Verification of Probabilistic Systems with Faulty Communication [⋆]

P. A. Abdulla [a], N. Bertrand [b],
A. Rabinovich [c], and Ph. Schnoebelen [b]

[a] *Uppsala University, Sweden*

[b] *LSV, CNRS & ENS de Cachan, France*

[c] *Tel Aviv University, Israel*

Abstract

Many protocols are designed to operate correctly even in the case where the underlying communication medium is faulty. To capture the behavior of such protocols, *Lossy Channel Systems* (LCS's) have been proposed. In an LCS the communication channels are modeled as unbounded FIFO buffers which are unreliable in the sense that they can nondeterministically lose messages.

Recently, several attempts have been made to study *Probabilistic Lossy Channel Systems* (PLCS's) in which the probability of losing messages is taken into account. In this article, we consider a variant of PLCS's which is more realistic than those studied previously. More precisely, we assume that during each step in the execution of the system, each message may be lost with a certain predefined probability. We show that for such systems the following model-checking problem is decidable: to verify whether a linear-time property definable by a finite-state ω-automaton holds with probability one. We also consider other types of faulty behavior, such as corruption and duplication of messages, and insertion of new messages, and show that the decidability results extend to these models.

1 Introduction

Finite-state machines which communicate asynchronously through unbounded buffers have been popular in the modeling of communication protocols [7,8].

[⋆] This article is based on [4] and [6].

Email addresses: `parosh@it.uu.se` (P. A. Abdulla), `bertrand@lsv.ens-cachan.fr` (N. Bertrand), `rabino@math.tau.ac.il` (A. Rabinovich), `phs@lsv.ens-cachan.fr` (Ph. Schnoebelen).

Preprint submitted to Elsevier Science *7 September 2005*

One disadvantage with such a model is that it has the full computational power of Turing machines [8], implying undecidability of all nontrivial verification problems. On the other hand, many protocols are designed to operate correctly even in the case where the underlying communication medium is faulty. To capture the behavior of such protocols, *lossy channel systems* (LCS's) [3,9] have been proposed as an alternative model. In an LCS the communication channels are modeled as FIFO buffers which are unbounded but also unreliable in the sense that they can nondeterministically lose messages. For LCS's it has been shown that the reachability problem is decidable [3] while progress properties are undecidable [2].

Since we are dealing with unreliable communication media, it is natural to consider models where the probability of errors is taken into account. Recently, several attempts have been made to study *Probabilistic Lossy Channel Systems* (PLCS's) which introduce randomization into the behavior of LCS's [17,5,1]. The decidability of model checking for the proposed models depends heavily on the semantics provided. The works in [5,1] define different semantics for PLCS's depending on the manner in which the messages may be lost inside the channels.

Baier and Engelen consider a model where it is assumed that at most one single message may be lost during each step of the execution of the system [5]. They show decidability of model checking under the assumption that the probability of losing messages is at least 0.5. This implies that, along each computation of the system, there are almost surely infinitely many points where the channels of the system are empty, and therefore the model-checking problem reduces to checking decidable properties of the underlying (non-probabilistic) LCS.

The model in [1] assumes that messages can only be lost during send operations. Once a message is successfully sent to a channel, it continues to reside inside the channel until it is removed by a receive operation. Both the reachability and repeated reachability problems are shown to be undecidable for this model of PLCS's. The idea of the proof is to choose sufficiently low probabilities for message losses to enable the system to simulate the behavior of (non-probabilistic) systems with perfect channels.

In this article, we consider a variant of PLCS's which are more realistic than that in [5,1]. More precisely, we assume that, during each step in the execution of the system, each message may be lost with a certain predefined probability. This means that the probability of losing a certain message will not decrease with the length of the channels (as it is the case with [5]). As a consequence, and in contrast to [5], our method is not dependent on the precise transition probabilities for establishing the qualitative properties of the system. For this model, we show decidability of both the reachability and repeated reachability problems.

The decidability results are achieved in two steps. First, we prove general theorems about (infinite-state) Markov chains which serve as sufficient conditions for decidability of model checking [1]. To do that, we introduce the concept of *attractor sets*: all computations of the system eventually visit the attractor almost surely. The existence of finite attractors implies that deciding reachability and repeated reachability in the PLCS can be reduced to checking reachability problems in the underlying LCS. Next, we show that all PLCS's, when interpreted according to our semantics, have finite attractors. More precisely, we prove the existence of an attractor defined as the set of all configurations where the channels are empty. In fact, the systems considered in [5] have the same attractor (when the probability of losing messages is at least 0.5), and therefore the decidability results in [5] can be seen as a consequence of the properties we show for attractors.

