The Verification of Probabilistic Lossy Channel Systems

Ph. Schnoebelen

http://www.lsv-ens-cachan.fr/~phs

Lab. Specification & Verification (LSV)
ENS de Cachan & CNRS
Cachan, France

Outline of the Talk

Channel Systems With Unreliable Channels

Probabilistic Lossy Channel Systems

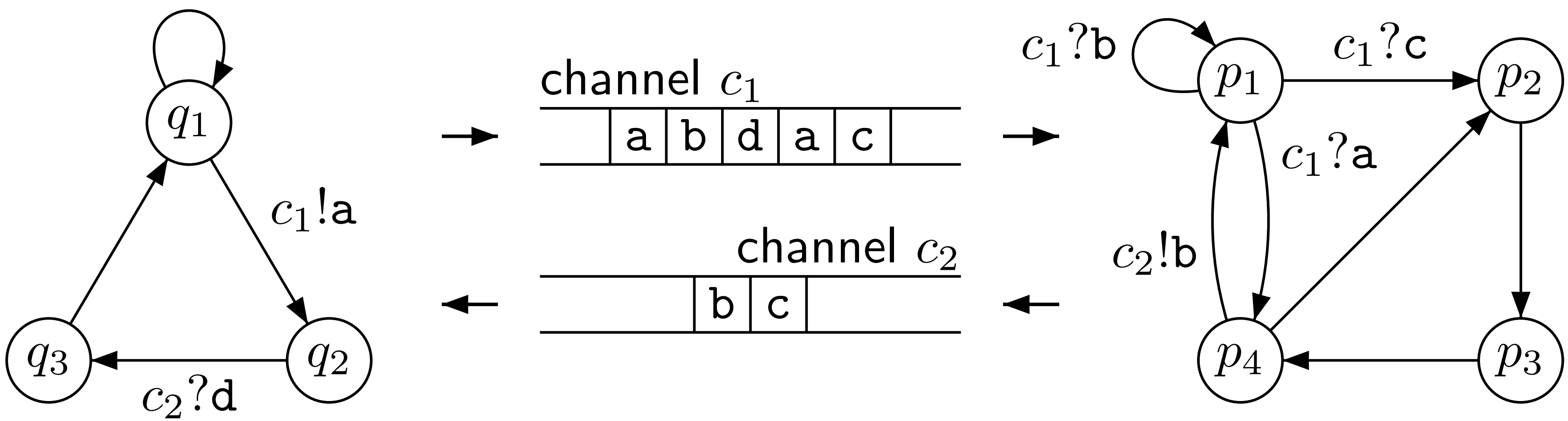
Qualitative Verification

Quantitative Verification

Adversarial Verification

Channel Systems

A.k.a. "communicating finite-state machines"

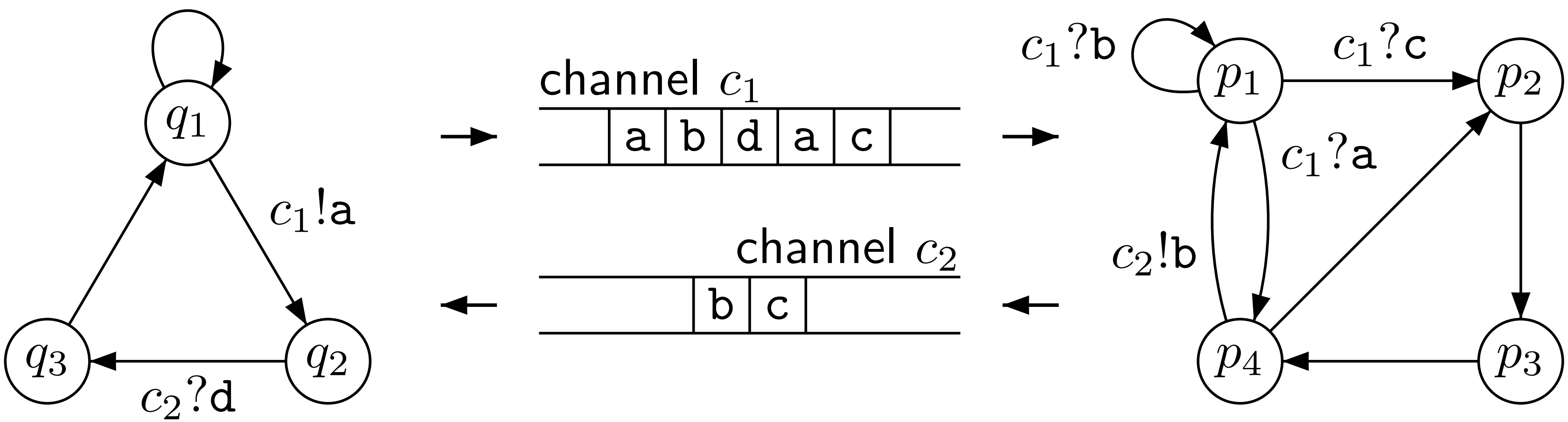


Natural model for asynchronous communication protocols: SDL, Estelle

[von Bochmann 1978; Brand & Zafiropulo 1983]

Channel Systems

A.k.a. "communicating finite-state machines"



Natural model for asynchronous communication protocols: SDL, Estelle

[von Bochmann 1978; Brand & Zafiropulo 1983]

Turing powerful!

Hence fully automated verification, aka model checking, is impossible on principle grounds.

The Paradox of Lossy Channel Systems

Model checking becomes possible when you assume channels are unreliable (can lose messages). [Finkel 94; Abdulla & Jonsson 96b]

The Paradox of Lossy Channel Systems

Model checking becomes possible when you assume channels are unreliable (can lose messages). [Finkel 94; Abdulla & Jonsson 96b]

Termination and more general eventuality properties are decidable.

Reachability and more general safety properties are decidable.

The Paradox of Lossy Channel Systems

Model checking becomes possible when you assume channels are unreliable (can lose messages). [Finkel 94; Abdulla & Jonsson 96b]

Termination and more general eventuality properties are decidable.

Reachability and more general safety properties are decidable.

These lossy channel systems are well-suited to the analysis of asynchronous protocols that are designed to cope with message losses.

Channel Systems: Perfect

W.l.o.g., we only consider systems made of one component and several channels.

$S = \langle Q, \Sigma, C, \Delta \rangle$ with
- $Q = \{q, \ldots\}$, the *control states*
- $\Sigma = \{a, b, \ldots\}$, the *messages*
- $C = \{c_1, c_2, \ldots, c_n\}$, the *channels*
- $\Delta \subseteq Q \times C \times \{?, !\} \times \Sigma \times Q$, the *rules*

Rules in Δ written e.g. as "$(q, c!a, q')$"

A *configuration* of S: $\sigma = \langle q, u_1, \ldots, u_n \rangle$

Perfect steps: $\langle q, u_1, \ldots, u_n \rangle \rightarrow \langle r, v_1, \ldots, v_n \rangle$ if
— $(q, c_i?a, r)$ is a rule, $u_i = a.v_i$ and $v_j = u_j$ for $j \neq i$, or
— $(q, c_i!a, r)$ is a rule, $v_i = u_i.a$ and $v_j = u_j$ for $j \neq i$.
NB: no test for emptiness

Channel Systems: Lossy

Subword ordering: $abba \sqsubseteq abracadabra$

Subword relation for configurations: $\sigma \sqsubseteq \sigma'$

Lossy steps: $\sigma \rightarrow_{\text{lossy}} \sigma' \overset{\text{def}}{\Leftrightarrow} \sigma \sqsupseteq \delta \rightarrow_{\text{perf}} \delta' \sqsupseteq \sigma'$

Corollary: If $\sigma_1 \rightarrow \sigma_2$ then $\sigma_1' \rightarrow \sigma_2'$ for any $\sigma_1' \sqsupseteq \sigma_1$ and $\sigma_2' \sqsubseteq \sigma_2$.

Channel Systems: Lossy

Subword ordering: $abba \sqsubseteq abracadabra$

Subword relation for configurations: $\sigma \sqsubseteq \sigma'$

Lossy steps: $\sigma \rightarrow_{\text{lossy}} \sigma' \overset{\text{def}}{\Leftrightarrow} \sigma \sqsupseteq \delta \rightarrow_{\text{perf}} \delta' \sqsupseteq \sigma'$

Corollary: If $\sigma_1 \rightarrow \sigma_2$ then $\sigma_1' \rightarrow \sigma_2'$ for any $\sigma_1' \sqsupseteq \sigma_1$ and $\sigma_2' \sqsubseteq \sigma_2$.

