ASYNCHRON 2006

Verification of Communication Protocols with Messages Carrying Values

Tristan Le Gall

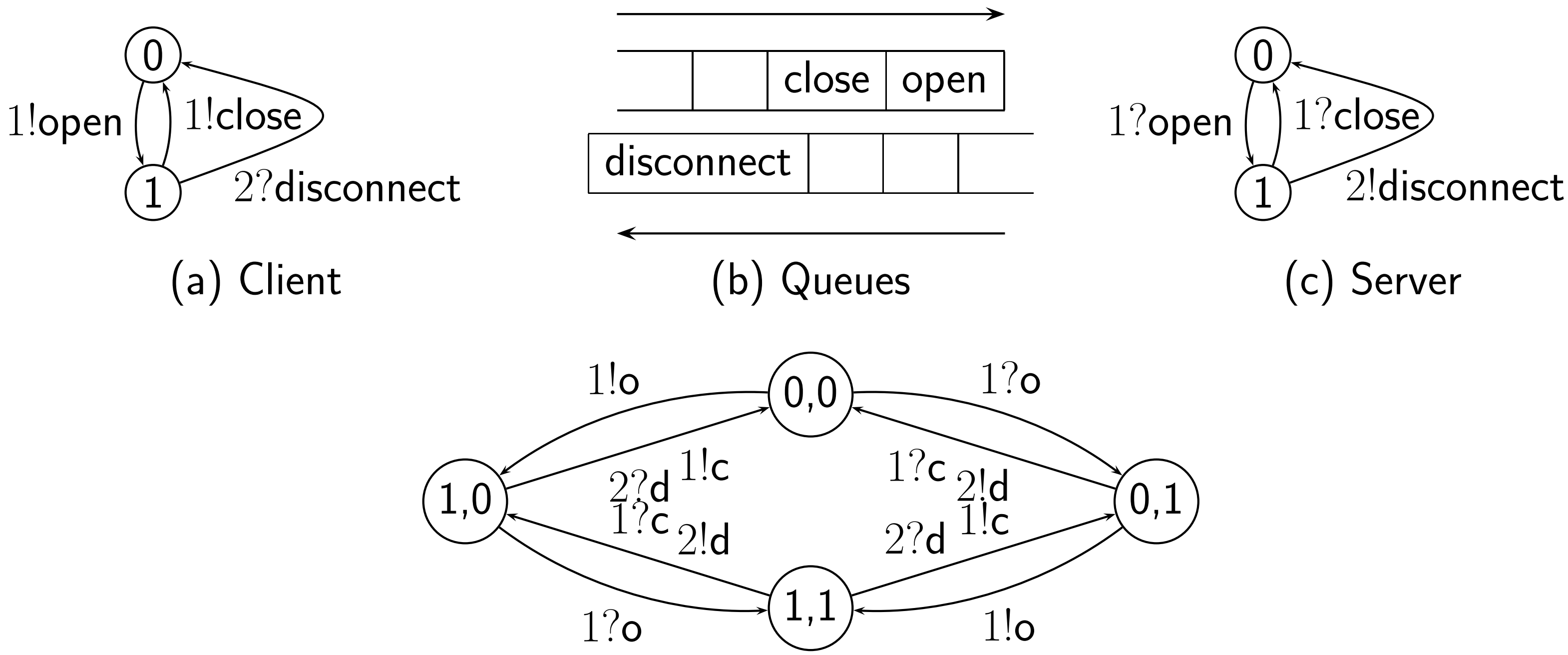
joint work with
Bertrand Jeannet and Thierry Jéron

Vertecs team
IRISA/INRIA Rennes

Comunication protocols

- Communication protocols are widely used with the developpement of the internet and other networks.
- Formal verification uses models like Messages Sequence Charts(MSCs), Communicating Finite-State Machines(CFSMs)
- Our goal : the verification of protocols modeled by an extention of CFSM, using the abstract interpretation framework.
- This work can be applied to process/components of a systems using queues or large buffers, Kahn networks, etc.

