# 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**

**AFOSR Dynamics and Control Meeting**
**5 August 2008**

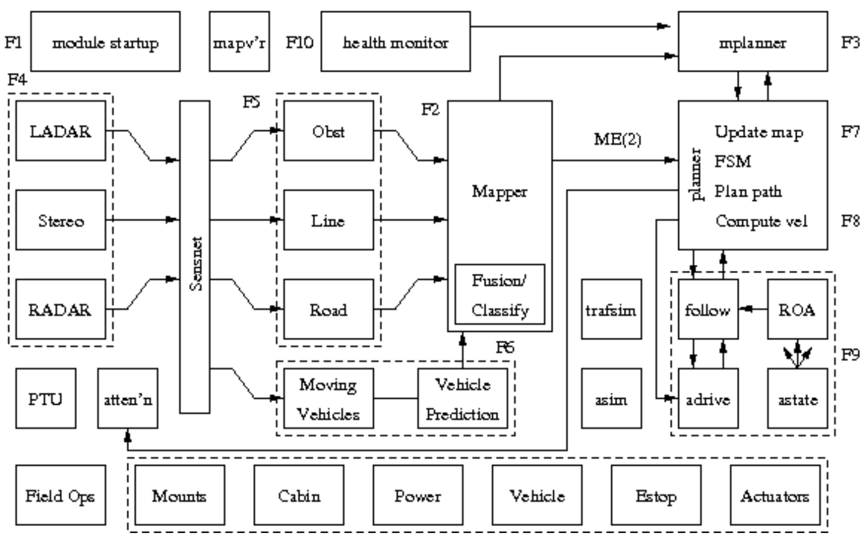
# Motivating Example: Alice (DGC07)

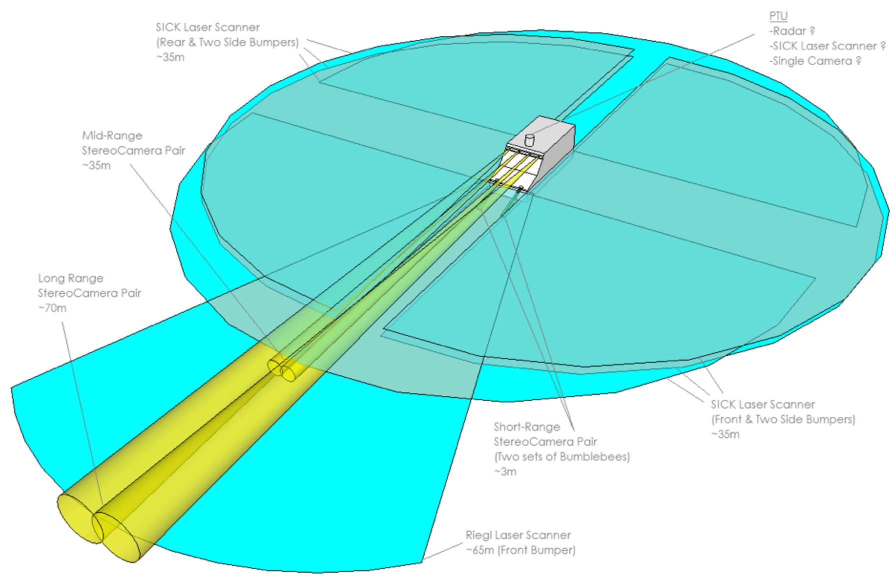
## Alice

- 300+ miles of fully autonomous driving
- 8 cameras, 8 LADAR, 2 RADAR
- 12 Core 2 Duo CPUs + Quad Core
- ~75 person team over 18 months