Lemma [Higman 1952]: the subword ordering is a well-quasi-order (wqo), i.e. any infinite sequence $u_0, u_1, u_2, \ldots$ of words has an infinite increasing subsequence $u_{i_0} \sqsubseteq u_{i_1} \sqsubseteq u_{i_2} \sqsubseteq \cdots$

$\Rightarrow$ Applies equivalently to configurations of S ordered by $\sqsubseteq$.

Corollary. Any set of configurations has a finite number of minimal elements.

Lossy Channel Systems Are Not Trivial

Recurrent reachability is undecidable [Abdulla & Jonsson 1996a].

(Hence model checking of temporal properties is undecidable too.)

Boundedness is undecidable [Mayr 2003] (see also [DJS 1999]).

All behavioral equivalences are undecidable [S. 2001].

Lossy Channel Systems Are Not Trivial

Recurrent reachability is undecidable [Abdulla & Jonsson 1996a].

(Hence model checking of temporal properties is undecidable too.)

Boundedness is undecidable [Mayr 2003] (see also [DJS 1999]).

All behavioral equivalences are undecidable [S. 2001].

Additionally, all decidable problems are nonprimitive recursive [S. 2002].

Lossy Channel Systems Are Not Trivial

Recurrent reachability is undecidable [Abdulla & Jonsson 1996a].

(Hence model checking of temporal properties is undecidable too.)

Boundedness is undecidable [Mayr 2003] (see also [DJS 1999]).

All behavioral equivalences are undecidable [S. 2001].

Additionally, all decidable problems are nonprimitive recursive [S. 2002].

In practice, the main limitation for verification is the undecidability of model checking properties involving liveness and/or fairness.

Probabilistic Lossy Channel Systems

Basic idea is to assume that *message losses follow probabilistic rules*, e.g. there is a known “failure rate” [PN 1997].

More realist than just non-deterministic losses (protocols are designed with the idea that losses are not that likely).

Randomization helps in general.
(Here by ruling out unrealistically nasty behaviors).

Probabilistic Lossy Channel Systems

Basic idea is to assume that *message losses follow probabilistic rules*, e.g. there is a known "failure rate" [PN 1997].

More realist than just non-deterministic losses (protocols are designed with the idea that losses are not that likely).

Randomization helps in general.
(Here by ruling out unrealistically nasty behaviors).

Definition: A Probabilistic LCS (a "PLCS") is
a LCS equipped with
– positive weights on rules, and
– a constant probability $p_{\text{loss}} \in (0, 1)$.

Probabilistic Lossy Channel Systems

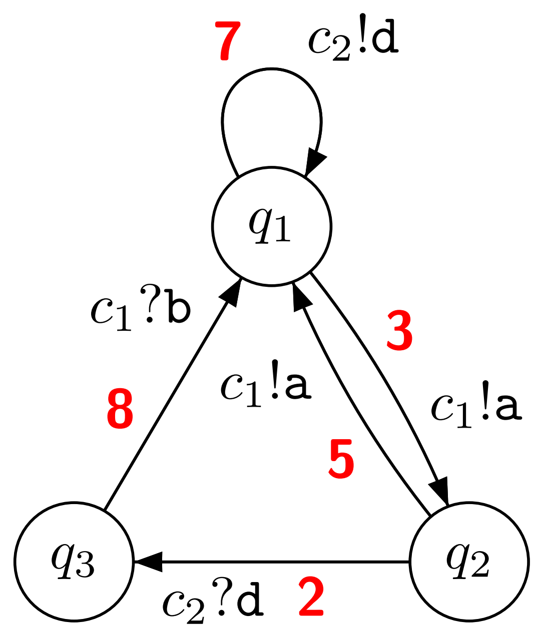
Basic idea is to assume that *message losses follow probabilistic rules*, e.g. there is a known "failure rate" [PN 1997].

More realist than just non-deterministic losses (protocols are designed with the idea that losses are not that likely).

Randomization helps in general.
(Here by ruling out unrealistically nasty behaviors).

Definition: A Probabilistic LCS (a "PLCS") is
a LCS equipped with
– positive weights on rules, and
– a constant probability $p_{\text{loss}} \in (0, 1)$.

$p_{\text{loss}} = \mathbf{.01} +$



Markovian Semantics

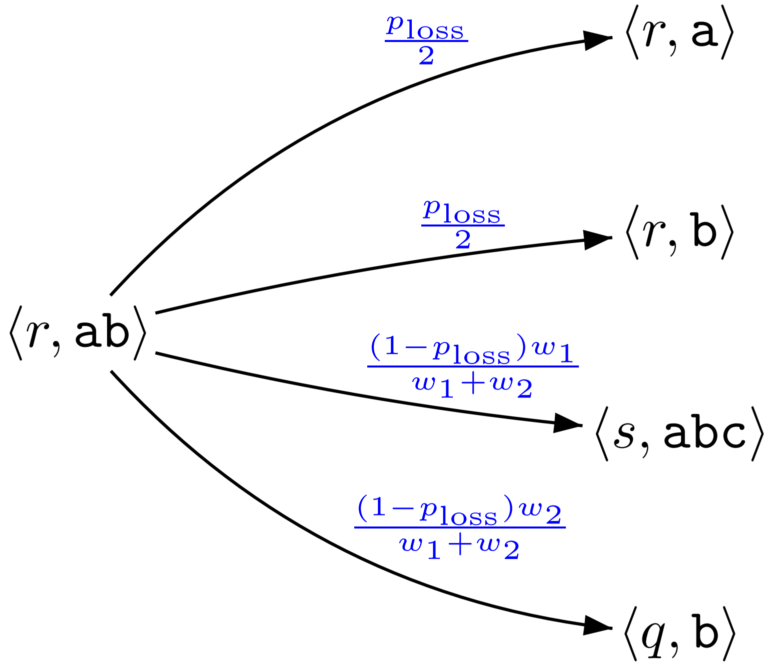
Semantics in form of a countable Markov chain.

Two interpretations of p_{loss}:
global-fault model [PN97, BE99] vs. local-fault model [BS03,AR03].

Markovian Semantics

Semantics in form of a countable Markov chain.

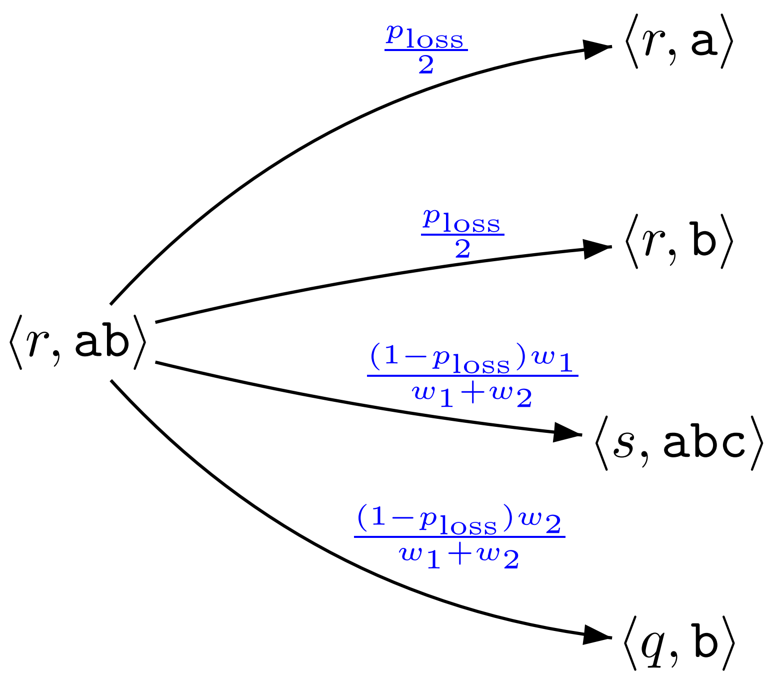
Two interpretations of p_{loss}:
global-fault model [PN97, BE99] vs. local-fault model [BS03,AR03].

