Verifying Nondeterministic Channel Systems With Probabilistic Message Losses[3]

N. Bertrand[1] and Ph. Schnoebelen[2]

Lab. Spécification & Vérification, ENS de Cachan & CNRS UMR 8643, 61, av. Pdt. Wilson, 94235 Cachan Cedex France

Abstract

Lossy channel systems (LCS's) are systems of finite state automata that communicate via unreliable unbounded fifo channels. In order to circumvent the undecidability of model checking for nondeterministic LCS's, probabilistic models have been introduced, where it can be decided whether a linear-time property holds almost surely. However, such fully probabilistic systems are not a faithful model of nondeterministic protocols.

We study a hybrid model for LCS's where losses of messages are seen as faults occurring with some given probability, and where the internal behavior of the system remains nondeterministic. Thus the semantics is in terms of infinite-state reactive Markov chains (equivalently, Markovian decision processes). A similar model was introduced in the second part of (Bertrand & Schnoebelen, FOSSACS'2003, LNCS 2620, pp. 120–135): we continue this work and give a complete picture of the decidability of qualitative model checking for MSO-definable properties and some relevant subcases.

Key words: Lossy channel systems, model checking, probabilistic systems, verification of asynchronous communication protocols.

1 Introduction

Verification of Channel Systems. Channel systems [8] are systems of finite state components that communicate via asynchronous unbounded fifo channels. See Fig. 1 for an example. They are a natural model for asynchronous communication protocols, used as the semantical basis of protocol specification languages such as SDL and Estelle.

[1] Email: bertrand@lsv.ens-cachan.fr

[2] Email: phs@lsv.ens-cachan.fr

[3] This research was supported by *Persée*, a project funded by the *ACI Sécurité Informatique* of the French Ministry for Scientific Research.

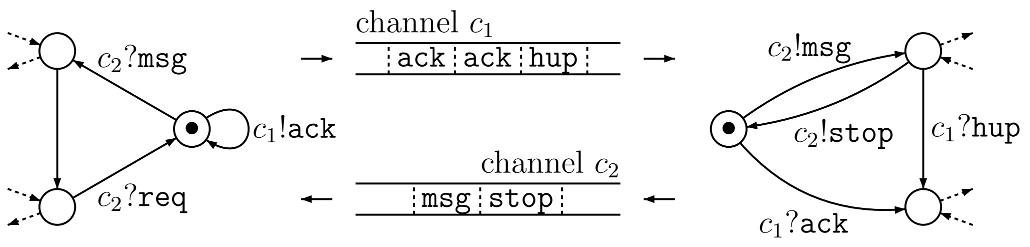

Fig. 1. A channel system

Lossy channel systems [10,4] are a special class of channel systems where messages can be lost while they are in transit, without any notification. Considering lossy systems is natural when modeling fault-tolerant protocols where the communication channels are not supposed to be reliable.

Surprisingly, while channel systems are Turing-powerful [8], several verification problems become decidable when one assumes channels are lossy: reachability, safety properties over traces, inevitability properties over states, and fair termination are decidable for lossy channel systems [10,9,4,11].

Unfortunately, several important verification problems are undecidable: recurrent reachability, liveness properties, boundedness, and all behavioral equivalences are undecidable for these systems [3,12,16]. Furthermore, none of the decidable problems listed in the previous paragraph can be solved in primitive recursive time [17]!

Verifying Liveness Properties. In practice, lossy channel systems are a convenient model for verifying safety properties of asynchronous protocols [1]. However, they are not so adequate for liveness properties. One first difficulty here is the undecidability of liveness properties. A second difficulty is that the model itself is too pessimistic when liveness is considered.

Protocols that have to deal with unreliable channels usually have some coping mechanism combining resends and acknowledgments. But, without any assumption limiting message losses, no mechanism can ensure that some communication will eventually be initiated. The classical solution to this problem is to impose some fairness assumptions on the behavior of the lossy channel, e.g. "if infinitely many messages are sent to the channels, infinitely many will not be lost". However, fairness assumptions also make decidability more elusive [3,11].

Probabilistic Losses. When modeling real-life protocols, it is natural to see message losses as some kind of faults having a probabilistic behavior. Following this idea, Purushothaman Iyer introduced the first Markov chain model for lossy channel systems [14]. In this model, not only the message losses are probabilistic, but the choice between different available transitions is also made probabilistically. Qualitative verification is decidable when message losses have a high probability [6] and undecidable when they are less likely [2].

An improved model was introduced in [5] and [7] where the probability of losses is modeled more faithfully. Additionally it makes qualitative verification [5,7] and approximate quantitative verification [15] decidable independently of the likelihood of message losses. See the survey [18] for more details.

These models are rather successful in bringing back decidability. However, they assume that the system is fully probabilistic, i.e. the choice between different actions is made probabilistically. But when verifying channel systems, nondeterminism is an essential feature: one uses it to include (part of) an unknown environment in the model, to delay implementation choices at early stages of the design, and to abstract away from complex control structures at later stages.

Our Contribution. We consider a new model where channel systems behave nondeterministically while messages are lost probabilistically. This gives rise to infinite-state Markovian decision processes, for which we study the decidability of qualitative properties. This study was initiated in the second part of [7] and we now provide complete results with full proofs.

As was already apparent in [7], qualitative verification is in general not decidable for the Markovian decision process model of LCS's. It appears that this is because schedulers are very powerful (e.g. they need not be recursive). In order to recover decidability without sacrificing too much of the model, we advocate restricting oneself to finite-memory schedulers, as explained in [7].

Outline of the Paper. Sections 2 and 3 provide the necessary definitions on, respectively, Markovian decision processes and lossy channel systems. Our results are given in Sections 4 and 5.

2 Markovian Decision Processes

We assume familiarity with Markov chains and Markovian decision processes, as they are used in the verification of probabilistic systems (introductory texts are [13] for the semantical issues, and [19] for the verification issues).

Let $\mathit{Dist}(\Omega)$ be the set of all discrete probability distributions on Ω.

Definition 2.1 A *Markovian decision process* (a MDP) is a pair $M = \langle S, \delta \rangle$ s.t. $S = \{s, s', \ldots\}$ is a countable set of *states*, and $\delta : S \to 2^{\mathit{Dist}(S)}$ maps each state $s \in S$ to a finite nonempty subset of $\mathit{Dist}(S)$.

The behavior of an MDP combines nondeterminism and probabilities. This is formalized through the concept of a *scheduler* (also called *opponent*, or *adversary*). Formally, a scheduler for $M = \langle S, \delta \rangle$ is a mapping u from S^+ (the set of sequences of states) to $\mathit{Dist}(S)$ such that $u(s_0 \ldots s_n) \in \delta(s_n)$. The intuition is that, based on the history $s_0 \ldots s_n$ of the computation (i.e. the sequence of states that have already been visited), the scheduler picks one distribution f among those that are allowed in the current state s_n. Then the next state s_{n+1} is chosen probabilistically, according to f.