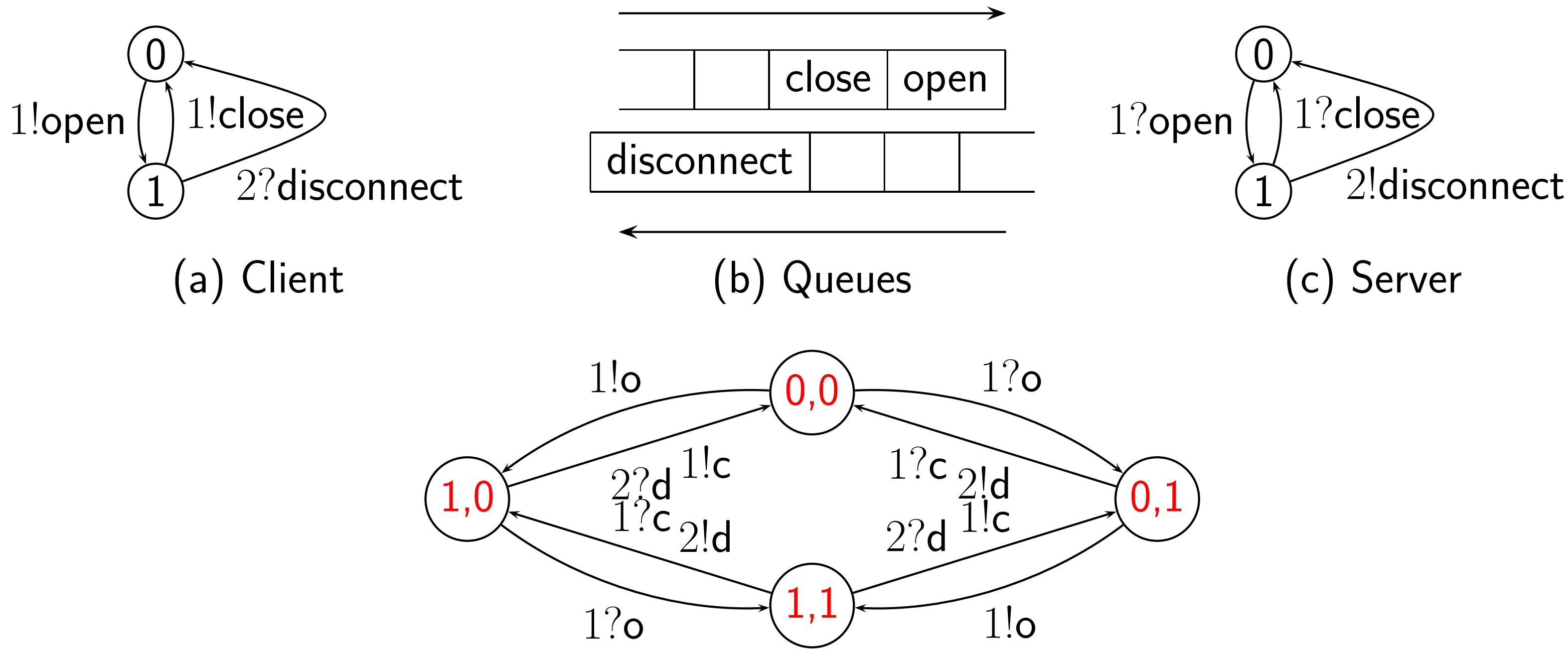
The CFSM model



(a) Client (b) Queues (c) Server

(d) Global CFSM: product of client and server processes

The CFSM model

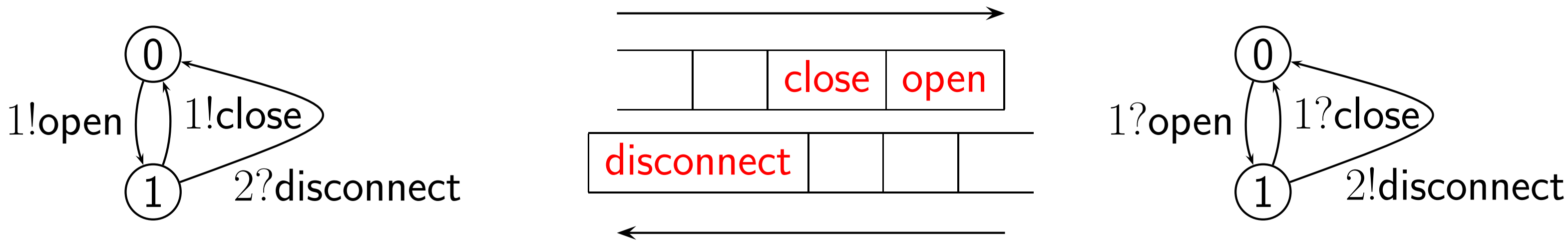


(a) Client (b) Queues (c) Server

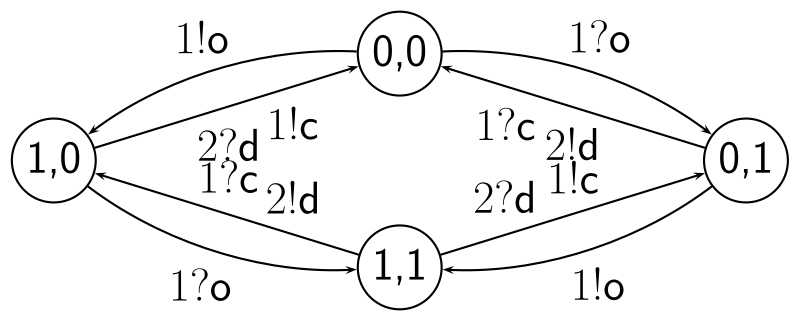
(d) Global CFSM: product of client and server processes

$C = \{(\mathbf{0},\mathbf{0}),(\mathbf{1},\mathbf{0}),(\mathbf{0},\mathbf{1}),(\mathbf{1},\mathbf{1})\}$