## Software

- 25 programs with ~200 exec threads
- 237,467 lines of executable code

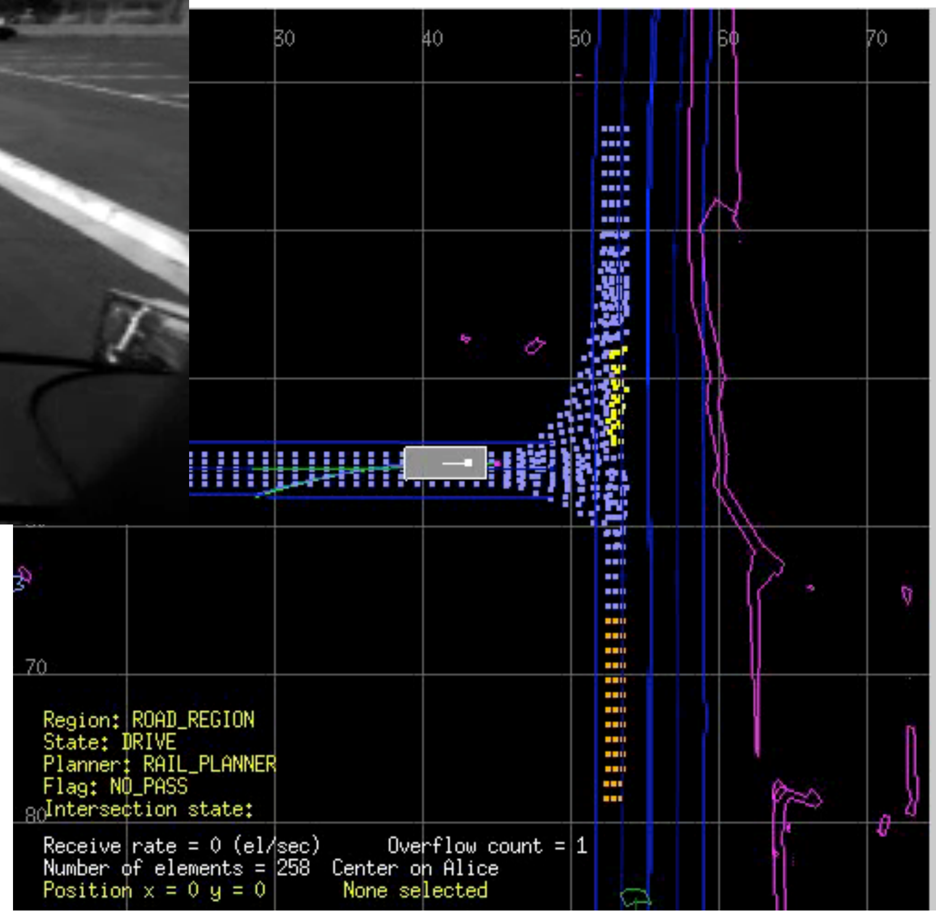


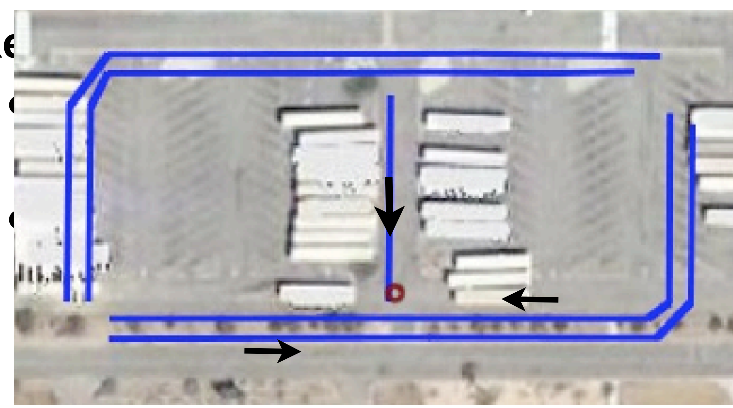



# 2007 National Qualifying Event

**Merging test**

- 10-12 cars circling past inters'n
- Count "perfect runs" in 30 min



# MURI Goals + Talk Outline

**Overall Goal:** Develop methods and tools for designing control policies, specifying the properties of the resulting distributed embedded system and the physical environment, and proving that the specifications are met

*Specification*

- How does the user specify---in a single formalism---continuous and discrete control policies, communications protocols and environment models (including faults)?

*Design and reasoning*

- How can engineers reason that their designs satisfy the specifications?
- In particular, can engineers reason about the performance of computations and communication, and incorporate real-time constraints, dynamics, and uncertainty into that reasoning?

*Implementation* (joint with Boeing, JPL, AFRL)

- What are the best ways of mapping detailed designs to hardware artifacts, running on specific operating systems? What languages are suitable for specifying systems so that the specifications can be verified more easily?

**Outline**

I. Embedded Graph Grammars (EGGs)
II. SOS extensions for hybrid and networked systems
III. Combining temporal logic and dynamics
IV. Reasoning about stochastic & adversarial environments
V. Summary

Klavins and M
IEEE PC, 2004

# CCL: Computation and Control Language

*Formal Language for Provably Correct Control Protocols*

**Guarded command language:**

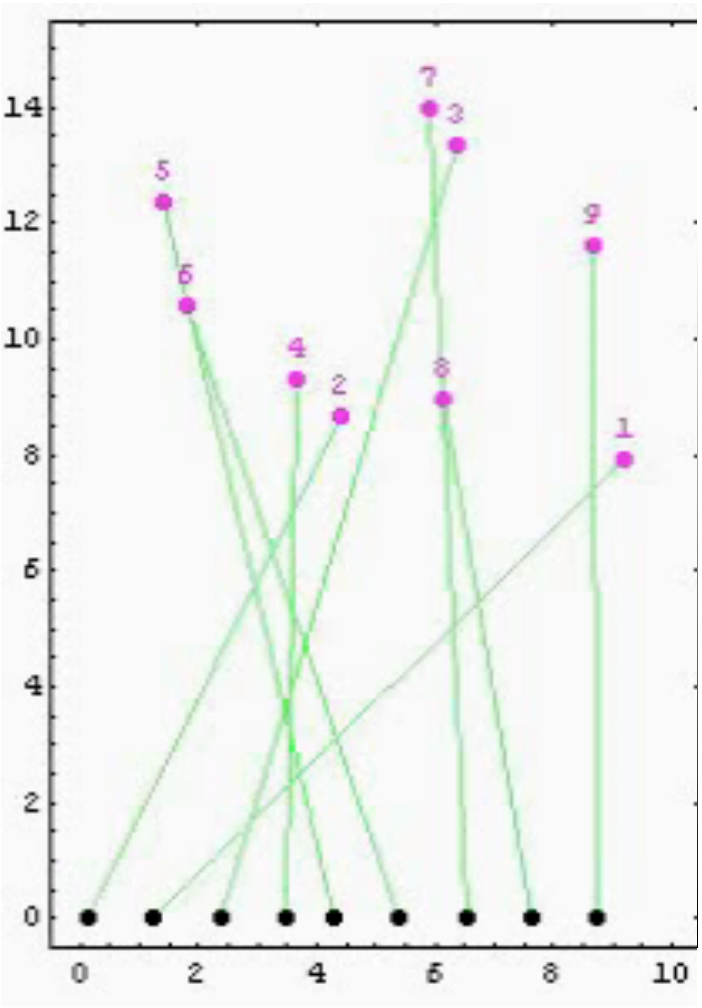
```
P(k1,k2) := {
  initializers
  guard1:rule1
  guard2:rule2
  ...
}
```

