Progress on Powertrain Verification Challenge with C2E2

Chuchu Fan, Parasara Sridhar Duggirala, Sayan Mitra, and Mahesh Viswanathan
Air-Fuel Ratio Control: Motivation

- Control exhaust gas-emissions
  - Environment
  - Government regulation

- Air-Fuel Ratio
  - Important measure for pollution
  - Powertrain control system
    - Input: throttle angle, engine speed, other events…
    - Output: Air-Fuel ratio
Powertrain Control and C2E2

- A suite of powertrain verification benchmarks were presented in ARCH 2014 [Jin, et al. 2014]
  - Nonlinear, hybrid, control system models
  - Used to evaluate various technologies
    - Falsification: S-TaLiRo [Annpureddy, 2011], Breach [Donzé, 2010]
    - Requirement Mining [Jin, 2014]
    - Simulation-guided Lyapunov Analysis [Kapinski, 2014]

- C2E2 Verification tool is being developed at Illinois [EmSoft 2013, TACAS 2015]
  - Simulation data + Model → Proofs
Outline

• Overview of Powertrain Control Models

• Simulation-driven Verification
  • Overview
  • Local discrepancy computation

• Nuts, bolts, and Results

• Conclusion
Mode Structure of Powertrain Models

**Sensor_fail**
\[ \lambda_{\text{ref}} = 14.7 \]

**normal**
\[ \lambda_{\text{ref}} = 14.7 \]

**startup**
\[ \lambda_{\text{ref}} = 14.7 \]

**power**
\[ \lambda_{\text{power\ ref}} = 12.5 \]

- \( \theta_{\text{in}} \geq 70^\circ \)
- \( \theta_{\text{in}} \leq 50^\circ \)

**Startup Time** \( \tau_i \)

Closed-loop control + estimator

Open Loop, feedforward estimator
Model 1
Delay Differential Equations
+ Look up table
+ ...

Transport delay → first order filter
$2^{nd}$ order effects → first order filter
Look up table → polynomial fits
...

Model 2
Continuous-Time Plant
+ Discrete-Time Controller

Make controller continuous-time
Polynomialize
Compose plant with controller

Model 3
Continuous-Time Plant
+ Continuous-Time Controller
+ Polynomial RHS

Powertrain Control Models
Powertrain Control Models

Model 1

Delay
Differential Equations
+
Look up table
+
⋯
Powertrain Control Models

Model 3
Continuous-Time Plant
+ Continuous-Time Controller
+ Polynomial RHS
\[ \dot{\theta} = 10(\theta_{in} - \theta) \]

\[ \dot{p} = c_1(2\theta(c_{20}p^2 + c_{21}p + c_{22}) - c_{12}(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)) \]

\[ \dot{\lambda} = c_{26}(c_{15} + c_{16}c_{25}F_c + c_{17}c_{25}^2F_c^2 + c_{18}\dot{m}_c + c_{19}\dot{m}_cc_{25}F_c - \lambda) \]

\[ \dot{p}_e = c_1(2c_{23}\theta(c_{20}p^2 + c_{21}p + c_{22}) - (c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)) \]

\[ i = c_{14}(c_{24}\lambda - c_{11}) \]
Requirements: Signal Temporal Logic (STL)

- Allows tests over continuous-valued signal variables
- Examples:
  \[ \square_{[0,100]} \lambda \in (1,3) \]
  \[ \square_{[60,100]} |\lambda| < 0.1 \]
- Verification goals: Air-Fuel ratio should be contained in given interval

\[ \text{rise} \Rightarrow \square_{[\eta,\zeta]} [0.98 \lambda_{\text{ref}}, 1.02 \lambda_{\text{ref}}] \]
OVERVIEW OF SIMULATION-DRIVEN VERIFICATION
Consider n-dimensional ODE $\dot{x} = f(x)$

$\xi(x_0, t)$: for any $t$, solution/trajectory from $x_0$

A **validated simulation** is a sequence of time-stamped sets $(R_0, t_0), (R_1, t_1), \ldots, (R_n, t_n)$ that contains the solution:

- $R_i$ contains solution at $t_i$
- convex hull of $R_{i-1}$ and $R_i$ contain solution between $t_{i-1}$ to $t_i$

**Tools:** CAPD [Wilczak`10], V-Node LP [Nedialkov`06]
Definition. A \((D, \tau, T)\)-reachtube: a sequence of time-stamped sets \((\tilde{R}_0, t_0),\ (\tilde{R}_1, t_1), \ldots, (\tilde{R}_n, t_n)\) satisfying:

- \(\tilde{R}_i \subseteq \mathbb{R}^n\) is a compact set.
- \(t_n = T\) and, \(0 < t_i - t_{i-1} \leq \tau\).
- \(\forall x_0 \in D, \forall t \in [t_{i-1}, t_i], \xi(x_0, t) \in \tilde{R}_i\).
Simulation-driven Verification

- Given Initial set $\Theta$ and Unsafe set $U$

$$\dot{x} = f(x)$$
$$\xi : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$$
Simulation-driven Verification

• Given Initial set $\Theta$ and Unsafe set $U$
• **Partition**: compute finite cover of initial set

\[
\dot{x} = f(x) \\
\xi : \mathbb{R}^n \times \mathbb{R}^{\geq 0} \to \mathbb{R}^n
\]
Simulation-driven Verification

- Given initial set $\Theta$ and Unsafe set $U$
- Partition: compute finite cover of initial set
- **Simulate** from the vector $x_0$ of each cover element

\[
\dot{x} = f(x) \\
\xi: \mathbb{R}^n \times \mathbb{R}^2 \rightarrow \mathbb{R}^n
\]
Simulation-driven Verification

- Given initial set $\Theta$ and Unsafe set $U$
- Partition: compute finite cover of initial set
- Simulate from the vector $x_0$ of each cover element
- **Bloat** simulation so that bloated tube contains all trajectories from the cover element

\[
\begin{align*}
\dot{x} &= f(x) \\
\xi : \mathbb{R}^n \times \mathbb{R}^2 &\to \mathbb{R}^n
\end{align*}
\]
Simulation-driven Verification

- Given initial set Θ and Unsafe set U
- Partition: compute finite cover of initial set
- Simulate from the vector $x_0$ of each cover element
- Bloat simulation so that bloated tube contains all trajectories from the cover element
- Check intersection/containment with U

$$\dot{x} = f(x)$$
$$\xi : \mathbb{R}^n \times \mathbb{R}^0 \rightarrow \mathbb{R}^n$$
Simulation-driven Verification

- Given initial set $\Theta$ and Unsafe set $U$
- Partition: compute finite cover of initial set
- Simulate from the vector $x_0$ of each cover element
- Bloat simulation so that bloated tube contains all trajectories from the cover element
- Check intersection/containment with $U$
- **Take union** to get the reachtube from $\Theta$

$$\dot{x} = f(x)$$
$$\xi : \mathbb{R}^n \times \mathbb{R}^{\geq 0} \to \mathbb{R}^n$$

Key step: how to choose the bloating factor
**Definition:** A continuous function \( \beta : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0} \) is a discrepancy function [Duggirala 2013] of the system \( \dot{x} = f(x) \) if

- \( \forall x, x' \in \mathbb{R}^n, \forall t \geq 0 \), we have \( \|\xi(x, t) - \xi(x', t)\| \leq \beta(x, x', t), \)
- \( \forall t > 0 \), as \( x \rightarrow x' \), \( \beta(x, x', t) \rightarrow 0 \),

**Related concepts:** Lipchitz constant, Contraction metric, Incremental Lyapunov function…

**Example:** If \( f(x) \) has Lipschitz constant \( L \), then a discrepancy function: \(|\xi(x, t) - \xi(x', t)| \leq |x - x'|e^{Lt} = \beta(x, x', t)\)
Powertrain control model

- Example differential equation of the powertrain control model

\[
\dot{\theta} = 10(\theta_{in} - \theta)
\]

\[
\dot{p} = c_1 \left(2\theta(c_{20}p^2 + c_{21}p + c_{22}) - c_{12}(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)\right)
\]

\[
\dot{\lambda} = c_{26}(c_{15} + c_{16}c_{25}F_c + c_{17}c_{25}F_c^2 + c_{18}\dot{m}_c + c_{19}\dot{m}_cc_{25}F_c - \lambda)
\]

\[
\dot{p}_e = c_1 \left(2c_{23}\theta(c_{20}p^2 + c_{21}p + c_{22}) - (c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)\right)
\]

\[
i = c_{14}(c_{24}\lambda - c_{11})
\]

where

\[
F_c = \frac{1}{c_{11}}(1 + i + c_{13}(c_{24}\lambda - c_{11}))(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)
\]

\[
\dot{m}_c = c_{12}(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega p^2)
\]

- None of the related concepts can provide algorithm to produce discrepancy function
Local Piece-wise Exponential Discrepancy

- **Definition**: Consider a compact set $C \subseteq \mathbb{R}^n$ and a sequence of time points $0 = t_0 < t_1 < t_2 < \cdots < t_k = T$. For $\forall x_1, x_2 \in C$, $\forall t \in [0, T]$, a $C$-local piece-wise exponential discrepancy function $\beta: C \times C \times [0, T] \to \mathbb{R}_{\geq 0}$ is defined as

$$
\beta(x, x', t) = \begin{cases} 
\|x - x'\| & \text{if } t = t_0 \\
\beta(x, x', t_{i-1})e^{b[i](t-t_{i-1})} & \text{if } t \in (t_{i-1}, t_i)
\end{cases}
$$

where $b[i]$ are constants.