The CFSM model



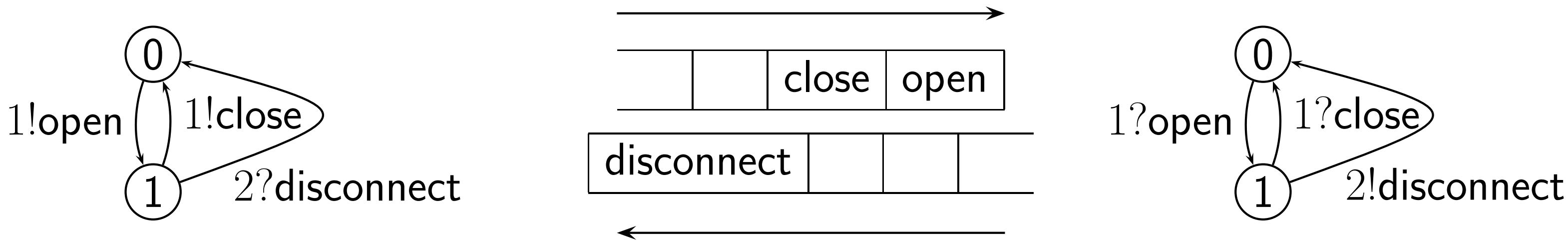
(a) Client (b) Queues (c) Server



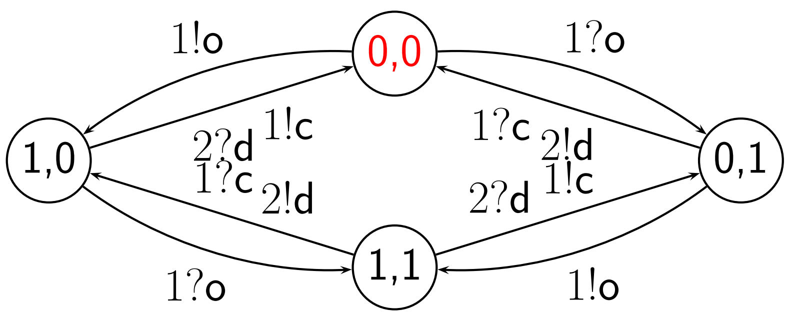
(d) Global CFSM: product of client and server processes

$\Sigma = \{\text{open}, \text{close}\} \cup \{\text{disconnect}\}$

The CFSM model



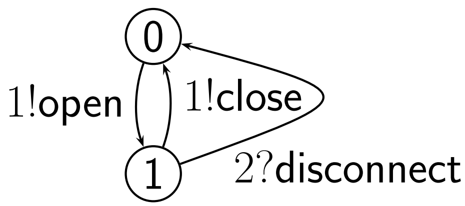
(a) Client (b) Queues (c) Server



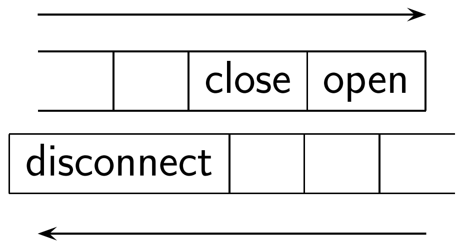
(d) Global CFSM: product of client and server processes

initial location $c_0 = (\mathbf{0}, \mathbf{0})$

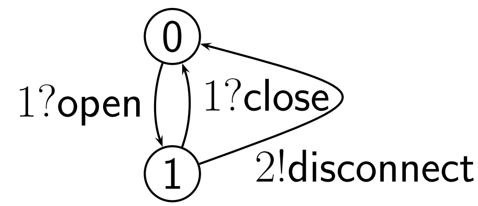
The CFSM model



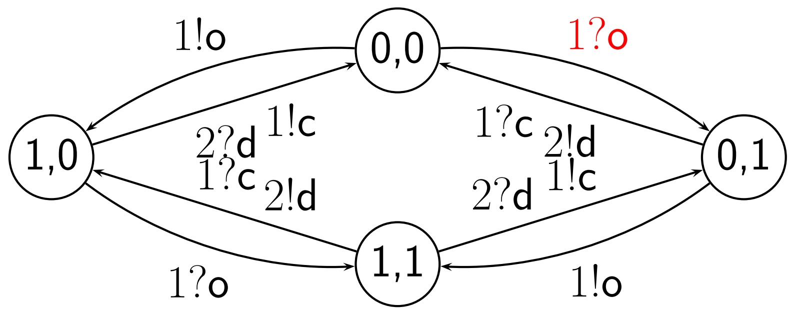
(a) Client



(b) Queues



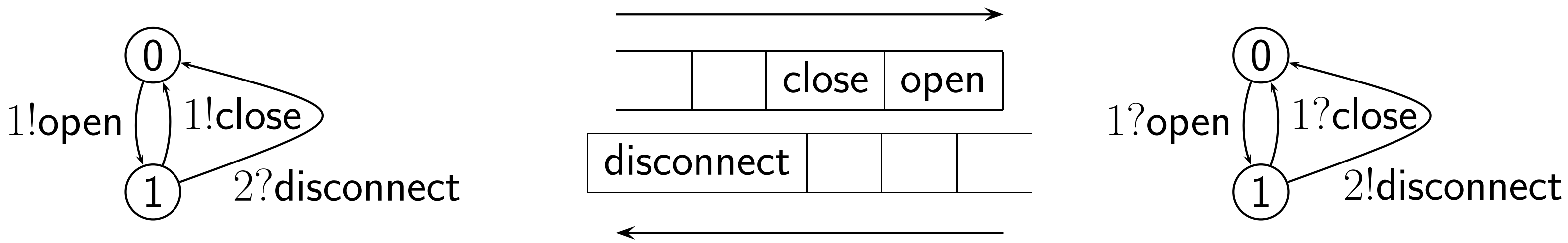
(c) Server



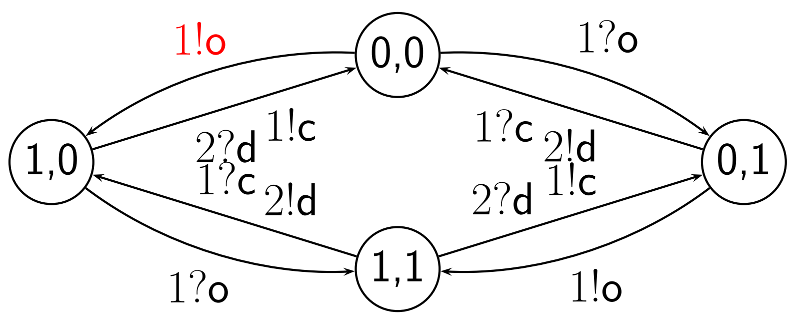
(d) Global CFSM: product of client and server processes

