Verifying Broadcast Protocols

Javier Esparza

Institut für Informatik
Technische Universität München

Automatic verification

- Initiated in the early 80s.
- Explicit construction of the state space $\rightarrow$ finiteness constraint.
- Very successful in hardware, not so much in software or protocols.

Sources of infinity

- Data manipulation.

  Unbounded counters, integer variables, queues ...

- Control structures.

  Procedures $\rightarrow$ stack, process creation $\rightarrow$ bag.

- Asynchronous communication.

  Unbounded FIFO queues.

- Parameters.

  Number of processes, of input gates, of buffers, ...

- Real-time.

  Discrete or dense domains.

Parametrized protocols

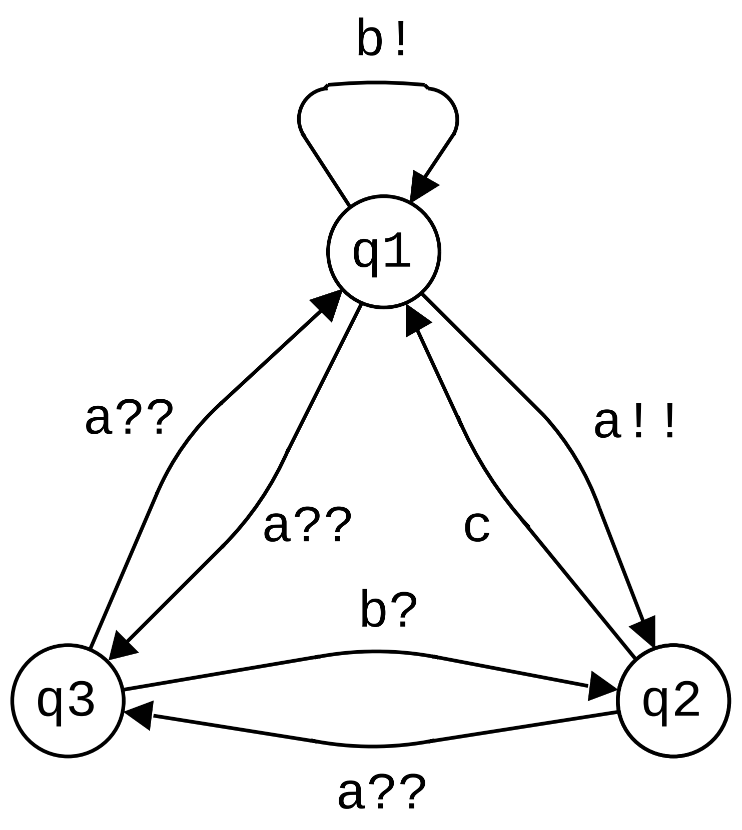
- Defined for n processes.
- Correctness: the desired properties hold for every n.
- Processes modelled as communicating finite automata.
- For each value of n the system has a finite state space (only one source of infinity).
- Turing powerful, and so further restrictions sensible:

Broadcast Protocols

Broadcast protocols

- Introduced by Emerson and Namjoshi in LICS ’98.
- All processes execute the same algorithm, i.e., all finite automata are identical.
- Processes are undistinguishable (no IDs).
- Two communication mechanisms:
  - Rendezvous: two processes exchange a message and move to new states.
  - Broadcasts: a process sends a message to all others; all processes move to new states.

Syntax



$a!!$: broadcast a message along (channel) a

$a??$: receive a broadcasted message along a

$b!$: send a message to one process along b

$b?$: receive a message from one process along b

c : change state without communicating with anybody

Semantics

The global state of a broadcast protocol is completely determined by the number of processes in each state.

- Configuration: mapping $c\colon S \to \mathbb{N}$, seen as element of $\mathbb{N}^n$, where $n = |S|$.
- Semantics for each n: finite transition system with configurations as nodes.