- Satisfies all criteria of Discrepancy function locally

- `ComputeLDF` returns coefficients vector $b$

\[\forall i = 1, \ldots, k\]

$b[i]$ is the bloating factor for $[t_{i-1}, t_i]$
ComputeLDF Subroutine

- **Input:**
  - A simulation trajectory $\psi = \{R_i, t_i\}_{i=0}^n$
  - Jacobian matrix $J_f$
  - Lipschitz constant $L_f$
  - Size of initial set $\delta$
  - Simulation precision $\epsilon$

- **Output:**
  - Coefficients to construct reachtube

**Algorithm 2: Algorithm ComputeLDF.**

```
input: $\psi = \{(R_i, t_i)\}_{i=0}^n, J_f, L_f, \delta, \epsilon$
1 $\Delta \leftarrow 0, b \leftarrow \text{zeros}(n)$
2 for $i = 1:n$ do
3     $\tau \leftarrow t_i - t_{i-1}$
4     $d \leftarrow (\Delta + \epsilon) e^{L_f \tau}$
5     $S \leftarrow \text{hull}(R_{i-1}, R_i) \oplus B_d(0)$
6     $J \leftarrow J_f(\text{center}(S))$
7     $\lambda \leftarrow \max(eig(J + J^T)/2)$
8     error $\leftarrow \text{upper}_{x \in S} \|(J_f(x) + J_f^T(x)) - (J + J^T)\|$
9     $b[i] \leftarrow \lambda + \text{error}/2$
10    $\Delta \leftarrow (\Delta + \epsilon) e^{b[i] \tau}$
11 end
12 return $b$
```

![Diagram showing reachtube construction](illinois.edu)
ComputeLDF Subroutine

- Coarse overapproximation $S$
  - Use Lipschitz constant
  - Contain all trajectories from initial set during $[t_0, t_1]$

```
Algorithm 2: Algorithm ComputeLDF.

input: $\psi = \{(R_i, t_i)\}_{i=0}^n, J_f, L_f, \delta, \epsilon$
1. $\Delta \leftarrow \delta, b \leftarrow \text{zeros}(n)$
2. for $i = 1:n$ do
3.   $\tau \leftarrow t_i - t_{i-1}$
4.   $d \leftarrow (\Delta + \epsilon)e^{L_f \tau}$
5.   $S \leftarrow \text{hull}(R_{i-1}, R_i) \oplus B_d(0)$
6.   $J \leftarrow J_f(\text{center}(S))$
7.   $\lambda \leftarrow \max(eig(J + J^T)/2)$
8.   $\text{error} \leftarrow \text{upper}_x \in S \| (J_f(x) + J_f^T(x)) - (J + J^T)\|
9.   $b[i] \leftarrow \lambda + \text{error}/2$
10.  $\Delta \leftarrow (\Delta + \epsilon)e^{b[i] \tau}$
11. end
12. return $b$
```
Local Convergence/Divergence rate

- **Lemma:** In region $S$, if $\forall x \in S, \frac{J(x) + J^T(x)}{2} \preceq bI$ then $\|\xi(x, t) - \xi(x', t)\| \leq \|x - x'\|e^{b(t-t_0)}$

- $J(x) = \left(\frac{\partial f_i(x)}{x_j}\right)_{ij}$ is the Jacobian matrix

- $b$ is an upper bound on the eigenvalues of the symmetric part of Jacobian matrix $\frac{J(x) + J^T(x)}{2}$
Compute Local Convergence/Divergence rate

\[ \dot{x} = f(x) \]
\[ \xi : \mathbb{R}^n \times \mathbb{R}^{\geq 0} \rightarrow \mathbb{R}^n \]

- Lemma: If \(X\) and \(Y\) are symmetric matrices, then [Teschl, 99]