We also show that our decidability results extend to PLCS's with different sources of unreliability, such as duplication, corruption, and insertion combined with lossiness [9]. Furthermore, we extend our decidability results to more general properties specified by finite-state automata or equivalently by formulas of the monadic logic of order.

Outline. In the next two Sections we recall basic notions on transition systems and Markov chains respectively, and we introduce the concept of attractors. In Section 4 we present sufficient conditions for checking reachability and repeated reachability for Markov chains. In Section 5 we extract from these conditions algorithms for PLCS's. In Section 6 we consider models involving different sources of unreliability combined with lossiness. In Section 7 we generalize our results to the verification of properties definable by the ω-behavior of finite-state automata (or equivalently by formulas in the monadic logic of order). Finally, we give conclusions and directions for future work in Section 8.

2 Transition Systems

In this section, we recall some basic concepts of transition systems.

A *transition system* T is a pair $(S, \longrightarrow)$ where S is a (possibly infinite) set of states, and $\longrightarrow$ is a binary relation on S. We write $s_1 \longrightarrow s_2$ to denote that $(s_1, s_2) \in \longrightarrow$ and use $\xrightarrow{*}$ and $\xrightarrow{+}$ to denote the reflexive transitive (resp. transitive) closure of $\longrightarrow$. We say that s_2 is *reachable* from s_1 if $s_1 \xrightarrow{*} s_2$. For sets $Q_1, Q_2 \subseteq S$, we say that Q_2 is *reachable* from Q_1, denoted $Q_1 \xrightarrow{*} Q_2$, if

[1] Existing works on the verification of infinite-state Markov chains, e.g., the probabilistic pushdown automata considered in [12], rely on other methods.

there are $s_1 \in Q_1$ and $s_2 \in Q_2$ with $s_1 \xrightarrow{*} s_2$. A *path* p from s to s' is of the form $s_0 \longrightarrow s_1 \longrightarrow \cdots \longrightarrow s_n$, where $s_0 = s$ and $s_n = s'$. For a set $Q \subseteq S$, we say that p *reaches* Q if $s_i \in Q$ for some $i : 0 \leq i \leq n$. For $Q_1, Q_2 \subseteq S$, we define the set $\mathit{Until}(Q_1, Q_2)$ to be the set of all states s_0 such that there is a path $s_0 \longrightarrow s_1 \longrightarrow \cdots \longrightarrow s_n$ from s_0 satisfying the following property: there is an $i : 0 \leq i \leq n$ such that $s_i \in Q_2$ and for each $j : 0 \leq j < i$ we have $s_j \in Q_1$.

For $Q \subseteq S$, we define the *graph* of Q, denoted $\mathit{Graph}(Q)$, to be the subgraph of $(S, \xrightarrow{+})$ induced by Q, that is, the transition system $(Q, \longrightarrow')$ where $s_1 \longrightarrow' s_2$ iff $s_1 \xrightarrow{+} s_2$.

A *strongly connected component (SCC)* in T is a maximal set $C \subseteq S$ such that $s_1 \xrightarrow{*} s_2$ for each $s_1, s_2 \in C$. We say that C is a *bottom SCC (BSCC)* if there is no other SCC C' in T with $C \xrightarrow{*} C'$. In other words, the BSCCs are the leaves in the acyclic graph of SCCs (ordered by reachability).

We shall later refer to the following two problems for transition systems:

Reachability

Instance: A transition system $T = (S, \longrightarrow)$, and sets $Q_1, Q_2 \subseteq S$.

Question: Is Q_2 reachable from Q_1?

Constrained Reachability

Instance: A transition system $T = (S, \longrightarrow)$, a state s, and sets $Q_1, Q_2 \subseteq S$.

Question: Does $s \in \mathit{Until}(Q_1, Q_2)$?

3 Markov Chains and Their Attractors

In this section, we recall some basic concepts of *Markov chains* and introduce *attractors* which will later play a key role in our analysis.

