Modélisation et vérification de systèmes probabilistes

Nathalie Bertrand

Projet VerTeCs, INRIA Rennes

30 septembre 2008

Outline

Formal methods for verification

Model-based testing : automatically generate a set of testing scenarios, given mathematical representations for system under test and specification.

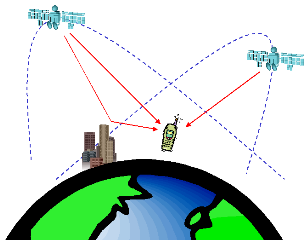
Static analysis : analyze properties of source code in a static manner, *i.e.* without unfolding all possible behaviours.

Automated proof : (partially automatically) prove correctness of a program through a logical reasoning using deduction rules.

Model checking : automatically prove that mathematical representation for the system satisfies model for the specification.

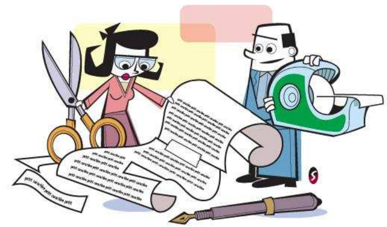
Principles of model checking

Does



system

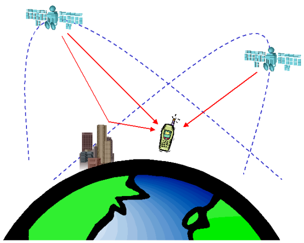
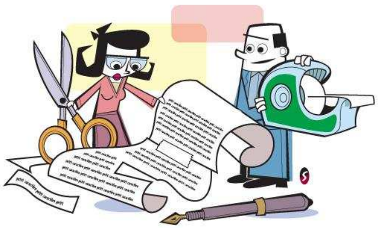
satisfy



specification

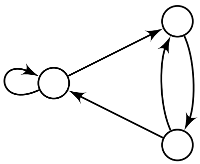
?

Principles of model checking

Does  satisfy  ?

system

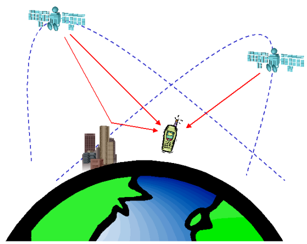
specification



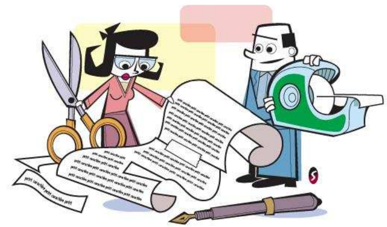
model

Principles of model checking

Does



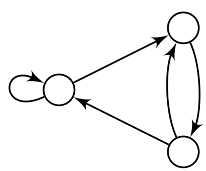
satisfy



?

system

specification



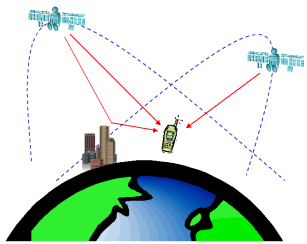
φ

model

formula

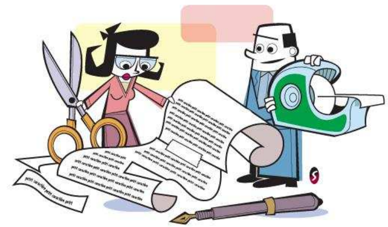
Principles of model checking

Does



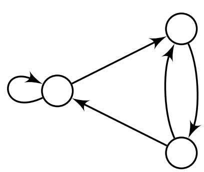
system

satisfy



specification

?



model

$\models$

model-checker

φ

formula

?

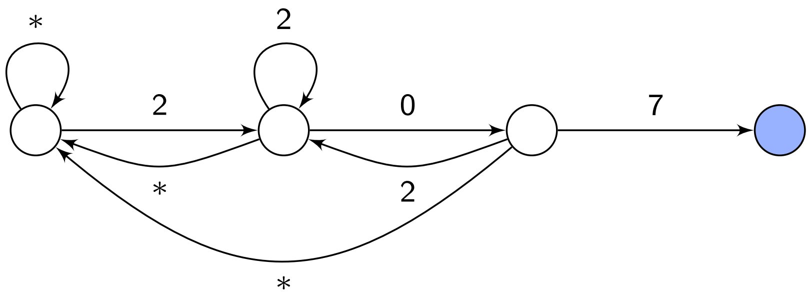
Models for systems

Systems under analysis are represented by transition systems.

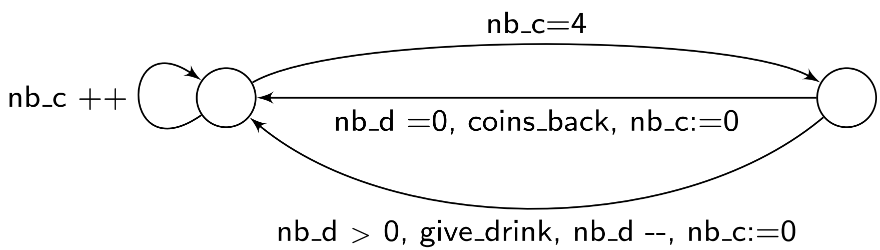
- finite automata
- pushdown automata
- counter automata
- timed automata
- hybrid automata
- Petri nets
- channel systems
- message sequence charts
- process algebra
- ...

