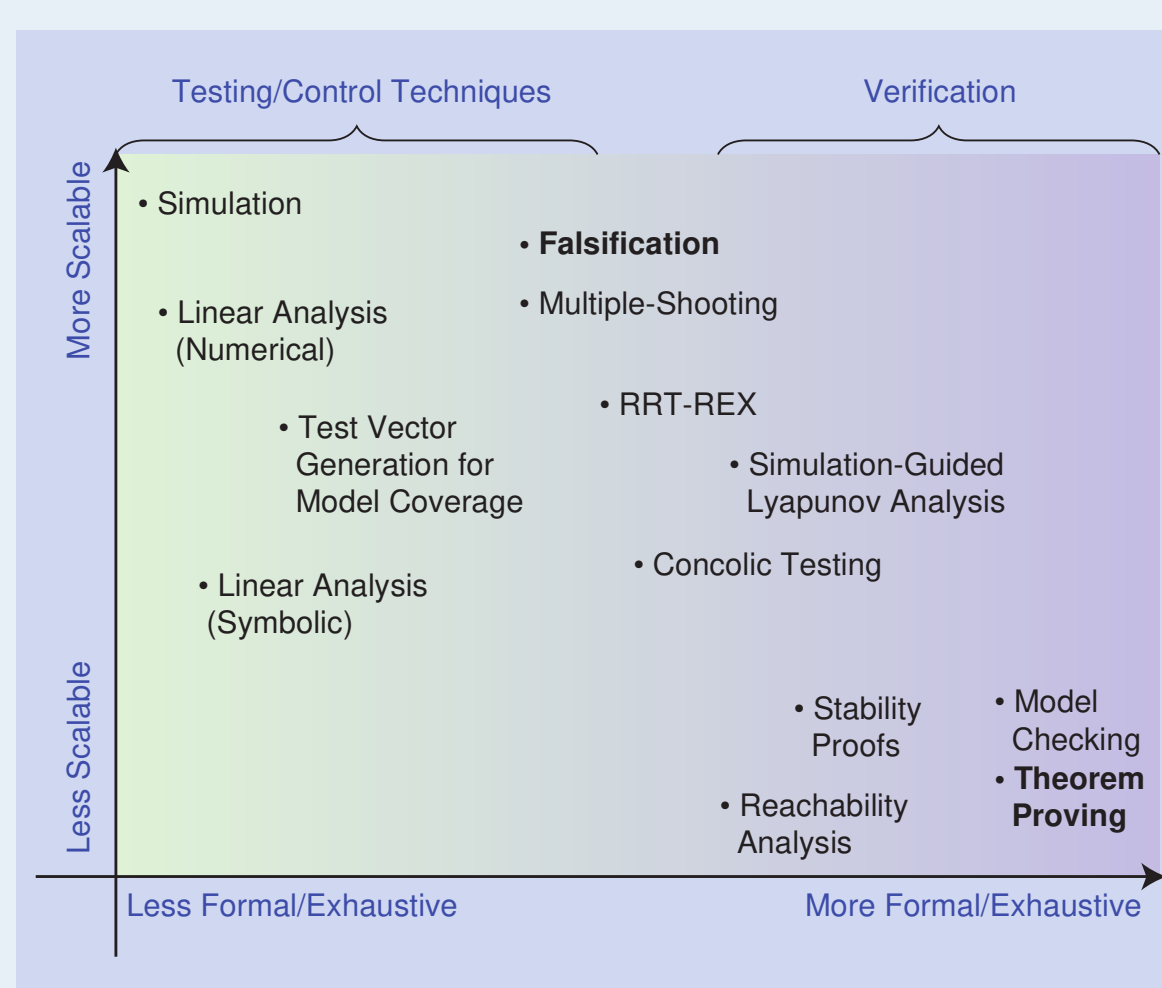
Autonomous and cyber-physical systems operate increasingly often in **safety-critical** domains (e.g., robot navigation, aircraft collision avoidance, or autonomous driving), which makes safety guarantees a necessity when aiming for trustworthy systems. **Theorem proving provides highest safety guarantees** with human-inspectable mathematical proofs in a logic system. The correctness arguments in cyber-physical systems require a logic system that can express the **interaction between computation, control, and physics**. Such hybrid systems models **make strong correctness guarantees**, but are **challenging to simulate** when providing counterexamples for incorrect models. Proof attempts therefore often iterate between progress in the proof and finding and correcting modeling mistakes.

## Approach Overview

To facilitate **finding modeling mistakes**, we propose an approach based on an **integration of theorem proving with falsification**.