A *Markov chain* M is a pair (S, P) where S is a countable (possibly infinite) set of states and P is a mapping from $S \times S$ to the real interval $[0, 1]$, such that $\sum_{s' \in S} P(s, s') = 1$ for each $s \in S$. A *computation* π *(from* s_0*)* of M is an infinite sequence $s_0, s_1, \ldots$ of states. We use $\pi(i)$ to denote s_i.

A Markov chain induces a transition system, where the transition relation consists of pairs of states related by strictly positive probabilities. Formally, the *underlying transition system* of M is $(S, \longrightarrow)$ where $s_1 \longrightarrow s_2$ iff $P(s_1, s_2) > 0$. In this manner, the concepts defined for transition systems can be lifted to Markov chains. For instance, an SCC in M is an SCC in the underlying transition system.

A Markov chain (S, P) induces a natural measure on the set of computations from every state s (see, e.g., [14] or [15]).

Let us recall some basic notions from probability theory.

A *measurable space* is a pair (Ω, Δ) consisting of a non empty set Ω and a σ-algebra Δ of its subsets that are called *measurable sets* and represent random events in some probability context. A *σ-algebra over* Ω contains Ω and is closed under complementation and countable union. Adding to a measurable space a *probability measure* $\mathbb{P} : \Delta \to [0, 1]$ that is countably additive and such that $\mathbb{P}(\Omega) = 1$, one obtains a *probability space* $(\Omega, \Delta, \mathbb{P})$.

Consider a state s of a Markov chain (S, P). Over the set of computations that start at s, the probability space $(\Omega, \Delta, \mathbb{P})$ is defined as follows:

- $\Omega = sS^\omega$ is the set of all infinite sequences of states starting from s,
- Δ is the σ-algebra generated by the basic cylindric sets $D_u = uS^\omega$, for every $u \in sS^*$,
- $\mathbb{P}$, the probability measure, is defined by $\mathbb{P}(D_u) = \prod_{0 \leq i < n} P(s_i, s_{i+1})$ where $u = s_0 s_1 ... s_n$; it is well known that this measure is extended in a unique way to the elements of the σ-algebra generated by the basic cylindric sets.

Let $Q \subseteq S$ be a set of states. Using standard temporal logic notations, we write $\pi \models \Diamond Q$ to denote that π visits Q (i.e., $\pi(i) \in Q$ for some $i \in \mathbb{N}$) and $\pi \models \Box\Diamond Q$ to denote that π visits Q infinitely many times (i.e., $\pi(i) \in Q$ for infinitely many $i \in \mathbb{N}$). For singleton sets, we shortly write, e.g., "$\Diamond s_1$" instead of "$\Diamond\{s_1\}$".

It is well known (and easily seen) that the set of executions in sS^ω that satisfy some linear-time formula φ of the form $\Diamond Q$ or $\Box\Diamond Q$ is measurable in $(\Omega, \Delta, \mathbb{P})$ [21]. When φ is such a property, or a Boolean combination of these, we write $\mathbb{P}_s(\varphi)$ for the measure $\mathbb{P}(\{\pi : \pi \text{ starts from } s \text{ and satisfies } \varphi\})$ and call it *the probability that φ will be satisfied (starting from s).*

Consider a Markov chain (S, P). A *recurrent* state is a state $s \in S$ such that $\mathbb{P}_s(\Box\Diamond s) = 1$, i.e., starting from s one visits s infinitely often with probability 1. A *transient* state is a state $s \in S$ such that $\mathbb{P}_s(\Box\Diamond s) = 0$, i.e., starting from s one visits s infinitely often with probability 0. Since, starting from s, the probability of visiting s again is either 1 or less than 1, all states are either recurrent or transient. Furthermore, all states reachable from a recurrent state are recurrent.

Similarly, when state s_2 is reachable from s_1, the probability of visiting s_1 infinitely often coincides with the probability of visiting s_1 and s_2 infinitely often (starting from a given s). Or, using temporal logic notation:

Lemma 3.1 *If* $s_1 \xrightarrow{*} s_2$ *then* $\mathbb{P}_s(\Box\Diamond s_1 \wedge \Box\Diamond s_2) = \mathbb{P}_s(\Box\Diamond s_1)$.