Thus, once a scheduler u is provided, the MDP M gives rise to a Markov chain (denoted M^u) and there is a canonical probability measure for its sets of paths. It is thus meaningful to speak of the probability that M^u satisfies some linear-time probability φ. The statements we want to decide have the form "for all schedulers u, M^u satisfies φ with probability 1" (what one calls "qualitative verification").

Remark 2.2 Formally, the states of M^u are *sequences* of states of M. However, these sequences denote a current state with its history. It is thus convenient (and customary) to informally speak of M^u as if it had the same state space as M and leave the history implicit. $\square$

Remark 2.3 Our definition considers *deterministic* schedulers. One sometimes considers more general schedulers (called *stochastic* schedulers) that pick a distribution in $\mathit{Dist}(\delta(s_n))$. It can be argued that, for the qualitative verification problems we consider, stochastic schedulers do not bring additional power. $\square$

3 Lossy Channel Systems

Lossy channel systems are finite-state processes operating on a finite set of channels, and where each channel behaves as an unbounded FIFO buffer which is unreliable in the sense that it can lose messages [10,4].

Formally, a *lossy channel system* (a LCS) is a tuple $\mathcal{L} = \langle \mathtt{Q}, \mathtt{C}, \mathtt{M}, \Delta \rangle$ where $\mathtt{Q}$ is a finite set of *local states*, $\mathtt{C}$ is a finite set of *channels*, $\mathtt{M}$ is a finite *message alphabet*, and Δ is a set of *transition rules* each of the form $\langle \mathtt{r}, \mathtt{op}, \mathtt{s} \rangle$, where $\mathtt{r}, \mathtt{s} \in \mathtt{Q}$, and $\mathtt{op}$ is an *operation* of one of the forms $\mathtt{c}!\mathtt{m}$ (sending message $\mathtt{m}$ to channel $\mathtt{c}$), or $\mathtt{c}?\mathtt{m}$ (receiving message $\mathtt{m}$ from channel $\mathtt{c}$). A *configuration* (also, a *global state*) σ of $\mathcal{L}$ is of the form $\langle \mathtt{r}, \mathtt{w} \rangle$ where $\mathtt{r} \in \mathtt{Q}$ and $\mathtt{w}$ is a mapping from $\mathtt{C}$ to $\mathtt{M}^*$ that gives the current contents of each channel. We abuse notation and write ε for denoting both the empty word in $\mathtt{M}^*$ and the "empty" map that associates ε with each $\mathtt{c} \in \mathtt{C}$.

Remark 3.1 Channel systems usually have several cooperating finite-state components (see Fig. 1 for example), but there is no loss of generality in combining all of these components into a single one. $\square$

3.1 Operational Semantics

A LCS $\mathcal{L}$ induces a transition system $T_{\mathcal{L}} \overset{\text{def}}{=} \langle S, \rightarrow \rangle$, where $S \overset{\text{def}}{=} \mathtt{Q} \times (\mathtt{C} \rightarrow \mathtt{M}^*)$ is the set of configurations, and $\rightarrow \subseteq S \times S$ is the transition relation.

The transition relation contains both *normal steps* (also called *perfect steps*, where no losses occur) and *lossy steps*. Assume $\sigma_1 = \langle \mathtt{r}_1, \mathtt{w}_1 \rangle$ and $\sigma_2 = \langle \mathtt{r}_2, \mathtt{w}_2 \rangle$ are two configurations. There is a normal step $\sigma_1 \rightarrow \sigma_2$ if either

writing step: there exists a rule in Δ of the form $\langle \mathtt{r}_1, \mathtt{c!m}, \mathtt{r}_2 \rangle$, and $\mathtt{w}_2$ is obtained from $\mathtt{w}_1$ by appending $\mathtt{m}$ to the end of $\mathtt{w}_1(\mathtt{c})$, or

reading step: there exists a rule in Δ of the form $\langle \mathtt{r}_1, \mathtt{c?m}, \mathtt{r}_2 \rangle$ and $\mathtt{w}_2$ is obtained from $\mathtt{w}_1$ by removing $\mathtt{m}$ from the front of $\mathtt{w}_1(\mathtt{c})$ (and therefore such a step is only possible if $\mathtt{w}_1(\mathtt{c})$ starts with $\mathtt{m}$), or

idling step: $\sigma_2 = \sigma_1$.

We now take into account the possibility of losing messages. Say $\sigma = \langle \mathtt{r}, \mathtt{w} \rangle$ is a *subconfiguration* of $\sigma' = \langle \mathtt{r}', \mathtt{w}' \rangle$, written $\sigma \sqsubseteq \sigma'$, if $\mathtt{r} = \mathtt{r}'$ and $\mathtt{w}$ can be obtained from $\mathtt{w}'$ by removing (arbitrarily many) messages at arbitrary positions (hence every $\mathtt{w}(\mathtt{c})$ is a subword of $\mathtt{w}'(\mathtt{c})$).

In addition to normal steps, there is a step $\sigma_1 \to \sigma_3$ in $T_{\mathcal{L}}$ if

lossy step: there is a normal step $\sigma_1 \to \sigma_2$ and $\sigma_3 \sqsubseteq \sigma_2$.

Hence normal steps can always be followed by arbitrary message losses.

Remark 3.2 Compared to other ways of defining lossy channel systems, the only important novelty here is that we always allow idling steps. This feature comes from [7]. It is convenient in that it ensures our transition systems have no deadlock states. But it is also a very useful tool when programming schedulers as we see in the next sections. It can be argued that it makes the schedulers too powerful: some of our decidability results rely in an essential way on the possibility of idling (see Section 4). $\square$

Observe that the transition system $T_{\mathcal{L}}$ sees message losses as occurring *nondeterministically* after normal steps[4].

3.2 Reachability and Control State Reachability

We write $\sigma \xrightarrow{*} \sigma'$ when there is a sequence of steps in $T_{\mathcal{L}}$ that reach σ' from σ. For $A \subseteq \mathtt{Q}$, we write $\sigma \xrightarrow{*} A$ when some configuration $\langle \mathtt{s}, \mathtt{w} \rangle$ with $\mathtt{s} \in A$ is reachable from σ.

The *reachability problem* is the set of all $(\mathcal{L}, \sigma, \sigma')$ s.t. $\sigma \xrightarrow{*} \sigma'$ in $T_{\mathcal{L}}$. Similarly, the *control state reachability problem* is the set of all $(\mathcal{L}, \sigma, A)$ s.t. $\sigma \xrightarrow{*} A$. It is known (from [4]) that reachability and control state reachability are decidable for LCS's.