Examples

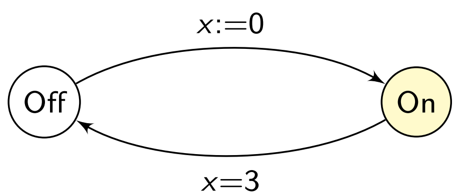
- A numerical code door lock



- A vending machine



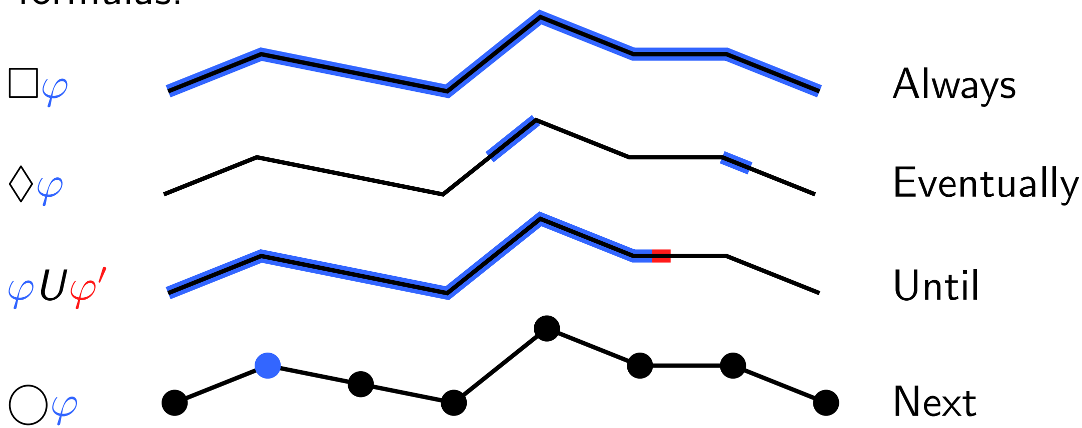
- A time-switch



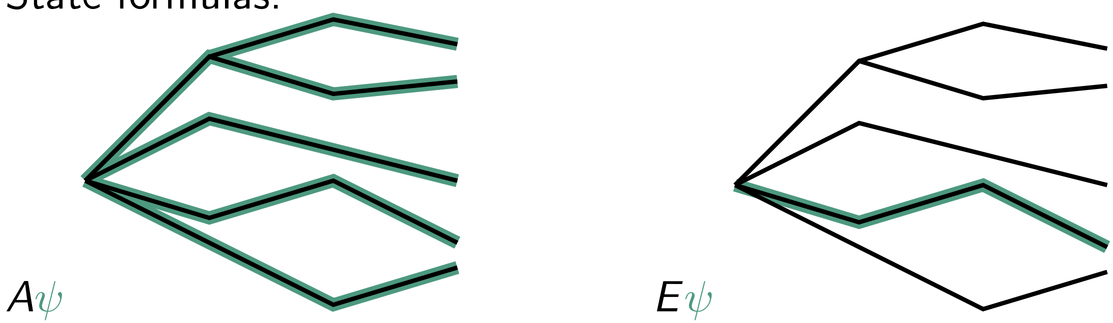
Models for specifications

Specifications are given by logical formulas (*e.g.* temporal logic [Pnueli 77])

- Path formulas:



- State formulas:



Examples

- A bad state is reachable.

$$E\Diamond(\text{bad\_state})$$

- Two processes cannot be in critical section simultaneously.

$$A\Box((\neg\text{critical\_section\_1}) \vee (\neg\text{critical\_section\_2}))$$

- The ATM does not give money as long as the pin code is uncorrect.

$$A((\neg\text{give\_money})U(\text{correct\_pin}))$$

- If the lift is called on the 6th floor, it will stop there.

$$A\Box(\text{call\_6} \Rightarrow (A\Diamond\ \text{stop\_6}))$$

- The barrier at the train crossing opens infinitely often.

$$A\Box\ A\Diamond(\text{barrier\_open})$$

The quest for good models

Tradeoff between expressivity and tractability

- Expressivity
  - representation of most aspects of systems
  - if possible concise representation
- Tractability
  - efficient algorithms
  - ... provided there is any

Outline

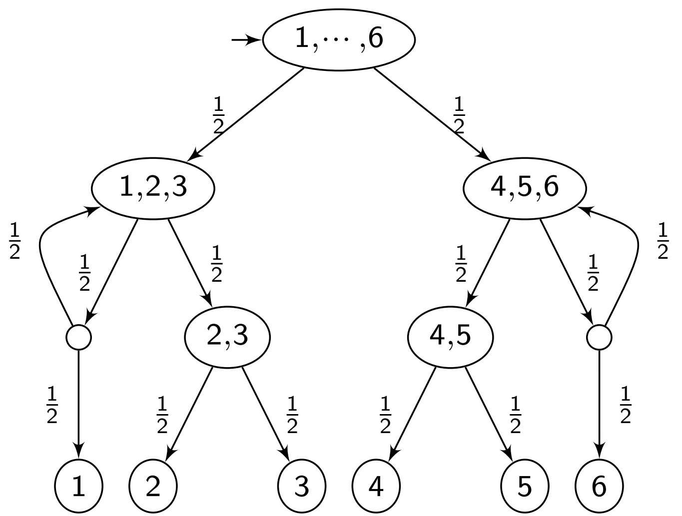
Discrete time Markov chains

Discrete time Markov chain (DTMC)

$\mathcal{M} = (S, \mathbb{P}, s_{\text{init}}, AP, L)$ with

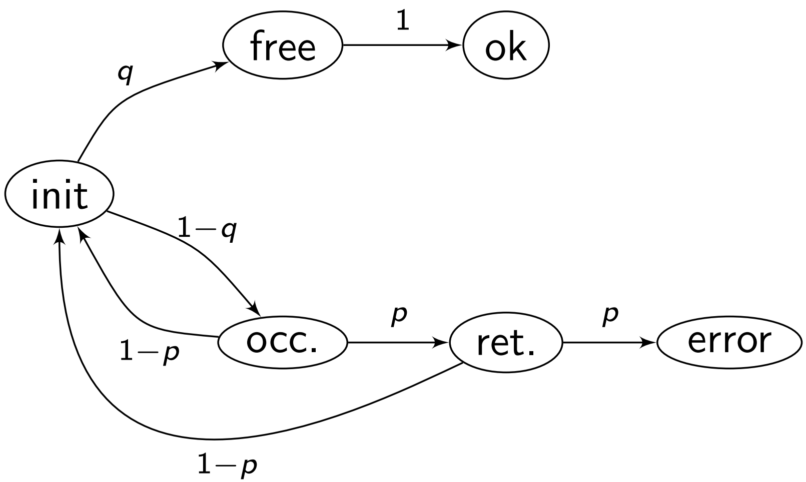
- S finite set of states, $\mathbb{P}$ probability matrix, s_{init} initial state,
- AP set of atomic propositions, $L : S \rightarrow 2^{AP}$ labeling function.

Example: Die simulated by a fair coin.



Zeroconf protocol

IP address automatic allocation



- with high probability (q), a free IP address is randomly chosen;
- otherwise, the host with the same address send an alert, which can be lost (with probability p)
- the new host sends n (here $n = 2$) probes to increase the reliability of the protocol.

Measure on DTMC paths

Probability of a finite path $\pi = s_0 s_1 \cdots s_n$:

$$Pr(\pi) = \prod_{i=0..n-1} \mathbb{P}_{i,i+1}.$$

Cylinder $\mathsf{Cyl}(\pi) = \{\pi_{max} | \pi \text{ prefix of } \pi_{max}\}$

$$Pr(\mathsf{Cyl}(\pi)) = Pr(\pi).$$

Probability measure

Pr is the unique probability measure on the σ-algebra generated by all $\mathsf{Cyl}(\pi)$, such that $Pr(\mathsf{Cyl}(\pi)) = Pr(\pi)$.

Reachability probabilities

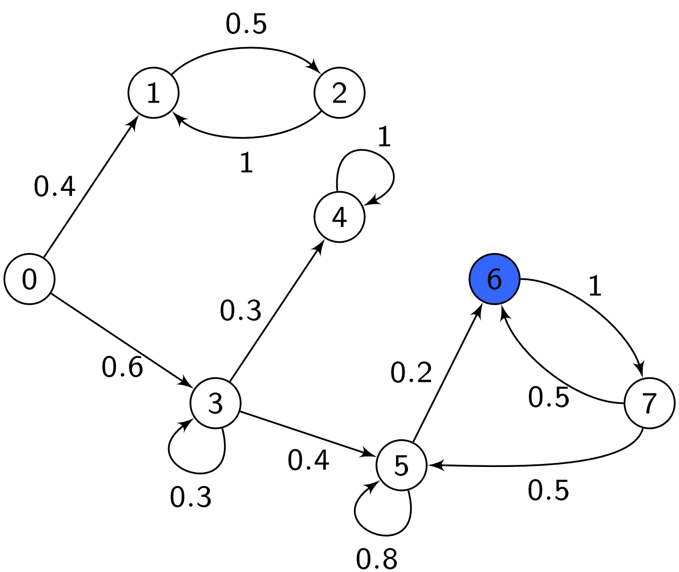
Goal: Compute $Pr(s_0 \models \Diamond T)$, for T set of target states.

For state $s \in S$, let $x_s = Pr(s \models \Diamond T)$.