PROOF. (Idea) Since s_2 is reachable from s_1, every time one visits s_1 there is a strictly positive probability that s_2 will be visited before a given number of steps. Thus if one visits s_1 infinitely often, then almost surely s_2 is visited eventually, and then almost surely visited infinitely often. □

We now introduce *attractors*, which will play a key role in our analysis:

Definition 3.2 *A set* $A \subseteq S$ *of states is an* attractor *if* $\mathbb{P}_s(\Diamond A) = 1$ *for all* $s \in S$.

In other words, regardless of the state in which we start, we will almost surely enter the attractor eventually. Observe that if A is an attractor, then for all $s \in S$, $\mathbb{P}_s(\Box\Diamond A) = 1$: we will almost surely visit A infinitely many times.

The next Lemma describes a property of the BSCCs of the graph of a finite attractor A, which will be useful in our algorithms (to prove Lemma 4.1 and Lemma 4.2).

Lemma 3.3 *Consider a finite attractor* A*, a BSCC* C *in Graph*(A)*, and a state* $s \in C$*. Then, for all* $s' \in C$*,* $\mathbb{P}_s(\Box\Diamond s') = 1$.

PROOF. $\mathbb{P}_s(\Box\Diamond A) = 1$ since A is an attractor. Since C is a BSCC of *Graph*(A), $A \setminus C$ is not reachable from C. Thus $\mathbb{P}_s(\Box\Diamond A) = 1$ translates into $\mathbb{P}_s(\Box\Diamond C) = 1$ (since $s \in C$). Now, A being finite, C is finite too and there must be some $s' \in C$ s.t. $\mathbb{P}_s(\Box\Diamond s') = 1$. Since C is a BSSC of *Graph*(A) every state in C is reachable from every other state, so that $\mathbb{P}_s(\Box\Diamond s') = 1$ for some $s' \in C$ entails $\mathbb{P}_s(\Box\Diamond s') = 1$ for all $s' \in C$ (by Lemma 3.1). □

The next Lemma enables us to characterize certain properties of the sets of reachable states in the systems of Section 5 through Section 6

Lemma 3.4 *Consider a finite attractor* A *and a set* A'*. If* A' *is reachable from each state* $s \in A$*, then* A' *is also an attractor.*

PROOF. Consider $s \in S$. We have $\mathbb{P}_s(\Box\Diamond A) = 1$. Since A is finite, there must be $s_1 \in A$ such that $\mathbb{P}_s(\Box\Diamond s_1) = 1$. By assumption, there is $s_2 \in A'$ reachable from s_1. By Lemma 3.1, $\mathbb{P}_s(\Box\Diamond s_2) = 1$, hence $\mathbb{P}_s(\Box\Diamond A') = 1$. (Observe that s_1 and s_2 depend on s.) □

Lemma 3.5 *Assume A is a finite attractor and write $C_1, \ldots, C_p$ for the BSCCs of Graph(A). For any $s \in S$*

$$\mathbb{P}_s(\Diamond C_1) + \cdots + \mathbb{P}_s(\Diamond C_p) = \mathbb{P}_s(\Box\Diamond C_1) + \cdots + \mathbb{P}_s(\Box\Diamond C_p) = 1.$$

PROOF. Since $C_1 \cup \cdots \cup C_p$ is reachable from any state in A, it is an attractor (Lemma 3.4). For $i \neq j$, C_i is not reachable from C_j, hence $\mathbb{P}_s(\Diamond C_i \wedge \Diamond C_j) = 0$. Thus $\mathbb{P}_s(\Diamond(C_1 \cup \cdots \cup C_p)) = 1$ entails $\mathbb{P}_s(\Diamond C_1) + \cdots + \mathbb{P}_s(\Diamond C_p) = 1$. We conclude by observing that, for any i, $\mathbb{P}_s(\Diamond C_i) = \mathbb{P}_s(\Box\Diamond C_i)$ (Lemma 3.3). □

This can be refined in

Lemma 3.6 *Assume A is a finite attractor and $s \in S$ is some state. Let $C_1, \ldots, C_k$ be the BSCCs in Graph(A) that are reachable from s. Then*

$$\mathbb{P}_s(\Diamond C_1) + \cdots + \mathbb{P}_s(\Diamond C_k) = \mathbb{P}_s(\Box\Diamond C_1) + \cdots + \mathbb{P}_s(\Box\Diamond C_k) = 1.$$

PROOF. From Lemma 3.5, relying on the fact that $\mathbb{P}_s(\Diamond C) = \mathbb{P}_s(\Box\Diamond C) = 0$ when C is not reachable from s. □

4 Reachability Analysis for Markov Chains

In this section we explain how to check reachability and repeated reachability for Markov chains. We show how to reduce qualitative properties of the above two types into the analysis of the underlying (non-probabilistic) transition system of the Markov chain.