- Calculate (bound on eigenvalues):
  - Pick center point \(x\) in \(S\), get the largest eigenvalue of the symmetric part of the Jacobian matrix at \(x\):

\[
\forall x \in S, \quad \left| \lambda_{\text{max}} \left( \frac{J^T(x) + J(x)}{2} \right) - \lambda_{\text{max}} \left( \frac{J^T(s_0) + J(s_0)}{2} \right) \right| \leq \left\| J^T(x) + J(x) - (J^T(s_0) + J(s_0)) \right\|_2 / 2
\]

\[
e \geq \max_{x \in S} \| J^T(x) + J(x) - (J^T(s_0) + J(s_0)) \|_2
\]

\[
b = l + e
\]
ComputeLDF Subroutine

- Coarse overapproximation $S$
  - Use Lipschitz constant
  - Contain all trajectories from initial set during $[t_0, t_1]$

```
Algorithm 2: Algorithm ComputeLDF.

input: $\psi = \{(R_i, t_i)\}_{i=0}^{n}, J_f, L_f, \delta, \epsilon$

1. $\Delta \leftarrow \delta, b \leftarrow \text{zeros}(n);$
2. for $i = 1:n$ do
3.   $\tau \leftarrow t_i - t_{i-1};$
4.   $d \leftarrow (\Delta + \epsilon)e^{L_f \tau};$
5.   $S \leftarrow \text{null}(R_{i-1}, R_i) + R_d(0);$
6.   $J \leftarrow J_f(\text{center}(S));$
7.   $\lambda \leftarrow \max(eig(J + JT)/2);$
8.   $\text{error} \leftarrow \text{upper}_{x \in S}\| (J_f(x) + J_f^T(x)) - (J + JT)\|;$
9.   $b[i] \leftarrow \lambda + \text{error}/2;$
10. $\Delta \leftarrow (\Delta + \epsilon)e^{L_f \tau};$
11. end
12. return $b;$
```
ComputeLDF Subroutine

- At time $t_1$, reachtube $\subseteq (R_1 \text{ bloated by } (\Delta + \epsilon)e^{b_1\tau})$
- During $[t_0, t_1]$, reachtube $\subseteq (\text{hull}(R_0, R_1) \text{ bloated by } \max\{\Delta + \epsilon, (\Delta + \epsilon)e^{b_1\tau}\})$
- Update $\Delta$ for next iteration

```
Algorithm 2: Algorithm ComputeLDF.

input: $\psi = \{(R_i, t_i)\}_{i=0}^{n}, J_f, L_f, \delta, \epsilon$
1 $\Delta \leftarrow \delta, b \leftarrow \text{zeros(n)}$
2 for $i = 1:n$ do
3     $\tau \leftarrow t_i - t_{i-1}$
4     $d \leftarrow (\Delta + \epsilon)e^{L_f\tau}$
5     $S \leftarrow \text{hull}(R_{i-1}, R_i) \oplus B_d(0)$
6     $J \leftarrow J_f(\text{center}(S))$
7     $\lambda \leftarrow \max(\text{eig}(J + J^T)/2)$
8     error $\leftarrow \text{upper} \max_{x \in S} \left\| (J_f(x) + J^T_f(x)) - (J + J^T) \right\|$
9     $b[i] \leftarrow \lambda + \text{error}/2$
10    $\Delta \leftarrow (\Delta + \epsilon)e^{b[i]\tau}$
11 end
12 return $b$
```
ComputeLDF Subroutine

Algorithm 2: Algorithm ComputeLDF.

\[
\text{ Computes the LDF parameters.}
\]

**input:** \( \psi = \{(R_i, t_i)\}_{i=0}^{n}, J_f, L_f, \delta, \epsilon \)

1. \( \Delta \leftarrow \delta, b \leftarrow \text{zeros}(n) ; \)

2. for \( i = 1:n \) do

3. \( \tau \leftarrow t_i - t_{i-1}; \)

4. \( d \leftarrow (\Delta + \epsilon)e^{L_f \tau}; \)

5. \( S \leftarrow \text{hull}(R_{i-1}, R_i) \oplus B_d(0) ; \)

6. \( J \leftarrow J_f(\text{center}(S)); \)

7. \( \lambda \leftarrow \max(\text{eig}(J + J^T)/2); \)

8. \( \text{error} \leftarrow \text{upper}_{x \in S} \| J_f(x) + J_f^T(x) \| - \| J + J^T \|; \)

9. \( b[i] \leftarrow \lambda + \text{error}/2; \)

10. \( \Delta \leftarrow (\Delta + \epsilon)e^{b[i] \tau}; \)

end

12. return \( b \);
NUTS, BOLTS, AND RESULTS
Model Transformation: Simulink Model

Fuel Control System Model

This model uses only the ODEs to implement the dynamics.