- $x_s = 1$ if $s \in T$
- $x_s = 0$ if $s \not\models E\Diamond T$
- $x_s = \sum_{t \in S \setminus T} \mathbb{P}(s,t)\ x_t + \sum_{u \in T} \mathbb{P}(s,t)$

  $\longrightarrow$ resolution of a linear equations system

Example of probability computation



$Pr(0 \models \Diamond 6)$?

$x_6 = 1$

$x_1 = x_2 = x_4 = 0$

$x_0 = 0.6\ x_3$

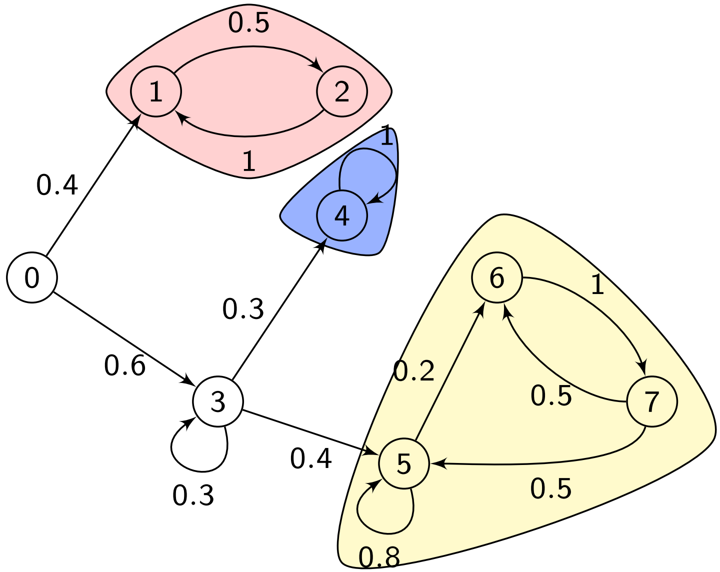
$x_3 = 0.3\ x_3 + 0.4\ x_5$

$x_5 = 0.8\ x_5 + 0.2$

$x_7 = 0.5\ x_5 + 0.5$

$x_0 = \frac{12}{35}$

Basic strongly connected components



Properties of BSCC

Let $\mathcal{C}$ be the set basic strongly connected components in $\mathcal{M}$.

- $Pr(s_0 \models \Diamond \bigcup_{C \in \mathcal{C}} C) = 1$,
- $\forall s \in C (\in \mathcal{C}),\ Pr(s \models \bigwedge_{t \in C} \Box \Diamond t) = 1$.

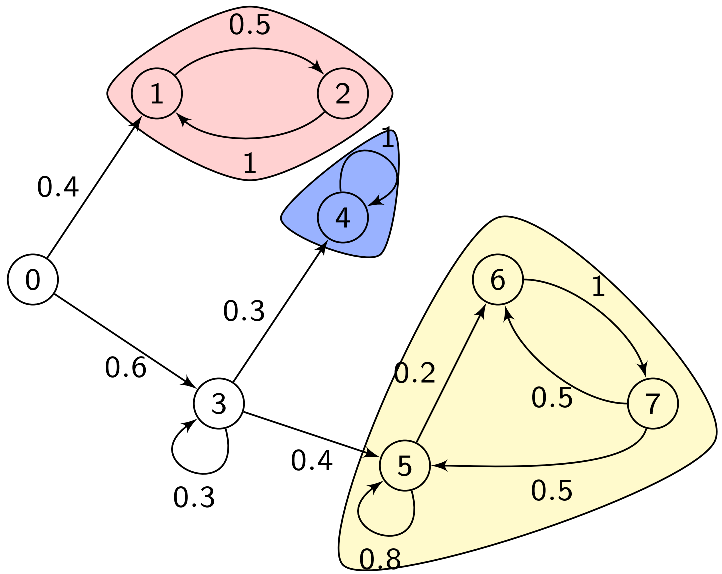
Prefix independent properties

Prefix independent

A property is said *prefix independent* if its validity only depends on the set of states that are visited infinitely often along a path.

For φ prefix independent,

$$Pr(s_0 \models \varphi) = Pr(s_0 \models \Diamond\{C \in \mathcal{C} | C \models \varphi\}).$$



$Pr(0 \models \Box\Diamond \text{ odd})?$

$Pr(0 \models \Box\Diamond \text{ odd}) =$
$Pr(0 \models \Diamond C_{1,2}) + Pr(0 \models \Diamond C_{5,6,7})$

$Pr(0 \models \Box\Diamond \text{ odd}) = \frac{26}{35}$

Outline

1. Introduction
2. Verification of probabilistic systems
   - Discrete time Markov chains
   - Markov decision processes
3. Channel systems with probabilistic losses
   - Channel systems
   - Probabilistic LCS
   - Nondeterministic and Probabilistic LCS
4. Conclusion

Markov decision processes

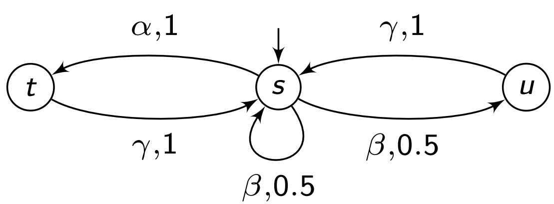
Markov decision process (MDP)

$\mathcal{M} = (S, Act, \mathbb{P}, s_{\text{init}}, AP, L)$ with

- S finite set of states, s_{init} initial state, Act set of actions,
- AP set of atomic propositions, $L : S \to 2^{AP}$ labeling function,
- $\mathbb{P} : S \times Act \times S \to [0, 1]$ transition probability function s.t.

$$\forall s \in S,\ \forall \alpha \in Act,\ \sum_{t \in S} \mathbb{P}(s, \alpha, t) \in \{0, 1\}.$$

Example

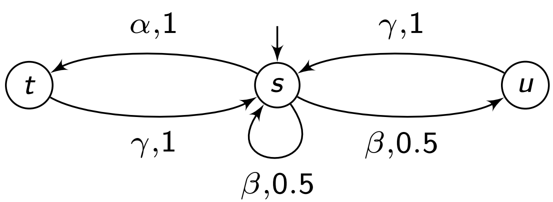


➜ nondeterministic choice in s between α and β.

Resolution of nondeterminism

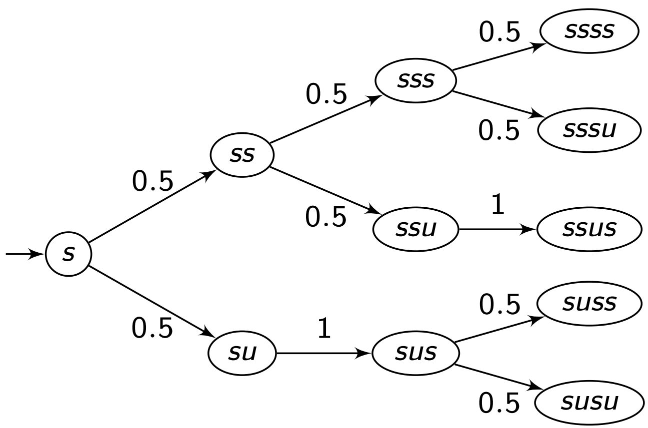
Scheduler

Let $\mathcal{M} = (S, Act, \mathbb{P}, s_{\text{init}}, AP, L)$ be an MDP. A scheduler for $\mathcal{M}$ is a function $\sigma : S^+ \to Act$ s.t. $\sigma(s_0 s_1 \cdots s_n)$ is enabled in s_n.



