Computational Modeling and Analysis For Complex Systems
NSF Expedition in Computing

CMACS
Embedded Systems Challenge Problem

Bruce H. Krogh
Carnegie Mellon University

2nd Year Review Meeting, Carnegie Mellon University
November 3, 2011
Embedded Control Systems

*Design Flow*

- requirements
- algorithms
- detailed design
- code
Embedded Control Systems

**Design Flow**
- requirements
- algorithms
- detailed design
- code

**Challenges**
- consistency
- correctness
- complexity
- coverage
- run-time correctness
Embedded Control Systems

**Design Flow**
- requirements
- algorithms
- detailed design
- code

**Challenges**
- consistency
- correctness
- complexity
- coverage
- run-time correctness

**CMACS Research**
- requirements reconstruction
- analysis of hybrid systems
  - theorem proving
  - compositionality
  - reachability
  - statistical model checking
- code verification
  - abstract interpretation
  - analysis-aware design
  - run-time verification
Impossible Without An Expeditions Project
Requirements Reconstruction

Challenge Problem
Outdated requirements documents for automotive embedded systems
  • due to system evolution
  • limits ability to apply formal verification in future development

Approach
Use test data to re-create high-level descriptions of system behavior.
  • apply machine learning: association-rule mining
  • identify possible invariants satisfied by the system.

Technical Challenges
  • quickly detecting and eliminating false invariants
  • ensuring that correct invariants are indeed detected

Research Team: UMD: Chris Ackermann, Rance Cleaveland, Sam Huang; Fraunhofer: Arnab Ray; Robert Bosch: Beth Latronico, Charles Shelton
Requirements Reconstruction – cont’d.

Major Advances

Applied *instrumentation-based verification* (model checking technique)
  • identifies false invariants
  • ensures test data satisfies coverage constraints
  • ensures coverage of proposed invariants

Results to date

For a large production automotive control subsystem
  • 41 of 42 invariants recovered for one module
  • found 2 invariants not stated in the requirements
  • only 1 incorrectly declared invariant not detected.

Current work

  • genetic algorithms for inferring temporal properties
  • larger pilot study involving 10 automotive control subsystems

Composition of Hybrid Systems

Challenge Problem
How can component-based models of automotive embedded control systems be composed and analyzed in a rigorous way based on formal methods?

Approach
Generalize ideas of Process Algebra to hybrid (dynamical) systems to analyze/verify complex systems in terms of simpler, reusable subsystems

Technical Challenges
• theories of composition has received relatively little attention for hybrid systems
• need new mathematical frameworks supporting the rich array of mechanisms used to build composite embedded systems in practice

Research Team: UMD: Rance Cleaveland (CS), Steve Marcus (ECE), Peter Fontana (CS), James Ferlez (ECE)
Composition of Hybrid Systems – cont’d.

Major Advances
New mathematical model of system behavior that generalizes methods of Process Algebra to hybrid systems
• asynchronous parallel composition
• synthesis of ideas from computer science (process algebra) and control (the behavioral methodology of Willems, van der Schaft, etc.)

Results to date
• generalized synchronization trees (GSTs) for hybrid systems
• preliminary algebraic properties of GSTs
• paper in progress

Current work
• further algebraic properties of GSTs
• types of generalized composition
• control law synthesis
Design Verification

Challenge Problem
Verification of stochastic Stateflow/Simulink models
E.g.
\[ \Phi = \neg F_{100} G_1(\text{FuelFlowRate} = 0) \]
\[ \text{Prob } (\text{Sys} \models \Phi) = 0.9779 \pm 0.01 \]

Approach
\[ \text{Prob } (\Phi)? \]
simulation + model checking + statistical estimation
\[ \text{Prob } (\Phi) > \theta? \]
simulation + model checking + statistical hypothesis testing

Research Team: CMU: Ed Clarke, Paolo Zuliani, Andre Plazer; TU Dresden: Christel Baier
Major Advances
- Efficient Bayesian estimation and hypothesis testing techniques
- Importance Sampling (IS) and Cross-Entropy (CE) with statistical MC

Results to date
- Improvement of 2-3 orders of magnitude in speed over previous methods (techniques based on Chernoff bound)
- Verified a fault-tolerant controller for an aircraft elevator system

P. Zuliani, C. Baier, E.M. Clarke. Rare-Event Verification for Stochastic Hybrid Systems. Submitted
Embedded Software Verification

Challenge Problems
Scale model checking algorithms to handle unmodified industrial size software as used for safety critical embedded systems (aerospace/automotive/medical)
Improve runtime verification techniques by creating more expressive specification languages with efficient monitoring algorithms, and designing specification learning and trace visualization techniques.

