Simulation-Guided Formal Analysis

Jyotirmoy V. Deshmukh
Toyota Technical Center

Joint work with: Jim Kapinski, Xiaoqing Jin (TTC);
Sriram Sankaranarayanan, Aditya Zutshi (Colorado);
Nikos Aréchiga (CMU);
Alexandre Donzé, Sanjit Seshia (UC Berkeley).
Why Model-Based Development?

• US legislation: 54.5 mpg by 2025
• EU: 65mpg by 2020
• How??

Google Image search, © unknown
Earlier phase: Focus on control algorithms, high-level requirements, easier and cheaper to debug

Later phase: Focus on control implementations, real-time/platform-aware requirements, harder and more expensive to debug
Left of V is where the action is, but ...

- Plant models:
  - nonlinear dynamics
  - big look-up tables
  - copious amounts of switching
  - black-box components with no models
  - modeling language semantics

- Controllers:
  - Time and Event-triggered modules (that look like a lot like C code)

- Requirements:
  - Natural language/Evolving

© https://blogs.olin.edu/studentblog/2008/11/pi-umpkin.html
Engineers like Simulation

- Helps design validation
- Provides visual feedback
- Can uncover bugs
- Does not require knowledge of:
  - Temporal Logic, SAT modulo theories, Bounded Model Checking; not even
    Hoare logic, Turing machines or Lambda-calculus!
- Simulations are cheap and usually fast
- Test-suites can be shared and built up across models
How can we assist them?

• Idea: **Inject formal analysis into simulation environments**

• NOT a fundamentally new idea:
  – Proofs from tests: Gupta et al (TACAS 2009)
  – Predictive analysis: Farzan et al (TACAS 2009)
  – .... (please pardon the omissions)

• But, a new domain requiring new techniques
  – Closed-loop Control Systems
Simulation Guided Formal Analysis

• Learn Signal/Metric Temporal Logic (STL/MTL) requirements from simulations [HSCC 2013]
• Learn Lyapunov functions, barrier certificates from simulations [HSCC 2014, AMMCS 2013]
• Falsification Analysis using optimization and trajectory splicing [CDC 2013]
SImulation Guided Formal Analysis

• Learn Signal/Metric Temporal Logic (STL/MTL) requirements from simulations [HSCC 2013]
• Learn Lyapunov functions, barrier certificates from simulations [HSCC 2014, AMMCS 2013]
• Falsification Analysis using optimization and trajectory splicing [CDC 2013]
Lyapunov Analysis

• Ranking functions
  – Used to show program termination
  – Decreasing function \( r(x') < r(x) \) over a well-founded set
  – If exists, shows program terminates
  – No general way to find \( r(\cdot) \)

• Lyapunov functions
  – Used to show stability for dynamical system
    \[
    \frac{dx}{dt} = f(x)
    \]
  – Decreasing function \( V(x) > 0; \frac{dV(x(t))}{dt} < 0 \)
  – If exists, shows system is stable
  – No general way to find \( v(\cdot) \)
Simulations to learn Lyapunov functions

• Topcu et al [1] used simulations to assist *region of attraction* computation & proving stability

• New idea:
  
  ![Diagram](attachment:diagram.png)

  Simulations help find candidate Lyapunov functions

  Global optimizer to refine candidate

  SMT solvers + arithmetic decision procedures for proofs/counterexamples

---

How it works

• Fix a SoS function template: \( V(\mathbf{x}) = \mathbf{z}^T \mathbf{Pz} \)

• \( \mathbf{z} \) is a vector of monomials, e.g. \( [x_1 \ x_1^2 x_2^3 \ x_2^4]^T \)

• When using simulations, only \( \mathbf{P} \) is unknown

• Get constraints from simulations:

\[
\begin{align*}
- & V(\mathbf{x}_0(0)) > 0 ; \\
- & V(\mathbf{x}_N(0)) > 0 \\
- & V(\mathbf{x}_0(\Delta)) - V(\mathbf{x}_0(0)) < 0 ; \\
- & V(\mathbf{x}_N(\Delta)) - V(\mathbf{x}_N(0)) < 0
\end{align*}
\]