σ defined by:

- $\sigma(*s) = \beta$
- $\sigma(*t) = \sigma(*u) = \gamma$



MDP $\mathcal{M}$ + scheduler σ = Markov chain $\mathcal{M}_\sigma$

Reachability properties

Goal: Compute $Pr^{\max}(s \models \Diamond T) = \sup_\sigma Pr^\sigma(s \models \Diamond T)$.

Memoryless schedulers

There exists a memoryless (*i.e.* based only on the current state) scheduler that maximizes the probability to reach T.

For state $s \in S$, let $x_s = Pr^{\max}(s \models \Diamond T)$.

- $x_s = 1$ if $s \in T$
- $x_s = 0$ if $s \not\models E\Diamond T$
- $x_s = \max_{\alpha \in Act} \sum_{t \in S} \mathbb{P}(s, \alpha, t)\ x_t$

  $\longrightarrow$ resolution of a linear program

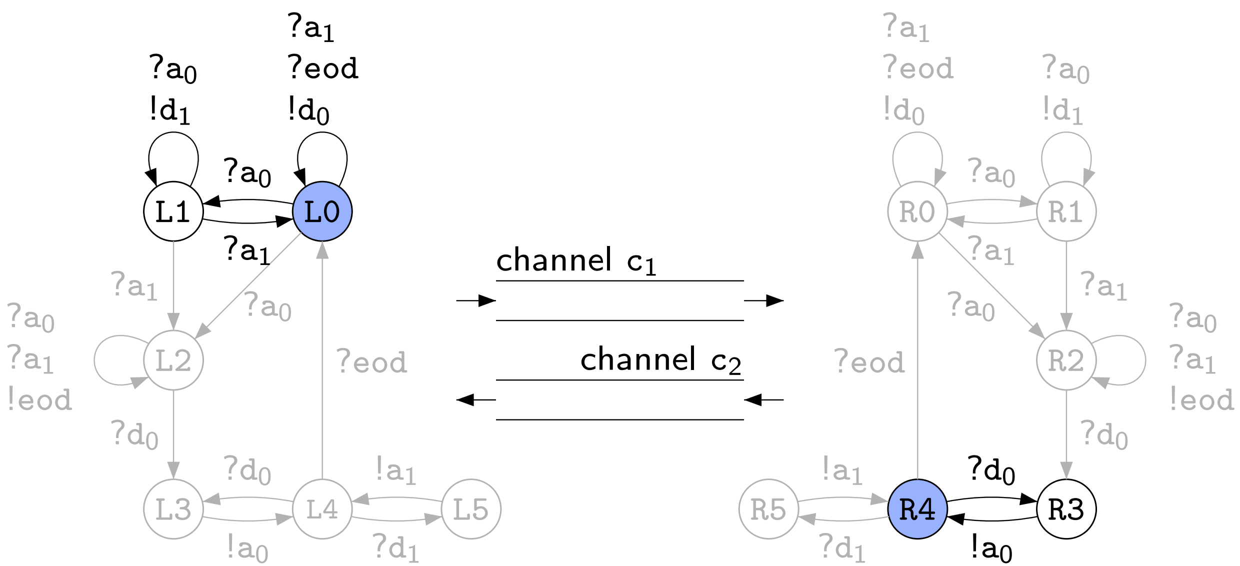
Outline

Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]

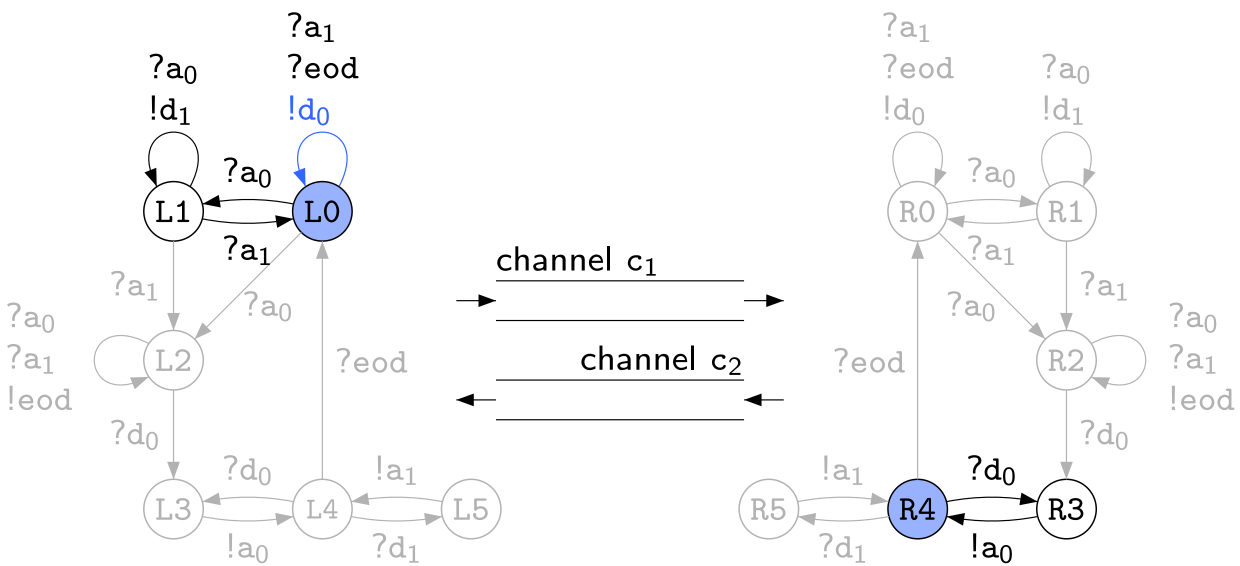
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



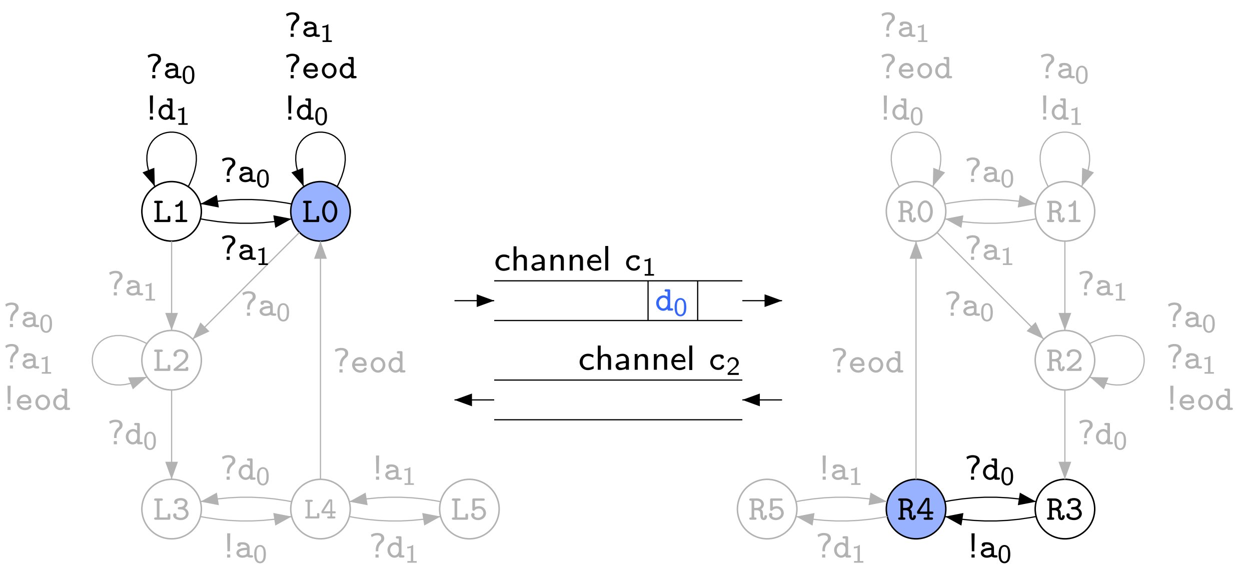
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