Formally, the problems we consider are:

<u>**Probabilistic Reachability**</u>

Instance: A Markov chain $M = (S, P)$, a state $s \in S$, and a set $Q \subseteq S$.

Question: Does $\mathbb{P}_s(\Diamond Q) = 1$?, i.e., is Q almost surely reached from s?

Probabilistic Repeated Reachability

Instance: A Markov chain $M = (S, P)$, a state $s \in S$, and a set $Q \subseteq S$.

Question: Does $\mathbb{P}_s(\Box\Diamond Q) = 1$?, i.e., is Q almost surely repeatedly reached from s?

Observe that the above problems are not yet *algorithmic* problems since we did not specify how an instance is to be finitely encoded (we do not assume that the Markov chain (S, P) is finite). In Sections 5–7 we consider reachability and repeated reachability problems when countable Markov chains are described by probabilistic lossy channel systems. For such finite descriptions we investigate the corresponding algorithmic problems.

For a countable Markov chain (S, P) containing a finite attractor A, the following Lemma reduces probabilistic reachability problems in (S, P) to reachability problems in *Graph*(A).

Lemma 4.1 *Assume A is a finite attractor, $s \in S$ is some state and $Q \subseteq S$ is some set of states. Then $\mathbb{P}_s(\Diamond Q) < 1$ iff there exists a BSCC C in Graph(A) such that:*
(1) Q is not reachable from C, and
(2) it is possible to reach C from s without traversing Q.

PROOF. ($\Leftarrow$): Let u be a finite path leading from s to C without visiting Q. Since Q is not reachable from C, any run with prefix u never visits Q. The set of such runs has measure $\mathbb{P}(D_u) > 0$. Thus $\mathbb{P}_s(\neg\Diamond Q) \geq \mathbb{P}(D_u) > 0$, entailing $\mathbb{P}_s(\Diamond Q) < 1$.

($\Rightarrow$): Write $C_1, \ldots, C_p$ for the BSCCs of *Graph*(A) and UC for $C_1 \cup \cdots \cup C_p$. Since UC is an attractor, $\mathbb{P}_s(\Box\Diamond UC) = 1$, so that $\mathbb{P}_s(\Diamond Q) = \mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond UC)$. Since a C_i is not reachable from a C_j when $i \neq j$, the events $\Box\Diamond C_1$, $\Box\Diamond C_2$, $\ldots$, $\Box\Diamond C_p$ form a partition of $\Box\Diamond UC$. Hence $\mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond UC) = \mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond C_1) + \cdots + \mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond C_p)$. Thus $\mathbb{P}_s(\Diamond Q) < 1$ entails that $\mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond C) < \mathbb{P}_s(\Box\Diamond C)$ for one C among $C_1, \ldots, C_p$.

If Q is reachable from C, then $\mathbb{P}_s(\Box\Diamond C) = \mathbb{P}_s(\Box\Diamond C \wedge \Box\Diamond Q)$ (by Lemma 3.1). Similarly, if all runs from s that reach C visit Q, then $\mathbb{P}_s(\Box\Diamond C) = \mathbb{P}_s(\Box\Diamond C \ \wedge \ \Diamond Q)$. Thus, if $\mathbb{P}_s(\Diamond Q \ \wedge \ \Box\Diamond C) < \mathbb{P}_s(\Box\Diamond C)$, then C satisfies (1) and (2). □

From Lemma 4.1 we conclude that we can define a scheme for solving the probabilistic reachability problem as follows.

```
Scheme – Probabilistic Reachability

Input: Markov chain $M = (S, P)$ with an underlying transition
system $T = (S, \to)$, a state $s \in S$, and a set $Q \subseteq S$.
Output: true if $Q$ is reached from $s$ with probability one.
begin
    1. construct a finite attractor $A$
    2. construct $\mathit{Graph}(A)$ and list its BSCCs $C_1, \ldots, C_p$
    3. for each BSCC $C$ in $\mathit{Graph}(A)$
        3a. if $s \in \mathit{Until}(S \setminus Q, C)$
        3b. and $\neg\left(C \xrightarrow{*} Q\right)$
        3c. then return(false)
    4. return(true)
end
```