In our example:

$$
\begin{array}{llll}
(3,1,2) & \xrightarrow{c} & (4,0,2) & \text{(silent move)} \\
(3,1,2) & \xrightarrow{b} & (3,2,1) & \text{(rendezvous)} \\
(3,1,2) & \xrightarrow{a} & (2,1,3) & \text{(broadcast)}
\end{array}
$$

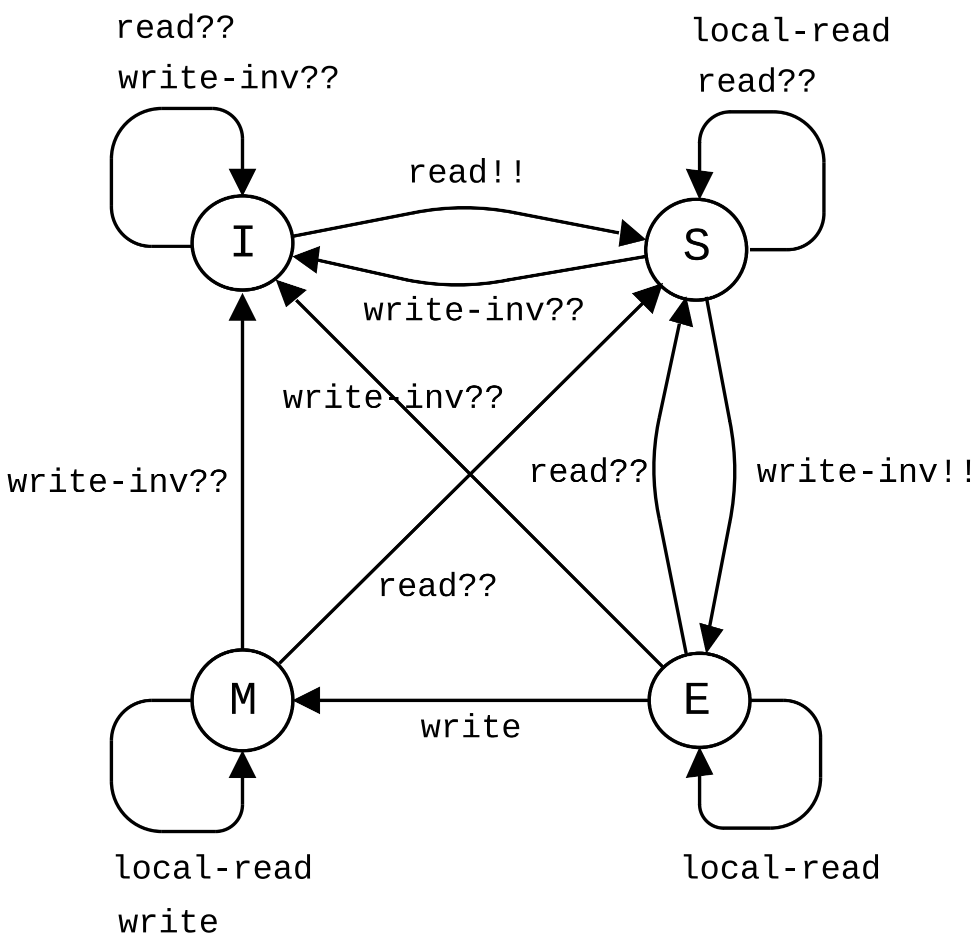
Semantics (continued)

- Parametrized configuration:

  Partial mapping $p : Q \to \mathbb{N}$, seen as a set of configurations (total mappings matching p).

- Transition system of the broadcast protocol:
  - Fix an initial parametrized configuration p_0.
  - Take the union of all finite transition systems T_c for each configuration $c \in p_0$.

A MESI-protocol



Safety properties

U is an upward-closed set of configurations if

$$c \in U \text{ and } c' \geq c \text{ implies } c' \in U$$

where $\geq$ is the pointwise order on $\mathbb{N}^n$.