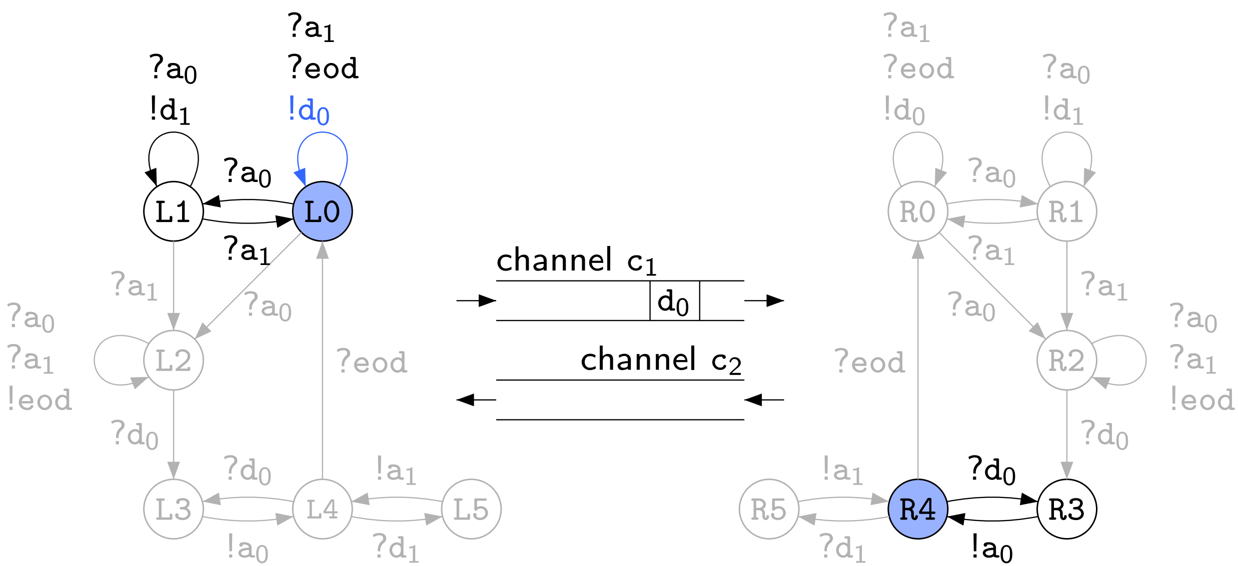
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



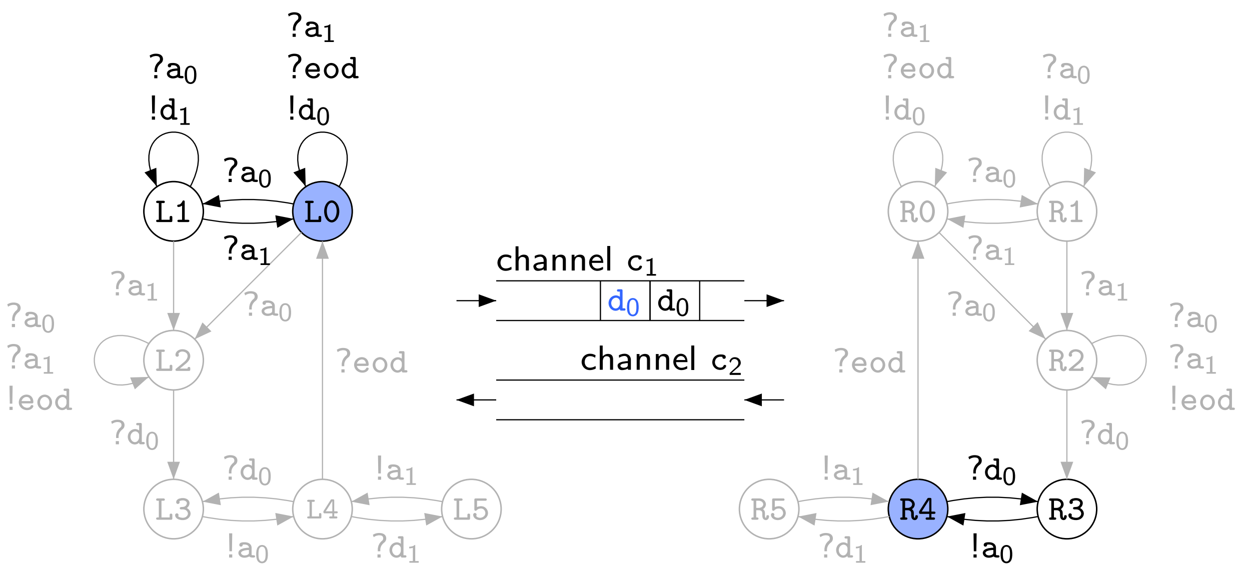
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



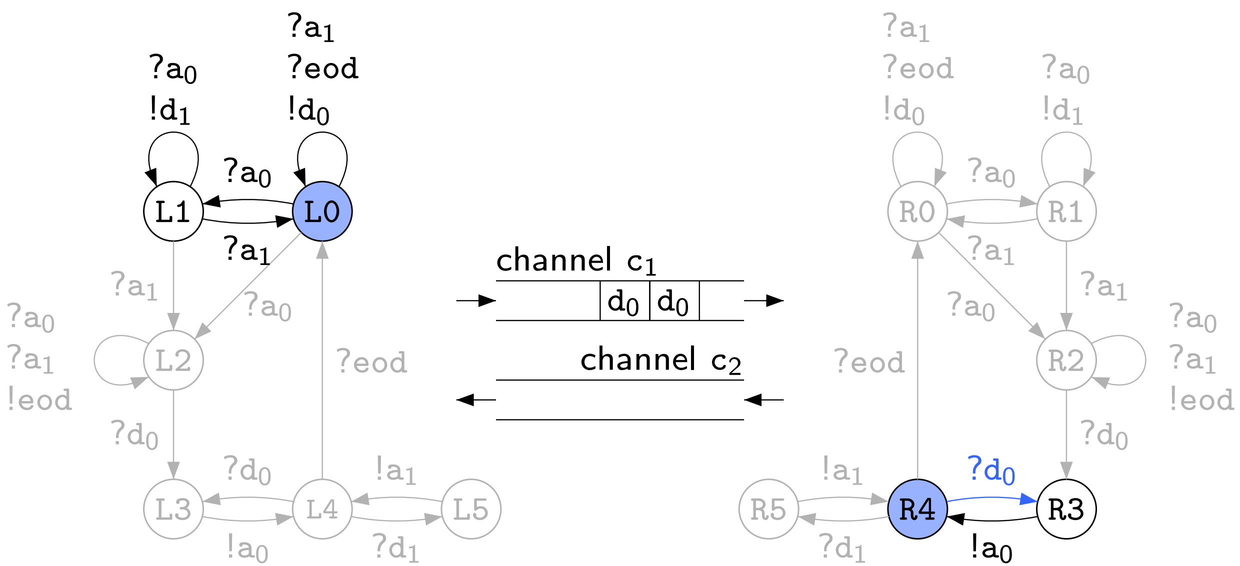
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