3.3 Channel Systems With Probabilistic Losses

We now define a model for LCS's where message losses are probabilistic instead of nondeterministic. We adopt the local-fault model for probabilistic losses: this was introduced in [7,5] where it is argued that it is more faithful than earlier probabilistic models for lossy channels.

Formally, let $\tau \in (0, 1)$ be a fixed *fault rate*: it is assumed that after every normal step of the LCS, each message present in the channels is lost with

[4] Nondeterminism also occurs in the choice between several normal steps.

probability τ (and kept with probability $1-\tau$). Every message is lost or kept independently of what happens to the other messages. Therefore, the probability that, during a step, the contents of the channels move from $\mathtt{w}$ to $\mathtt{w}'$ via message losses is exactly

$$\mathbb{P}_l(\mathtt{w},\mathtt{w}') \overset{\text{def}}{=} \tau^{|\mathtt{w}|-|\mathtt{w}'|} \times (1-\tau)^{|\mathtt{w}'|} \times \binom{\mathtt{w}}{\mathtt{w}'} \tag{1}$$

where $|\mathtt{w}|$ is the size of $\mathtt{w}$ (the total number of messages present in the channels), and $\binom{\mathtt{w}}{\mathtt{w}'}$ is the number of different ways one can obtain $\mathtt{w}'$ by erasing letters from $\mathtt{w}$ (e.g., in the case of a single channel, $\binom{\mathtt{abbab}}{\mathtt{ab}} = 4$ and $\binom{\mathtt{ab}}{\mathtt{bab}} = 0$).

We extend this to a probability between configurations via

$$\mathbb{P}_l(\langle \mathtt{r},\mathtt{w}\rangle, \langle \mathtt{s},\mathtt{w}'\rangle) \overset{\text{def}}{=} \begin{cases} \mathbb{P}_l(\mathtt{w},\mathtt{w}') \text{ if } \mathtt{r}=\mathtt{s}, \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

Now, for a configuration σ, we denote by δ_l^σ the distribution on S defined by $\delta_l^\sigma(\sigma') \overset{\text{def}}{=} \mathbb{P}_l(\sigma,\sigma')$.

Definition 3.3 The MDP $M_\mathcal{L}$ associated with a LCS $\mathcal{L}$ is $\langle S,\delta\rangle$ where S is the set of configurations of $\mathcal{L}$ and where $\delta(\sigma) \overset{\text{def}}{=} \{\delta_l^{\sigma'} \mid \sigma \to \sigma' \text{ is a } \textit{normal} \text{ step}\}$.

Thus the behavior of $M_\mathcal{L}$ is given by a succession of nondeterministic choices (what normal step next?) interleaved with probabilistic message losses.

Remark 3.4 The specific values appearing in δ_l^σ are not so relevant for qualitative verification. What is important is that all steps allowed in $T_\mathcal{L}$ can happen in $M_\mathcal{L}$. Additionally, an important feature of the local-fault model we adopted is that it ensures that, with probability 1, the channels will be (simultaneously) empty infinitely often [7]. This does not depend on the choices made by the scheduler, and does not depend on the exact value of the fault-rate τ. $\square$

A construction in section 5 relies on the following:

Fact 3.5 *For any* $\tau \in (0,1)$

$$0 < \prod_{n=1}^{\infty} 1-\tau^n. \tag{3}$$

4 Qualitative Verification of Streett Properties

For a LCS $\mathcal{L} = \langle \mathtt{Q},\mathtt{C},\mathtt{M},\Delta\rangle$ we consider Streett properties α of the form $\bigwedge_{i=1}^k \Box\Diamond A_i \Rightarrow \Box\Diamond B_i$ where the A_i's and B_i's are subsets of $\mathtt{Q}$. Here the modalities "$\Box$" and "$\Diamond$" have their usual temporal meaning: a run $\langle \mathtt{s}_0,\mathtt{w}_0\rangle \to$

$\langle \mathtt{s}_1, \mathtt{w}_1\rangle \to \dots \langle \mathtt{s}_n, \mathtt{w}_n\rangle \to \dots$ satisfies $\Diamond A$ (resp. $\Box A$, $\Diamond\Box A$, $\Box\Diamond A$) if $\mathtt{s}_i \in A$ for some i (resp. all i's, all i's after some point, infinitely many i's).

Remark 4.1 It is well known that more complex properties defined e.g. in temporal logic, or in the second-order monadic logic of order, can be reduced to Streett properties by building the product of the system under study with a deterministic finite-state automaton [19].

However, our choice of allowing idling steps in LCS's has the unfortunate consequence that we cannot synchronize a LCS with a finite-state automaton (since idling is in general not allowed in these automata). Hence our results summarized in Fig. 5 do not provide all the answers we are interested in (they do not really consider all LTL formulae). □

We are interested in *qualitative verification*, i.e. checking whether for all schedulers u, and starting from some initial configuration $\langle \mathtt{s}, \varepsilon\rangle$, $M^u_{\mathcal{L}}$ satisfies α with probability equal to 1, written "$\forall u\ \mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models \alpha) = 1$". We often leave the LCS and/or its initial configuration implicit, writing e.g. "$\forall u\ \mathbb{P}(u \models \alpha) = 1$". We also consider the problems of checking whether, for all schedulers, the probability $\mathbb{P}(u \models \alpha)$ is strictly less than 1, or equal to 0, or strictly more than 0.

Since our problems are undecidable in general (see section 5), we shall consider restricted classes of Streett properties, such as *reachability* (of the form $\Diamond A$), *safety* (of the form $\Box A$), generalized *Büchi properties* (of the form $\bigwedge_i \Box\Diamond A_i$), and their duals (of the form $\bigvee_i \Diamond\Box A_i$). Our results are summarized in Fig. 5 (page 16), where "D" and "U" stand for "Decidable" and "Undecidable" respectively.

4.1 Some Easy Cases For Reachability

Let $A \subseteq \mathtt{Q}$ be a set of control states. The following implications hold:

$$\begin{aligned} \mathtt{s} \in A &\Rightarrow \exists u\ \mathbb{P}(M^u, \langle \mathtt{s}, \varepsilon\rangle \models \Box A) = 1 && (4)\\ &\Rightarrow \exists u\ \mathbb{P}(M^u, \langle \mathtt{s}, \varepsilon\rangle \models \Box A) > 0 && (5)\\ &\Rightarrow \mathtt{s} \in A && (6)\end{aligned}$$

Implication (4) is proved by picking a scheduler that always idles, and the other implications are obvious.