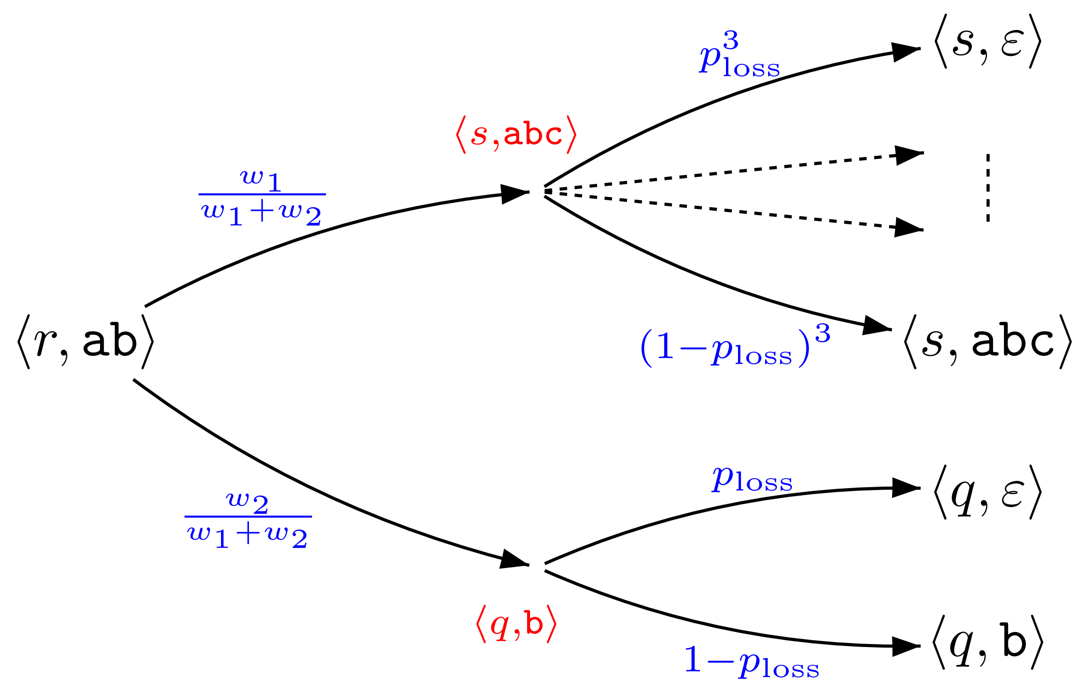


Markovian Semantics

Semantics in form of a countable Markov chain.

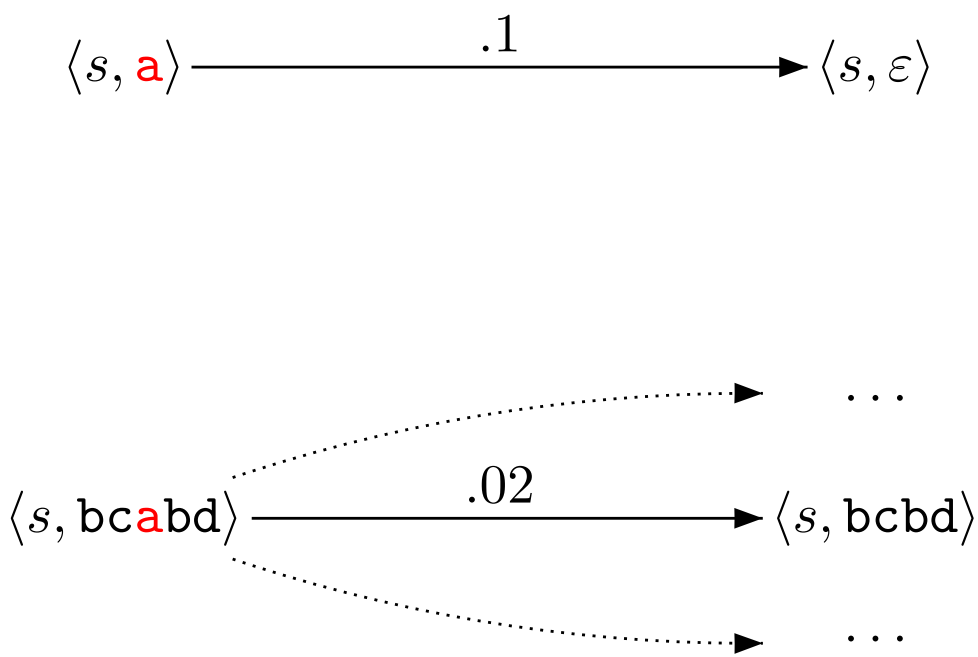
Two interpretations of p_{loss}:
global-fault model [PN97, BE99] vs. local-fault model [BS03,AR03].





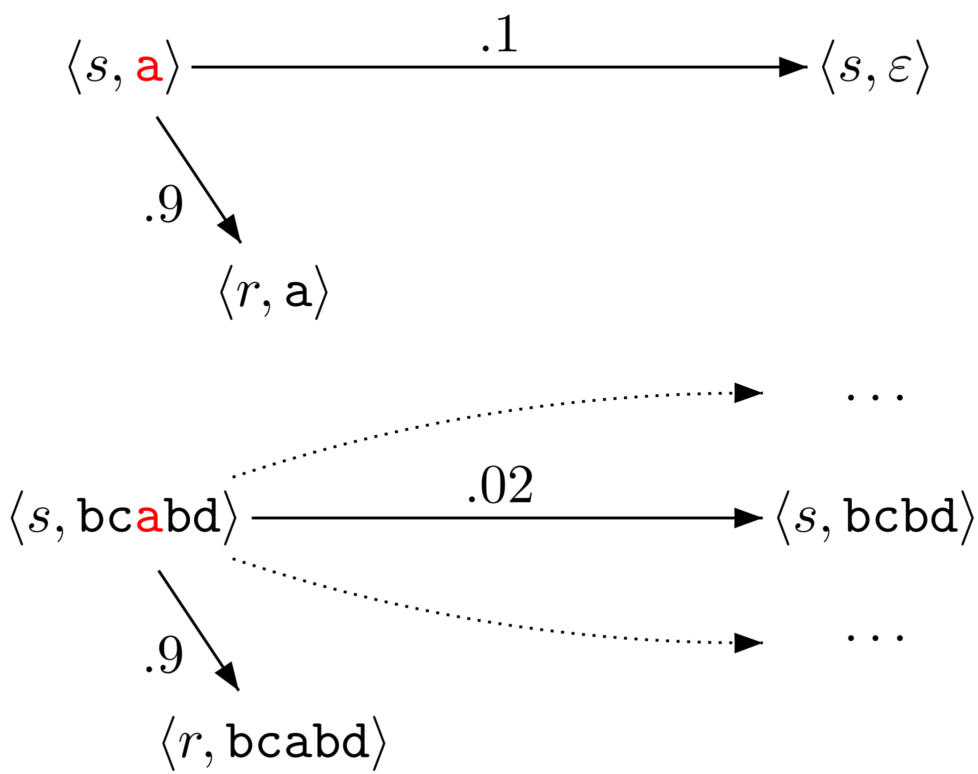
Global-Fault Model Is Less Realist

Assume $p_{\text{loss}} = .1$.



Global-Fault Model Is Less Realist

Assume $p_{\mathrm{loss}} = .1$.



Message losses are not independent events!

Qualitative Verification of PLCS's

Question: can we check whether $\mathbb{P}(S \models \varphi) = 1$, i.e. whether φ holds almost surely, for φ an LTL property?

Qualitative Verification of PLCS's

Question: can we check whether $\mathbb{P}(S \models \varphi) = 1$, i.e. whether φ holds almost surely, for φ an LTL property?

Theorem [BE99]: Qualitative verification is decidable for the global-fault model, assuming $p_{\text{loss}} \geq .5$.

NB: When $p_{\text{loss}} < .5$, qualitative verification is undecidable [ABPJ00].

Qualitative Verification of PLCS's

Question: can we check whether $\mathbb{P}(S \models \varphi) = 1$, i.e. whether φ holds almost surely, for φ an LTL property?

Theorem [BE99]: Qualitative verification is decidable for the global-fault model, assuming $p_{\text{loss}} \geq .5$.

NB: When $p_{\text{loss}} < .5$, qualitative verification is undecidable [ABPJ00].

Theorem [BS03,AR03]: Qualitative verification is decidable for the local-fault model, whatever $p_{\text{loss}} > 0$.

Qualitative Verification of PLCS's

Question: can we check whether $\mathbb{P}(S \models \varphi) = 1$, i.e. whether φ holds almost surely, for φ an LTL property?

Theorem [BE99]: Qualitative verification is decidable for the global-fault model, assuming $p_{\text{loss}} \geq .5$.