Safety property $\rightarrow$ upward-closed set D of dangerous configurations that should not be reachable.
Example: in the MESI protocol the states M and S should be mutually exclusive.

$$D = \{(m,e,s,i) \mid m \geq 1 \wedge s \geq 1\}$$

Backward search

$$\mathit{add\_pre}(C) \quad = \quad \text{union of } C \text{ and its immediate predecessors}$$

Procedure for solving the reachability problem:

Proc $\mathit{BackReach}(I, F)$

  $C := F$;

  repeat

    $\mathit{old\_C} := C$;

    $C := \mathit{add\_pre}(\mathit{old\_C})$;

    if $C \cap I \neq \emptyset$ **then return** "reachable"

  until $C = \mathit{old\_C}$;

return "non-reachable"

The procedure is effective if there exists a family $\mathcal{C}$ of sets such that

(1) each $C \in \mathcal{C}$ has a finite representation;

(2) $F \in \mathcal{C}$;

(3) if $C \in \mathcal{C}$, then $add\_pre(C) \in \mathcal{C}$ and $add\_pre(C)$ can be effectively computed;

(4) emptyness of $C \cap I$ is decidable, and

(5) $C_1 \subseteq C_2$ is decidable.

The procedure terminates if

(6) any chain $C_1 \subseteq C_2 \subseteq C_3 \ldots$ of elements of $\mathcal{C}$ reaches a fixpoint after finitely many steps.

Backward search for broadcast protocols

E., Finkel, and Mayr LICS '99, basing on
Abdulla, Čerāns, Jonsson, and Tsay LICS '96.
The family of all upward-closed sets satisfies (1)–(6):

(1) An upward-closed set can be represented by its set of minimal elements w.r.t. the pointwise order $\leq$ (Dickson's Lemma).

(3) If U is upward-closed then so is $add\_pre(U)$.

$$\begin{array}{ccc} c & \xrightarrow{a} & u \in U \\ \leq & & \leq \\ c' & \xrightarrow{a} & u' \in U \end{array}$$

(6) Any chain $U_1 \subseteq U_2 \subseteq U_3 \ldots$ of upwards closed sets reaches a fixpoint after finitely many steps (Dickson's lemma + some reasoning).

Complexity

The number of iterations of *BackReach* can be

non-primitive-recursive

in the number of states of the protocol.

However, the algorithm is useful in practice!

Application to the MESI-protocol

Are the states M and S mutually exclusive?
Check if the upward-closed set with minimal element

$$m = 1, e = 0, s = 1, i = 0$$

can be reached from the initial p-configuration

$$m = 0, e = 0, s = 0, i = \sqcup\ .$$

Proceed as follows:

$$\begin{aligned}
U:&\quad m \geq 1 \wedge s \geq 1\\
add_pre(U):&\quad (m \geq 1 \wedge s \geq 1) \vee\\
&\quad (m = 0 \wedge e = 1 \wedge s \geq 1)\\
add_pre^2(U):&\quad add_pre(U)
\end{aligned}$$

Implementing *BackReach*

Linear constraints as finite representation of sets of configurations.

- The variable x_i represents the number of processes in state q_i.
- Set of configurations $\rightarrow$ set of constraints over $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ (interpreted disjunctively).

Proc *SymbBackReach*(Φ_I, Φ_F)

  $\Phi := \Phi_F$;

  repeat

    $old\_\Phi := \Phi$;

    $\Phi := Add\_pre(old\_\Phi)$;

    if $\Phi \wedge \Phi_I$ satisfiable **then return** "reachable"

  until $Entail(\Phi, old\_\Phi)$;

  return "non-reachable"

- Label $a \rightarrow$ linear transformation with guard.
  Label a of our example:
  Guard $\mathbf{G_a}$: $x_1 \geq 1$.
  Linear transformation $\mathbf{M_a x} + \mathbf{b_a}$:

$$M_a = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \quad b_a = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$$

- Symbolic computation of *add_pre* must satisfy:

$$Add\_pre(\Phi)$$

$$\equiv$$

$$\Phi \vee \bigvee_{a \in \Sigma, \phi \in \Phi} \mathbf{G_a} \wedge \phi[\mathbf{x}/\mathbf{M_a x} + \mathbf{b_a}]$$

Which class of constraints?