Corollary 4.2 *It is decidable whether, for given $\mathcal{L}$ and A,*

- $\forall u\ \mathbb{P}(u \models \Box A) = 0$,
- $\forall u\ \mathbb{P}(u \models \Box A) < 1$.

With duality, we obtain

Corollary 4.3 *It is decidable whether, for given $\mathcal{L}$ and A,*

- $\forall u\ \mathbb{P}(u \models \Diamond A) = 1$,

- $\forall u\ \mathbb{P}(u \models \Diamond A) > 0$.

Other subcases reduce to reachability in the underlying $T_{\mathcal{L}}$:

$$\exists u\ \mathbb{P}(M^u_{\mathcal{L}}, \sigma \models \Diamond A) > 0 \ \Leftrightarrow\ \sigma \xrightarrow{*} A \tag{7}$$

The "($\Rightarrow$)" direction is obvious, and the converse implication is proved by considering a scheduler that just tries to mimic the path $\sigma \xrightarrow{*} A$: there is a non-zero probability that the message losses will just match what is required by that path.

Since reachability is decidable for LCS's, using duality we deduce

Corollary 4.4 *It is decidable whether, for given $\mathcal{L}$ and A:*

- $\forall u\ \mathbb{P}(u \models \Diamond A) = 0$,
- $\forall u\ \mathbb{P}(u \models \Box A) = 1$.

4.2 Harder Cases For Reachability

Theorem 4.5 *It is decidable whether, for given $\mathcal{L}$ and A:*

- $\forall u\ \mathbb{P}(u \models \Diamond A) < 1$.

For the proof of Theorem 4.5 we introduce a new notion:

Definition 4.6 A set of control states $X \subseteq \mathtt{Q}$ is *safe for A* if for all $\mathtt{x}$ in X, $\langle \mathtt{x}, \varepsilon\rangle \xrightarrow{*}_X A$.

Here the X subscript in "$\xrightarrow{*}_X$" denotes *constrained reachability.* More precisely, $\sigma \xrightarrow{*}_X A$ iff there is a path from σ to A along which only control states from X are visited (including at the extremities of the path). Obviously constrained reachability is decidable for a LCS $\mathcal{L}$ because it corresponds to reachability in the LCS $\mathcal{L}_{|X}$, also denoted $\mathcal{L} - (\mathtt{Q} \setminus X)$, obtained from $\mathcal{L}$ by only keeping the control states from X.

Lemma 4.7 *There exists a scheduler u s.t. $\mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models \Diamond A) = 1$ iff there is a $X \subseteq \mathtt{Q}$ that contains $\mathtt{s}$ and is safe for A.*

Proof. ($\Leftarrow$) Assuming $\mathtt{s}$ belongs to some set X safe for A, we describe a scheduler ensuring $\mathbb{P}(u \models \Diamond A) = 1$. It has two modes. Starting from a $\langle \mathtt{x}, \varepsilon\rangle$ with $\mathtt{x} \in X$, u is in *normal mode* and tries to reach A by mimicking the path witnessing $\langle \mathtt{x}, \varepsilon\rangle \xrightarrow{*}_X A$. u goes on with this strategy as long as the message losses occur according to what the path requires. Whenever the message losses are not as expected, u is in some unwanted $\langle \mathtt{y}, \mathtt{w}\rangle$. Here it switches to *recovery mode* and stays idling, remaining in control state $\mathtt{y}$ and only losing messages. When the configuration $\langle \mathtt{y}, \varepsilon\rangle$ is reached (which will eventually happen almost surely), u switches back to normal mode (observe that $\mathtt{y} \in X$) and aims again for A. With this strategy, u will eventually reach A almost surely.
($\Rightarrow$) Assume that $\mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models \Diamond A) = 1$ for some u. We say a configuration σ is *required* if $\mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models (\neg A)\ \mathit{Until} \sigma) > 0$, i.e. it is possible that u visits

σ before reaching A. Any subconfiguration σ' of a required σ is required too since the path that brings u to σ could, with more losses, have reached σ' instead. Furthermore, A is reachable from any required σ, otherwise u could not ensure $\Diamond A$ almost surely. Let now X be the set of control states that appear in the required configurations: X is safe for A and contains $\mathtt{s}$. $\square$

Thus, and because control state reachability is decidable in LCS's, we can decide whether $\mathbb{P}(u \models \Diamond A) = 1$ for some u, proving Theorem 4.5.

By duality, Theorem 4.5 entails:

Corollary 4.8 *It is decidable whether, for given $\mathcal{L}$ and A:*

- $\forall u\ \mathbb{P}(u \models \Box A) > 0.$

4.3 Generalized Büchi properties

Theorem 4.9 *It is decidable whether, for given $\mathcal{L}$ and A_i's:*

- $\forall u\ \mathbb{P}(u \models \bigwedge_{i=1}^{n} \Box\Diamond A_i) < 1.$

For the proof of Theorem 4.9, write α for $\bigwedge_{i=1}^{n} \Box\Diamond A_i$ and say a control state $\mathtt{x} \in \mathtt{Q}$ is *allowed* if $\langle \mathtt{x}, \varepsilon\rangle \xrightarrow{*} A_i$ for all $i = 1, \ldots, n$ (otherwise, $\mathtt{x}$ is *forbidden*). Clearly, it is decidable whether a control state is allowed.

Lemma 4.10 *Suppose all states in $\mathtt{Q}$ are allowed. Then, there exists a scheduler u s.t. $\mathbb{P}(u \models \alpha) = 1$.*

Proof. [Idea] The scheduler is built like in the proof of Lemma 4.7, only now it aims at the A_i's in turn, moves to the next target A_{i+1} when A_i is eventually reached, and loops back to A_1 when A_n is reached. Since all states are allowed, the scheduler is never blocked. $\square$

Assume now that $\mathtt{x}$ is forbidden and consider $\mathcal{L} - \mathtt{x}$, the LCS obtained by removing $\mathtt{x}$ from $\mathcal{L}$.

Lemma 4.11 *The following statements are equivalent:*

(i) *There exists a scheduler u s.t. $\mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models \alpha) = 1$.*

(ii) *$\mathtt{s} \neq \mathtt{x}$ and there exists a scheduler u' s.t. $\mathbb{P}(M^{u'}_{\mathcal{L}-\mathtt{x}}, \langle \mathtt{s}, \varepsilon\rangle \models \alpha) = 1$.*

Proof. ($\Leftarrow$) u' works for $\mathcal{L}$ too.
($\Rightarrow$) Assume $\mathbb{P}(M^u_{\mathcal{L}}, \langle \mathtt{s}, \varepsilon\rangle \models \alpha) = 1$. Then each A_i is reachable from $\langle \mathtt{s}, \varepsilon\rangle$. Hence $\mathtt{s} \neq \mathtt{x}$ since $\mathtt{x}$ is forbidden. Observe that u never picks a rule moving to $\mathtt{x}$; otherwise there is a non-zero probability to end up in $\langle \mathtt{x}, \varepsilon\rangle$ and fail to fulfill α. Thus u can be defined on $\mathcal{L} - \mathtt{x}$ and makes for a valid u'. $\square$

These two lemmas suffice to prove Theorem 4.9. One removes forbidden states until $\mathtt{s}$ itself is forbidden, or all states are allowed.