"soup" of guarded commands

composition = union

non-shared variables remain local to component programs

```
S(k1,k2):=P(k1,k2)⊕C(k1+1)
```




**Execution semantics and properties**

- Any rule whose guard is true can be executed at any time; no synchronization between agents
- Specify desired properties using temporal logic
  - $\square p$ ≡ **always** *p* (invariance)
  - $\lozenge p$ ≡ **eventually** *p* (guarantee)
  - $p \rightarrow q\ \mathcal{U}\ r$ ≡ *p* **implies** *q* **until** *r* (precedence)
  - $\square\lozenge p$ ≡ **always eventually** *p* (progress)
  - $\lozenge\square p$ ≡ **eventually always** *p* (stability)

# Embedded Graph Grammars (McNew et al)

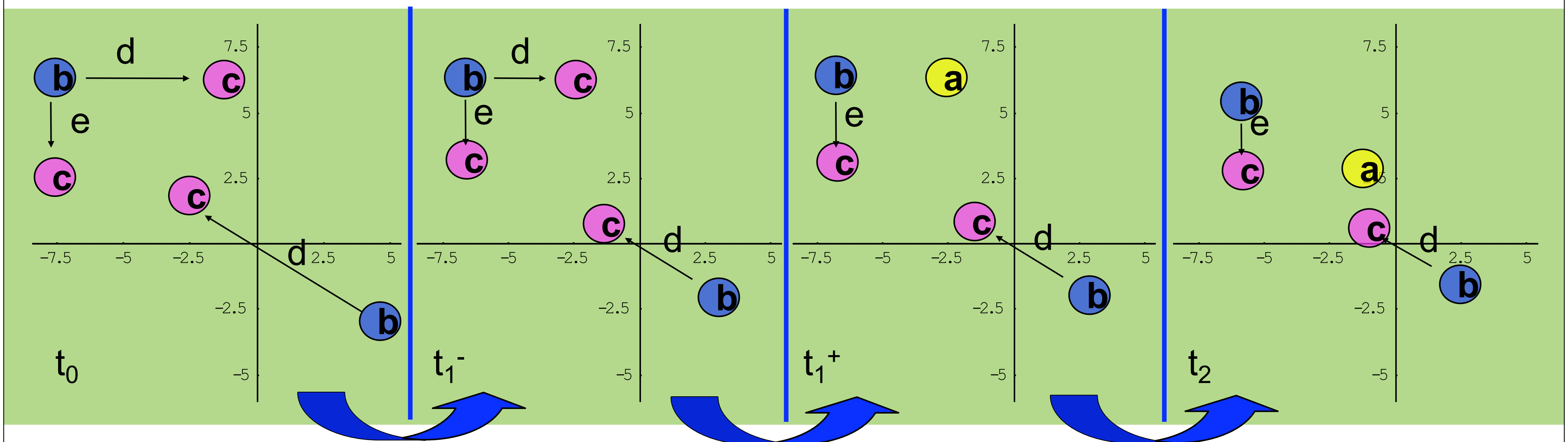
**Defn** An *embedded graph* is a tuple G = (V, E, z, e) such that

- V is a set of vertices (agents)
- E is a set of edges representing agents that can communicate with each other
- z is a set of vertex variables (properties; eg, location)
- e is a set of edge variables (properties; eg, relationship)

**Defn** An *embedded graph grammar* is a pair (F, u) where

- F is a set of local rules (for each agent)
- u is a set of local controllers (for each agent)

Design problem: find (F, u) such that the dynamics g(t) ∊ G have desired set of properties under asynchronous execution




Continuous flow via (local) control laws

Discrete, instantaneous (local) update to network topology

Continuous flow via (local) new control laws

# Lexicographically Ordered Lyapunov Functions

**Defn** Let $A \subset G$ be closed under a graph grammar $\Phi$ and let $\preccurlyeq$ be an ordering on $R^k$ with a unique zero element. A function $U: A \to R^k$ is a *discrete Lyapunov function* for the graph grammer $\Phi$ if

- $U(G) \succ 0$ implies at least one rule is applicable
- $U(G) = 0$ implies no rule is applicable
- When $U(G) \succ 0$, ever applicable rule decreases $U$