Conditions:

- Able to express all upward-closed sets.
- Efficient computation of *Add_pre*.
- Efficient computation of *Entail*.
  Entail is co-NP-complete for arbitrary constraints.

Natural candidates

- L-constraints:
  Conjunction of $x_{i_1} + \ldots + x_{i_n} \geq c$ where $x_i \neq x_j$.
  - Closed under broadcast transformations.
  - Entailment is co-Np-complete for single constraints.
- WA-constraints: Conjunction of $x_i \geq c$
  - Entailment is polynomial (quadratic)
  - Not closed under broadcast transformations.
    L-constraints equivalent to sets of WA-constraints, but with exponential blow-up:

$$x_{i_1} + \ldots + x_{i_m} \geq c$$
$$\equiv$$
$$\bigvee_{c_1+\ldots+c_m=c} x_{i_1} \geq c_1 \wedge x_{i_2} \geq c_2 \wedge \ldots \wedge x_{i_m} \geq c_m$$

Using WA-constraints:

Delzanno and Raskin '00

- Represent the constraint $x_1 \geq c_1 \wedge \ldots \wedge x_n \geq c_n$ by $(c_1, \ldots c_n)$.
- Use sharing trees to represent sets of constraints.

A sharing tree is an acyclic graph with one root and one terminal node such that:

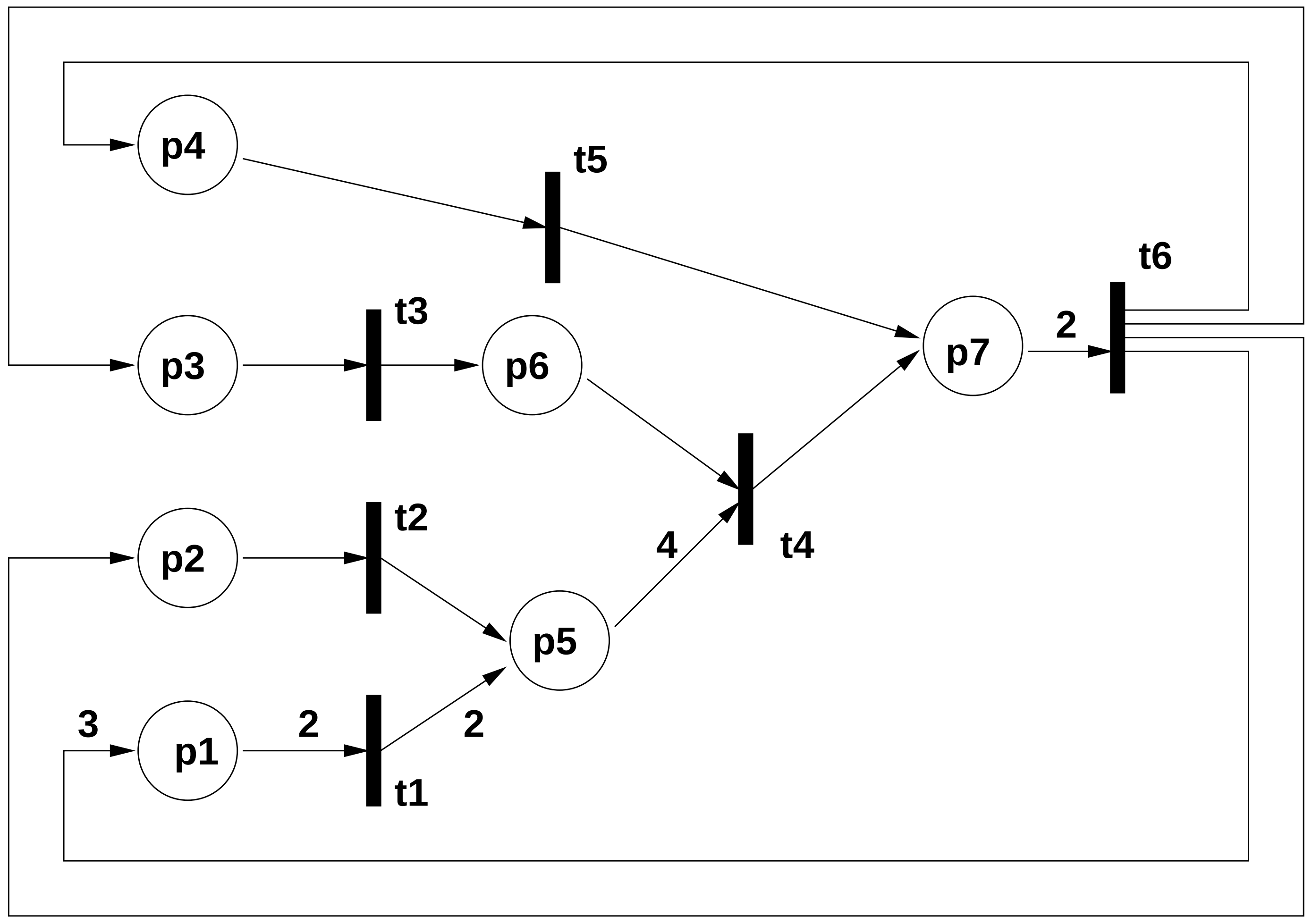
- all nodes of layer i have successors in the layer $i+1$;
- a node cannot have two successors with the same label;
- two nodes with the same label in the same layer do not have the same set of successors

