Specification, Design and Verification of Distributed Embedded Systems

Mani Chandy    John Doyle    Richard Murray (PI)
California Institute of Technology

Eric Klavins    Pablo Parrilo
U. Washington   MIT

Project Overview: S5
June 2 2009
Caltech/MIT/UW V&V MURI Team

Principal Investigators

• Mani Chandy (Caltech CS)
• John Doyle (Caltech CDS)
• Gerard Holzmann (JPL CS)*
• Eric Klavins (U. Washington, EE/CS)
• Richard Murray (Caltech CDS)
• Pablo Parrilo (MIT EE)

Partners

• Air Force Research Laboratory: IF, MN, VA, VS
• Boeing Corporation - Systems of Systems Integration
• Honeywell Corporation - Guidance and Control
• Jet Propulsion Laboratory (JPL) - Laboratory for Reliable Software (LARS)
Problem Focus

Verification and validation of multi-agent systems operating in extreme environments

- State space and state transitions have continuous and discrete components
- Communication between agents may be continuous (analog) or discrete (messages);
- Messages may be delayed, lost, or overtaken
- Environment may be stochastic and/or adversarial

(Steve Drager’s terminology of research quadrant: Transformational Technology)
Outcomes

Verification and validation of multi-agent systems

• **Theory**
  • Game theory; stochastic processes; hybrid systems; optimization using SoS

• **Tools**
  • Model checkers (SPIN); theorem provers (PVS); optimization and algebraic packages

• **V&V methodologies**
  • Exploiting concurrent architectures; libraries of PVS theorems; modular designs of distributed system

• **Educational material**
  • Online courses; tools workshops
Overview of Applications of Theorem Provers

Using PVS and hybrid automata

- State space and state transitions have continuous and discrete components
- Communication between agents may be continuous (analog) or discrete (messages);
- Messages may be delayed, lost, or overtaken
- Environment may be stochastic and/or adversarial
Periodically Controlled Hybrid Automata (PCHA)

State space and transitions have discrete and continuous components

PCHA setup

- Continuous dynamics with piecewise constant inputs
- Controller executes with period \( T \in [\Delta_1, \Delta_2] \)
- Input commands are received asynchronously
- Execution consists of trajectory segments + discrete updates
- Verify safety (avoid collisions) + performance (turn corner)

Proof technique: verify invariant (safe) set via barrier functions

- Let \( I \) be an (safe) set specified by a set of functions \( F_i(x) \geq 0 \)
- Step 1: show that the control action renders \( I \) invariant
- Step 2: show that between updates we can bound the continuous trajectories to live within appropriate sets
- Step 3: show progress by moving between nested collection of invariant sets \( I_1 \rightarrow I_2 \), etc
State Space and Transitions have Discrete and Continuous Components

System consists of Agents

System executes in Rounds
- Each agent stores some value
- Reads the current value of some other active agents
- Computes a new value using some function

State of the Multi-Agent System

June 2 2009

Caltech/MIT/UW V&V MURI: S5
Communication Medium may be Faulty

Broadcast Channel

• Agents: send, receive
• Internal Actions: duplicate, drop

Assumptions

• Messages are eventually dropped or received
• Total number of copies is finite
• For all $i, j$: if $j$ sends infinitely often then $i$ receives messages from $j$ infinitely often
Receding Horizon Control for Linear Temporal Logic

Find planner (logic + path) to solve general control problem

\[(\varphi_{\text{init}} \land \Box \varphi_{\text{e}}) \implies (\Box \varphi_{\text{s}} \land \Diamond \varphi_{\text{g}})\]

- \(\varphi_{\text{init}} = \) init conditions
- \(\varphi_{\text{e}} = \) envt description
- \(\varphi_{\text{s}} = \) safety property
- \(\varphi_{\text{g}} = \) planning goal

• Can find automaton to satisfy this formula in \(O((nm|\Sigma|^3) \text{ time})\) (!)

Basic idea

• Discretize state space into regions \(\{\mathcal{V}_i\} + \) interconnection graph
• Organize regions into a partially ordered set \(\{\mathcal{W}_i\}; \mathcal{W}_j \preceq_{\varphi_{\text{g}}} \mathcal{W}_i \implies \) if state starts in \(\mathcal{W}_i\), must transition through \(\mathcal{W}_j\) on way to goal
• Find a finite state automaton \(A_i\) satisfying

\[\Psi_i = ((v \in \mathcal{W}_i) \land \Phi \land \Box \varphi_{\text{e}}) \implies (\Box \varphi_{\text{s}} \land \Diamond (v \in \mathcal{W}_{g_i}) \land \Box \Phi)\]

- \(\Phi\) describes receding horizon invariants (eg, no collisions)
- Automaton states describe sequence of regions we transition through; \(\mathcal{W}_{g_i} \preceq_{\varphi_{\text{g}}} \mathcal{W}_i\) is intermediate (fixed horizon) goal
- Planner generates trajectory for each discrete transition
- Partial order condition guarantees that we move closer to goal

Properties

• Provably correct behavior according to spec
Applying Temporal Logic and Hybrid Automata to Continuous Games

State: \((x, y)\): Ben's state is \(x\); Sam's state is \(y\)
Applying Temporal Logic and Hybrid Automata to Continuous Games

How do you model continuous and discrete movements as hybrid automata, and map hybrid automata to PVS?