For solving the probabilistic repeated reachability problem, we rely on the following Lemma:

Lemma 4.2 *Consider a finite attractor A, a state $s \in S$, and a set $Q \subseteq S$. Then $\mathbb{P}_s(\Box\Diamond Q) = 1$ iff Q is reachable from each BSCC C of Graph(A) that is reachable from s.*

PROOF. ($\Leftarrow$) Let $C_1, \ldots, C_k$ be the BSCCs that are reachable from s. Write UC for $C_1 \cup \cdots \cup C_k$. We have $\mathbb{P}_s(\Box\Diamond UC) = 1$ (by Lemma 3.6) and then $\mathbb{P}_s(\Box\Diamond UC \wedge \Box\Diamond Q) = 1$ (by Lemma 3.1) since Q is reachable from any state in UC.
($\Rightarrow$) If Q is not reachable from C then $\mathbb{P}_s(\Box\Diamond Q) \leq 1 - \mathbb{P}_s(\Diamond C) < 1$. $\Box$

From Lemma 4.2 we conclude that we can define a scheme for solving the repeated reachability problem by modifying the previous algorithmic scheme as follows:

```
        3a. if $C$ is reachable from $s$
```

The correctness of the two schemes follows immediately from Lemma 4.1 and Lemma 4.2. Furthermore, we observe that, in order to turn these schemes into algorithms for checking the reachability and repeated reachability problems, it is sufficient to establish the following three effectiveness properties for the operations involved:

(1) Existence and computability of a finite attractor. This condition allows

computing the set A.

(2) Decidability of the reachability problem for the underlying transition system T. This condition allows computing *Graph*(A) and checking the various reachability conditions like "$C \xrightarrow{*} Q$" or "$s \xrightarrow{*} C$".

(3) Decidability of the constrained reachability problem for the underlying transition system. This condition is only used in the reachability algorithm.

5 Lossy Channel Systems

In this section we consider (probabilistic) lossy channel systems: processes with a finite set of local states operating on a number of unbounded and unreliable channels. We use the schemes defined in Section 4 to solve the problem of whether a set of local states is (repeatedly) reachable from a given initial state with probability one.

5.1 Basic Notions

Structure of Channel Systems. A *lossy channel system* consists of a finite-state process 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 nondeterministically lose messages. Formally, a *lossy channel system (LCS)* $\mathcal{L}$ is a tuple $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$ where $\mathtt{S}$ is a finite set of *local states*, $\mathtt{C}$ is a finite set of *channels*, $\mathtt{M}$ is a finite *message alphabet*, and $\mathtt{T}$ is a set of *transitions* each of the form $(\mathtt{s}_1, \mathtt{op}, \mathtt{s}_2)$, where $\mathtt{s}_1, \mathtt{s}_2 \in \mathtt{S}$, and $\mathtt{op}$ is an *operation* of one of the forms $\mathtt{c!m}$ (sending message $\mathtt{m}$ to channel $\mathtt{c}$), or $\mathtt{c?m}$ (receiving message $\mathtt{m}$ from channel $\mathtt{c}$). A *global state* s is of the form $(\mathtt{s}, \mathtt{w})$ where $\mathtt{s} \in \mathtt{S}$ and $\mathtt{w}$ is a mapping from $\mathtt{C}$ to $\mathtt{M}^*$ that gives the current contents of each channel. By abuse of notations, we write ε for denoting both the empty word in $\mathtt{M}^*$ and the "empty" map that associates ε with each $\mathtt{c} \in \mathtt{C}$.