- Throttle input (deg) [0, 81.2]
- Throttle [0.9]
- Base opening angle [0.8]
- Engine speed (rpm) [0.0, 110.0]
- Startup Mode
- Sensor Failure Detection Latch

Powertrain Control Benchmark Model
Toyota Technical Center
2014

This is a model of a hybrid automaton with polynomial dynamics, and an implementation of the 3rd model that appears in "Powertrain Control Verification Benchmark", 2014 Hybrid Systems: Computation and Control, X. Jin, J. V. Deshmukh, J. Asadi, K. Ueda, and K. Butler
Model Transformation: Stateflow Model
Sensor_fail
\[ \dot{x} = f_{sf}(x) \]

normal
\[ \dot{x} = f_{n}(x) \]

power
\[ \dot{x} = f_{p}(x) \]

\[ \theta_{in} \geq 70^\circ \]
\[ \theta_{in} \leq 50^\circ \]

Model Transformation: Stateflow Model

\[ \text{Sensor_fail} \]
\[ \text{startup} \]
\[ \dot{x} = f_{s}(x) \]

\[ \text{normal} \]
\[ \dot{x} = f_{n}(x) \]

\[ \text{power} \]
\[ \dot{x} = f_{p}(x) \]
Engineering and Artifacts

- Encoding Drivers and Properties
  - Hybrid system to time switch system
  - Transition between modes is defined by start time $T_s$, sensor failure event and throttle angle $\theta_{in}$
  - Explicit construction of a family of signals that determines the timing of mode switches

- Coordinate Transformation
  - Symmetric part of Jacobian may have positive eigenvalues when the trajectories are converging
  - Local coordinate transformation give less conservative bounds
  - Results should be multiplied by condition number

- Model Reduction
  - In start up and power mode, variable $i$ is “isolated”
  - Drop $i$ and reduce the model to 3 continuous variables in corresponding modes
Initial condition: $\lambda_{ref} \times (1 \pm 1\%)$
Verified many key specification for a given set of driver behaviors

<table>
<thead>
<tr>
<th>Property</th>
<th>Mode</th>
<th>Sat</th>
<th>Sim.</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>all modes</td>
<td>all modes</td>
<td>Yes</td>
<td>53</td>
<td>11m58s</td>
</tr>
<tr>
<td>startup</td>
<td>startup</td>
<td>Yes</td>
<td>50</td>
<td>10m21s</td>
</tr>
<tr>
<td>normal</td>
<td>normal</td>
<td>Yes</td>
<td>50</td>
<td>10m21s</td>
</tr>
<tr>
<td>power</td>
<td>power</td>
<td>Yes</td>
<td>53</td>
<td>11m12s</td>
</tr>
<tr>
<td>power</td>
<td>power</td>
<td>No</td>
<td>4</td>
<td>0m43s</td>
</tr>
<tr>
<td>normal</td>
<td>normal</td>
<td>Yes</td>
<td>50</td>
<td>10m15s</td>
</tr>
<tr>
<td>power</td>
<td>power</td>
<td>Yes</td>
<td>53</td>
<td>11m35s</td>
</tr>
<tr>
<td>power</td>
<td>power</td>
<td>No</td>
<td>4</td>
<td>0m45s</td>
</tr>
</tbody>
</table>
Extension to Model 2

Input to state discrepancy

\[ \| \xi(x, u, t) - \xi(x', u', t) \| \leq \beta(x, x', t) + \int_0^t \gamma(|u(s) - u'(s)|) ds \]

**Sensor_fail**

\[ \begin{align*}
  t &= 1 \\
  \dot{x}_p &= f_{sf}(x_p) \\
  \dot{x}_c &= 0 \\
  t &= h \\
  \{x_c := g_{sf}(x_c); t := 0\}
\end{align*} \]

**normal**

\[ \begin{align*}
  t &= 1 \\
  \dot{x}_p &= f_n(x_p) \\
  \dot{x}_c &= 0 \\
  t &= h \\
  \{x_c := g_n(x_c); t := 0\}
\end{align*} \]

**power**

\[ \begin{align*}
  t &= 1 \\
  \dot{x}_p &= f_p(x_p) \\
  \dot{x}_c &= 0 \\
  t &= h \\
  \{x_c := g_p(x_c); t := 0\}
\end{align*} \]
Conclusions

• Meet the powertrain verification challenge
  – Simulation-guided verification
  – New feature in C2E2: local piece-wise discrepancy function

• Learn from the verification problem
  – Engineering and artifacts
  – Experimental results

• Next steps
Special Thanks to

- Xiaoqing Jin
- Jyotirmoy Deshmukh
- Jim Kapinski
- Koichi Ueda
- Ken Butts