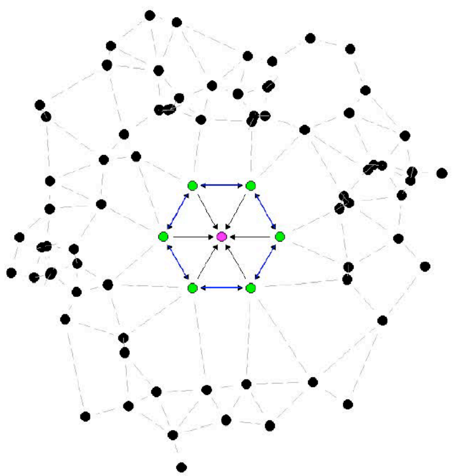
**Theorem** (McNew et al) Suppose $(G_0, \Phi)$ is a system, $P$ is a set of desired final graphs, $A$ is a set of $\Phi$ invariant graphs and $U$ is a discrete Lyapunov function so that $A \cap U^{-1}(0) \subset P$. If $G_0 \in A$, then every trajectory converges to a final graph in $P$.

**Defn** The *lexicographic ordering* $(R^n, \preccurlyeq)$ is defined as $(a_1, a_2, \cdots, a_n) \prec (b_1, b_2, \cdots, b_n)$ if $a_1 < b_1$ or there exists a $k$ such that $a_i = b_i$ for all $i \leq k$ and $a_{k+1} < b_{k+1}$.

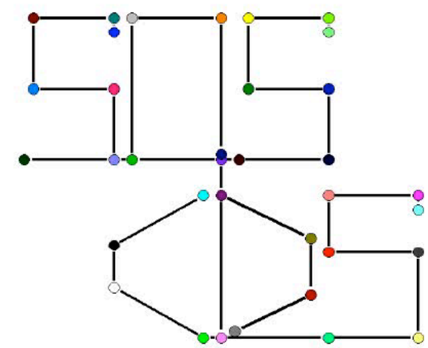
**Corollary** Suppose $\Phi_1$, $\Phi_2$ are two grammars with invariant sets $A$ and $B$ and discrete Lyapunov functions $U$ and $V$. If $A \cap B$ is closed under applications of rules in $\Phi_1 \cup \Phi_2$, and there exists a lexicographic order of elements of $U, V$ with respect to $\Phi_1 \cup \Phi_2$ then every trajectory converges to a final graph in $A \cap B \cap U^{-1}(0) \cap V^{-1}(0)$.

**Remark** Allows constructive techniques for combining basic behaviors...

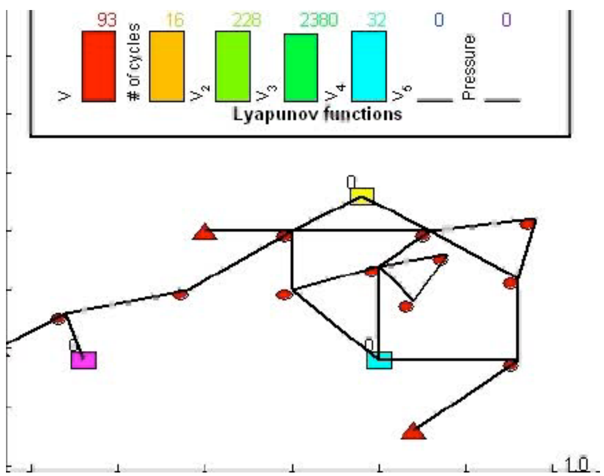
# Example Tasks



Triangulation: Vehicles achieve uniform coverage from arbitrary initial conditions (HSCC 07).



Reconfiguration: Vehicles change formations while maintaining network connectivity (ACC 08).




Load Balancing: Vehicles cover targets in equal numbers while maintaining connectivity (CDC 06).

- Each task <u>requires</u> mode-switching and communication.
- Solutions are completely decentralized.
- Each solution comes with safety and progress guarantees.
- More tasks and proof methodologies in J.M. McNew's thesis (Ph.D. in Sept. 2008).

# Proof Certificates for Stability and Sum of Squares

**Certifying stability for dynamical systems**

- Given a (controlled) dynamical system,

$$\dot{x} = f(x, \mu)$$

  determine whether the system is stable and estimate the region of attraction
- Traditional technique: find a Lyapunov function that serves as a "proof certificate" for stability and gives a set that is guaranteed to be in region of attraction