For words $x, y \in \mathtt{M}^*$, we let $x \bullet y$ denote the concatenation of x and y. We use $|x|$ to denote the length of x, and $x(i)$ to denote the i^{th} element of x where $1 \leq i \leq |x|$. We write $x \preceq y$ to denote that x is a (not necessarily contiguous) substring of y. Since $\mathtt{M}$ is finite, Higman's Lemma [13] implies that $\preceq$ is a well-quasi-ordering (a *wqo*), hence for each infinite sequence $x_0, x_1, x_2, \ldots$ of words from $\mathtt{M}^*$, there are i and j with $i < j$ and $x_i \preceq x_j$. For $\mathtt{w}, \mathtt{w}' \in (\mathtt{C} \mapsto \mathtt{M}^*)$, we define $|\mathtt{w}| = \sum_{\mathtt{c} \in \mathtt{C}} |\mathtt{w}(\mathtt{c})|$ and use $\mathtt{w} \preceq \mathtt{w}'$ to denote that $\mathtt{w}(\mathtt{c}) \preceq \mathtt{w}'(\mathtt{c})$ for each $\mathtt{c} \in \mathtt{C}$: this is again a wqo. We further extend this to a wqo on $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$, by defining $(\mathtt{s}_1, \mathtt{w}_1) \preceq (\mathtt{s}_2, \mathtt{w}_2)$ iff $\mathtt{s}_1 = \mathtt{s}_2$ and $\mathtt{w}_1 \preceq \mathtt{w}_2$.

Operational Semantics. The LCS $\mathcal{L}$ induces a transition system $(S, \longrightarrow)$, where S is the set of global states, i.e., $S = (\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*))$. We start by defining normal steps (where messages are not lost): there is a step $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2)$ if one of the following conditions is satisfied

- There is a $\mathtt{t} \in \mathtt{T}$, where $\mathtt{t}$ is of the form $(\mathtt{s}_1, \mathtt{c!m}, \mathtt{s}_2)$ and $\mathtt{w}_2$ is the result of appending $\mathtt{m}$ to the end of $\mathtt{w}_1(\mathtt{c})$.
- There is a $\mathtt{t} \in \mathtt{T}$, where $\mathtt{t}$ is of the form $(\mathtt{s}_1, \mathtt{c?m}, \mathtt{s}_2)$ and $\mathtt{w}_2$ is the result of removing $\mathtt{m}$ from the head of $\mathtt{w}_1(\mathtt{c})$ (thus $\mathtt{w}_1(\mathtt{c})$ must be of the form $\mathtt{m} \bullet x$).

In any of these cases we define $\mathtt{t}(\mathtt{s}_1, \mathtt{w}_1) = (\mathtt{s}_2, \mathtt{w}_2)$ and say that $\mathtt{t}$ is *enabled* at $(\mathtt{s}_1, \mathtt{w}_1)$. We let *enabled*$(\mathtt{s}, \mathtt{w}) = \{\mathtt{t} \; : \; \mathtt{t}$ is enabled at $(\mathtt{s}, \mathtt{w})\}$. A state $(\mathtt{s}, \mathtt{w})$ is a *deadlock state* if *enabled*$(\mathtt{s}, \mathtt{w})$ is empty. An LCS is *deadlock-free* if there are no deadlock states. It is easy to check whether an LCS is deadlock-free (see Section 8.3).

The definition of the transition system $(S, \longrightarrow)$ is complete after we take into account the possibility of message losses: if $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2)$ is a normal step, then for each $\mathtt{w}_2' \preceq \mathtt{w}_2$, $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2')$ is also a step.

For the rest of this section we assume an LCS $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$ whose behavior is given by the associated transition system $(S, \longrightarrow)$ [2].

For $\mathtt{Q} \subseteq \mathtt{S}$, we define a $\mathtt{Q}$-*state* to be a state of the form $(\mathtt{s}, \mathtt{w})$ where $\mathtt{s} \in \mathtt{Q}$.

A set $Q \subseteq S$ is said to be *upward closed* if $s_1 \in Q$ and $s_1 \preceq s_2$ imply $s_2 \in Q$. Notice that, for any $\mathtt{Q} \subseteq \mathtt{S}$, the set of $\mathtt{Q}$-states is an upward closed set. The upward closure $Q \uparrow$ of a set Q is the set $\{s \; : \; \exists s' \in Q.\; s' \preceq s\}$. We use $\min(Q)$ to denote the set of minimal elements of Q (with respect to $\preceq$). This set is unique and (by Higman's lemma) finite. Furthermore, if Q is upward closed then Q is completely characterized by $\min(Q)$ in the sense that $Q = \min(Q) \uparrow$.

Lemma 5.1 [3] *For states s_1 and s_2, it is decidable whether s_2 is reachable from s_1.*

Lemma 5.2 [3] *For a state s and a set $\mathtt{Q} \subseteq \mathtt{S}$, it is decidable whether the set of $\mathtt{Q}$-states is reachable from s.*

[2] **Remark on notation:** Observe that we use $\mathtt{s}$ and $\mathtt{S}$ to range over *local* states and sets of local states respectively, while we use s and S to range over states and sets of states of the induced transition system (states of the transition system are *global* states of the LCS).