An input : 1?o

The CFSM model



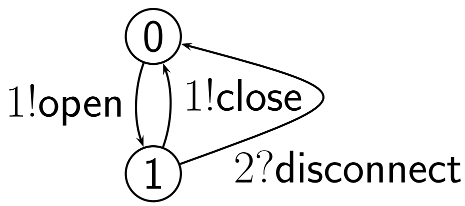
(a) Client (b) Queues (c) Server



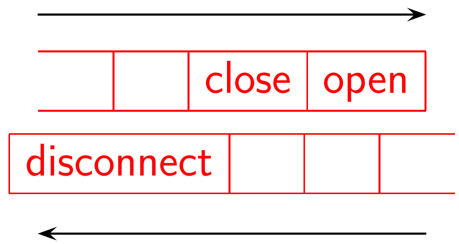
(d) Global CFSM: product of client and server processes

An output : 1!o

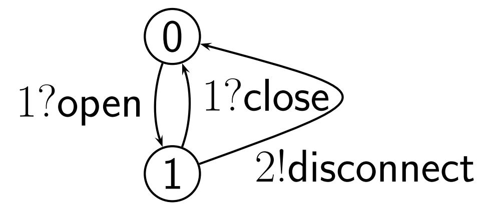
The CFSM model



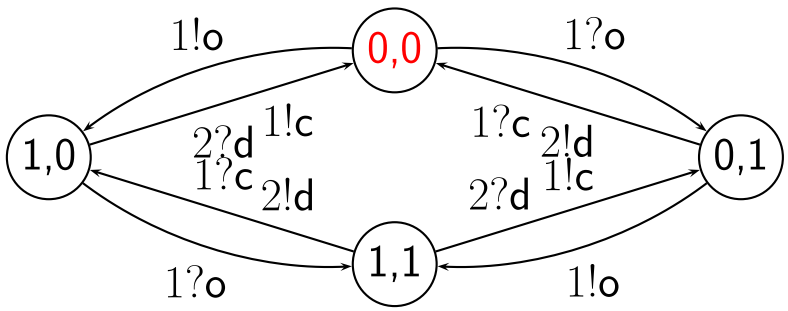
(a) Client



(b) Queues



(c) Server



(d) Global CFSM: product of client and server processes

A state of the CFSM : a location + contents of all queues

Problematics

- Verification of safety properties
- Main issue : reachability analysis
- Undecidable in the general case
- Our solution : compute an over-approximation of the reachability set

IRISA

Outline

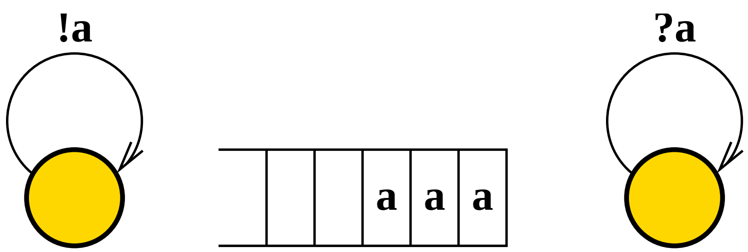
- Introduction : model and problematics
- Verification of CFSM
- Limitations and new model
- Representation of queues with messages carrying values
- Application to the verification of symbolic CFSM
- Conclusion

CFSM with a single queue

- Σ : alphabet of messages
- a content of a queue : a word $w \in \Sigma^*$
- $C \longrightarrow \mathcal{L}(\Sigma^*)$
- Operational semantics in terms of operations on languages :

$$(c_1, L) \xrightarrow{\langle c_1, !a, c_2 \rangle} (c_2, L.a)$$

- Reachability analysis : a fixpoint equation



- Fixpoint equation $L = \varepsilon \cup L.a \cup L/a$
- Solution : $L = a^*$

Abstraction

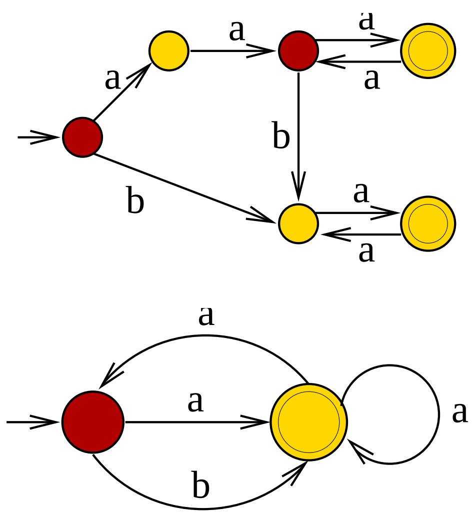
- Main idea : work with regular over-approximations of the content of the queue
- See the regular languages as an abstract lattice $(\boldsymbol{Reg}(\Sigma), \subseteq)$
- Compute an over-approximation of the least fix-point with iterations

$$\begin{aligned} L_0 &= \varepsilon \\ L_{i+1} &= L_i \cup L_i.a \cup L_i/a \end{aligned}$$

- Use a *widening operator* so that the computation terminates

Widening operator for regular languages(1)

- Working on the Minimal Deterministic Automaton (MDA) M_L
- Quotient automaton $\widetilde{M_L} = M_L / \simeq_k$ (fusion of states)



- $\simeq_k$: auto-bisimilarity of depth k
- $\rho_k(L)$: language recognized by this quotient automaton