**Theorem proving mathematically shows absence of bugs** in a correct model (i.e., a verified model provably satisfies a desired correctness property), while **falsification uses optimization to find bugs in an incorrect model** quickly and automatically (i.e., a falsified model violates a desired correctness property). We propose to find bugs in nondeterministic models by using falsification to steer their nondeterministic operators when attempting to find violations of a desired correctness property.

## Theorem Proving & Falsification



Approaches to CPS verification; adapted from [3]

### Falsification:

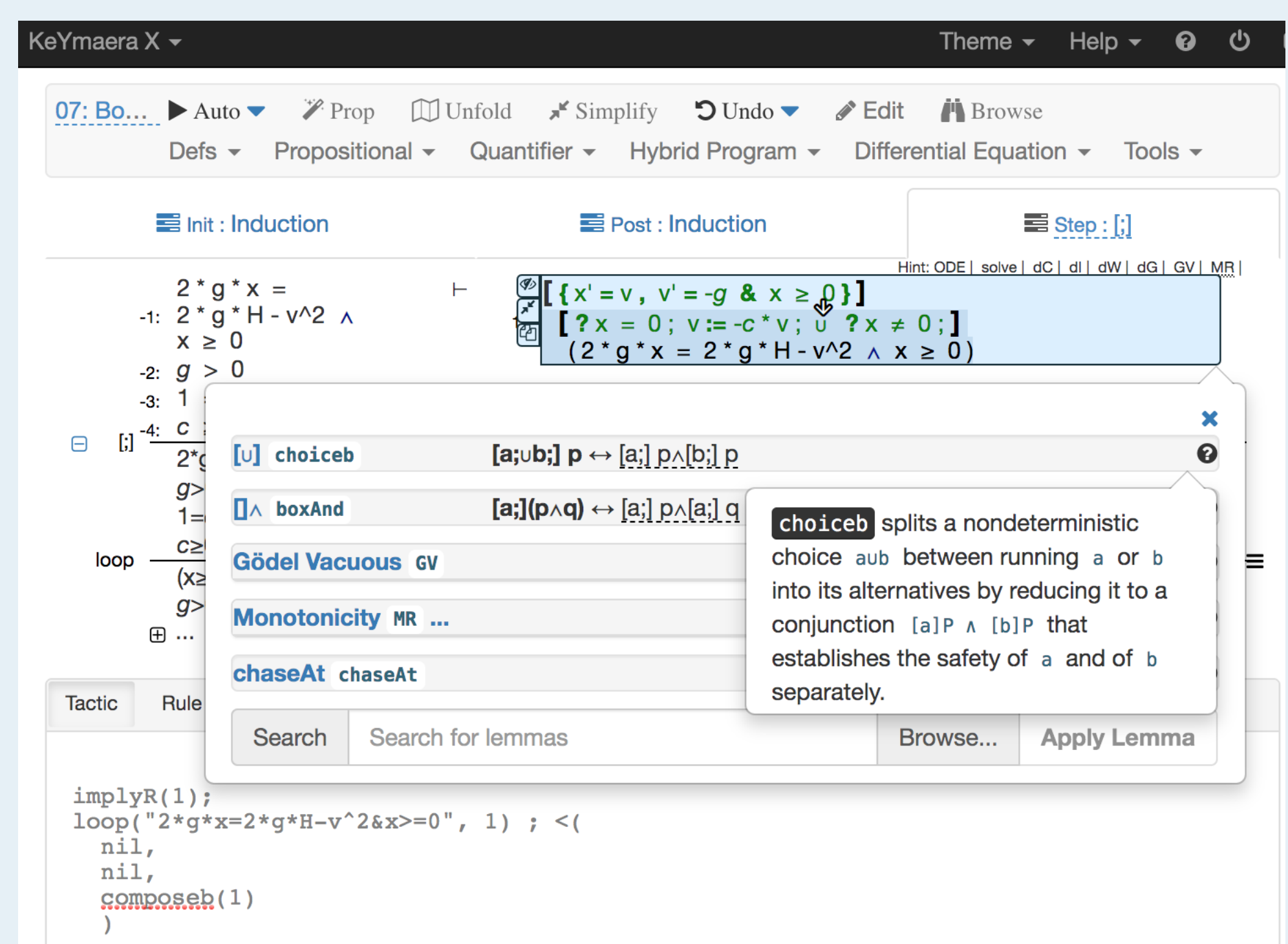
- Optimization to find bugs
- Scalable
- Deterministic  $\rightsquigarrow$  Easy to simulate

### Theorem Proving:

- Mathematically show absence of bugs
- Strong guarantees
- Nondeterministic  $\rightsquigarrow$  Hard to simulate