Using duality, we also obtain:

Corollary 4.12 *It is decidable whether, for given $\mathcal{L}$ and A_i's:*

- $\forall u\ \mathbb{P}(u \models \bigvee_{i=1}^{n} \Diamond\Box A_i) > 0$.

4.4 Decidable Cases for Streett Properties

Consider a formula of the form $\bigvee_{i=1}^{n}(\Diamond\Box A_i \wedge \Box\Diamond B_i)$. We have the following equivalences:

$$\exists u\ \mathbb{P}\Big(u \models \bigvee_{i=1}^{n}(\Diamond\Box A_i \wedge \Box\Diamond B_i)\Big) = 1 \tag{8}$$

$$\Leftrightarrow\ \exists u\ \mathbb{P}\Big(u \models \bigvee_{i=1}^{n}(\Box\Diamond(A_i \cap B_i))\Big) = 1 \tag{9}$$

$$\Leftrightarrow\ \exists u\ \mathbb{P}\Big(u \models \Diamond(\bigcup_{i=1}^{n}(A_i \cap B_i))\Big) = 1. \tag{10}$$

Idling is used to prove that (10) implies (8): with an arbitrary scheduler u, we associate a new scheduler u' that behaves like u as long as $\bigcup_{i=1}^{n}(A_i \cap B_i)$ is not reached, and idles when $\bigcup_{i=1}^{n}(A_i \cap B_i)$ is reached. Obviously,

$$\mathbb{P}\Big(u' \models \bigvee_{i=1}^{n}(\Diamond\Box A_i \wedge \Box\Diamond B_i)\Big)\ =\ \mathbb{P}\Big(u \models \Diamond(\bigcup_{i=1}^{n}(A_i \cap B_i))\Big). \tag{11}$$

Hence (10) implies (8). The rest is easy: (8) implies (9) since $\Diamond\Box A \wedge \Box\Diamond B$ entails $\Box\Diamond(A \cap B)$, and (9) implies (10) since $\Box\Diamond A$ entails $\Diamond A$.

Since it can be decided whether $\exists u\ \mathbb{P}(u \models \Diamond A) = 1$ (Theorem 4.5), the equivalence between (8) and (10) gives

Corollary 4.13 *It is decidable whether, for given $\mathcal{L}$ and A_i, B_i's:*

- $\forall u\ \mathbb{P}\big(u \models \bigvee_{i=1}^{n}(\Diamond\Box A_i \wedge \Box\Diamond B_i)\big) < 1$,
- $\forall u\ \mathbb{P}\big(u \models \bigwedge_{i=1}^{n}(\Box\Diamond A_i \Rightarrow \Box\Diamond B_i)\big) > 0$.

For a formula $\bigwedge_{i=1}^{n}(\Diamond\Box A_i \vee \Box\Diamond B_i)$, and a LCS $\mathcal{L}$ with initial configuration σ_0, we have the following equivalences:

$$\exists u\ \mathbb{P}\Big(u \models \bigwedge_{i=1}^{n}(\Diamond\Box A_i \vee \Box\Diamond B_i)\Big) < 1 \tag{12}$$

$$\Leftrightarrow\ \exists u\ \mathbb{P}\Big(u \models \bigwedge_{i=1}^{n}\Box(A_i \cup B_i)\Big) < 1 \tag{13}$$

$$\Leftrightarrow\ \sigma_0 \xrightarrow{*} \mathtt{Q} \setminus \bigcap_{i=1}^{n}(A_i \cup B_i). \tag{14}$$

That (14) implies (12) is easy to see: write C for $\mathtt{Q} \setminus \bigcap_{i=1}^{n}(A_i \cup B_i)$. Since C is reachable from σ_0, there exists a scheduler u that reaches C with strictly positive probability. If furthermore u idles once C is reached, u ensures that $\Diamond\Box C$, and hence $\neg \bigwedge_{i=1}^{n}(\Diamond\Box A_i \vee \Box\Diamond B_i)$ will be satisfied with strictly positive probability. The rest is easy: (12) implies (13) since $\Box(A \cup B)$ entails $\Diamond\Box A \vee \Box\Diamond B$, and (13) implies (14) since if we assume C is not reachable we conclude that $\mathbb{P}(u \models \Box\neg C) = 1$ whatever u may be.

Since it can be decided whether C is reachable, the equivalence between (12) and (14) gives

Corollary 4.14 *It is decidable whether, for given $\mathcal{L}$ and A_i, B_i's:*

- $\forall u\ \mathbb{P}\big(u \models \bigwedge_{i=1}^{n}(\Box\Diamond A_i \Rightarrow \Box\Diamond B_i)\big) = 1,$
- $\forall u\ \mathbb{P}\big(u \models \bigvee_{i=1}^{n}(\Diamond\Box A_i \wedge \Box\Diamond B_i)\big) = 0.$

5 Undecidable Qualitative Verification

We now prove the undecidability results that complete the table in Fig. 5. The proofs are by reduction from the boundedness problem (whether the set of configurations reachable from a given σ is finite or not) which is undecidable for LCS's. These reductions rely on a special gadget that we describe first.

5.1 The Cleaning Gadget

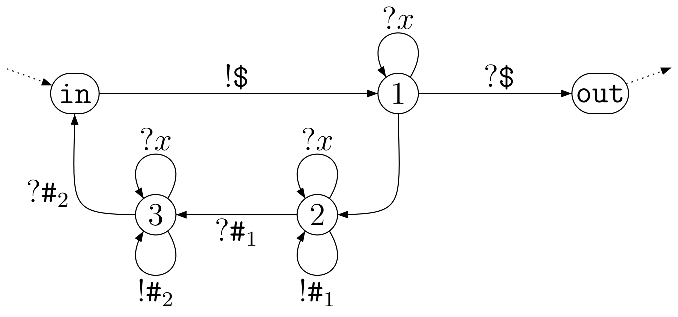

Fig. 2. Cleaning gadget, assuming $\texttt{\$}, \texttt{\#}_1, \texttt{\#}_2 \notin \texttt{M}$

For a given message alphabet M, the system described in Fig. 2 uses one channel (left implicit) and three new symbols: \$, $\texttt{\#}_1$ and $\texttt{\#}_2$. Its purpose is to force the channel to be emptied when moving from in to out, and do this *without introducing deadlocks.*

Starting from some configuration $\langle \texttt{in}, \texttt{w}\rangle$ with $\texttt{w} \in \texttt{M}^*$, the nominal behavior of the system is to move to $\langle 1, \texttt{w\$}\rangle$, to consume all letters from w with the loop on state 1 and to end in $\langle \texttt{out}, \varepsilon\rangle$ with channel empty. The role of the special symbol "\$" is to ensure the channel is empty when we reach out.