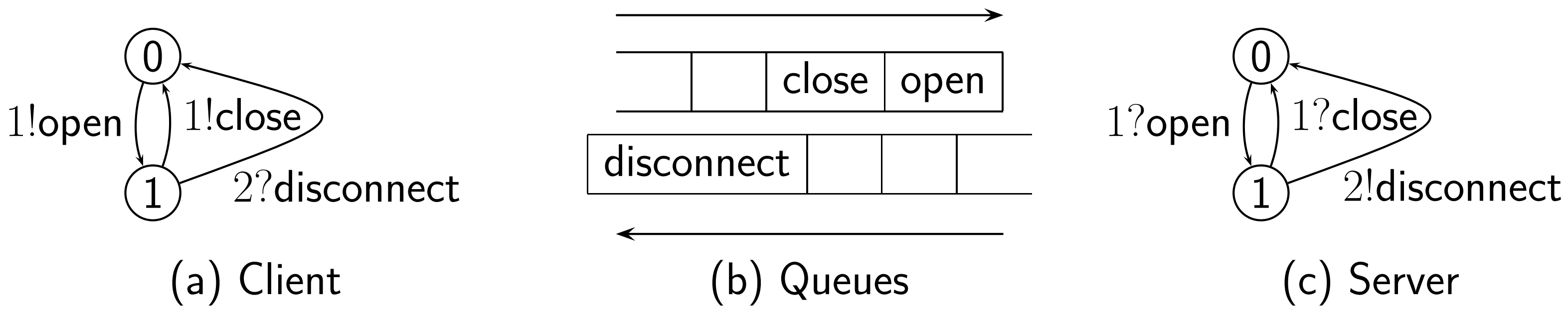
Widening operator for regular languages(2)

- Widening operator $L_1 \nabla_k L_2 \triangleq \rho_k(L_1 \cup L_2)$
- The following computation terminates and gives an over-approximation of the reachability set:

$$\begin{aligned} L_0 &= \varepsilon \\ L_{i+1} &= L_i \, \nabla_k (\, L_i.a \, \cup \, L_i/a) \end{aligned}$$

- Result : $L_\infty = a^*$

Connexion/deconnexion protocol



- The client can open and close a cession, or be forced to close the session if a disconnect message is received
- The serveur can ask for a client to terminate his session

Analysis of the connexion/deconnexion Protocol

Analysis with dependance

Client/ Server	Queue 1 # Queue 2
0/0	$(co)^*(oc)^*\#\varepsilon \ + \ c(oc)^*\#d$
1/0	$(co)^*(oc)^*o\#\varepsilon \ + \ (co)^*\#d$
0/1	$c(oc)^*\#\varepsilon$
1/1	$(co)^*\#\varepsilon$

Analysis without dependance

Client/ Server	Queue 1	Queue 2
0/0	$o^* + (o^*c)^+(\varepsilon + o^+ + o^+c)$	d^*
1/0	$(o^*c)^*o^+$	d^*
0/1	$o^* + (o^*c)^+(\varepsilon + o^+ + o^+c)$	d^*
1/1	$o^+ + o^*(co^+)^+$	d^*

- Analysing the queues alltogether gives the exact result
- Analysing each queue independently gives a very bad approximation

Protocol with non-regular reachability set

- The reachability set is non-regular
- Exact result : $L_{(0/0)} = a^n \sharp \varepsilon \sharp c^n \sharp \varepsilon$
- Relational analysis result :

$$L_{(0/0)} = \varepsilon \sharp \varepsilon \sharp \varepsilon \sharp \varepsilon + a \sharp \varepsilon \sharp c \sharp \varepsilon + aaa^* \sharp \varepsilon \sharp ccc^* \sharp \varepsilon$$

Outline

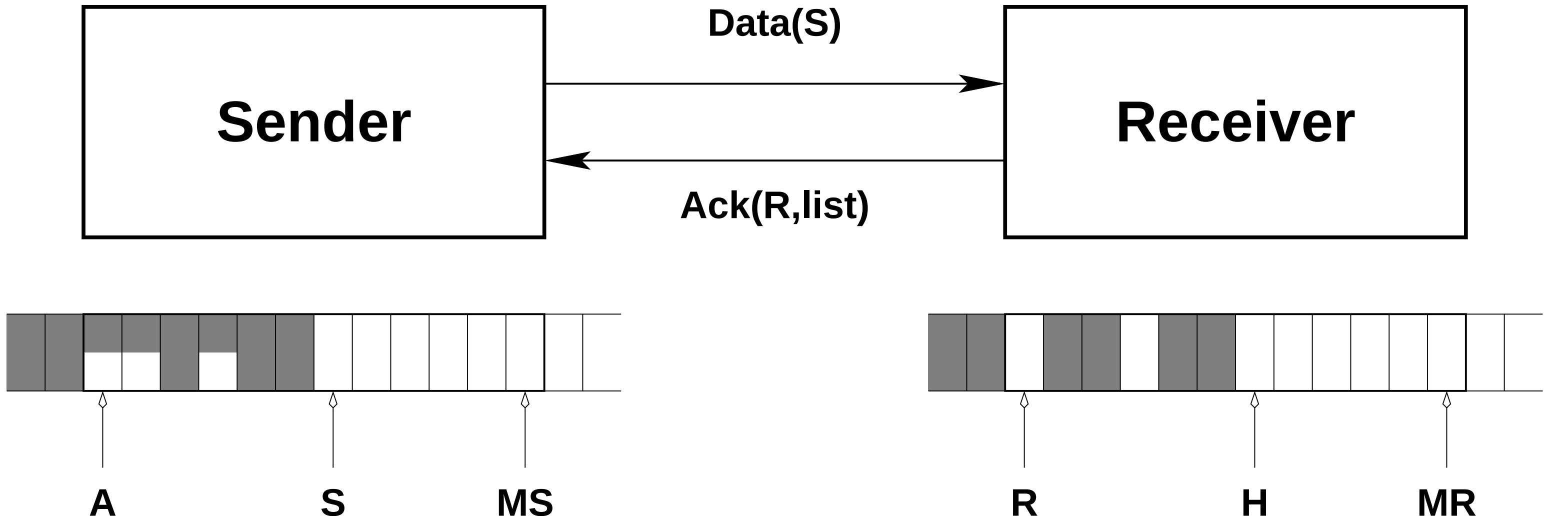
- Introduction : model and problematics
- Verification of CFSM
- Limitations and new model
- Representation of queues with messages carrying values
- Application to the verification of symbolic CFSM
- Conclusion

IRISA

Already finished ?