• Action is a trajectory over a finite time, and specified by a predicate on the trajectory.
Overview of Applications of Algebra

Using polynomials, semi-definite programming, and stochastic processes

• State space and state transitions have continuous and discrete components
• Communication between agents may be continuous (analog) or discrete (messages);
• Messages may be delayed, lost, or overtaken
• Environment may be stochastic and/or adversarial
Complex Stochastic Networks

Task 1: Formal specification of Network Control Algorithms

- Complex networked systems require a *domain specific language* for their specs.
- *Embedded Graph Grammars (EGGs)* allow specification of networked systems by describing changing network topology via local rewrite rules.

Task 2: Reasoning about complex stochastic processes

- Many complex networked systems can be characterized by stochastic processes with enormous state spaces.
- Verification of such systems by exhaustive search is impossible.
- The space of all stochastic processes can be given a metric, so that the Wasserstein distance between processes, can be determined.

Illustrative Example: Find k to minimize the Wasserstein distance between the following processes.

Results: Embedded Graph Grammars

- Formal definition of the *embedded graph grammar* specification language
- Examples of complex systems specified and proved correct.

Results: Verification of Stochastic Processes

- New efficient algorithms for computing an approximation of the Wasserstein distance from data and/or large models.
- Model reduction methods are based on finding simple models that explain complex data.
- Robustness of temporal logic statements to model structure investigated.
Relaxations for Reachability and Word Problems

Goal: efficient tests
- Can we transition between two states, using only moves from a given finite set? (word problem for finite semi-Thue systems, generally undecidable)
- Direct applications to graph grammars, infinite graph reachability, Petri nets, etc.
- What are the obstructions to reachability?

Approach: symbolic-numeric
- Relaxations: commutative and/or symmetric versions
- Algebraic reformulation in terms of ideal membership and nonnegativity
- Convexity enables duality-based considerations

Results to date
- Characterization in terms of polynomial identities and nonnegativity constraints
- Yields a hierarchy of linear programming (LP) conditions
- Zero-to-all reachability equivalent to finitely many point-to-point problems
- Progress towards higher-order relaxations, that do not rely on commutativity assumptions

Analysis via Non-monotonic Lyapunov Functions
Ahmadi, Parrilo (MIT)

Goal: stability and performance
• Traditional Lyapunov-based analysis relies on monotone invariants (e.g., energy)
• This often forces descriptions requiring high algebraic complexity
• Is it possible to relax the monotonicity assumption?

Approach: convexity-based
• Require nonnegativity of linear combinations of time derivatives
• Algebraic reformulation in terms of polynomial nonnegativity
• Yields tractable conditions, verifiable by convex optimization

Results to date
• Convexity-based conditions, checkable by SOS/semidefinite programming
• Easy to apply, more powerful than standard conditions
• Connections with other techniques (e.g., vector Lyapunov functions)
• Many extensions to discrete/continuous/hybrid/switched, etc.

Problem Focus

Verification and validation of multi-agent systems operating in extreme environments

- State space and state transitions have continuous and discrete components
- Communication between agents may be continuous (analog) or discrete (messages);
- Messages may be delayed, lost, or overtaken
- Environment may be stochastic and/or adversarial
Outcomes

Verification and validation of multi-agent systems

- **Theory**
  - Game theory; stochastic processes; hybrid systems; optimization using SoS

- **Tools**
  - Model checkers (SPIN); theorem provers (PVS); optimization and algebraic packages

- **V&V methodologies**
  - Exploiting concurrent architectures; libraries of PVS theorems; reliable component architecture (William McKeever, John Scott: S3)

- **Educational material**
  - Online courses; tools workshops
Number of lines of code in space flight is increasing exponentially. Next Mars flight may have over 2 million lines.

Moore’s law appears to be hitting a wall.