However \$ can be lost. In order to avoid deadlocks in these situations, additional "recovery" states let one go back to in and retry.

Lemma 5.1 (Absence of deadlock) *There exists a scheduler u such that $\mathbb{P}(u, \langle \texttt{in}, \texttt{w}\rangle \models \Diamond\texttt{out}) = 1$ for any* w.

Proof. [Idea] u tries to implement the nominal behavior. If $\texttt{\$}$ is lost, u moves from 1 to 2, clears the channel and eventually moves from 2 to 3. This step requires reading a $\texttt{#}_1$ marker but u can insert $\texttt{#}_1$'s if required. Then a similar clearing is made using $\texttt{#}_2$'s. Finally u goes back to `in` and tries again. ☐

The purpose of the several clearing stages is to prevent the possibility of using the recovery states to introduce extra $\texttt{\$}$'s and reach `out` with nonempty channel. The first clearing ensures that no $\texttt{\$}$ is left when we reach 3. However spurious $\texttt{#}_1$'s could remain and they have to be cleared before starting again, hence the second stage. When reaching `in` spurious $\texttt{#}_2$'s can remain but, unlike spurious $\texttt{#}_1$'s, these cause no problem.

Formally, let S be the set of configurations described by the following regular expression

$$\begin{aligned} S \stackrel{\text{def}}{=} & \langle \texttt{in}, (\texttt{M} + \texttt{#}_2)^* \rangle + \langle 1, (\texttt{M} + \texttt{#}_2)^* (\texttt{\$} + \varepsilon) \rangle \\ & + \langle 2, (\texttt{M} + \texttt{#}_2)^* (\texttt{\$} + \varepsilon) \texttt{#}_1^* \rangle \ + \ \langle 3, \texttt{#}_1^* \texttt{#}_2^* \rangle \ + \ \langle \texttt{out}, \varepsilon \rangle. \end{aligned} \tag{15}$$

Lemma 5.2 *S is an invariant: from a configuration in S, one only reach configurations in S.*

Proof. S is downward-closed (w.r.t. subconfigurations), so the invariant property is respected by message losses[5]. We also have to check that each transition rule respects it. We consider a few important cases and let the reader check the other rules.

$1 \xrightarrow{?\texttt{\$}}$ **`out`:** in state 1, S-configurations have the form $\langle 1, (\texttt{M} + \texttt{#}_2)^* (\texttt{\$} + \varepsilon) \rangle$. Out of these, the only one offering to read a $\texttt{\$}$ is $\langle 1, \texttt{\$} \rangle$. This leads to $\langle \texttt{out}, \varepsilon \rangle$ that is in S.

$2 \xrightarrow{?\texttt{#}_1}$ **3:** in state 2, S-configurations have the form $\langle 2, (\texttt{M} + \texttt{#}_2)^* (\texttt{\$} + \varepsilon) \texttt{#}_1^* \rangle$. Out of these, the only ones offering to read a $\texttt{#}_1$ have the form $\langle 2, \texttt{#}_1^+ \rangle$. They lead to configurations of the form $\langle 3, \texttt{#}_1^* \rangle$ that are in S.

$3 \xrightarrow{?\texttt{#}_2}$ **`in`:** in state 3, S-configurations have the form $\langle 3, \texttt{#}_1^* \texttt{#}_2^* \rangle$. Out of these, the only ones offering to read a $\texttt{#}_2$ have the form $\langle 3, \texttt{#}_2^+ \rangle$. They lead to configurations of the form $\langle \texttt{in}, \texttt{#}_2^* \rangle$ that are in S.

☐

Since S is an invariant, the gadget is correct: from $\langle \texttt{in}, \texttt{w} \rangle$ with $\texttt{w} \in \texttt{M}^*$, one can only reach `out` with empty channel.

5.2 Undecidability for $\forall u\ \mathbb{P}(\ldots) = 0$

Theorem 5.3 *It is undecidable whether, for given $\mathcal{L}$ and A, B:*

[5] Observe that the three clearing loops in states 1, 2 and 3 are not required since message losses subsume them. We only put them in for clarity.

- $\forall u\ \mathbb{P}(u \models \Box\Diamond A \wedge \Box\Diamond B) = 0.$

Let $\mathcal{L} = \langle \mathtt{Q}, \{\mathtt{c}\}, \mathtt{M}, \Delta\rangle$ be a LCS with a single channel and a designated initial configuration $\langle \mathtt{r}_0, \varepsilon\rangle$. We modify $\mathcal{L}$ by adding the cleaning gadget and two control states: `retry` and `success`. We also add rules allowing to jump from every "original" state in `Q` to `retry` or `success`. When in `success`, one can move to `retry` with a read or idle unconditionally. When in `retry`, one can go back to $\langle \mathtt{r}_0, \varepsilon\rangle$ through the cleaning gadget.

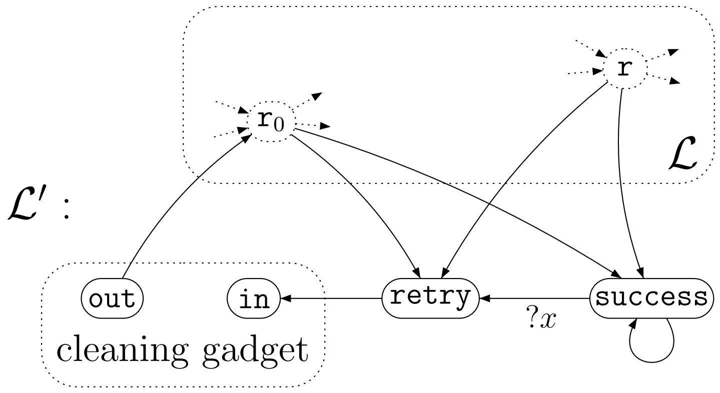

Fig. 3. The LCS $\mathcal{L}'$ associated with $\mathcal{L}$

Write $\mathcal{L}'$ for the resulting LCS. Since the cleaning gadget lets one go back to the initial configuration of $\mathcal{L}$, any behavior of $\mathcal{L}'$ is a succession of behaviors of $\mathcal{L}$ separated by visits to the additional states.