- The analysis terminates and return an over-approximation of the reachability set
- The approximations of the queue contents are quite precise
- We defined an abstract lattice for regular languages
- But the model is not expressive enough

Sliding window protocols



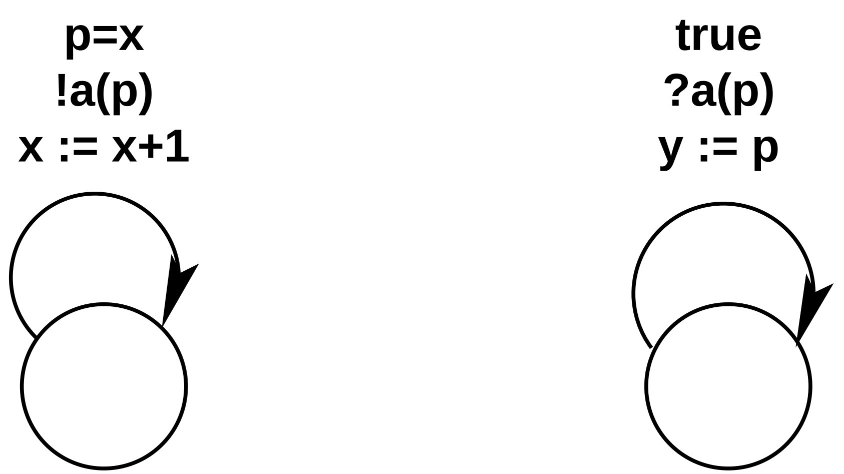
Can we check the (in)equations :

- $A \leq R$
- $S \geq H$
- $MR - R = MS - A$

What shall we do ?

- Do not care about real protocols
- Add variables and parameters to the CFSM model and use a similar method on the new model.

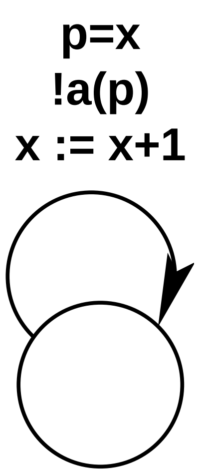
New Model : Symbolic CFSM



Each transition has

- a guard : predicate on the value of the variables and the parameter
- a comunication action with a parameter p : emission $!a(p)$ or reception $?a(p)$
- an affectation : gives the new value of the variables

Example of analysis



- Toy example : the producer
- Non-relational analysis : does not keep relation between the value of x and the values of the messages
- Relational analysis : keep relation between the value of x and the values of the messages

Non-relational analysis

Computation step	Set of reachable states
init	$[0,0] \times \varepsilon$
step 1	$[0,1] \times a([0,0])$
step 2	$[0,2] \times a([0,0]) + a([0,0]).a([0,1])$
step 3	$[0,3] \times a([0,0]) + a([0,0]).a([0,1]) + a([0,0]).a([0,1]).a([0,2])$
step4*	$[0,+\infty[\times a([0,1]) + a([0,1]).\,(a([0,+\infty[))^*$

We lose the relation $p \leq x$.

Relational analysis

Computation step	Set of reachable states	
init	$\{0 \leq x \leq 0\} \wedge$	ε
step 1	$\{0 \leq x \leq 1\} \wedge$	$a(\{0 \leq p = x - 1\})$
step 2	$\{0 \leq x \leq 2\} \wedge$	$a(\{0 \leq p = x - 1\}) + a(\{0 \leq p = x - 2\}).a(\{0 \leq p = x - 1\})$
step 3	$\{0 \leq x \leq 3\} \wedge$	$a(\{0 \leq p = x - 1\})+$
		$+a(\{0 \leq p = x - 2\}).a(\{0 \leq p = x - 1\})+$
		$+a(\{0 \leq p = x - 3\}).a(\{0 \leq p = x - 2\}).a(\{0 \leq p = x - 1\})$
step 4*	$\{0 \leq x\}$	$a(\{0 \leq p \leq x - 1\})+$
		$+a(\{0 \leq p \leq x - 1\}).\,(a(\{0 \leq p \leq x - 1\}))^*$

New abstractions

In both cases we need :

- Abstractions for the values of the variables : intervals, polyhedra,...
- Representation of languages on the infinite alphabet $\Sigma \times \Omega$
- Automata "with a lattice feature"

Outline

- Introduction : model and problematics
- Verification of CFSM
- Limitations and new model
- Representation of queues with messages carrying values
- Application to the verification of symbolic CFSM
- Conclusion