Contributions: Good ways to use networks of multi-core computers for model checking and testing.
S3 Concepts for Concurrent Computing

System A

Thread
Thread
Thread
Shared input/output queue
Multi-threaded sensor
Communication layer

System X

Communication layer
Thread
Thread
Thread
Shared input/output queue
Multi-threaded processing component
Multi-threaded responder
S3 Concepts for Concurrent Computing

Control Functions
- Registration
- Security
- Updates
- Load Balancing
- Snapshot
- Logging
- Clocking

Pattern Detection
- Pattern Detection
- Anomaly Detection
- Aberrant Behavior
- Load Balancing
- Snapshot
- Logging
- Clocking

Common Network Usage
- Web Service Processing
- Email Transmission
- Aberrant Behavior
- Load Balancing
- Snapshot
- Logging
- Clocking

Transformations
- Regex/Xpath Filter
- Time Window Calculations
- Aberrant Behavior
- Load Balancing
- Snapshot
- Logging
- Clocking

June 2 2009
Caltech/MIT/UW V&V MURI: S5
S3 Concepts for Concurrent Computing

Example:

Consensus

Consensus: Associative, Commutative, Idempotent Operators
Shared State

Program transformation

Consensus: Associative, Commutative, Idempotent Operators
Message Passing

Consensus: Message-Passing $max$, $min$, $gcd$, … in PVS

Agents coded in Java

Hand transformation from PVS to Java

Refinement using PVS to produce Reliable Java Code
Examples of mapping from PVS to Implementations

Mobile agent

PVS – \( f(\text{left}, \text{right}): \frac{(\text{left} + \text{right})}{2} \)

Java – \( f(\text{left}, \text{right}): \frac{(\text{left} + \text{right})}{2} \)

Consensus

PVS – \( f(x, m): \max (x, m) \)

Java – \( f(x, m): \max (x, m) \)
Case Study: Verification for Autonomous Driving

Goal: safe vehicle operation in multithreaded environment

• Vehicle operation controlled by networked interface; responsible for fail safe operation
• Requires careful reasoning about message passing, external events, internal failures
• Asynchronous operations (message passing, failures, environment) complicate verification

Approach: temporal logic + SOS

• Formulate control goal using temporal logic specification w/ continuous+ discrete vars

Results to date

• Verification of low level state machines and messaging with TLA+, TLC
• Verification of periodic controller using PCHA formulation
• Reformulation of logic (traffic) planner using RHTL

Program synergy

• Implementation and testing supported through DARPA 6.2 funds (18 FTE effort)
Transition Strategy

Toolbox development

- Develop and disseminate algorithms via publicly available toolboxes

Annual workshops/short courses

- Model after mutools workshops developed by Balas, Doyle and Packard
- Provide opportunity for researchers to learn about the toolboxes developed under the MURI and apply the design tools to simple problems
- Provide forum for feedback to MURI team and discussion of needed tools
- Develop new courses and new course materials that can be used to teach students the required background to be effective practitioners and researchers in distributed embedded systems

Personnel exchange

- Student internships at AFRL labs and industry
- Industry visitors: eg, Sonja Glavaski from Honeywell spending 1 month at Caltech

Additional workshops and tutorials

- CDC 2006: High Confidence Embedded Systems (Klavins and Murray)
- Hands-on workshop @ Caltech, 16-17 Sep 09 - PVS, LTV, PHAVer and more
  - Send e-mail to murray@cds.caltech.edu if interested in attending
Specification, Design and Verification of Distributed Embedded Systems
Caltech/MIT/UW, Murray (PI)/Chandy/Doyle/Klavins/Parrilo

FUNDING ($K)—Show all funding contributing to this project

<table>
<thead>
<tr>
<th></th>
<th>FY06</th>
<th>FY07</th>
<th>FY08</th>
<th>FY09</th>
<th>FY10</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFOSR Funds</td>
<td>417</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Boeing</td>
<td>310</td>
<td>390</td>
<td>390</td>
<td>370</td>
<td>[390]</td>
</tr>
<tr>
<td>DARPA GC</td>
<td></td>
<td></td>
<td>390</td>
<td>370</td>
<td>1200</td>
</tr>
</tbody>
</table>

OBJECTIVES
- Specification language for continuous & discrete control policies, communications protocols and environment models (including faults)
- Analysis tools to reason about designs and provide proof certificates for correct operation
- Implementation on representative testbeds

ACCOMPLISHMENTS/RESULTS
- Embedded graph grammars for cooperative control
- Lyapunov-based verification of temporal properties
- Stochastic games using semidefinite programming
- Tools for converting goal networks to hybrid FSM
- Applications examples with DARPA GC + JPL

TRANSITIONS
- Application to autonomous driving (DGC07)

STUDENTS, POST-DOCS
2006-08: 12 graduate students, 4 postdocs, 4 undergraduates

LABORATORY POINT OF CONTACT
Dr. Siva Banda, AFRL/RBCA, WPAFB, OH

APPROACH/TECHNICAL CHALLENGES
- Specification and reasoning using graph grammars
- Sum of squares analysis for certificates, invariants
- Extensions to probabilistic, adversarial and networked operations
Problem Focus

Verification and validation of multi-agent systems operating in extreme environments