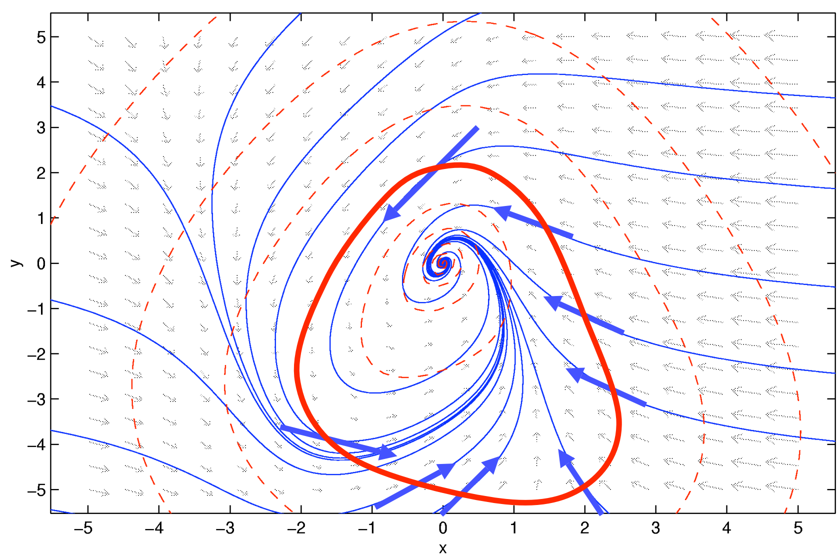
$$V(x) \succ 0, \quad \frac{\partial V}{\partial x} f(x) \preceq 0, \quad x \in \mathcal{S}$$

**Sum-of-squares approach**

- Approximate the Lyapunov certificate with a sum of squares; solve a convex programming problem
- Constructive algorithm for finding Lyapunov functions; rapidly becoming a standard computational approach

**Extensions**

- SOSTOOLS – MATLAB package for finding SOS certificates
- Proof certificates for hybrid dynamica systems (barrier certificates)
- Incorporating stochastic inputs (noise, disturbances)
- Current work: incorporating temporal logic specifications (focus of MURI)



# Non-Monotonic Lyapunov Functions (Ahmadi et al)

**Goal: easier conditions for stability and performance of hybrid systems**

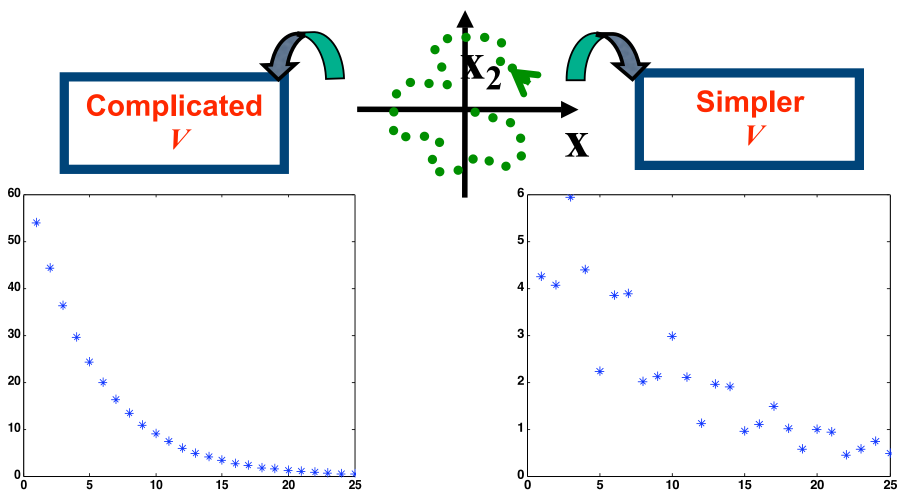
- Traditional Lyapunov-based analysis relies on monotone invariants (e.g., energy)
- This often forces descriptions requiring high algebraic complexity
- Is it possible to relax the monotonicity assumption?

**Thm** Consider a discrete-time linear system $x_{k+1} = f(x_k)$. If there exists a scalar $\tau \geq 0$ and a continuous radially unbounded function $V$ such that $V(x) > 0 \ \forall\, x \neq 0$, $V(0) = 0$ and $\tau(V_{k+2} - V_k) + V_{k+1} - V_k < 0$ then the origin is global asymptotically stable.

**Pf** Show that for any $V_k$, either $V_{k+1}$ or $V_{k+2}$ is less than $V_k$ and construct a converging subsequence

**Remarks**

- Can reformulate results as 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.



# Formal Reasoning for Dynamics + Protocols

**Asynchronous Iterative Processes (Tsitsiklis, 1987)**

- $S$ = states, $S_0$ = starting states (mixed continuous and discrete)
- $A$ = set of actions, $E$ = enabling predicate, $T$ = transition function
- $E(s, a)$ holds if an only if the transition labeled by $a$ can be applied to $s$
- $s' = T(a, s)$ if $a$ is enabled at $s$ or $s' = s$
- $d$ = distance function on $S^* \subseteq S$: $\forall\, s \in S^*, s' \in S^*, d(S^*, s) > d(S^*, s')$

**Defn** Let $\mathcal{A} = (S, A, S_0, E, T)$ be an automaton, $s^*$ a state in $S$ and $d$ a distance function for $s^*$. The automaton $\mathcal{A}$ is $(s^*, d)$-*stable* if $\forall\, \varepsilon > 0, \exists\, \delta > 0$ such that $\forall\, s \in S, a^\omega \in A^\omega, n \in \mathbb{N}, s \in B_\delta(s^*) \Rightarrow \mathrm{Trans}(s, a^\omega, n) \in B_\varepsilon(s^*)$.

