convergence verification

synchrony to partial synchrony

Sayan Mitra

mitras@crhc.uiuc.edu

joint work with Mani Chandy and Concetta Pilotto

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Computer Engineering Seminar Series 2008

ECE Illinois
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

consensus in distributed systems

system
n agents; agent i starts with value s_i

goal
agent i *converges* to value s_i^*

applications
consensus
pattern formation
flocking

consensus in distributed systems

system
n agents; agent i starts with value s_i

goal
agent i *converges* to value s_i^*

applications
consensus
pattern formation
flocking

ECE Illinois
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

synchronous communication model

system executes in rounds $\cdots$

- in each round k, a subset (e.g., $\{1, 2, 4\}$), of agents are *active*
- each active agent read the current values of the other active agents to compute their new values
  - $s_2(k+1) = f_2(s_1(k), s_2(k), s_4(k))$
  - $s_2(k+1) = \frac{2}{3}s_1(k) + \frac{1}{3}s_4(k)$

strong assumption

- lock step synchrony
- instantaneous communication

partially synchronous (PS) communication

agent i sends a message with its value $s_i(t)$ at time t

- the message may be dropped
- otherwise, it is delivered to an *arbitrary subset* of agents within time $t + b$

b is a constant (system parameter) that is unknown to agents
PS model captures latency and uncertainties

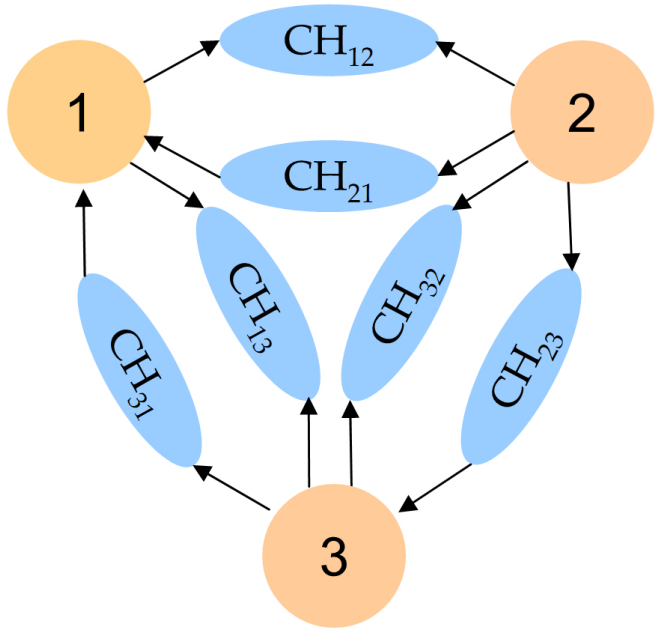
challenges in verifying PS systems

huge state space!

- $n(n-1)$ channels
- each channel stores (large number of) undelivered messages
- many possible message types as well

hence, limited success with model checking

generic techniques for compositional reasoning also not available



our main result[1]

Given the proof of convergence of a synchronous distributed system, under certain additional assumptions, it automatically guarantees the convergence of the corresponding PS system.

- proof of convergence: Lyapunov-like function
- additional assumptions about the structure of the lyapunov function
- correspondence defined by a natural transformation from synchronous to partially synchronous

. . . so what?

in designing algorithms for PS systems, if possible, verify the synchronous algorithm, check for additional conditions, and then use the natural transformation to get the PS system

ECE Illinois
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

[1] To appear in Intl. Conf. on Formal Modelling and Analysis of Timed Systems (FORMATS'08)

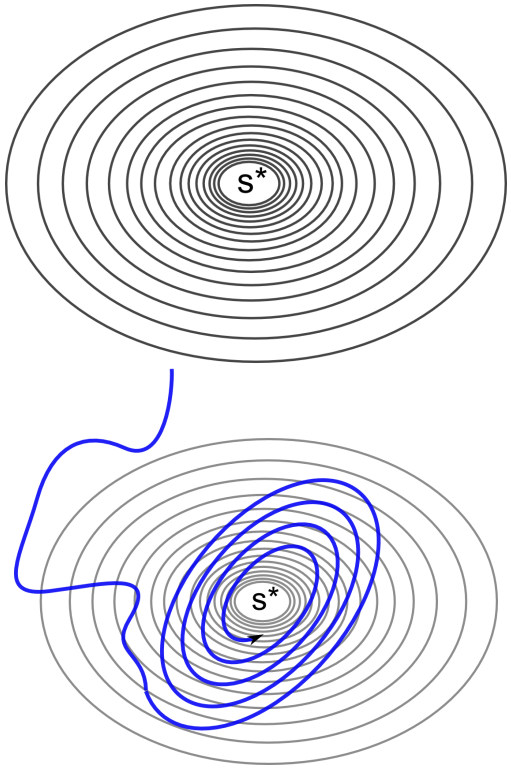
convergence

convergence to s^* is defined in terms of the set of neighborhood sets (topology around s^*) around s^*.

a discrete time system converges to a state s^* if for every neghborhood U around s^*, there exists $k_0 \in \mathbb{N}, k \geq 0$ such that for all $k > k_0$, $s(k) \in U$.