Approach
• develop new analysis-aware software design methods
• develop new context aware verification methods
• target massive use of parallelism

Research Team: JPL/CalTech: Klaus Havelund, Gerard Holzmann, Mihai Florian (Caltech CS, grad student), Ed Gamble

LARS
NASA/JPL Laboratory for Reliable Software
Current work

- direct verification of real-time priority-based scheduling algorithms
- new multi-core and cloud-based model checking algorithms
  - performance is expected to scale linearly with the number of available processing elements (cores, CPUs, and/or GPU engines),
  - potential for orders of magnitude improvements on large compute farms
- new efficient rule-based methods for runtime verification based on pattern matching

Advances in aerospace applications

- The paper
  Julien Bertrane, Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, & Xavier Rival.
  Static Analysis and Verification of Aerospace Software by Abstract Interpretation. In AIAA Infotech@Aerospace 2010, Atlanta, Georgia. American Institute of Aeronautics and Astronautics, 20—22 April 2010. © AIAA.
  received the AIAA intelligent systems best paper award 2010

- All control/command software of a European aircraft manufacturer now mandatorily verified by abstract-interpretation based static static analysis (in conformance with DO-178-C)

- Progress on the static verification of parallel processes
Advances in abstract interpretation

Significant advances on

- Under-approximation
- Combination of algebraic and logical abstractions
- Probabilistic abstraction
- Termination/liveness

have been done for infinite state systems.
Difficulty of the problems

• Abstraction to finite / bounded executions is impossible (unsound, ineffective, ...)

Example: [non]-termination of unbounded programs

• Abstraction must be infinite, which is extremely difficult
Under-approximation

- **Previously:** explore finite parts of a finite subset of executions
- **New:** algebraic approach to handle infinitely many infinite executions
- **Example:** pre-conditions ensuring the presence of errors
Combining algebraic & logical abstractions

• A new understanding of the Nelson-Oppen procedure to combine logical theories in SMT solvers/provers as an algebraic reduced product

• When checking satisfiability of $\varphi_1 \land \varphi_2 \land ... \land \varphi_n$, the Nelson-Oppen procedure generates (dis)-equalities that can be propagated by $\rho_{la}$ to reduce the $P_i$, $i=1,..,m$
• $\alpha_i(\varphi_1 \land \varphi_2 \land ... \land \varphi_n)$ can be propagated by $\rho_{la}$ to reduce the $P_i$, $i=1,..,m$
• The purification to theory $T_i$ of $\gamma_i(P_i)$ can be propagated to $\varphi_i$ by $\rho_{al}$ in order to reduce it to $\varphi_i \land \gamma_i(P_i)$ (in $T_i$)
Termination

• Previously: recent progress on automatic proof of termination for small, simple and pure programs (no abstraction needed)

• Challenge: scale automatic program termination methods to large, complex, and realistic programs by integrating abstraction

• New advances:
  
  • Trace segments as a new basis for inductively formulating program properties
  
  • Fixpoint definition of a collecting semantics for termination/liveness
  
  • Systematic ways for constructing termination proofs, by construction of abstract fixpoints (e.g. variant functions)
  
  • Includes weak fairness
Distributed and Compositional Hybrid Systems
Hierarchical and Compositional Verification

Hierarchical Modularity

Decompositions

\[ \phi_x^\theta \]

\[ [x := \theta] \phi \]

\[ \exists t \geq 0 \langle x := y_x(t) \rangle \phi \]

\[ \langle x' = f(x) \rangle \phi \]

\[ [\alpha] \phi \land [\beta] \phi \]

\[ [\alpha \cup \beta] \phi \]

\[ \vdash p \quad \vdash (p \rightarrow [\alpha] p) \]

\[ \vdash [\alpha^*] p \]
Sensor limits on actual cars are always local. Sometimes a maneuver may look safe locally... But is a terrible idea when implemented globally because of unsafe emergent behavior.
Car Control Proof Sketch