**Thm** *Let $S^*$ be a nonempty subset of $S$ and let $d$ be a distance function for $S^*$. Suppose there exists a totally ordered set* $(\mathrm{T}, <)$ *with sublevel sets $L_p$ and a function $f: S \to \mathrm{T}$ that satisfies the following conditions*

- $\forall\, \varepsilon \geq 0, \exists\, p \in \mathrm{T}$ *such that* $L_p \subseteq B_\varepsilon(S^*)$
- $\forall\, p \in \mathrm{T}, \exists\, \varepsilon \geq 0$ *such that* $B_\varepsilon \subseteq L_p$
- $\forall\, s \in S, a \in A, E(a, s) \Rightarrow f(T(a, s)) \leq f(s)$

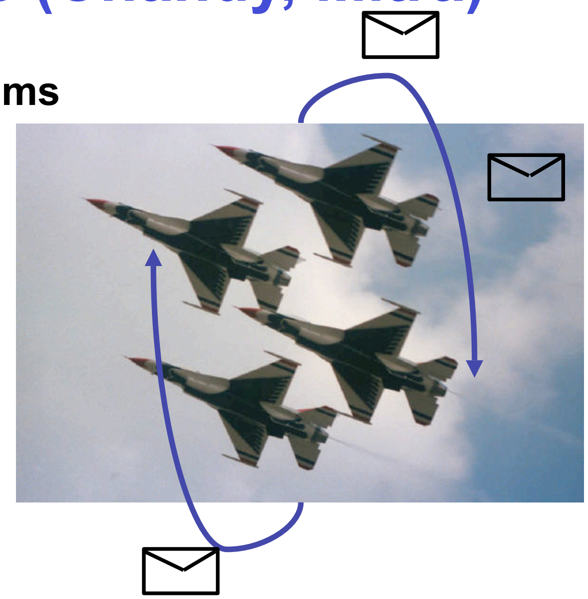
Proof via PVS metatheory
⇒ allows reasoning in theorem-proving environment

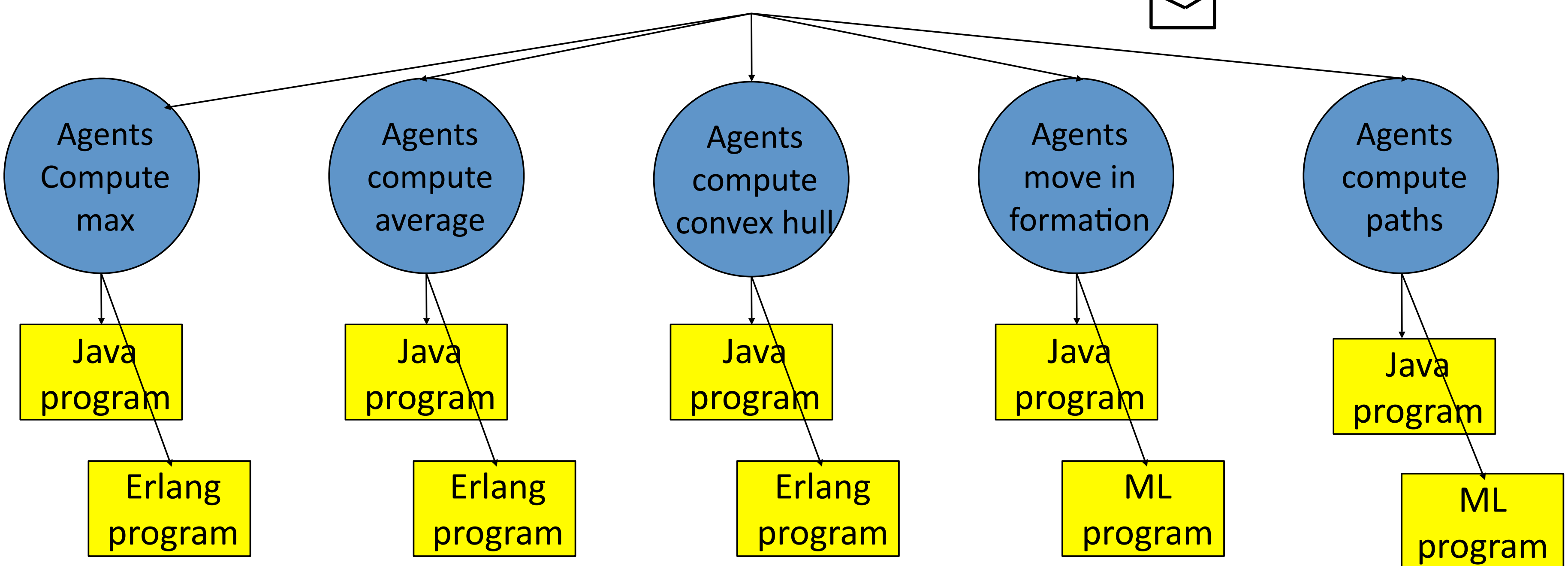
*Then $\mathcal{A}$ is $(S^*, d)$-stable*

# Distributed Control with Messages (Chandy, Mitra)

**Convergence verification for partially synchronous systems**

- Mechanical transformation from shared-memory algorithms to message-passing algorithms
- Use Tsitsiklis formalism + timed I/O automata (TIOA) to show that if an algorithm converges using shared mem-ory, it converges using message passing
  - Relatively modest assumptions on messages
- Show that programs (designs) are within the specified class of algorithms ⇒ can robustly distributed