Lattice structure

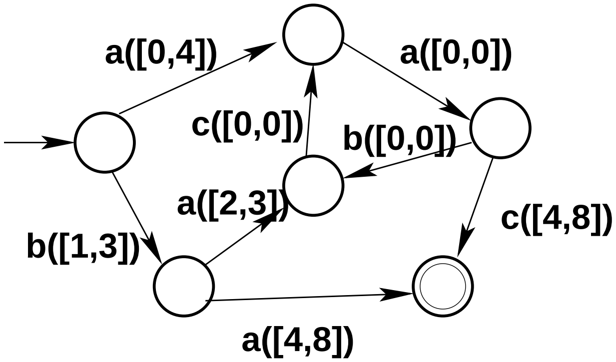
- $(\Omega, \sqsubseteq_\Omega)$ an abstract lattice (abstract values of the parameter)
- $\Sigma = \{a_1, \ldots, a_n\}$ finite alphabet of messages
- $\Lambda = \Sigma \times \Omega$ the lattice with

$$(a_1, P_1) \sqsubseteq (a_2, P_2) \;\Leftrightarrow\; a_1 = a_2 \wedge P_1 \sqsubseteq_\Omega P_2$$

- Example : lattice : $\Lambda = \Sigma \times \mathcal{I}$; value: $a([0, +\infty[)$

Lattice automaton

- Finite automaton with transitions labeled by $\lambda \in \Sigma \times \Omega$

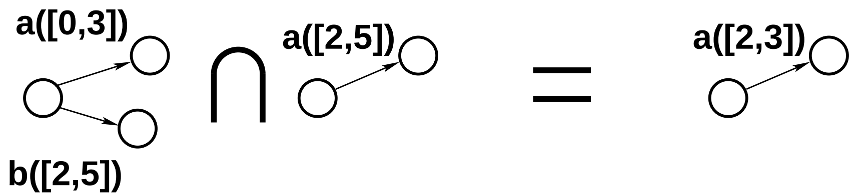


- $q_1 \xrightarrow{\lambda} q_2$ if
  1. $\lambda \neq \bot$
  2. there is a transition (q_1, λ', q_2) in the automaton with $\lambda \sqsubseteq \lambda'$
- Accepted words : as for classical finite automata.
- $Reg(\Lambda)$: languages recognized by a lattice automaton

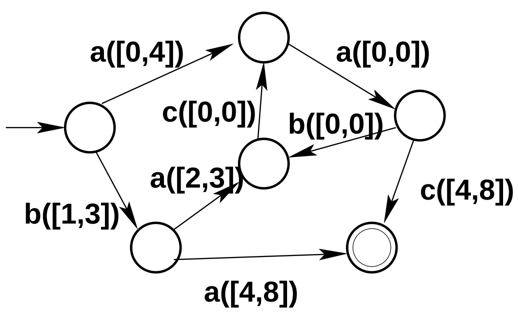
IRISA

Algorithms for lattice automata

- Union : as for classical finite automata
- Intersection :



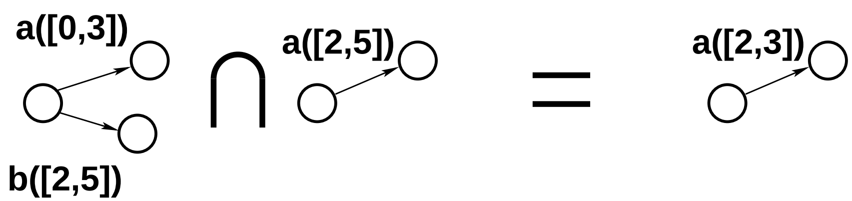
- Inclusion : simulation taking into account the lattice structure
- Determinisation, minimisation, quotient of $\mathcal{A}$: use $Shape(\mathcal{A})$



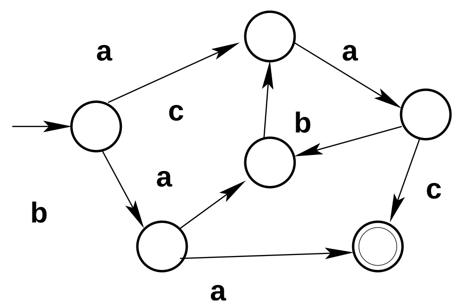
IRISA

Algorithms for lattice automata

- Union : as for classical finite automata
- Intersection :

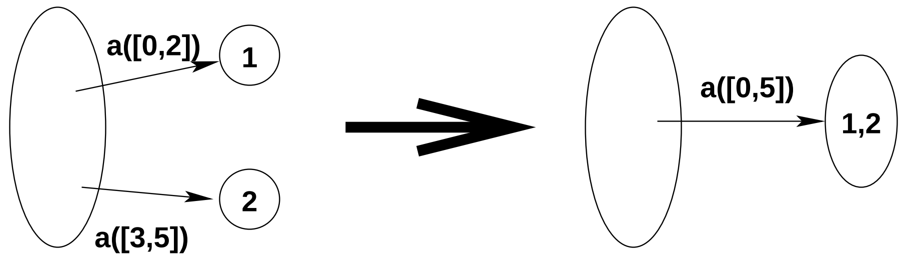


- Inclusion : simulation taking into account the lattice structure
- Determinisation, minimisation, quotient of $\mathcal{A}$: use $Shape(\mathcal{A})$



Determinisation of lattice automata

Idea of the algorithm :

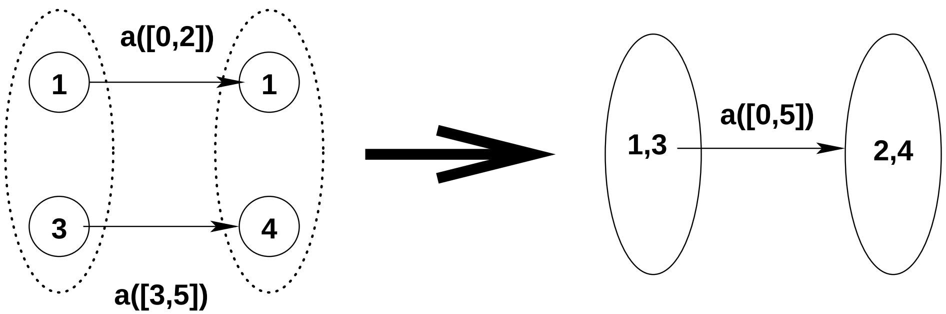


$\mathcal{A}$ a no-deterministic lattice automaton. We get $\mathcal{A}'$:

- $Shape(\mathcal{A}')$ is deterministic (so is $\mathcal{A}'$)
- $\mathcal{A}'$ is an over-approximation of $\mathcal{A}$:
  $L_{\mathcal{A}} \subseteq L_{\mathcal{A}'}$
- $\mathcal{A}'$ is the best approximation :
  $\forall \mathcal{B}$ such as $Shape(\mathcal{B})$ is deterministic and $L_{\mathcal{A}} \subseteq L_{\mathcal{B}}$, then $L_{\mathcal{A}'} \subseteq L_{\mathcal{B}}$

Quotient and minimisation

Same principle as before



L recognized by $\mathcal{A}$. There exists an unique $\mathcal{A}'$:

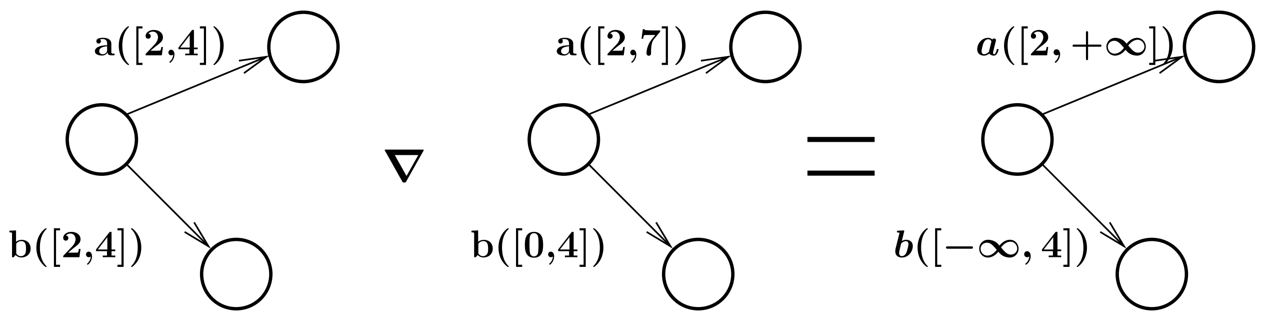
- $Shape(\mathcal{A}')$ is deterministic and minimal(so is $\mathcal{A}'$)
- $\mathcal{A}'$ is an over-approximation of $\mathcal{A}$:
  $L_{\mathcal{A}} \subseteq L_{\mathcal{A}'}$
- $\mathcal{A}'$ is the best approximation :
  $\forall \mathcal{B}$ such as $Shape(\mathcal{B})$ is deterministic and minimal, and $L_{\mathcal{A}} \subseteq L_{\mathcal{B}}$,
  then $L_{\mathcal{A}'} \subseteq L_{\mathcal{B}}$

IRISA

Widening operator

If $L_{\mathcal{A}_1} \subseteq L_{\mathcal{A}_2}$:

1. consider the quotient automaton $\mathcal{A}_2/\simeq_k$
2. If $\mathcal{A}_1$ and $\mathcal{A}_2/\simeq_k$ have the same shape :

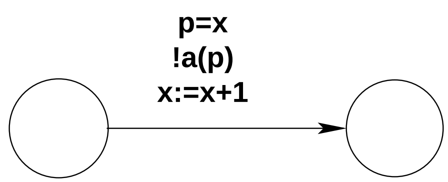


Outline

- Introduction : model and problematics
- Verification of CFSM
- Limitations and new model
- Representation of queues with messages carrying values
- Application to the verification of symbolic CFSM
- Conclusion

Non-relational analysis : the lattice of intervalls

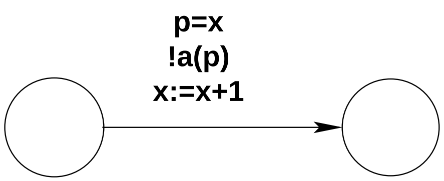
- The value of each variable is represented by an interval
- Lattice automata on $\Lambda = \Sigma \times \mathcal{I}$
- Abstract semantics :



$$\begin{array}{cc} x & \text{queue} \\ [0,2] & a([0,1]) \end{array} \longrightarrow \begin{array}{cc} x & \text{queue} \\ [1,3] & a([0,1]).a([0,2]) \end{array}$$

Relational analysis : the lattice of polyhedra

- The value of the varaibles are represented by a polyhedron
- Lattice automata on $\Lambda = \Sigma \times \mathcal{P}$
- Abstract semantics :



$$\overset{x}{\{0 \leq x \leq 2\}} \;\; \overset{\text{queue}}{a(\{0 \leq p = x-1\})} \longrightarrow \overset{x}{\{1 \leq x \leq 3\}} \;\; \overset{\text{queue}}{a(\{0 \leq p = x-2\}).a(\{0 \leq p = x-1\})}$$

- The lattice automaton is modified each time the value of x changes

Conclusion and ongoing works

- Sumary of results
  - Approximate analysis of protocols using mesaages carying values
  - Definition of a kind of automata dealing with infinite alphabets
- Ongoing work
  - Implementation of these algorithms
  - Experimentations
  - A new version of NBAC ?