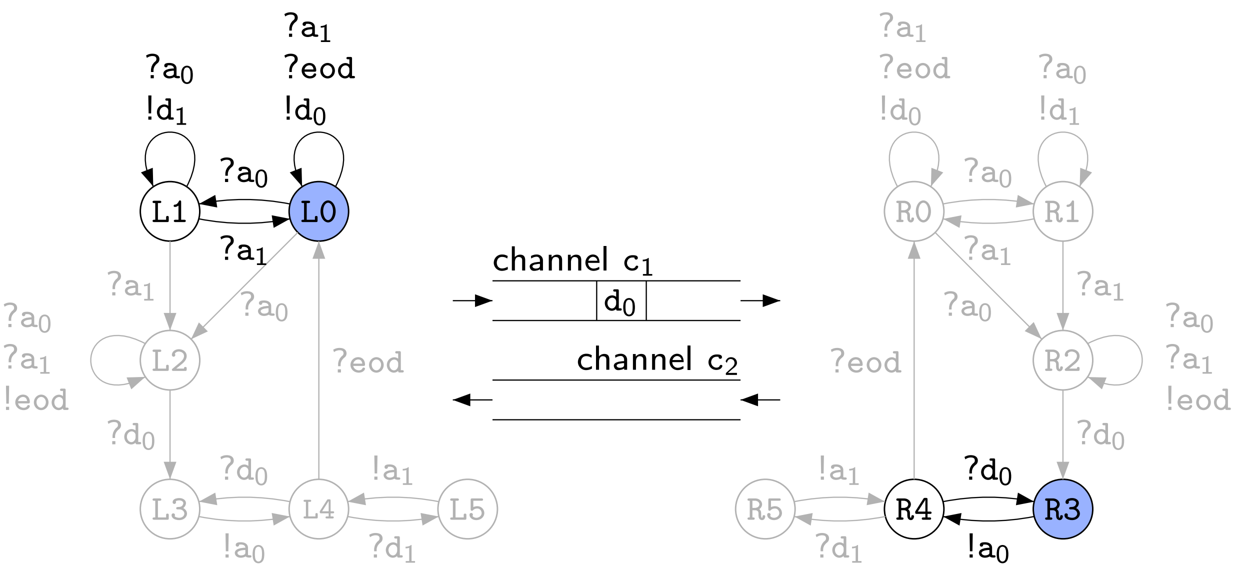
Channel systems

Finite processes that communicate via unbounded FIFO channels [Brand Zafiropulo 83]



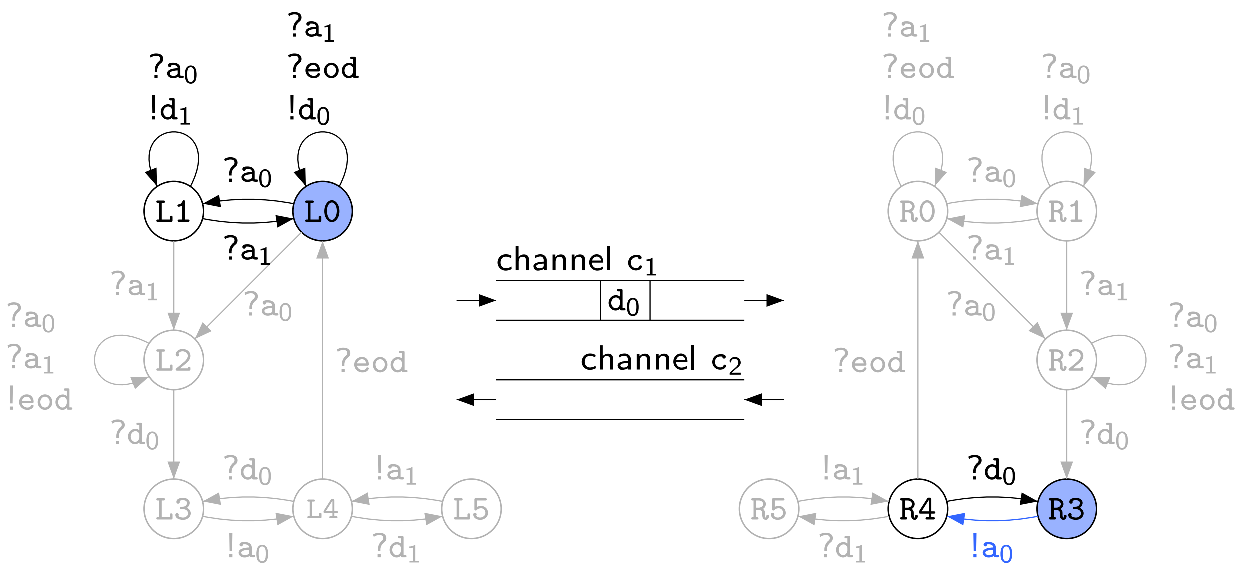
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



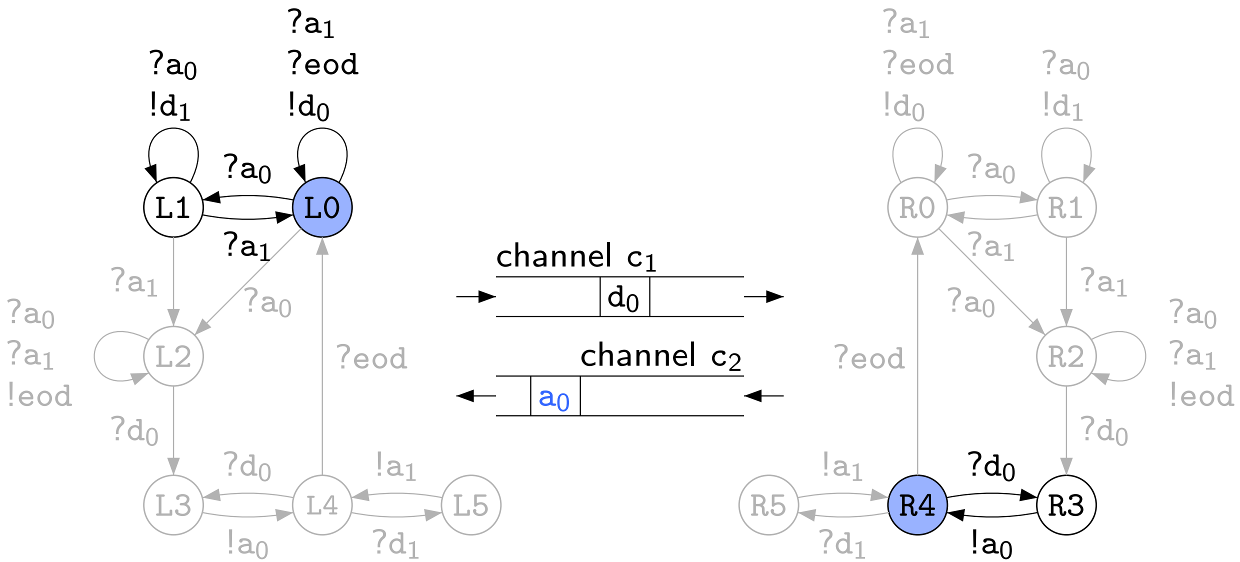
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