NB: When $p_{\text{loss}} < .5$, qualitative verification is undecidable [ABPJ00].

Theorem [BS03,AR03]: Qualitative verification is decidable for the local-fault model, whatever $p_{\text{loss}} > 0$.

Furthermore, whether $\mathbb{P}(S \models \varphi) = 1$ does not depend on p_{loss}, on the weights, on the choice of a model.

Finite attractors play an essential rôle...

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Examples of finite attractors:

1. Random walk on the grid $\mathbb{Z}^2$: any point.

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Examples of finite attractors:
1. Random walk on the grid $\mathbb{Z}^2$: any point.
2. Finite Markov chain: any set with one configuration from each bottom strongly connected component.

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Examples of finite attractors:
1. Random walk on the grid $\mathbb{Z}^2$: any point.
2. Finite Markov chain: any set with one configuration from each bottom strongly connected component.
3. PLCS's assuming global-fault model: when $p_{\text{loss}} \geq .5$, the set of all empty configurations is an attractor.

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Examples of finite attractors:
1. Random walk on the grid $\mathbb{Z}^2$: any point.
2. Finite Markov chain: any set with one configuration from each bottom strongly connected component.
3. PLCS's assuming global-fault model: when $p_{\text{loss}} \geq .5$, the set of all empty configurations is an attractor.
4. PLCS's assuming local-fault model: the set of all empty configurations is an attractor whatever $p_{\text{loss}} > 0$.

Finite attractors

An *attractor* is a set $W_0 \subseteq W$ of configurations s.t.
for all $\sigma \in W$, $\mathbb{P}(\sigma \models \Diamond W_0) = 1$.

NB: Then $\mathbb{P}(\sigma \models \Box\Diamond W_0) = 1$ for all σ.

Examples of finite attractors:
1. Random walk on the grid $\mathbb{Z}^2$: any point.
2. Finite Markov chain: any set with one configuration from each bottom strongly connected component.
3. PLCS's assuming global-fault model: when $p_{\text{loss}} \geq .5$, the set of all empty configurations is an attractor.
4. PLCS's assuming local-fault model: the set of all empty configurations is an attractor whatever $p_{\text{loss}} > 0$.

We show how finite attractors provide finitary conditions for properties of countable Markov chains.

Algorithmic Ideas

A method for checking $\mathbb{P}(S \models \varphi) = 1$:

1. Reduce $\mathbb{P}(S \models \varphi)$ to $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i)$ for $S' = S \otimes \mathcal{A}_\varphi$.

Algorithmic Ideas

A method for checking $\mathbb{P}(S \models \varphi) = 1$:

1. Reduce $\mathbb{P}(S \models \varphi)$ to $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i)$ for $S' = S \otimes \mathcal{A}_\varphi$.

2. Build the *finite* graph $G_{S'}$ of all configurations from W_0 with an edge $\langle s, \varepsilon \rangle \longrightarrow \langle r, \varepsilon \rangle$ if $\langle r, \varepsilon \rangle$ is reachable from $\langle s, \varepsilon \rangle$.

Algorithmic Ideas

A method for checking $\mathbb{P}(S \models \varphi) = 1$:

1. Reduce $\mathbb{P}(S \models \varphi)$ to $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i)$ for $S' = S \otimes \mathcal{A}_\varphi$.

2. Build the *finite* graph $G_{S'}$ of all configurations from W_0 with an edge $\langle s, \varepsilon \rangle \rightarrow \langle r, \varepsilon \rangle$ if $\langle r, \varepsilon \rangle$ is reachable from $\langle s, \varepsilon \rangle$.

3. Since an infinite run of S' almost surely visits W_0 infinitely often, it almost surely ends up visiting a BSSC B of $G_{S'}$, and then it almost surely visits all configurations reachable from B infinitely often.

Algorithmic Ideas

A method for checking $\mathbb{P}(S \models \varphi) = 1$:

1. Reduce $\mathbb{P}(S \models \varphi)$ to $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i)$ for $S' = S \otimes \mathcal{A}_\varphi$.

2. Build the *finite* graph $G_{S'}$ of all configurations from W_0 with an edge $\langle s, \varepsilon\rangle \to \langle r, \varepsilon\rangle$ if $\langle r, \varepsilon\rangle$ is reachable from $\langle s, \varepsilon\rangle$.

3. Since an infinite run of S' almost surely visits W_0 infinitely often, it almost surely ends up visiting a BSSC B of $G_{S'}$, and then it almost surely visits all configurations reachable from B infinitely often.

4. Hence $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i) = 1$ iff all BSSCs B of $G_{S'}$ reachable from σ_0 satisfy $\bigwedge_i (B \xrightarrow{*} Q_i \;\Rightarrow\; B \xrightarrow{*} Q'_i)$.

Algorithmic Ideas

A method for checking $\mathbb{P}(S \models \varphi) = 1$:

1. Reduce $\mathbb{P}(S \models \varphi)$ to $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i)$ for $S' = S \otimes \mathcal{A}_\varphi$.

2. Build the *finite* graph $G_{S'}$ of all configurations from W_0 with an edge $\langle s, \varepsilon \rangle \longrightarrow \langle r, \varepsilon \rangle$ if $\langle r, \varepsilon \rangle$ is reachable from $\langle s, \varepsilon \rangle$.

3. Since an infinite run of S' almost surely visits W_0 infinitely often, it almost surely ends up visiting a BSSC B of $G_{S'}$, and then it almost surely visits all configurations reachable from B infinitely often.

4. Hence $\mathbb{P}(S' \models \bigwedge_i \Box\Diamond Q_i \Rightarrow \Box\Diamond Q'_i) = 1$ iff all BSSCs B of $G_{S'}$ reachable from σ_0 satisfy $\bigwedge_i (B \xrightarrow{*} Q_i \;\Rightarrow\; B \xrightarrow{*} Q'_i)$.

5. All this only needs reachability analysis of S'. Hence decidability.

An Assessment Of Qualitative Verification

+ Circumvents the undecidability of model checking lossy channel systems.

+ That precise values for weights etc. do not change the result is a bonus point: this means we only assumed some kind of fairness property.

+ The global-fault model is vindicated!

An Assessment Of Qualitative Verification

+ Circumvents the undecidability of model checking lossy channel systems.

+ That precise values for weights etc. do not change the result is a bonus point: this means we only assumed some kind of fairness property.

+ The global-fault model is vindicated!

– We'll see later that the fairness assumption is sometimes not realistic.

– What about properties that do not hold almost surely but, say, 99% of the time?

Quantitative Verification of PLCS's

Question: can we compute $\mathbb{P}(S \models \varphi)$ when it is not $= 1$?

Quantitative Verification of PLCS's

Question: can we compute $\mathbb{P}(S \models \varphi)$ when it is not $= 1$?

Theorem [Rab03]: There is a way to compute, for any *tolerance* $\tau > 0$, a probability p s.t. $p - \tau \leq \mathbb{P}(S \models \varphi) \leq p + \tau$.

NB: Earlier solution by [PN97] is flawed.

Quantitative Verification of PLCS's

Question: can we compute $\mathbb{P}(S \models \varphi)$ when it is not $= 1$?

Theorem [Rab03]: There is a way to compute, for any *tolerance* $\tau > 0$, a probability p s.t. $p - \tau \leq \mathbb{P}(S \models \varphi) \leq p + \tau$.

NB: Earlier solution by [PN97] is flawed.

This approximability result holds for the local-fault model (and the global-fault model when $p_{\text{loss}} \geq .5$).