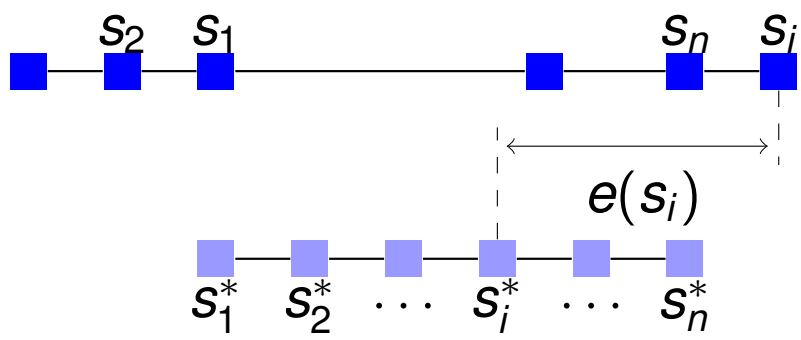
a continuous time system converges to a state s^* if for every neghborhood U around s^*, there exists $t_0 \in \mathbb{R}_{\geq 0}$ such that for all $t > t_0$, $s(t) \in U$.

convergence is a property of infinite executions



convergence in line formation

- s_i: position of i^{th} robot
- s_i^*: ideal position of i^{th} robot
- $e(s_i)$: deviation of i^{th} robot
- $U_p \triangleq$ states in which $\max_i e(s_i) \leq p$



fairness

some executions do not converge for silly reasons
e.g. an execution in which

- (*failure or perpetual message loss*) agent 3 is never active
- (*partition*) agents $\{1, 3\}$ never see $\{2, 4\}$

A *fairness condition* $\mathcal{F}$ restricts the set of executions by requiring that certain actions occur infinitely often.

e.g. “no permanent partition” is a fairness condition and we will look at convergence of executions that satisfy it.

proof of convergence

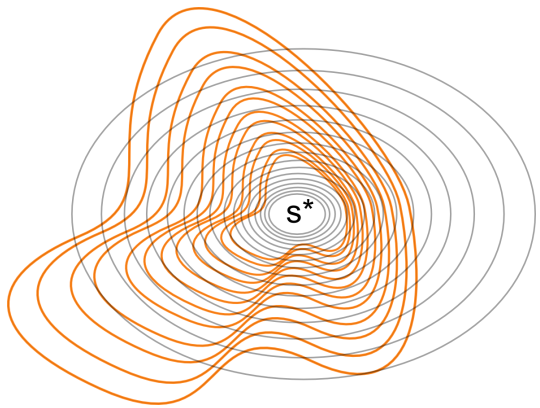
convergence is proved by establishing the existence of a (Lyapunov) function that is non-increasing along all executions of the system

Theorem (Tsitsiklis 1987)

Suppose there exists a collection of sets $\{P_k \subseteq S | \ k \in \mathbb{N}\}$ *satisfying:*

- *(monotonicity)* $k > l \Rightarrow P_k \subsetneq P_l$.
- *(granularity)* $\forall\ U, \exists\ k \in \mathbb{N}$, *such that* $P_k \subseteq U$.
- *(initial)* $S_0 \subseteq P_0$.
- *(invariance) if* $s \xrightarrow{a} s'$ *and* $s \in P_k$ *then* $s' \in P_k$.
- *(progress) If* $P_k \neq \{s^*\}$ *then there is a* $\mathcal{F}$*-fair action* a, $\forall\ s \in P_k, s' \in S$, $s \xrightarrow{a} s' \ \Rightarrow \ s' \in P_l$, *for some* $l > k$.

Then all $\mathcal{F}$*-fair executions of* $\mathcal{A}$ *converge to* s^*.



turns out to be necessary as well!

“additional assumptions”

simple assumption

For any $k \in \mathbb{N}$, P_k is of the form $p_{k1} \wedge p_{k2} \wedge \ldots p_{kn}$ where each p_{ki} is a predicate on the state of the i^{th} agent.