Proposition 5.4 *Assume that $\mathcal{L}$ is bounded. Then, for all schedulers u, $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry}) = 0$.*

Proof. Let u be any scheduler and consider the runs consistent with u that visit `success` infinitely often. Let π be one such run: either π jumps from $\mathcal{L}$ to `success` infinitely many times, or it ends up idling in `success`. In the last case, π does not satisfy $\Box\Diamond\mathtt{retry}$. In the first case, and since $\mathcal{L}$ is bounded, π can only jump to `success` from finitely many different configurations. Hence, for each such jump, the probability that it ends in $\langle\mathtt{success}, \varepsilon\rangle$ is at least τ^m, where m is the size of the largest reachable configuration in $\mathcal{L}$. Therefore $\langle\mathtt{success}, \varepsilon\rangle$ will be visited almost surely, or more formally,

$$\begin{aligned} &\mathbb{P}(u \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathcal{L}) \\ = \ &\mathbb{P}(u \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathcal{L} \wedge \Diamond\langle\mathtt{success}, \varepsilon\rangle). \end{aligned} \tag{16}$$

Now, since one cannot reach $\mathcal{L}$ from $\langle\mathtt{success}, \varepsilon\rangle$, the probability in (16) is 0. Finally, $\mathbb{P}(u \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry}) = 0$. □

Proposition 5.5 *Assume that $\mathcal{L}$ is unbounded. Then there exists a scheduler u s.t. $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry}) > 0$.*

Proof. We describe the required scheduler. Because $\mathcal{L}$ is unbounded, we can pick a sequence $(\langle r_n, w_n\rangle)_{n=1,2,\ldots}$ of reachable configurations s.t. $|w_n| \geq n$.

The scheduler works in phases numbered $1, 2, \ldots$ When phase n starts, u is in the initial configuration $\langle \mathtt{r}_0, \varepsilon\rangle$ and tries to reach $\langle r_n, w_n\rangle$. In principle, this can be achieved (since $\langle r_n, w_n\rangle$ is reachable) but it requires that the right messages are lost at the right times. These losses are probabilistic and u cannot control them. Thus u aims for $\langle r_n, w_n\rangle$ and hopes for the best. It goes on according to plan as long as losses occur as hoped. When a "wrong" loss occurs, u resigns temporarily, jumps directly to `retry`, reaches $\langle \mathtt{r}_0, \varepsilon\rangle$ via the cleaning gadget, and then tries again to reach $\langle r_n, w_n\rangle$. When $\langle r_n, w_n\rangle$ is eventually reached (which will happen almost surely given enough retries), u jumps to `success`, from there to `retry`, and initiates phase $n+1$. With these successive phases, u tries to visit `success` (and `retry`) an infinite number of times. We now show that it succeeds with nonzero probability.

When jumping from $\langle r_n, w_n\rangle$ to `success`, there is a probability $\mathbb{P}_l(w_n, \varepsilon)$ that all messages in the channel are lost, leaving us in $\langle \mathtt{success}, \varepsilon\rangle$. When this happens, u is not able to initiate phase $n+1$ (moving from `success` to `retry` requires a nonempty channel) and will not visit `retry` again. Finally,

$$\mathbb{P}(u \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry}) = \prod_{n=1}^{\infty} 1 - \mathbb{P}_l(w_n, \varepsilon) \geq \prod_{n=1}^{\infty} 1 - \tau^n > 0 \quad (17)$$

using Fact 3.5. $\Box$

Corollary 5.6 *$\mathcal{L}$ is unbounded if, and only if there exists a scheduler u s.t. $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry}) > 0$.*

This proves Theorem 5.3 since it is undecidable whether a given LCS is bounded.

With duality we obtain:

Corollary 5.7 *It is undecidable whether, for given $\mathcal{L}$ and A, B:*

- $\forall u\ \mathbb{P}(u \models \Diamond\Box A \vee \Diamond\Box B) = 1$.

5.3 Undecidability for $\forall u\ \mathbb{P}(\ldots) < 1$

Theorem 5.8 *It is undecidable whether, for given $\mathcal{L}$ and A, B, C:*

- $\forall u\ \mathbb{P}(u \models \Box\Diamond A \wedge \Box\Diamond B \wedge \Diamond\Box C) < 1$.

For Theorem 5.8, we use a slightly modified reduction. The general principle is like in the previous reduction, except that the configuration $\langle \mathtt{success}, \varepsilon\rangle$ is not a sink state anymore: one can escape through `fail` and go on with the game.

Proposition 5.9 *Assume that $\mathcal{L}$ is bounded. Then, for all schedulers u, $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\mathtt{success} \wedge \Box\Diamond\mathtt{retry} \wedge \Diamond\Box\neg\mathtt{fail}) < 1$.*

Proof. [Idea] Similar to the proof of Prop. 5.4. Visiting both `success` and `retry` infinitely many times requires jumping infinitely many times from $\mathcal{L}$ to

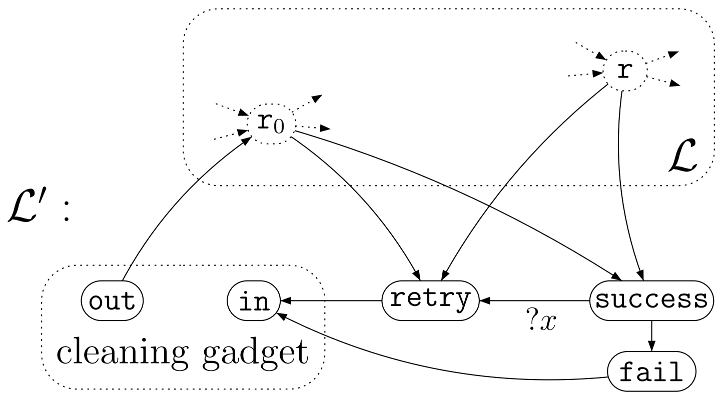

Fig. 4. The LCS $\mathcal{L}'$ associated with $\mathcal{L}$

success. Since $\mathcal{L}$ is bounded, the probability that each such jump ends in $\langle \texttt{success}, \varepsilon \rangle$ cannot be made arbitrarily low and one will almost surely visit $\langle \texttt{success}, \varepsilon \rangle$ infinitely many times. Then escape through fail will have to be used infinitely many times. □

Proposition 5.10 *Assume that $\mathcal{L}$ is unbounded. Then there exists a scheduler u s.t. $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\texttt{success} \wedge \Box\Diamond\texttt{retry} \wedge \Diamond\Box\neg\texttt{fail}) = 1$.*

Proof. We consider a scheduler similar to the one we used in the proof of Prop. 5.5. The difference is that, if u ends in $\langle \texttt{success}, \varepsilon \rangle$ after it jumps from $\langle \texttt{r}_n, \texttt{w}_n \rangle$ to success at the end of phase n, it simply uses fail to avoid deadlock and manage to initiate phase $n+1$ anyway.