**Combine strengths of theorem proving and falsification**

## KeYmaera X

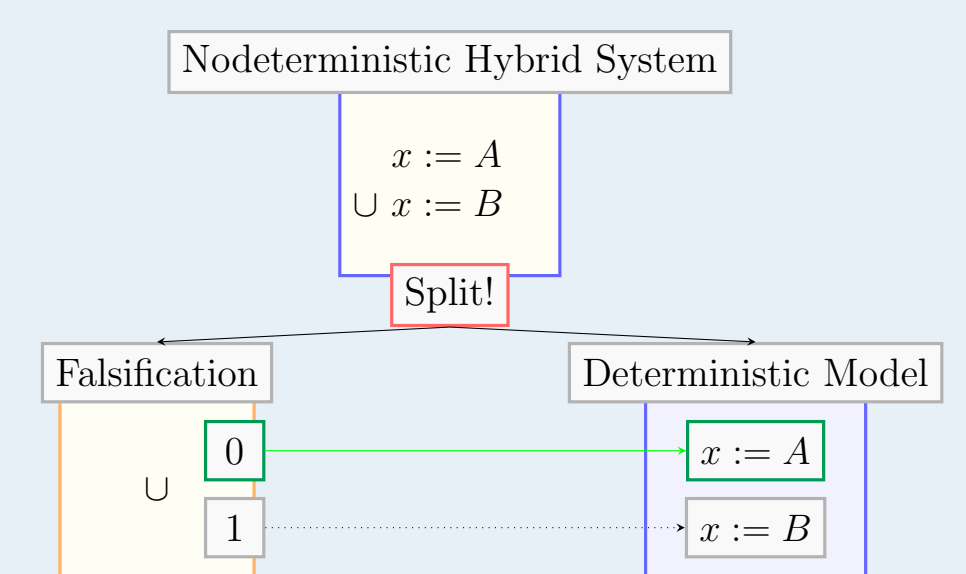


- Automated and interactive theorem proving [2, 4, 5]
- Verification of non-linear differential equations (Lyapunov functions, control barrier certificates, etc.)
- Safety and stability proofs [6]

## Falsification Steers Models

### Specification in KeYmaera X

- Assigning  $A$  or  $B$  to  $x$  satisfies  $x = B$
- $[x := A \cup x := B]x = B$
- $\text{Program is wrong} \rightsquigarrow$  Proof fails
- $\text{Find a program trace}$



### Resolve Nondeterministic Choice

A nondeterministic choice is resolved by falsification to seek violations of the specification. In the above figure the nondeterministic choice resolved to  $A$ .

Our proposed method uses falsification with FalStar [1] to **steer the nondeterministic operators of a theorem proving model**. To accomplish this, nondeterministic elements of the theorem prover language (e.g., choice, repetition, assignment, duration of differential equations) are split out of the hybrid system and **resolved with falsification to produce counterexample traces**.

## Bibliography

- [1] ERNST, G., SEDWARDS, S., ZHANG, Z., AND HASUO, I. Falsification of hybrid systems using adaptive probabilistic search. *ACM Trans. Model. Comput. Simul.* 31, 3 (2021), 18:1–18:22.
- [2] FULTON, N., MITSCH, S., QUESEL, J., VÖLP, M., AND PLATZER, A. Keymaera X: an axiomatic tactical theorem prover for hybrid systems. In *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings (2015)*, A. P. Felty and A. Middeldorp, Eds., vol. 9195 of *Lecture Notes in Computer Science*, Springer, pp. 527–538.
- [3] KAPINSKI, J., DESHMUKH, J. V., JIN, X., ITO, H., AND BUTTS, K. Simulation-based approaches for verification of embedded control systems: An overview of traditional and advanced modeling, testing, and verification techniques. *IEEE Control Systems Magazine* 36, 6 (2016), 45–64.
- [4] MITSCH, S. Implicit and explicit proof management in keymaera X. In *Proceedings of the 6th Workshop on Formal Integrated Development Environment, F-IDE@NFM 2021, Held online, 24-25th May 2021 (2021)*, J. Proença and A. Paskevich, Eds., vol. 338 of *EPTCS*, pp. 53–67.
- [5] MITSCH, S., AND PLATZER, A. A retrospective on developing hybrid system provers in the keymaera family - A tale of three provers. In *Deductive Software Verification: Future Perspectives - Reflections on the Occasion of 20 Years of KeY*, W. Ahrendt, B. Beckert, R. Babel, R. Hähle, and M. Ulbrich, Eds., vol. 12345 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 21–64.
- [6] TAN, Y. K., MITSCH, S., AND PLATZER, A. Verifying switched system stability with logic. In *HSCC '22: 25th ACM International Conference on Hybrid Systems: Computation and Control, Milan, Italy, May 4 - 6, 2022 (2022)*, E. Bartocci and S. Putot, Eds., ACM, pp. 2:1–2:11.