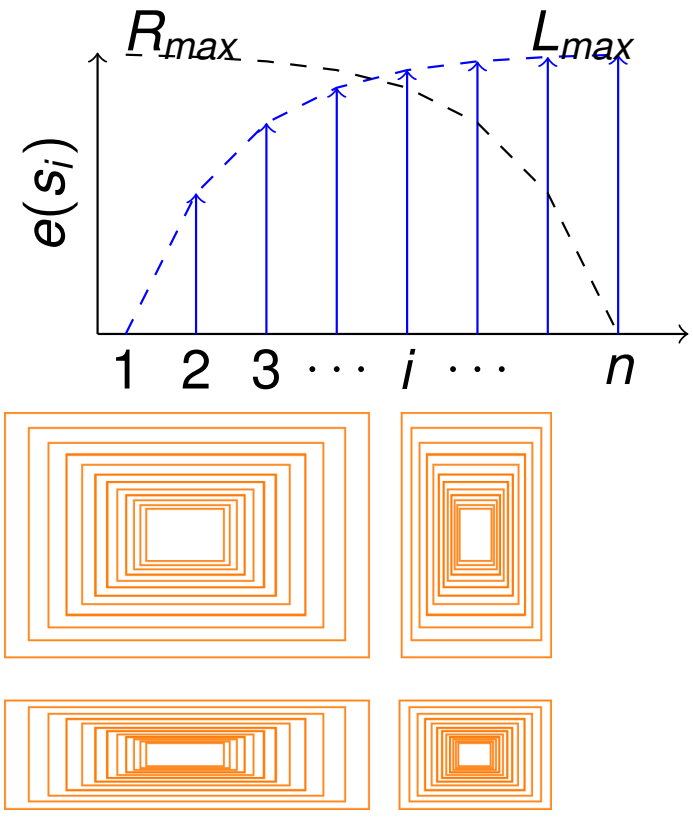
a more general assumption

appears in the paper

P_k's in line formation

$$L_{max}(s) \triangleq \forall l \quad e(s_l) \leq C\beta^m \left(1 - \frac{1}{2^{l-1}}\right)$$

$$R_{max}(s) \triangleq \forall r \quad e(s_r) \leq C\beta^m \left(1 - \frac{1}{2^{n-r}}\right)$$



transformation from synchronous to PS

variables

- $s_i : S := s_i(0)$
- $y_i : \texttt{Array}[[n] \to S_\perp]$ initially $\perp$

actions

- $\text{receive}_{ij}(m)$ sets $y_i[j] := m$
- $\text{update}_i(a)$ occurs periodically (a is chosen fairly) and it set $s_i := f_{ia}(x, y)$
- $\text{send}_i(m)$ occurs periodically

trajectories

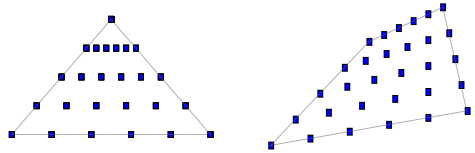
time progresses without violating task and message delivery deadlines

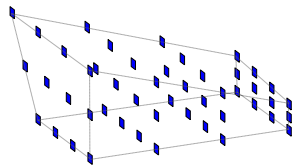
proof idea

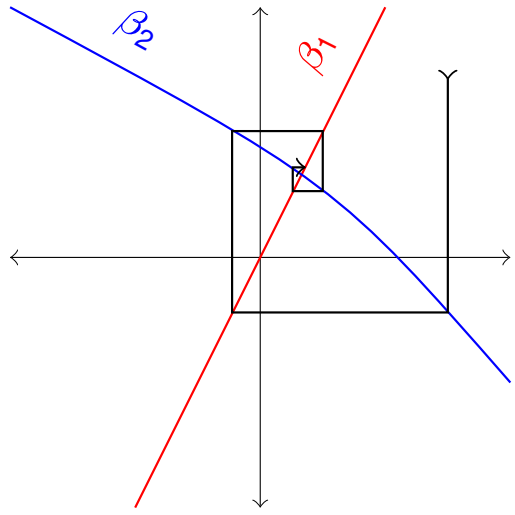
- from $\{P_k\}$ for $\mathcal{A}$ we construct $\{Q_k\}$ for $\mathcal{B}$ which satisfies all the conditions in Tistiklis' theorem
- define $\mathcal{R} \subseteq X \times S$, $\mathbf{x}\mathcal{R}s$ if and only if for each i one of the following holds
  - $\mathbf{x}.s_i = s_i$
  - there is j such that $\mathbf{x}.y_j[i] = s_i$
  - there is a message from i inbound to some j with value s_i
- monotonicity, granularity, and initial conditions are easy to check
- for invariance we use the additional assumption about $\{P_k\}$
- for progress we hae to use (a) additional assumptions about $\{P_k\}$, (b) the fairness of choosing the actions, (c) a weak assumption about message delivery.

applications

- **pattern formation.** in 2 and higher dimensions by simple extensions of the synchronous 1-D rule.
  agents with realistic dynamics
- **multi-player dynamic games.** each agent maximizes some utility based on its own state and (possibly old) messages from other agents
  do they agents converge to an equilibrium ?
- **aggregation and gossip protocols.** agents update their values using associative, commutative, and idempotent operations
  do they terminate ?
- **stationary distributions.** computing stationary distribution of big Markov chains using message passing over parallel processes (Miranker 1970s).







verification using theorem prover[2]

- there is an existing formalization of timed, untimed, and hybrid I/O automata in the PVS theorem prover
- these theories provide support for proving invariants and implementation relations—which requires only modelling and reasoning about finite executions
- to this existing body of theories we add notions of infinite executions, fairness, neighborhood sets, and prove the theorems for proving convergence
- several case studies

ECE Illinois
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

[2] In proc. of Theorem Proving in High Order Logics (TPHOLs'08)

implementations[3]

- synthesis of Java programs for partially synchronous networks from synchronous (PVS) specifications
- currently using an arbiter for ensuring fairness

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

[3] joint work with Brian Go and Jerome White submitted for review