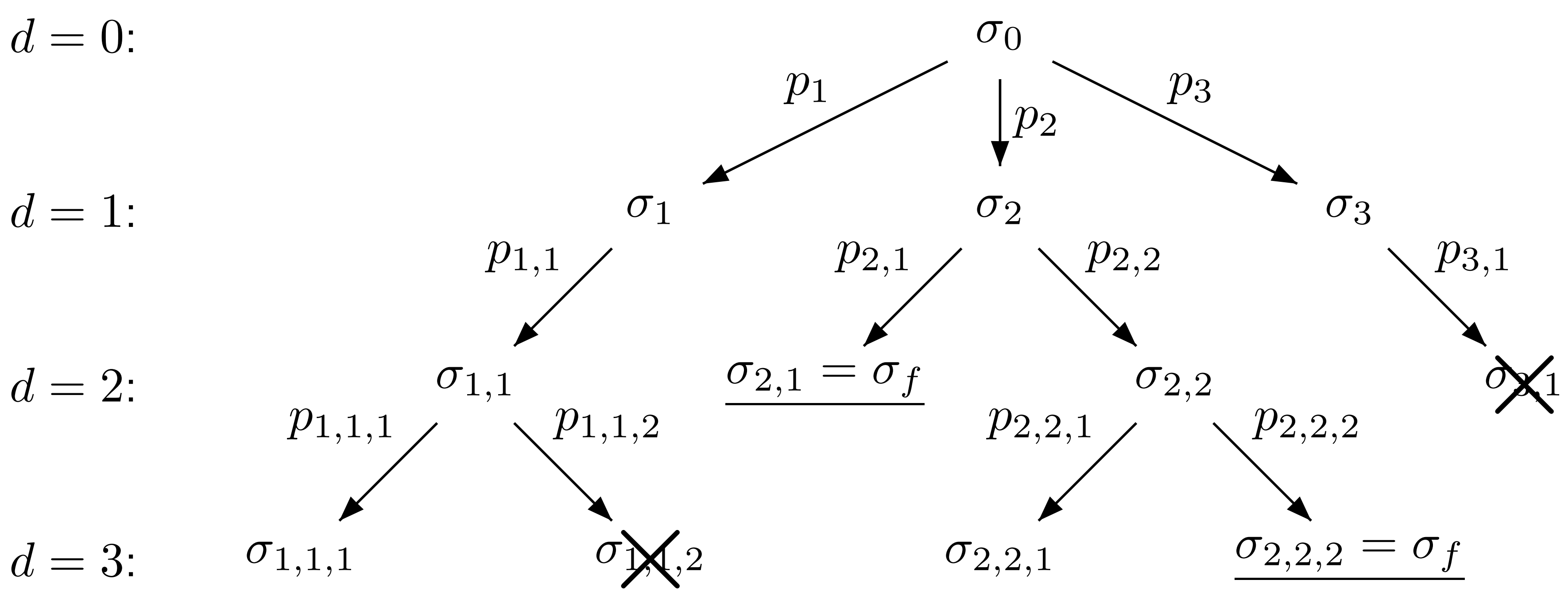
Again, finite attractors play an essential rôle...

Algorithmic Ideas

What is $\mathbb{P}(\sigma_0 \models \Diamond \sigma_f)$?

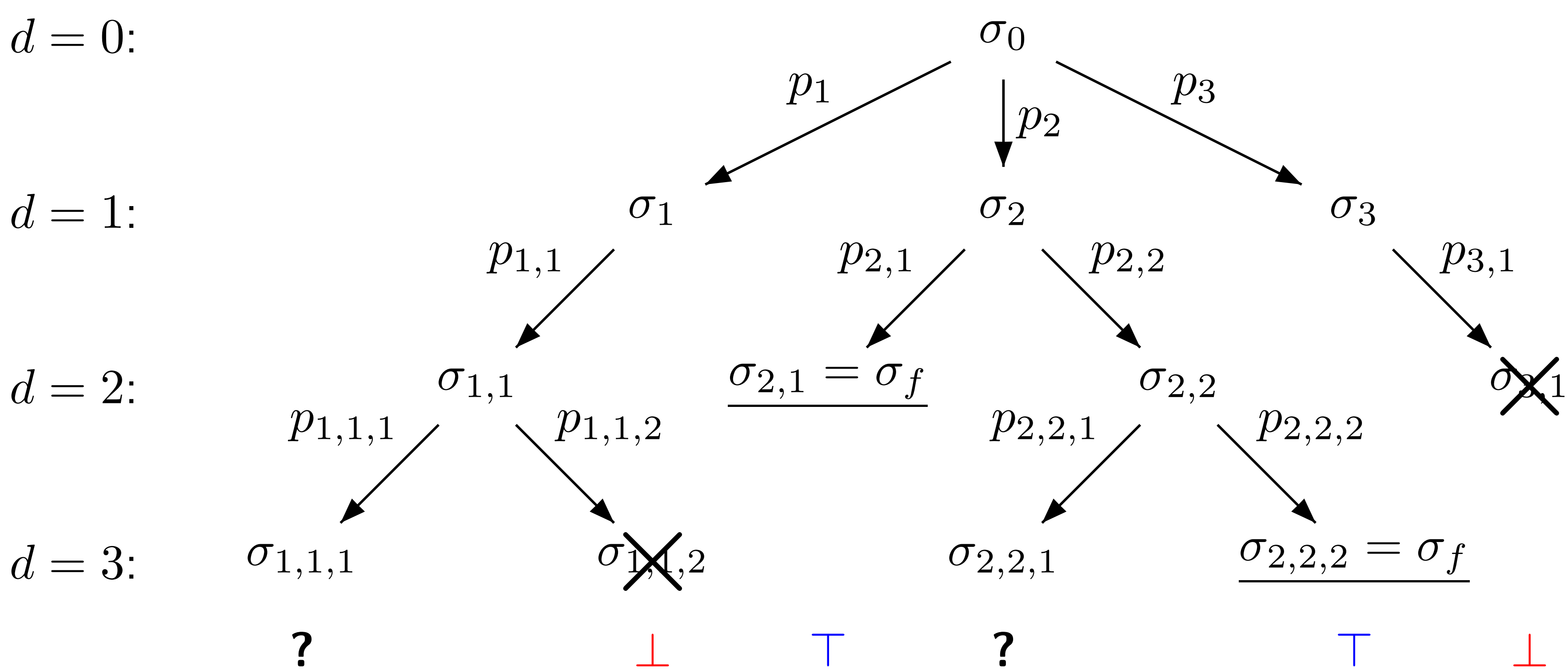
Algorithmic Ideas

What is $\mathbb{P}(\sigma_0 \models \Diamond \sigma_f)$?



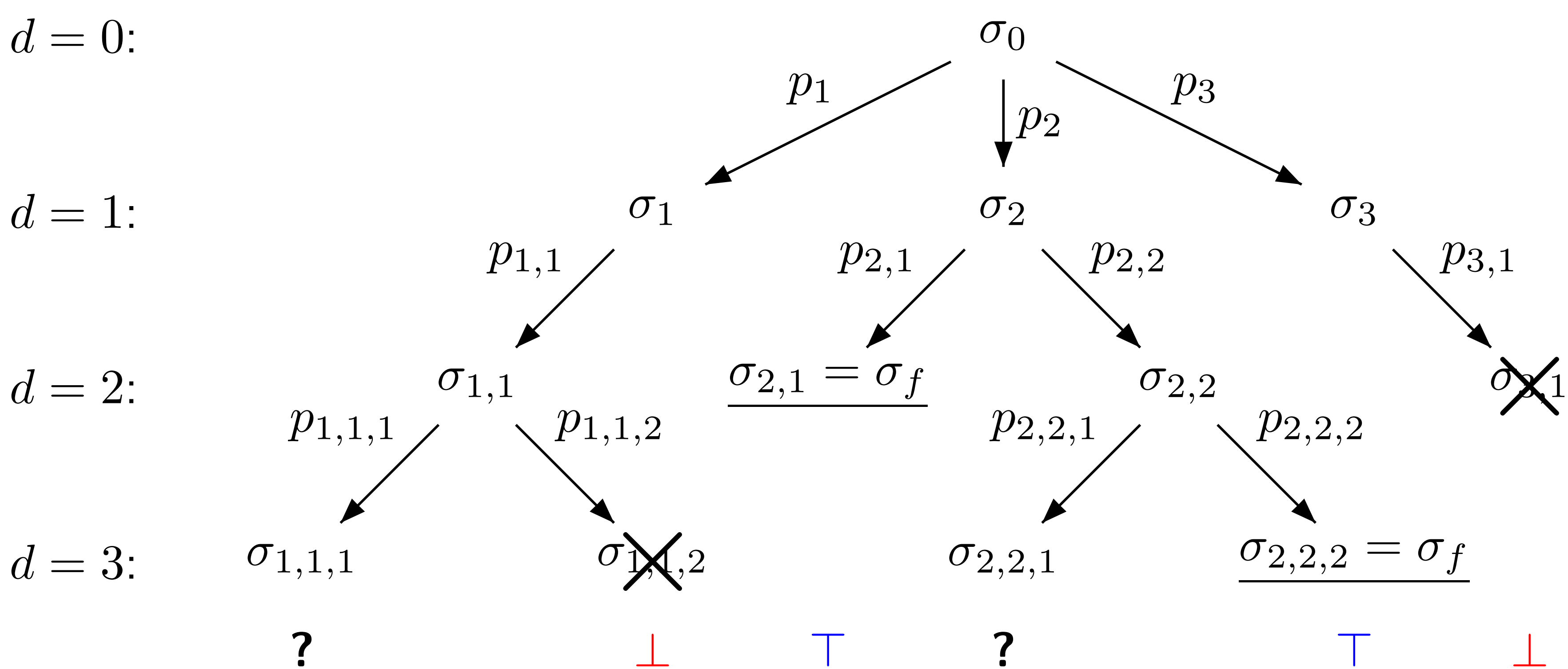
Algorithmic Ideas

What is $\mathbb{P}(\sigma_0 \models \Diamond \sigma_f)$?