Local Lane Control

Global Lane Control

Local Highway Control

Global Highway Control

2 vehicles
1 lane
no lane change

n vehicles
1 lane
no lane change

n vehicles
1 lane
lane changes

n vehicles
m lanes
lane changes
\[ \forall i : C(i \ll L(i)) \rightarrow [lhc]\forall i : C(i \ll L^*(i)) \]

\[
lhc \equiv (\text{delete}^*; \text{create}^*; \text{ctrl}^n; \text{dyn}^n)^* \\
\text{create} \equiv n := \text{new}; \ ?((F(n) \ll n) \land (n \ll L(n))) \\
(n := \text{new}) \equiv n := *; \ ?(E(n) = 0); \ E(n) := 1 \\
(F(n) \ll n) \equiv \forall j : C (L(j) = n \rightarrow (j \ll n)) \\
\text{delete} \equiv n := *; \ ?(E(n) = 1); \ E(n) := 0 \\
\text{ctrl}^n \equiv \forall i : C (\text{ctrl}(i)) \\
\text{ctrl}(i) \equiv (a(i) := *; \ ?(-B \leq a(i) \leq -b)) \\
\bigcup (\text{Safe}_c(i); \ a(i) := *; \ ?(-B \leq a(i) \leq A)) \\
\text{Safe}_c(i) \equiv x(i) + \frac{v(i)^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2} \epsilon^2 + \epsilon v(i)\right) < x(L(i)) + \frac{v(L(i))^2}{2B} \\
\text{dyn}^n \equiv (t := 0; \ \forall i : C (\text{dyn}(i)), t' = 1, t \leq \epsilon) \\
\text{dyn}(i) \equiv \dot{x}(i) = v(i), \dot{v}(i) = a(i), v(i) \geq 0 \]
Proof: Local Highway Control

∀\(i\) \(x(i) \ll x(L(i))\) → [\(\text{glc}\)] \(\forall i\) \(x(i) \ll x(L(i))\)

Transitivity

∀\(i\) \(x(i) \ll L(x(i))\) → [\(\text{create}^*\)] \(\forall i\) \(x(i) \ll L^*(x(i))\)

Transitivity

∀\(i\) \(x(i) \ll L(x(i))\) → [\(\text{delete}^*\)] \(\forall i\) \(x(i) \ll L^*(x(i))\)

∀\(i\) \(x(i) \ll L(x(i))\) → [\(\text{delete}^*\)][\(\text{create}^*\)][\(\text{glc}\)] \(\forall i\) \(x(i) \ll L^*(x(i))\)

∀\(i\) \(x(i) \ll L(x(i))\) → [\(\text{lhc}\)] \(\forall i\) \(x(i) \ll L^*(x(i))\)

(cut)

\([\cdot]\) split

\([\cdot;\cdot]\)
Conclusions

Challenges

• Infinite, continuous, and evolving state space, $\mathbb{R}^\infty$
• Continuous dynamics
• Discrete control decisions
• Distributed dynamics
• Arbitrary number of cars, changing over time
• Emergent behaviors

Solutions

• Quantifiers for distributed dynamics of cars
• Compositionality – using small problems to solve the big ones
• Hierarchical and modular proofs
• Variations in system design
• Future work: curved road dynamics and using differential invariants
Rollover Verification of a Truck

**Problem:** Prove that truck cannot roll over under all possible maneuvers when the truck is braking \(a_x = -7 \text{ m/s}^2\) and the lateral acceleration is bounded by \(a_y \in [-4, 4] \text{ m/s}^2\)

- Infinitely many maneuvers including all steering frequencies.
- Cannot be exhaustively tested by real experiments and simulations.

**Challenges:**

- Nonlinear continuous dynamics (8 continuous state variables)
- Uncertainty: Steering input
- Hybrid dynamics (gain scheduled controller)
Capturing Nonlinear Dynamics and Uncertain Inputs

Inherit problem: Only linear maps are structure-preserving for common set representations (ellipsoids, polyhedra, zonotopes, etc.)

Solution: Abstract nonlinear dynamics to linear dynamics ($x$: state, $u$: input):

$$\dot{x} = f(x(t), u(t)) \in \left\{ A(t)x(t) + u(t) + v(t) \mid A(t) \in \mathcal{A}, v(t) \in \mathcal{V} \right\}$$

Dynamic abstraction using

- uncertain system matrix $\mathcal{A}$: [Althoff, Le Guernic, Krogh 2011]
- uncertain additional input $\mathcal{V}$: [Dang, Le Guernic, Maler 2011; Althoff et al. 2008]

Old technique: Static abstraction (coarser abstraction, guard intersection required):

New technique: Dynamic abstraction (tighter abstraction, no guard intersection required):
Capturing Switching Dynamics

Hybrid reachability is limited by geometric intersections with guard sets, which is

- exact for polyhedra, but does not scale and is numerically unstable,
- efficient for other representations (template polyhedra, etc.), but conservative.