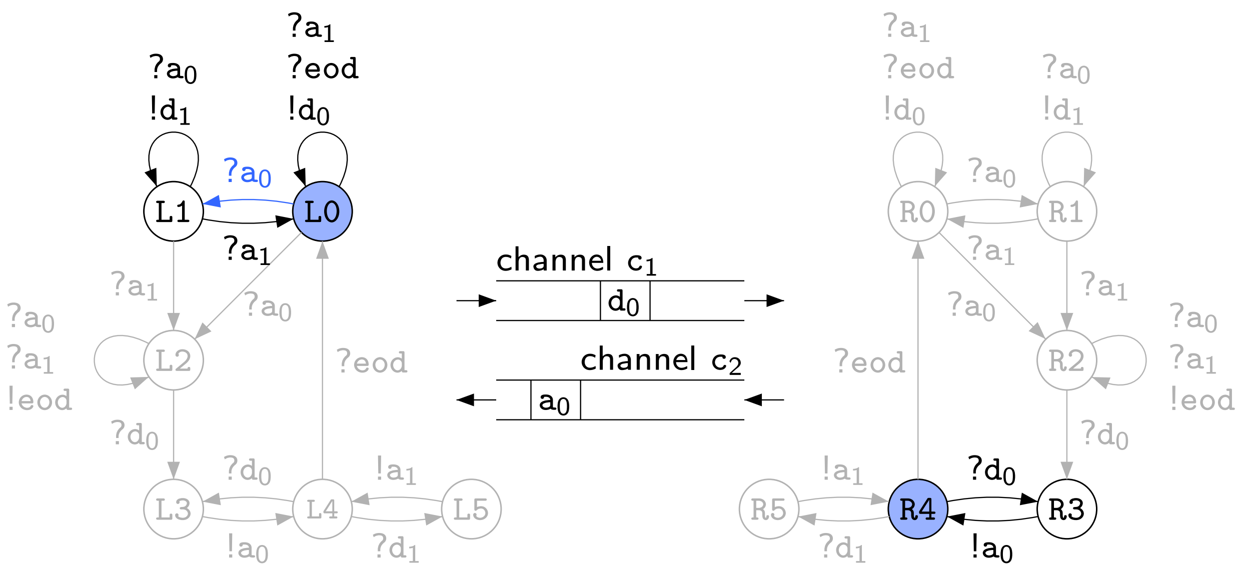
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



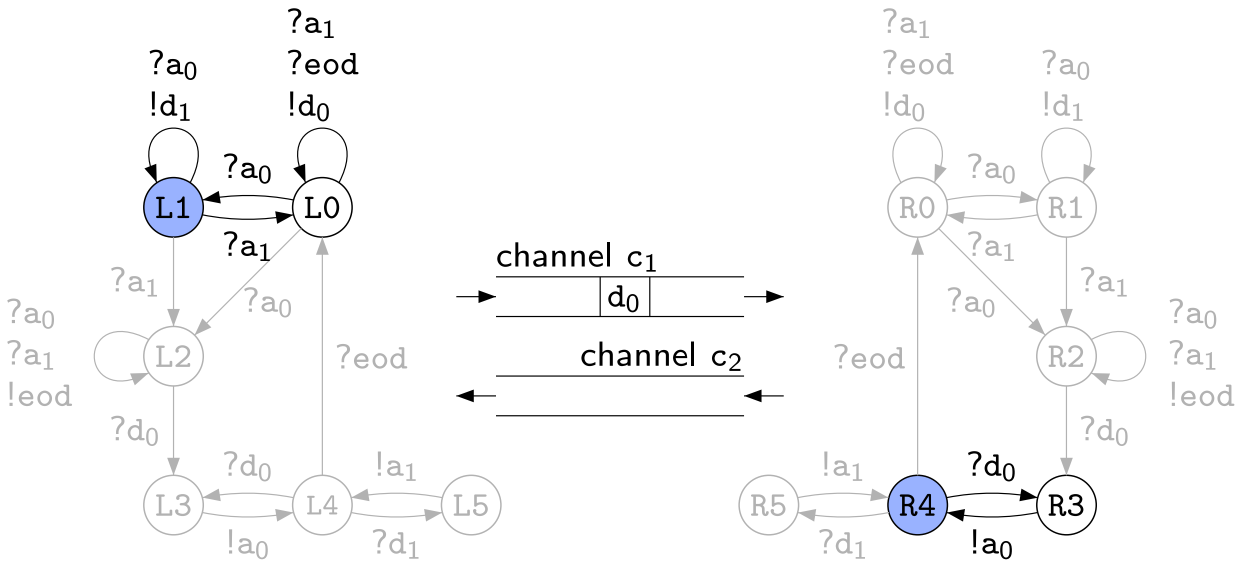
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



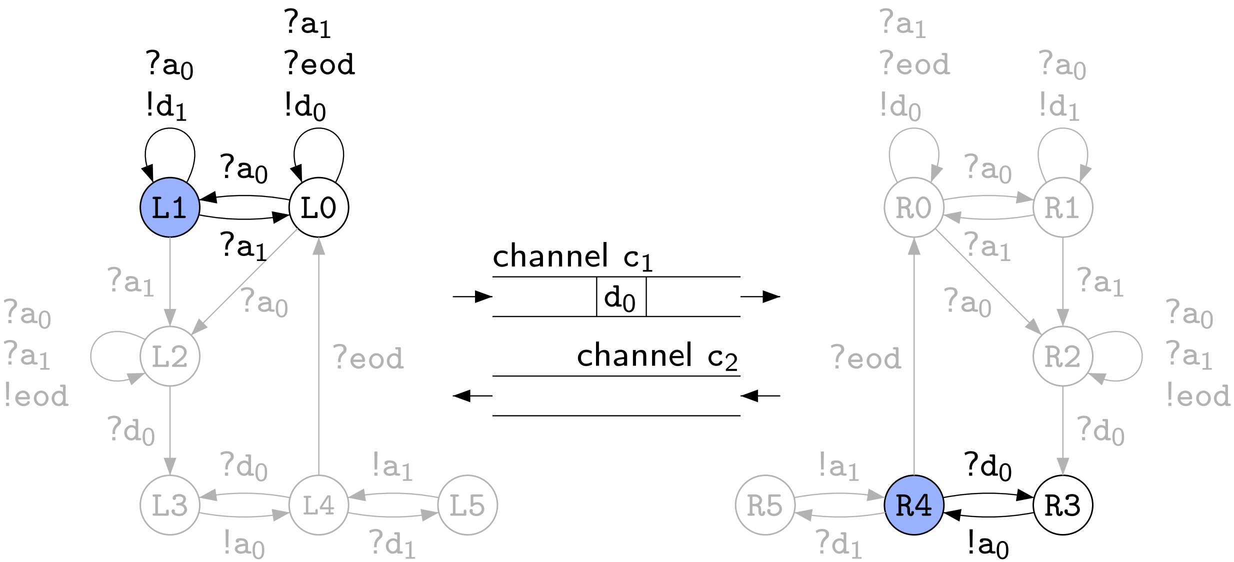
Channel systems

Finite processes that communicate via unbounded FIFO channels
[Brand Zafiropulo 83]