Algorithmic Ideas

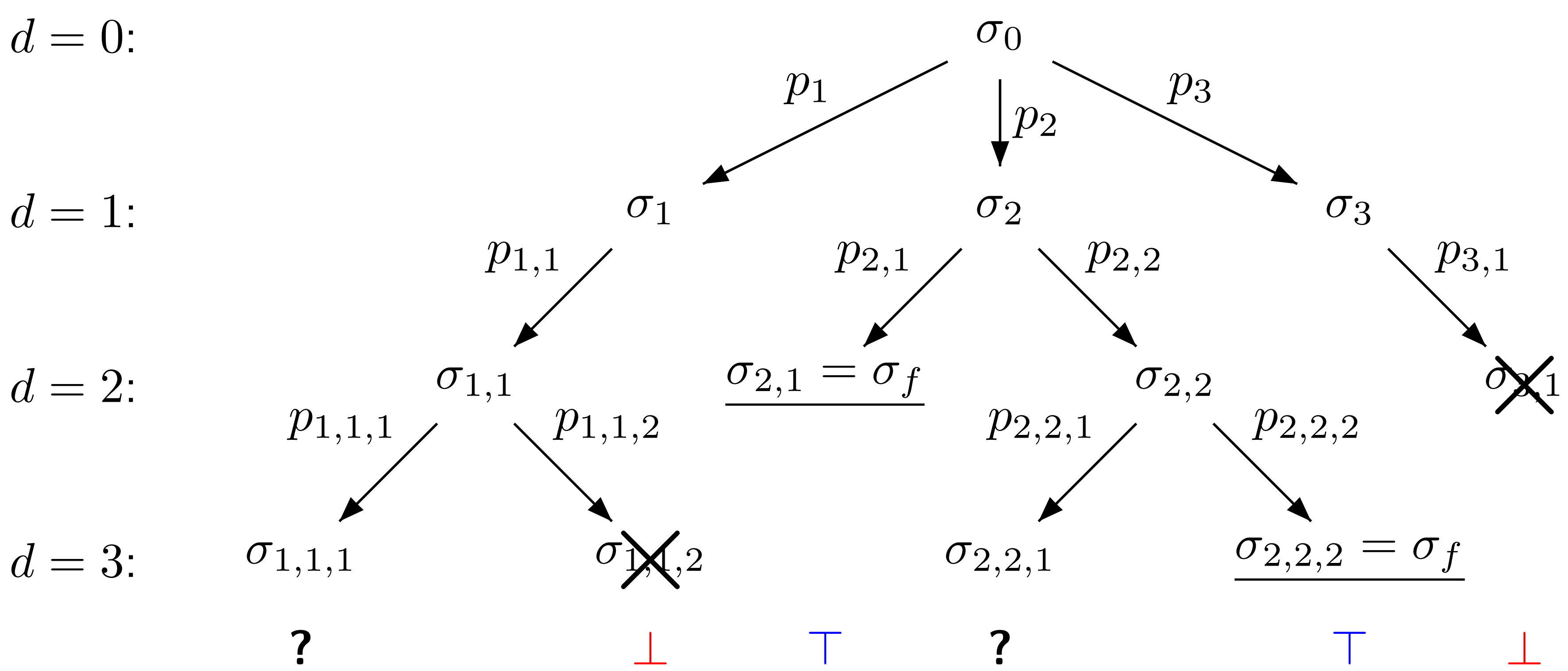
What is $\mathbb{P}(\sigma_0 \models \Diamond\sigma_f)$?



$$\mathbb{P}^d_{\top} \leq \mathbb{P}(\sigma_0 \models \Diamond\sigma_f) \leq \mathbb{P}^d_{\top} + \mathbb{P}^d_{?}.$$

Algorithmic Ideas

What is $\mathbb{P}(\sigma_0 \models \Diamond \sigma_f)$?



$$\mathbb{P}^d_\top \leq \mathbb{P}(\sigma_0 \models \Diamond \sigma_f) \leq \mathbb{P}^d_\top + \mathbb{P}^d_?.$$

$\lim_{d \to \infty} \mathbb{P}^d_? = 0$ for systems with a finite attractor!

An Assessment Of Quantitative Verification

+ Allows performance evaluation.

? Some open problems remain.

An Assessment Of Quantitative Verification

+ Allows performance evaluation.

? Some open problems remain.

– Requires that rules be given weights: where do these come from?

Beyond Markov Chains

The problem with PLCS's is that you have to view rules as probabilistic instead of nondeterministic.

Classically, nondeterminism in rules comes from:

– arbitrary interleaving of asynchronous components

– abstraction of real-life programs

– open systems

– early designs

Beyond Markov Chains

The problem with PLCS's is that you have to view rules as probabilistic instead of nondeterministic.

Classically, nondeterminism in rules comes from:

– arbitrary interleaving of asynchronous components

– abstraction of real-life programs

– open systems

– early designs

This cannot be realistically modeled by probabilities.

Beyond Markov Chains

The problem with PLCS's is that you have to view rules as probabilistic instead of nondeterministic.

Classically, nondeterminism in rules comes from:

– arbitrary interleaving of asynchronous components

– abstraction of real-life programs

– open systems

– early designs

This cannot be realistically modeled by probabilities.

You want a *Markov decision process* model, where rules are nondeterministic and losses are probabilistic!! [Bertrand & S. 2003].

Beyond Markov Chains

The problem with PLCS's is that you have to view rules as probabilistic instead of nondeterministic.

Classically, nondeterminism in rules comes from:

– arbitrary interleaving of asynchronous components

– abstraction of real-life programs

– open systems

– early designs

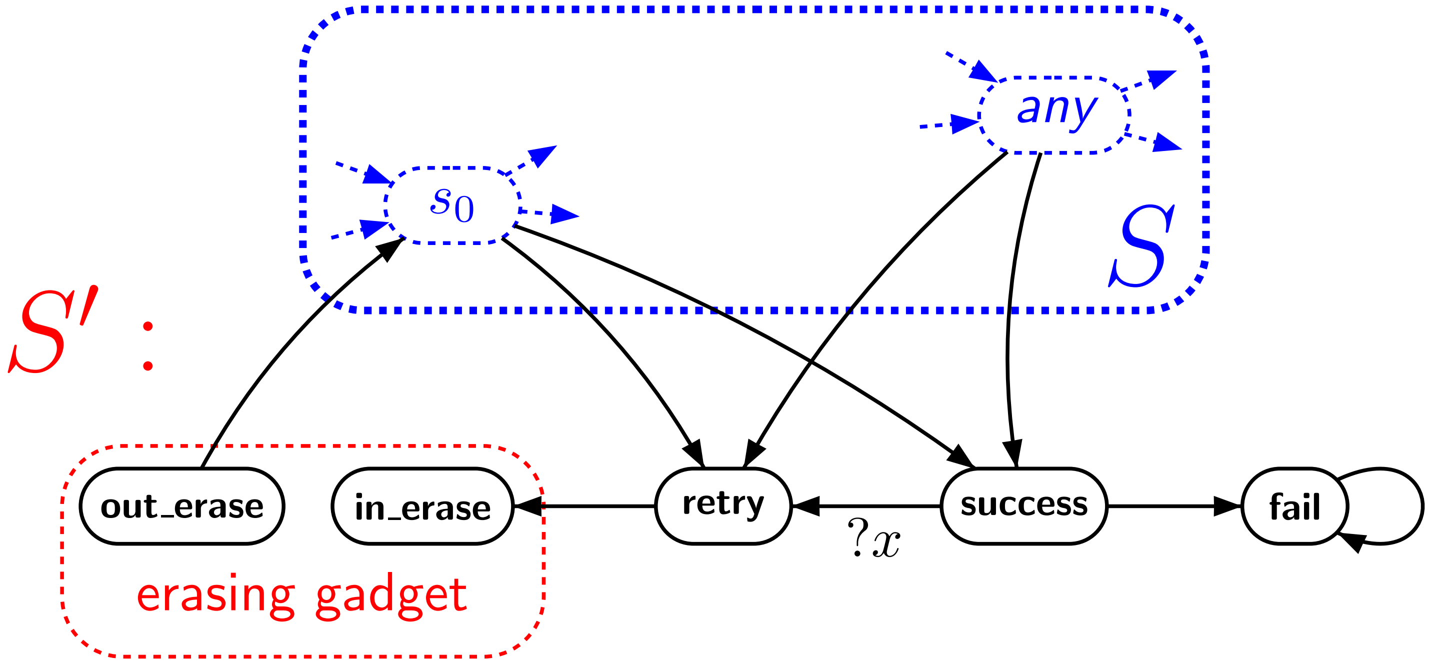
This cannot be realistically modeled by probabilities.