• State space and state transitions have continuous and discrete components
• Communication between agents may be continuous (analog) or discrete (messages);
• Messages may be delayed, lost, or overtaken
• Environment may be stochastic and/or adversarial
Outcomes

Verification and validation of multi-agent systems

• **Theory**
  • Game theory; stochastic processes; hybrid systems; optimization using SoS

• **Tools**
  • Model checkers (SPIN); theorem provers (PVS); optimization and algebraic packages

• **V&V methodologies**
  • Exploiting concurrent architectures; libraries of PVS theorems; reliable component architecture

• **Educational material**
  • Online courses; tools workshops; transition strategy
Backup Slides
Partial orders and decentralized control

Goal: understand information flow
- A new framework to reason about information flow in terms of partially ordered sets (posets).
- What are the structures amenable to decentralized control design?

Approach: incidence algebras
- Posets and incidence algebras
- Abstract flow of information, generalize notions of causality
- Yields convexity of the underlying control problems. Relations with quadratic invariance.

Results to date
- Generalizes sequential and partially nested structures (e.g., leader-follower)
- Convex characterization of poset-preserving controllers, via Youla
- Captures the right level of abstraction, rich algebraic and combinatorial tools
- Extensions to more complicated situations, via Galois connections

Stability in Hybrid Automata

\[ s, s' \]: states of the automaton

\[ c \rightarrow \]: binary relation between states;

\[ s \xrightarrow{c} s' \]: there exists a sequence of actions that takes the automaton from \( s \) to \( s' \)

\[ B_l(s^*) \]: the ball with diameter \( l \) and center \( s^* \)

\[ B_\delta(s^*) \]

\[ \forall \epsilon > 0 \; \exists \delta > 0 : \]

\[ \forall s \in B_\delta(s^*), \; s' : \]

\[ s \xrightarrow{c} s' \Rightarrow s' \in B_\epsilon(s^*) \]
Stability in Hybrid Automata

\( s, s' \) : states of the automaton

\( c \rightarrow \) : binary relation between states;

\( s \xrightarrow{c} s' \) : there exists a sequence of actions that takes the automaton from \( s \) to \( s' \)

\( B_l(s^*) \) : the ball with diameter \( l \) and center \( s^* \)

\[ \forall \varepsilon > 0 \exists \delta > 0 : \square \left( B_\delta \Rightarrow \square B_\varepsilon \right) \]
**Sufficient Condition for Stability**

\[ T : \text{totally ordered set} < \]

\[ f : S \to T \]

\[ L_p : \{ s | f(s) < p \} \]

\[ a : \text{action} \]

\[
\forall \epsilon \geq 0 : \exists p \in T : L_p \subseteq B_\epsilon(S^*) \quad \wedge \\
\forall p \in T : \exists \epsilon \geq 0 : B_\epsilon(S^*) \subseteq L_p \quad \wedge \\
\forall s, a : f(\text{transition}(s, a)) \leq f(s)
\]
Outline: PVS Proof

Condition

\[ \forall \epsilon \geq 0 : \exists p \in T : L_p \subseteq B_\epsilon(S^*) \quad \wedge \quad (1) \]
\[ \forall p \in T : \exists \epsilon \geq 0 : B_\epsilon(S^*) \subseteq L_p \quad \wedge \quad (2) \]
\[ \forall s, a : f(\text{transition}(s, a)) \leq f(s) \quad (3) \]

1. Fix \( \epsilon \)

2. From (1): \[ \exists p : L_p \subseteq B_\epsilon(S^*) \]

3. From (2): \[ \exists \eta : B_\eta(S^*) \subseteq L_p \]

4. From steps 2, 3: \[ B_\eta(S^*) \subseteq B_\epsilon(S^*) \]

5. Result follows by induction on state transitions from states in \( B_\eta(S^*) \)
Program Thrusts

Specification and Reasoning Using Graph Grammars
• Build on Klavins’ Computation and Control Language (CCL) & SPIN (Holzmann)
• Use graph grammars to define interaction rules and reason about them

Sum of Squares Techniques (SOS)
• Unified framework for finding invariants and proof certificates for nonlinear and hybrid systems

Extensions
• Probabilistic techniques (specification + algorithms)
• Adversarial settings (including security issues)

Testbeds
• U. Washington Programmable Parts testbed
• Caltech Multi-Vehicle Wireless Testbed (hardware + sims)
• Alice: 2005 and 2007 DARPA Grand Challenge entry

• Allow temporal logic statements and verification of semi-algebraic conditions to coexist
• Develop design specification and design language plus reasoning tools
Proving Convergence

Show a \textit{(Lyapunov)} function that is non-increasing along all executions of the system.

Show a collection of Sets \{R_i\}_{i \in \mathbb{N}} satisfying

- C1. Monotonicity
- C2. Initial
- C3. Stability
- C4. Progress