# Stochastic Systems

**Increased interest in stochastic behavior**

- Need to eason about probability of events and stochastic performance measures
- Formal reasoning systems allow non-determinism (in events), but often don't include random variables and processes

**Model reduction using Wasserstein pseudometrics (Thorsley et al)**

- Define a formal distance between stochastic processes
- Enables reasoning about complicated systems by producing simpler models
- Details: Thorsely & Klavins (ACC, 2008)

**Polynomial stochastic games via sum of squares optimization (Shah et al)**

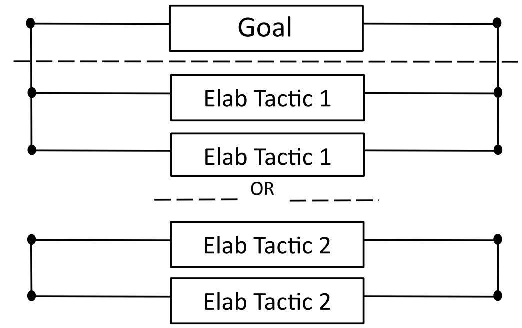
- Generalize Markov decision processes to game theoretic settings
- Can show that equilibria for certain classes of two-player, zero-sum, infinite strategy games can be solved via SDPs (eg, SOS-tools)
- Provides possible method to extend current results to adversarial environments
- Details: Shah & Parrilo (CDC, 2008)

http://www.cds.caltech.edu/~murray/VaVMURI

# Implementation Tools

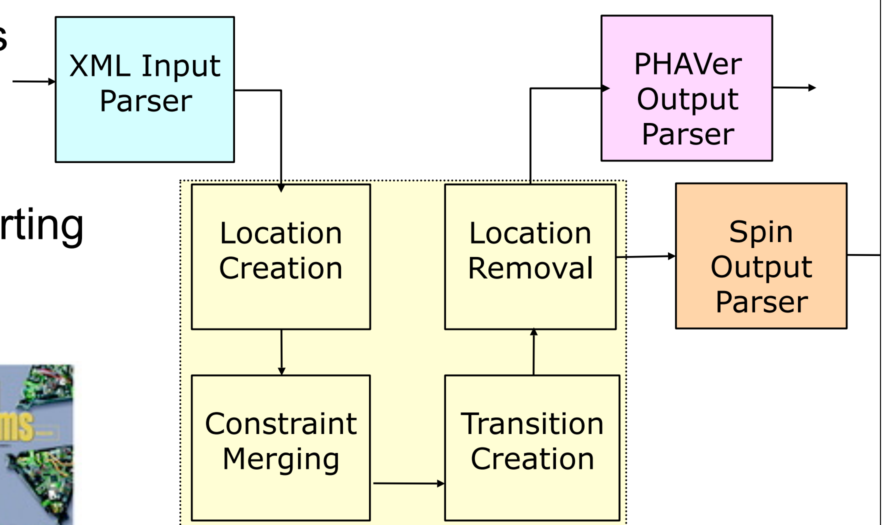
**Mission Data System (MDS) → Hybrid Automata**

- Conversion of goal network to hybrid automata that can be verified using PHAVer, SPIN, etc
- Joint work with JPL, applying to Titan mission



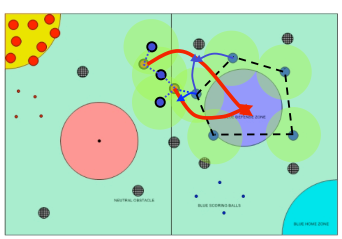

**PVS metatheory for asynchronous iterative processes**

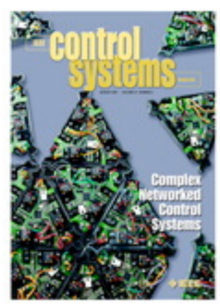
- “Library” for reasoning about stability in PVS
- Being used for verifying multi-robot protocols