Small experiment

- Application: computation of safe initial states in production plants.
- Usually no broadcasts, but multi-way synchronization.
- Computing the deadlock-free initial states of a small production plant (five machines, seven storage points).
- Seven different regions of initial states.
- Computation time (Sun Ultra Sparc):

Sharing trees	HyTech	Presburger
39s	> 24h	19h50m

A small experiment



Deadlock-free initial markings:

$P_1 \geq 10, P_2 \geq 1, P_3 \geq 2$ $\quad$ $P_1 \geq 8, P_3 \geq 3$ $\quad$ $P_1 \geq 12, P_3 \geq 2$

$P_1 \geq 6, P_2 \geq 5, P_3 \geq 2$ $\quad$ $P_1 \geq 8, P_3 \geq 1, P_4 \geq 1$ $\quad$ $P_1 \geq 6, P_4 \geq 2$

$P_1 \geq 6, P_2 \geq 1, P_3 \geq 1, P_4 \geq 1$

Computation time (Sun Ultra Sparc):

Sharing trees	HyTech	Presburger
39s	> 24h	19h50m

Using L-constraints

Delzanno, E., Podelski '99, Delzanno '00.

First simplification: *Entail* need only be computed for single constraints, not for sets of constraints: Replace

$$\textbf{until } \mathit{Entail}(\Phi, \mathit{old\_}\Phi)$$

by the stronger condition

$$\textbf{until forall } \phi \in \Phi \textbf{ exists } \psi \in \mathit{old\_}\Phi : \quad \mathit{Entail}(\phi, \psi)$$

Possibly slower, but still guaranteed termination!

But entailment for L-constraints co-Np-complete even for single constraints!

Second simplification: $\mathit{Entail}(\phi, \psi)$ can be interpreted over the reals. Again, stronger **until**-condition which doesn't spoil termination!

Case studies (by G. Delzanno)

- Broadcast protocols must be extended with more complicated guards.
- Termination guarantee gets lost.
- Berkeley RISC, Illinois, Xerox PARC Dragon, DEC Firefly.
  At most 7 iterations and below 100 seconds (SPARC5, Pentium 133).
- Futurebus +
  8 steps and 200 seconds (Pentium 133).

Related and future work

- Abdulla, Jonsson *et al.* on arbitrary well-quasi-orderings
- Abdulla, Pnueli, Jonsson *et al.* on linear arrays of processes.
- Advanced Validation Techniques for Telecommunication Protocols
  - Coordinated by Paris VII (Ahmed Bouajjani)
  - Participants: Grenoble, Liège, Munich, Paris VII, Weizmann, France Telecom (Ericsson, Siemens)