You want a *Markov decision process* model, where rules are nondeterministic and losses are probabilistic!! [Bertrand & S. 2003].

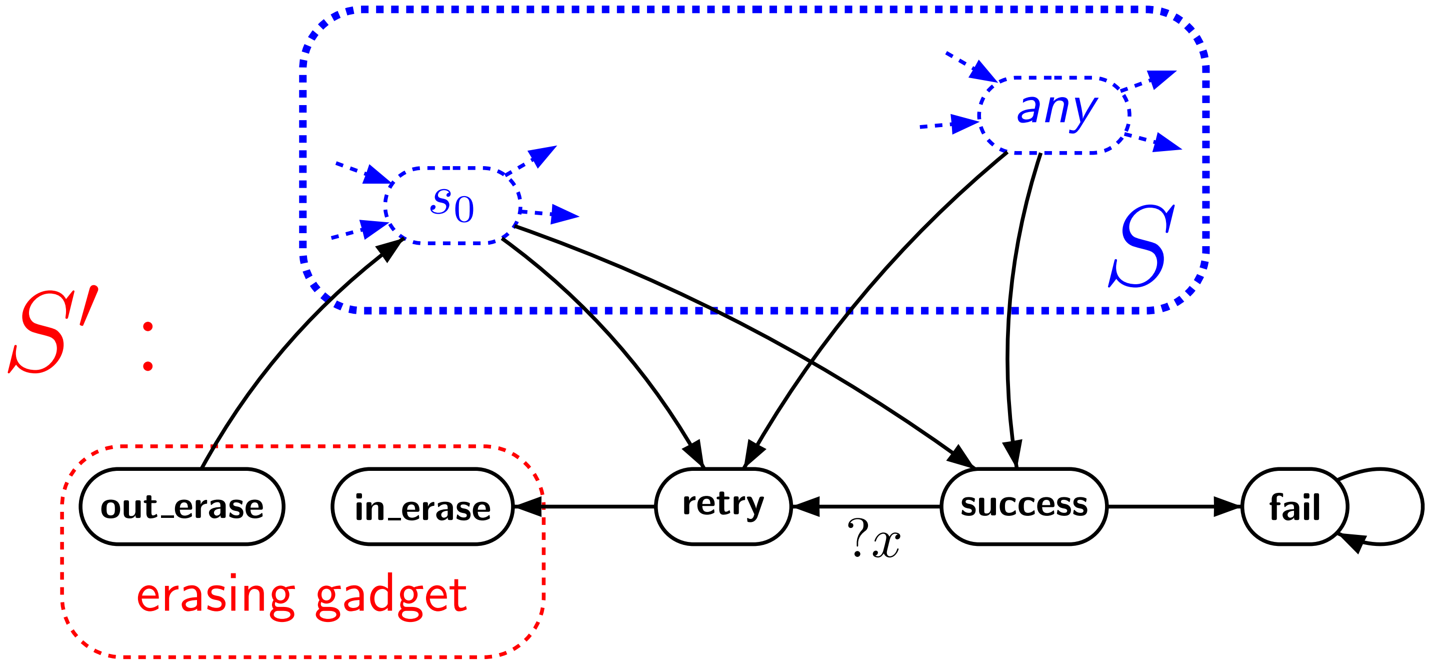
Then we can ask questions such as "*does* $\mathbb{P}(\varphi) = 1$ *under all scheduling policies?*" (This is the adversarial qualitative viewpoint).

What scheduling policies can do

What scheduling policies can do

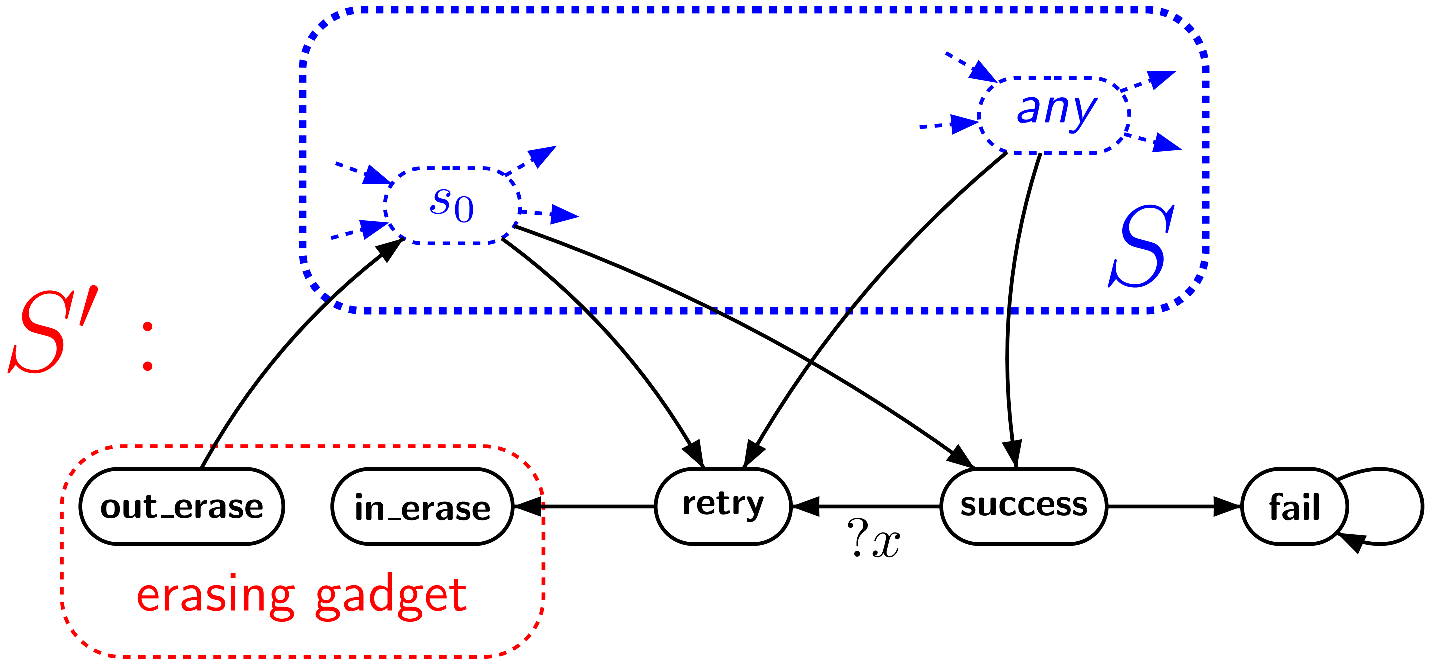


What scheduling policies can do



Question: is there a scheduling policy that makes S' visit **success** infinitely often with > 0 probability?