**Applications to Alice, MVWT**

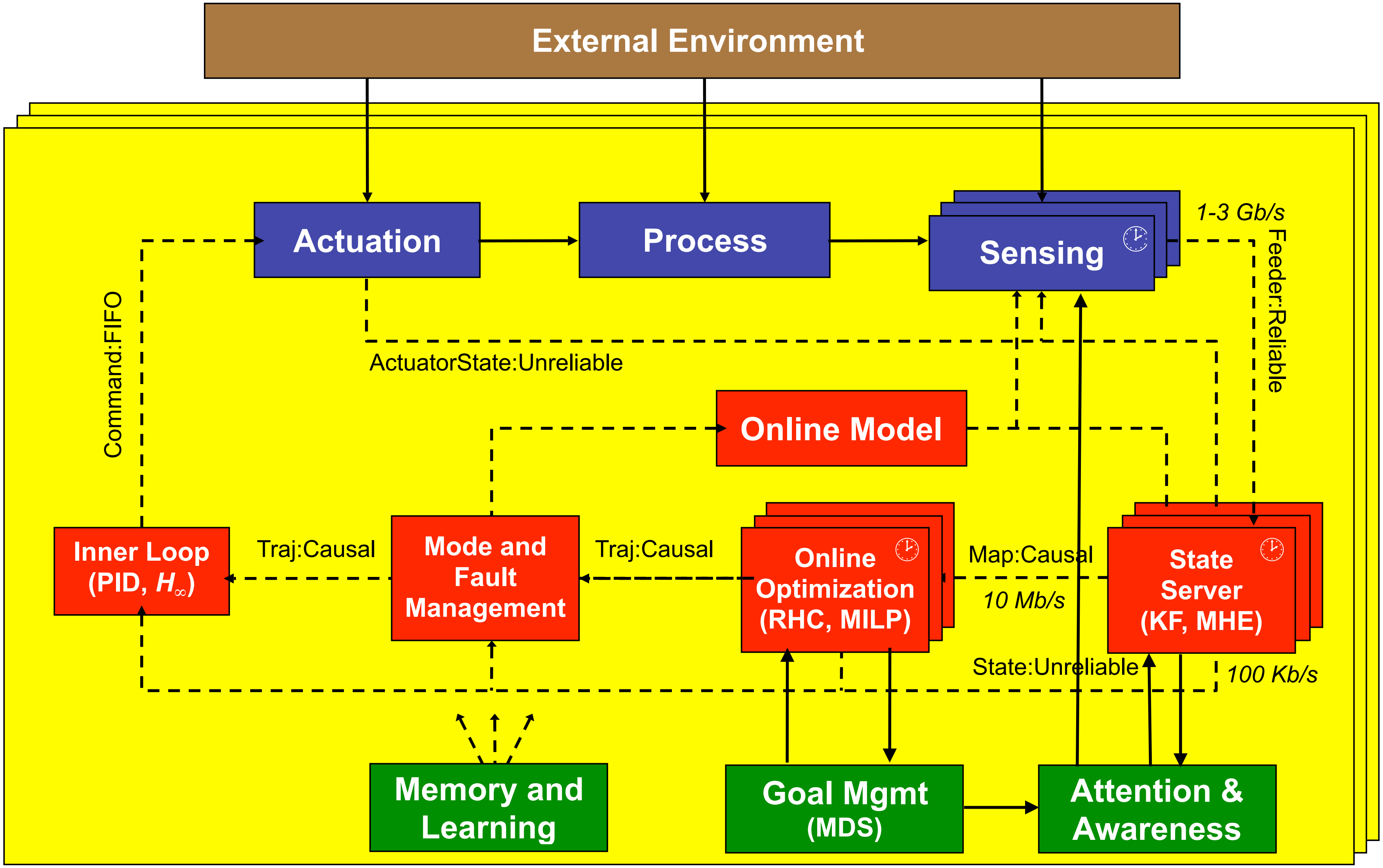
- Applying tools to verify behavior of Alice (starting with fixing DGC07 failure mode!)



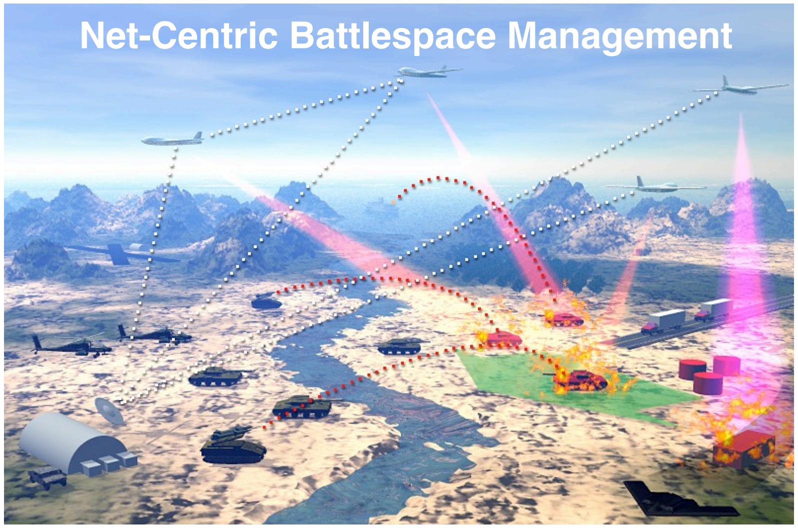


# Networked Control Systems

(following P. R. Kumar)



# Specification, Design and Verification of Distributed Embedded Systems
# Caltech/MIT/UW, Murray (PI)/Chandy/Doyle/Klavins/Parrilo




**Long-Term PAYOFF:** Rigorous methods for design and verification of distributed systems-of-systems in dynamic, uncertain, adversarial environments

**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

**APPROACH/TECHNICAL CHALLENGES**

- Specification and reasoning using graph grammars
- Sum of squares analysis for certificates, invariants
- Extensions to probabalistic, adversarial and networked operations

**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

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

| | FY06 | FY07 | FY08 | FY09 | FY10 |
|---|---|---|---|---|---|
| AFOSR Funds | 417 | 1000 | 1000 | 1000 | 1000 |
| Boeing | 310 | 390 | 390 | 390 | 390 |
| DARPA GC | | 1200 | | | |

**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