[Positive Definiteness]

[Approximation of Lie Derivative is Negative Definite]


Linear Program

Solution = candidate \( V(\mathbf{x}) \)
Refining Candidate

• Nelder-Mead Optimizer to find counterexamples:
  – maximize $V(x(\Delta)) - V(x(0))$
  – positive = counterexample

Initial conditions
$x_i(0)$ selected by optimizer

Domain, $\mathcal{D}$
Proving Validity

• Decision procedures to prove validity:
  – Arithmetic: Mathematica, QEPCAD, MetiTarski, z3
  – Interval Constraint Propagation based: iSAT, dReal [δ-complete decision procedure]
    • Negation is Unsat: Certificate is Valid
    • Negation is δ-Sat: Possible counterexample

```lisp
(set-logic QF_NRA)
(declare-fun x1 () Real)
(declare-fun x2 () Real)
(assert (<= 3.0 x1))
(assert (<= x1 3.14))
(assert (<= -7.0 x2))
(assert (<= x2 5.0))
(assert (<= (- (* 2.0 3.14159265) (* 2.0 (* x1 (arcsin (* (cos 0.797) (sin (/ 3.14159265 x1)))))))
          (+ (- 0.591 (* 0.0331 x2)) (+ 0.506 1.0))))
(check-sat)
(exit)
```
Hybrid System Example

\[ \dot{x} = \begin{cases} 
\begin{bmatrix} -0.1 & 1.0 \\ -10 & -0.1 \end{bmatrix} x & (x_1 \geq 0 \land x_2 \geq g(x_1)) \lor \\
\begin{bmatrix} -0.1 & 10 \\ -1 & -0.1 \end{bmatrix} x & (x_1 < 0 \land x_2 > g(x_1)) \lor \\
& (x_1 > 0 \land x_2 < g(x_1))
\end{cases} 
\]

\[ g(x_1) = 0.1e^{x_1} - 0.1 \]

Found Candidate:

\[ V(x) = 11x_1^2 + 2x_1x_2 + x_2^2 \]

in 171 seconds, with 9800 simulations; proved with Mathematica in 1.5 seconds
Air-to-Fuel ratio control

(See upcoming HSCC 2014 paper [1])

\[ \dot{p} = c_1 \left( 2\hat{u}_1 \sqrt{\frac{p}{c_{11}}} - \left( \frac{p}{c_{11}} \right)^2 - (c_3 + c_4 c_2 p + c_5 c_2 p^2 + c_6 c_2^2 p) \right) \]

\[ \dot{r} = 4 \left( \frac{c_3 + c_4 c_2 p + c_5 c_2 p^2 + c_6 c_2^2 p}{c_{13} (c_3 + c_4 c_2 p^2_{est} + c_5 c_2 p^2_{est} + c_6 c_2^2 p_{est})(1 + i + c_{14}(r - c_{16}))} \right) - r \]

\[ \dot{p}_{est} = c_1 \left( 2\hat{u}_1 \sqrt{\frac{p}{c_{11}}} - \left( \frac{p}{c_{11}} \right)^2 - c_{13} \left( c_3 + c_4 c_2 p_{est} + c_5 c_2 p^2_{est} + c_6 c_2^2 p_{est} \right) \right) \]

\[ \dot{i} = c_{15}(r - c_{16}) \]

r must be within 0.95 and 1.05 at steady state!


Found candidate in 25 mins., with 258K simulations and dReal proved safety in 20 mins.
Let’s ExCAPE from Controls for a bit

• Requirement Mining
  – Counterexamples for Inductive Synthesis of a Temporal Requirement

• Lyapunov Analysis:
  – Counterexamples for Inductive Synthesis of a Lyapunov function/Safety Certificates
Symbiosis: ExCAPE & SIGFA

ExCAPE → SiGFA

- CEGIS: pillar of ExCAPE
- SiGFA needs inductive synthesis/learning
  (some tasks expressible in SYNTH-LIB??)

SiGFA → ExCAPE

- SiGFA can be used for CAPE (e.g. parameter tuning)
- Industrial-scale benchmarks available
Thank You!