Channel systems

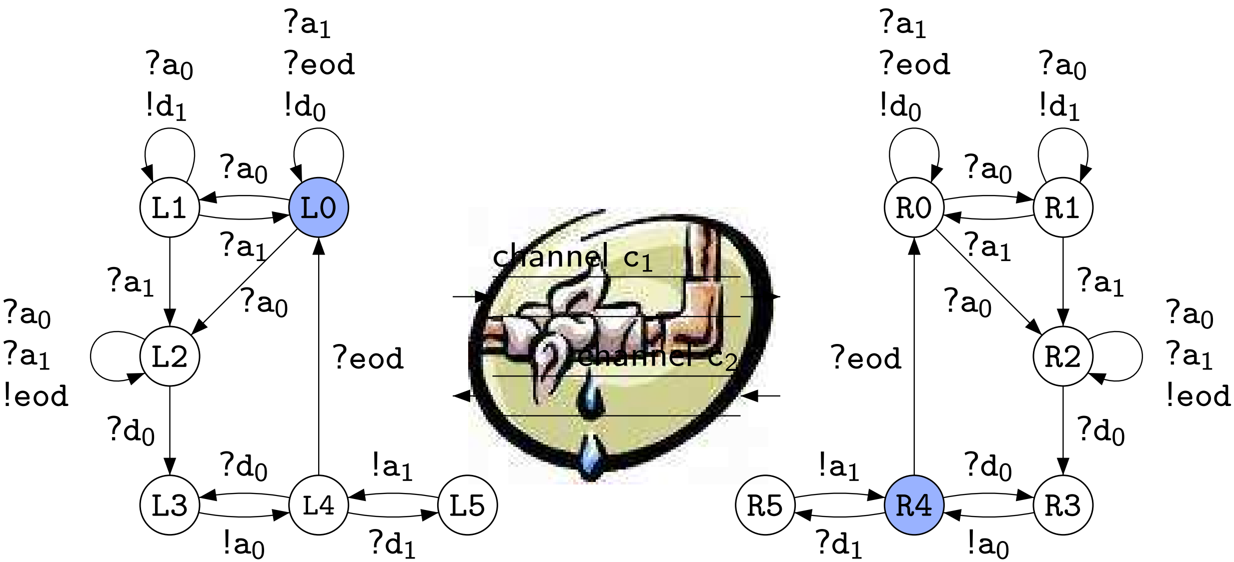
Finite processes that communicate via unbounded FIFO channels [Brand Zafiropulo 83]



- Turing-powerful

Lossy channel systems (LCS)

Unreliable channels: messages may be lost while in transit



- Safety properties are decidable (but with high complexity).

Outline

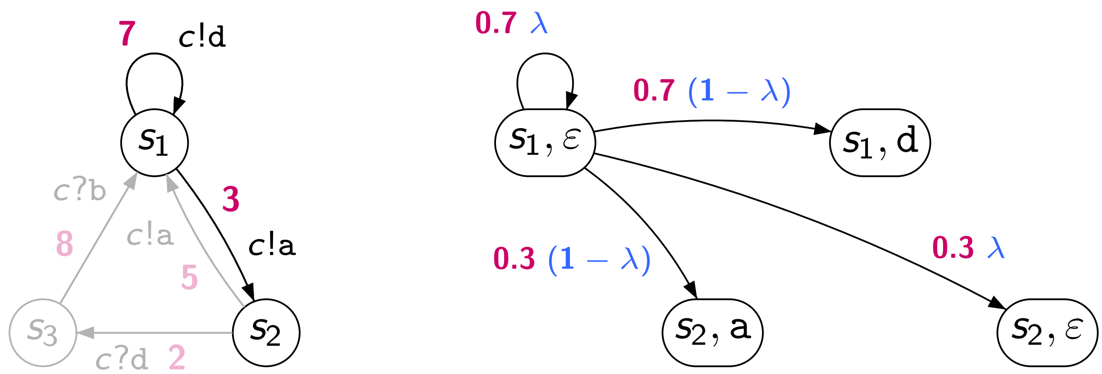
Probabilistic LCS

Markov chain model for channel systems with probabilistic losses.

Probabilistic LCS

A *Probabilistic LCS* is an LCS equipped with

- positive weights on rules, and
- a constant probability $\lambda \in]0, 1[$.



➜ Rules are chosen probabilistically according to weights.

➜ Message losses are independent events.

Qualitative verification of PLCS

PLCS are infinite-state Markov chains...

... but with a finite attractor!

Definition: Attractor

An attractor W in a Markov Chain M is a set of states that is visited almost surely from any starting state: $\forall s_0,\ \mathbb{P}(s_0 \models \Diamond W) = 1$
Hence $\forall s_0,\ \mathbb{P}(s_0 \models \Box\Diamond W) = 1$

Almost-sure model checking problem:

Given a PLCS $\mathcal{P}$, a configuration σ_0, an LTL formula φ

Question does $\mathbb{P}(\sigma_0 \models \varphi) = 1$?

Almost-sure model checking is decidable whatever $\lambda \in (0, 1)$.

Outline

Nondeterministic and Probabilistic LCS

Markov decision process model for channel systems.

- Choices between enabled actions are non-deterministic.
- Message losses are probabilistic.
- The two kinds of configurations (non-deterministic and probabilistic ones) alternate.

We are interested in qualitative questions such as:

$$\text{Does } \mathbb{P}(\varphi) = 1 \text{ under all schedulers ?}$$

Qualitative verification

- Bad news!
  Qualitative verification of LTL properties is undecidable for NPLCS.

- All is not lost...
  - Some problems are decidable for the full class of schedulers (mainly reachability and safety). Moreover, in these cases the two classes (full and finite-memory) coincide.
  - When restricting to finite-memory schedulers, qualitative probabilistic LTL model-checking is decidable

Outline

Conclusion

Much work has been done in model-checking of probabilistic systems

- probabilistic finite automata [Paz 71]
- probabilistic pushdown automata [Esparza Kučera Mayr 04]
- probabilistic counter automata
- probabilistic timed automata [Kwiatkowskia *et al.* 01]
- probabilistic Petri nets
- probabilistic channel systems [Iyer Narashima 97]
- ...

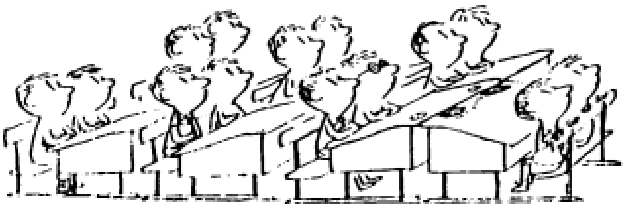
Conclusion

Much work has been done in model-checking of probabilistic systems

- probabilistic finite automata [Paz 71]
- probabilistic pushdown automata [Esparza Kučera Mayr 04]
- probabilistic counter automata
- probabilistic timed automata [Kwiatkowskia *et al.* 01]
- probabilistic Petri nets
- probabilistic channel systems [Iyer Narashima 97]
- ...

Recently, two topics of interest (at least for me)

- probabilistic Büchi automata [Baier Größer 05]
- probabilistic semantics for timed automata [Baier *et al.* 07]



Thank you for your attention!
Any questions?

Two references

BK08 Christel Baier and Joost-Pieter Katoen. Principles of Model-Checking. MIT Press, 2008.

BBF+01 Béatrice Bérard, Michel Bidoit, Alain Finkel, François Laroussinie, Antoine Petit, Laure Petrucci and Philippe Schnoebelen. Model-Checking techniques and Tools. Springer, 2001.