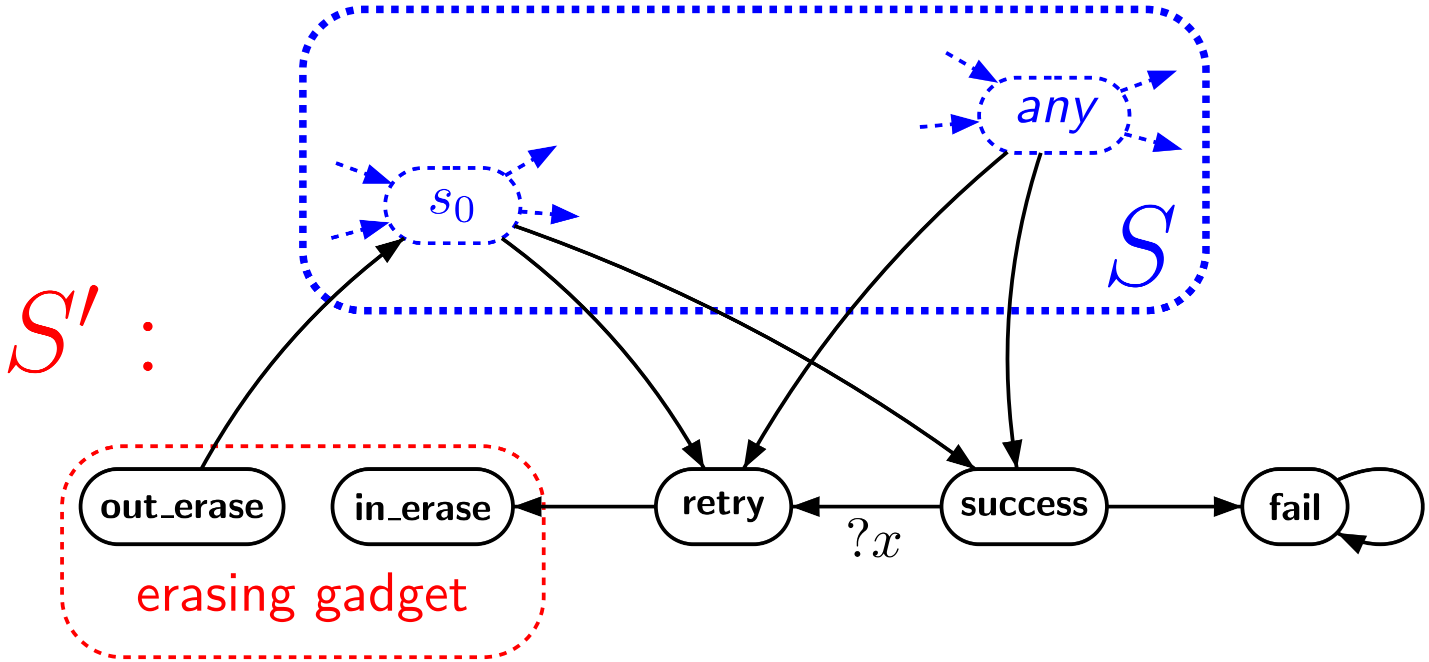
What scheduling policies can do



Question: is there a scheduling policy that makes S' visit **success** infinitely often with > 0 probability?

Answer: yes iff (nondeterministic) S is unbounded!

What scheduling policies can do



Question: is there a scheduling policy that makes S' visit **success** infinitely often with > 0 probability?

Answer: yes iff (nondeterministic) S is unbounded!

Corollary: model checking qualitative properties under all scheduling policies is undecidable.

All Is Not Lost!

In previous proof, the nasty scheduling policy is unrealistic.

E.g. it needs remember infinitely many things.

All Is Not Lost!

In previous proof, the nasty scheduling policy is unrealistic.

E.g. it needs remember infinitely many things.

Theorem (Bertrand & S. 2003): model checking qualitative properties under all *finite-memory policies* is decidable.

NB: Algorithm only needs examine G_S, the graph of empty configurations.

All Is Not Lost!

In previous proof, the nasty scheduling policy is unrealistic.

E.g. it needs remember infinitely many things.

Theorem (Bertrand & S. 2003): model checking qualitative properties under all *finite-memory policies* is decidable.

NB: Algorithm only needs examine G_S, the graph of empty configurations.

Some remaining open problems:

– What about cooperative qualitative model checking?

– What about computing minimal and maximal probabilities?

Concluding remarks

It is possible to analyze systems combining two hard features: probabilities and infinite state space.

Quantitative analysis is possible.

Qualitative analysis is possible.

Qualitative analysis of Markovian decision processes is a good substitute for traditional linear-time model checking (minus the undecidability!).

Randomization helps.

All this is still new and many open questions remain.

Bibliography – 1

[AJ96a] P. A. Abdulla and B. Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996. http://www.docs.uu.se/docs/avds/publications/icalp94_ic.ps.

[AJ96b] P. A. Abdulla and B. Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996. http://www.docs.uu.se/docs/avds/publications/lics93_ic.ps.

[ABPJ00] P. A. Abdulla, C. Baier, S. Purushothaman Iyer, and B. Jonsson. Reasoning about probabilistic lossy channel systems. In *Proc. 11th Int. Conf. Concurrency Theory (CONCUR'2000), University Park, PA, USA, Aug. 2000*, volume 1877 of *Lecture Notes in Computer Science*, pages 320–333. Springer, 2000. http://web.informatik.uni-bonn.de/I/papers/.

[AR03] P. A. Abdulla and A. Rabinovich. Verification of probabilistic systems with faulty communication. In *Proc. 6th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'2003), Warsaw, Poland, Apr. 2003*, volume 2620 of *Lecture Notes in Computer Science*, pages 39–53. Springer, 2003. http://www.math.tau.ac.il/~rabinoa/fossacs.ps.gz.

Bibliography – 2

[BE99] C. Baier and B. Engelen. Establishing qualitative properties for probabilistic lossy channel systems: An algorithmic approach. In *Proc. 5th Int. AMAST Workshop Formal Methods for Real-Time and Probabilistic Systems (ARTS'99), Bamberg, Germany, May 1999*, volume 1601 of *Lecture Notes in Computer Science*, pages 34–52. Springer, 1999. http://web.informatik.uni-bonn.de/I/papers/arts99.ps.

[Boc78] G. von Bochmann. Finite state description of communication protocols. *Computer Networks and ISDN Systems*, 2:361–372, 1978.

[BS03] N. Bertrand and Ph. Schnoebelen. Model checking lossy channels systems is probably decidable. In *Proc. 6th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'2003), Warsaw, Poland, Apr. 2003*, volume 2620 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2003. http://www.lsv.ens-cachan.fr/Publis/PAPERS/BerSch-fossacs2003.ps.

[BZ83] D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, 1983.

[DJS99] C. Dufourd, P. Jančar, and Ph. Schnoebelen. Boundedness of Reset P/T nets. In *Proc. 26th Int. Coll. Automata, Languages, and Programming (ICALP'99), Prague, Czech Republic, July 1999*, volume 1644 of *Lecture Notes in Computer Science*, pages 301–310. Springer, 1999. http://www.lsv.ens-cachan.fr/Publis/PAPERS/DJS-icalp99.ps.

Bibliography – 3

[Fin94] A. Finkel. Decidability of the termination problem for completely specificied protocols. *Distributed Computing*, 7(3):129–135, 1994.

[Hig52] G. Higman. Ordering by divisibility in abstract algebras. *Proc. London Math. Soc. (3)*, 2(7):326–336, 1952.

[May03] R. Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 297(1–3):337–354, 2003. http://www.informatik.uni-freiburg.de/~mayrri/lcmtcs.ps.

[MS02] B. Masson and Ph. Schnoebelen. On verifying fair lossy channel systems. In *Proc. 27th Int. Symp. Math. Found. Comp. Sci. (MFCS'2002), Warsaw, Poland, Aug. 2002*, volume 2420 of *Lecture Notes in Computer Science*, pages 543–555. Springer, 2002. http://www.lsv.ens-cachan.fr/Publis/PAPERS/MS-mfcs2002-long.ps.

Bibliography – 4

[Rab03] A. Rabinovich. Quantitative analysis of probabilistic lossy channel systems. In *Proc. 30th Int. Coll. Automata, Languages, and Programming (ICALP'2003), Eindhoven, NL, July 2003*, Lecture Notes in Computer Science. Springer, 2003. To appear.

[Sch01] Ph. Schnoebelen. Bisimulation and other undecidable equivalences for lossy channel systems. In *Proc. 4th Int. Symp. Theoretical Aspects of Computer Software (TACS'2001), Sendai, Japan, Oct. 2001*, volume 2215 of *Lecture Notes in Computer Science*, pages 385–399. Springer, 2001. http://www.lsv.ens-cachan.fr/Publis/PAPERS/Sch-tacs2001.ps.

[Sch02] Ph. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *Information Processing Letters*, 83(5):251–261, 2002. http://www.lsv.ens-cachan.fr/Publis/PAPERS/Sch-IPL2002.ps.