Because $\langle \texttt{success}, \varepsilon \rangle$ is not a deadlocked configuration, our scheduler ensures $\mathbb{P}(\Box\Diamond\texttt{success} \wedge \Box\Diamond\texttt{retry}) = 1$. The probability that fail is visited finitely many times only is the probability that u ends in $\langle \texttt{success}, \varepsilon \rangle$ only finitely often, that is, the probability that $\langle \texttt{success}, \varepsilon \rangle$ is not visited anymore after some phase k. This is

$$\lim_{k\to\infty} \left(\prod_{n=k}^{\infty} 1 - \mathbb{P}_l(\texttt{w}_n, \varepsilon) \right) \geq \lim_{k\to\infty} \left(\prod_{n=k}^{\infty} 1 - \tau^n \right) = 1 \tag{18}$$

using Fact 3.5 again. Hence $\mathbb{P}(u \models \Diamond\Box\neg\texttt{fail}) = 1$. □

Corollary 5.11 *$\mathcal{L}$ is unbounded if, and only if, there exists a scheduler u s.t. $\mathbb{P}(\mathcal{L}' \models \Box\Diamond\texttt{success} \wedge \Box\Diamond\texttt{retry} \wedge \Diamond\Box\neg\texttt{fail}) = 1$.*

This proves Theorem 5.8.

With duality we obtain:

Corollary 5.12 *It is undecidable whether, for given $\mathcal{L}$ and A, B, C:*

- $\forall u\ \mathbb{P}(u \models \Diamond\Box A \vee \Diamond\Box B \vee \Box\Diamond C) > 0.$

6 Conclusions and Perspectives

When verifying lossy channel systems, adopting a probabilistic view of losses is a way of enforcing progress and ruling out some unrealistic behaviors (under probabilistic reasoning, it is extremely unlikely that all messages will always be lost). Progress could be enforced with fairness assumptions, but assuming fair losses makes verification undecidable [3,11]. It seems this undecidability is an artifact of the standard rigid view asking whether no incorrect behavior exists, when we could be content with the weaker statement that incorrect behaviors are extremely unlikely.

In [7] we proposed a model for channel systems where at each step each message in transit can be lost with some fixed probability $\tau \in (0,1)$, and where the nondeterministic nature of the model is preserved. This model is more realistic than earlier proposals since it uses the local-fault model for probabilistic losses, and since it does not require to view the rules of the system as probabilistic.

For this model, we showed that qualitative verification is undecidable in general, but we exhibited many important subcases where decidability is regained (see table of results in Fig. 5). Decidability is obtained by reduc-

	$\mathbb{P}(\ldots) = 1$	$\mathbb{P}(\ldots) = 0$	$\mathbb{P}(\ldots) < 1$	$\mathbb{P}(\ldots) > 0$
$\Diamond A$	D (Cor. 4.2)	D (Cor. 4.4)	D (Th. 4.5)	D (Cor. 4.2)
$\Box A$	D (Cor. 4.4)	D (Cor. 4.3)	D (Cor. 4.3)	D (Cor. 4.8)
$\bigwedge_i \Box\Diamond A_i$	D (Cor. 4.14)	U (Th. 5.3)	D (Th. 4.9)	D (Cor. 4.13)
$\bigvee_i \Diamond\Box A_i$	U (Cor. 5.7)	D (Cor. 4.14)	D (Cor. 4.13)	D (Cor. 4.12)
$\bigwedge_i(\Box\Diamond A_i \Rightarrow \Box\Diamond B_i)$	D (Cor. 4.14)	U (Th. 5.3)	U (Th. 5.8)	D (Cor. 4.13)
$\bigvee_i(\Diamond\Box A_i \wedge \Box\Diamond B_i)$	U (Cor. 5.7)	D (Cor. 4.14)	D (Cor. 4.13)	U (Cor. 5.12)

Fig. 5. (Un)Decidability of qualitative verification

ing qualitative properties to reachability questions in the underlying non-probabilistic transition system. Since in this model qualitative properties do not depend on the exact value of the fault rate τ, the issue of what is a realistic value for τ is avoided, and one can establish correctness results that apply to all fault rates.

For qualitative properties that are undecidable, we advocate restricting the verification to finite-memory schedulers (see [7]).

An important open question is the decidability of quantitative properties, but [15] shows that already the fully deterministic case raises very difficult problems.

References

[1] P. A. Abdulla, A. Annichini, and A. Bouajjani. Symbolic verification of lossy channel systems: Application to the bounded retransmission protocol. In *Proc. 5th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99), Amsterdam, The Netherlands, Mar. 1999*, volume 1579 of *Lecture Notes in Computer Science*, pages 208–222. Springer, 1999.

[2] 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.

[3] P. A. Abdulla and B. Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996.

[4] P. A. Abdulla and B. Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996.

[5] 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.

[6] 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.

[7] 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.

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

[9] G. Cécé, A. Finkel, and S. Purushothaman Iyer. Unreliable channels are easier to verify than perfect channels. *Information and Computation*, 124(1):20–31, 1996.

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

[11] 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.

[12] R. Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 297(1–3):337–354, 2003.

[13] P. Panangaden. Measure and probability for concurrency theorists. *Theoretical Computer Science*, 253(2):287–309, 2001.

[14] S. Purushothaman Iyer and M. Narasimha. Probabilistic lossy channel systems. In *Proc. 7th Int. Joint Conf. Theory and Practice of Software Development (TAPSOFT'97), Lille, France, Apr. 1997*, volume 1214 of *Lecture Notes in Computer Science*, pages 667–681. Springer, 1997.

[15] A. Rabinovich. Quantitative analysis of probabilistic lossy channel systems. In *Proc. 30th Int. Coll. Automata, Languages, and Programming (ICALP'2003), Eindhoven, NL, July 2003*, volume 2719 of *Lecture Notes in Computer Science*, pages 1008–1021. Springer, 2003.

[16] 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.

[17] Ph. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *Information Processing Letters*, 83(5):251–261, 2002.

[18] Ph. Schnoebelen. The verification of probabilistic lossy channel systems. In *Validation of Stochastic Systems*. Springer, 2004. To appear.

[19] M. Y. Vardi. Probabilistic linear-time model checking: An overview of the automata-theoretic 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 265–276. Springer, 1999.