**Old technique**: Classical intersection computation possibly resulting in large overapproximation.

**New technique**: Compute with union of parameters when only the parameter set changes [Althoff, Le Guernic, Krogh 2011].

\[ P_{\text{total}} = P_1 \]

\[ P_{\text{total}} = \text{CH}(P_1 \cup P_2) \]
Dynamics of the Closed Loop System

truck dynamics (blue variables are states, red ones are inputs) taken from [Gaspar et al. 2004]:

\[
mx_7(\dot{x}_1 + x_2) - mS h \dot{x}_4 = Y_\beta x_1 + Y_\psi(x_7)x_2 + Y_\delta \delta \\
- l_{xz} \dot{x}_4 + l_{zz} \dot{x}_2 = N_\beta x_1 + N_\psi(x_7)x_2 + N_\delta \delta \\
(l_{xx} + mS h^2) \dot{x}_4 - l_{xz} \dot{x}_2 = mS gh x_3 + mS h x_7(\dot{x}_1 + x_2) - k_f(x_3 - x_5) \\
- b_f(x_4 - \dot{x}_5) - k_r(x_3 - x_6) - b_r(x_4 - \dot{x}_6) \\
-r(Y_\beta, f x_1 + Y_\psi, f x_2 + Y_\delta \delta) = m_{u,f}(r - h_{u,f}) x_7(\dot{x}_1 + x_2) + m_{u,f} g h_{u,f} x_5 \\
- k_{t,f} x_5 + k_f(x_3 - x_5) + b_f(x_4 - \dot{x}_5) \\
-r(Y_\beta, r x_1 + Y_\psi, r x_2) = m_{u,r}(r - h_{u,r}) x_7(\dot{x}_1 + x_2) - m_{u,r} g h_{u,r} x_6 \\
- k_{t,r} x_6 + k_r(x_3 - x_6) + b_r(x_4 - \dot{x}_6) \\
\dot{x}_7 = a_x.
\]

yaw controller: \( \delta = k_1 e + k_2 \int e(t) \, dt, \quad e = \dot{\psi}_d - \dot{\psi} = \dot{\psi}_d - x_2. \)

<table>
<thead>
<tr>
<th>velocity ( x_7 )</th>
<th>[10, 20] m/s</th>
<th>[20, 30] m/s</th>
<th>[30, ( \infty )] m/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>controller gains</td>
<td>( k_1 = 0.4 )</td>
<td>( k_1 = 0.5 )</td>
<td>( k_1 = 0.6 )</td>
</tr>
<tr>
<td></td>
<td>( k_2 = 1.5 )</td>
<td>( k_2 = 2 )</td>
<td>( k_2 = 2.5 )</td>
</tr>
</tbody>
</table>
Black lines: possible trajectories.
- Dark gray area: old technique; light gray area: new technique.
- Verification of safety only achieved by new technique.
- Computation time 38 s on an Intel i7 Processor with 6GB memory in MATLAB.
Other Advances for Hybrid Reachability Analysis

- Abstracting hybrid dynamics to uncertain linear dynamics. Allows verification of a phase-locked loop in the time of a few simulations [Althoff et al. 2011].
- Tightening the reachability results of linear system with uncertain parameters [Althoff, Krogh 2010].
- Introduction of zonotope bundles to mitigate shortcomings of zonotopes [Althoff, Krogh 2011].

Applications: phase-locked loop, RLC-circuits, autonomous cars, automotive powertrain, collision avoidance at intersections.
Embedded Control Systems

**Design Flow**
- requirements
- algorithms
- detailed design
- code

**Challenges**
- consistency
- correctness
- complexity
- coverage
- run-time correctness

**CMACS Research**
- requirements reconstruction
- analysis of hybrid systems
  - theorem proving
  - compositionality
  - reachability
  - statistical model checking
- code verification
  - abstract interpretation
  - analysis-aware design
  - run-time verification
Embedded Systems: Future Research Directions

- scalability for more complex systems
- compositional methods for hybrid systems
- advancing probabilistic/statistical methods
- integrated methods (theorem proving, model checking, abstract interpretation, probabilistic approaches)
- abstractions for real systems
- industry-scale case studies