A *probabilistic lossy channel system (PLCS)* $\mathcal{L}$ is of the form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$, where $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$ is an LCS, $\lambda \in (0, 1)$, and w is a mapping from $\mathtt{T}$ to the positive natural numbers. Intuitively, we derive a Markov chain from the PLCS $\mathcal{L}$ by assigning probabilities to the transitions of the underlying transition system $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$. The probability of performing a transition $\mathtt{t}$ from a global state $(\mathtt{s}, \mathtt{w})$ is determined by the weight $w(\mathtt{t})$ of $\mathtt{t}$ compared to the weights of the other transitions which are enabled at $(\mathtt{s}, \mathtt{w})$. Furthermore, after performing each transition, each message which resides inside one of the channels may be lost with a probability λ. This means that the probability of reaching $(\mathtt{s}_2, \mathtt{w}_2)$ from $(\mathtt{s}_1, \mathtt{w}_1)$ is equal to (the sum over all $(\mathtt{s}_3, \mathtt{w}_3)$ of) the probability of reaching some $(\mathtt{s}_3, \mathtt{w}_3)$ from $(\mathtt{s}_1, \mathtt{w}_1)$ through performing a transition of the underlying LCS, multiplied by the probability of reaching $(\mathtt{s}_2, \mathtt{w}_2)$ from $(\mathtt{s}_3, \mathtt{w}_3)$ through the loss of messages.

Now, we show how to define formally these probabilities. For simplicity, and throughout the rest of this article, we assume that PLCS's are deadlock-free. We refer to Section 8.3 for indications on how to deal with PLCS's having deadlock states.

First, we compute probabilities of reaching states through the loss of messages. For $x, y \in \mathtt{M}^*$, we define $\#\,(x, y)$ to be the size of the set

$$\{(i_1, \ldots, i_n) \;:\; i_1 < \cdots < i_n \text{ and } x = y(i_1) \bullet \cdots \bullet y(i_n)\}\,.$$

In other words, $\#\,(x, y)$ is the number of the different ways in which we can delete symbols in the word y in order to obtain x. We also define

$$P_L(x, y) = \#\,(y, x) \cdot \lambda^{|x|-|y|} \cdot (1-\lambda)^{|y|}. \tag{5.1}$$

$P_L(x, y)$ is the probability that the string x becomes y by losing some of its symbols when each symbol can be lost with probability λ. One readily checks that $\sum_{y \in \mathtt{M}^*} P_L(x, y) = 1$ for all $x \in \mathtt{M}^*$, using the following two combinatorial equalities:

$$\forall k \in \mathbb{N} : \sum_{y \in \mathtt{M}^k} \#\,(y, x) = \binom{|x|}{k}, \tag{5.2}$$

$$\sum_{k=0}^{|x|} \binom{|x|}{k} \cdot \lambda^{|x|-k} \cdot (1-\lambda)^k = [\lambda + (1-\lambda)]^{|x|} = 1. \tag{5.3}$$

We extend P_L to a probability of transforming a state to another state by message losses. For $\mathtt{w}_1, \mathtt{w}_2 \in (\mathtt{C} \mapsto \mathtt{M}^*)$, we define $P_L(\mathtt{w}_1, \mathtt{w}_2) = \prod_{\mathtt{c} \in \mathtt{C}} P_L\left(\mathtt{w}_1(\mathtt{c}), \mathtt{w}_2(\mathtt{c})\right)$.

Notice that $P_L(\mathtt{w}_1, \mathtt{w}_2) = 0$ in case $\mathtt{w}_1 \not\preceq \mathtt{w}_2$. We take

$$P_L((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = \begin{cases} P_L(\mathtt{w}_1, \mathtt{w}_2) & \text{if } \mathtt{s}_1 = \mathtt{s}_2, \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

We define $w(\mathtt{s}, \mathtt{w}) = \sum_{\mathtt{t} \in \mathit{enabled}(\mathtt{s},\mathtt{w})} w(\mathtt{t})$.

The PLCS $\mathcal{L}$ induces a Markov chain (S, P), where

$$S = \mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*), \tag{5.5}$$

$$P((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = \sum_{\mathtt{t} \in \mathtt{T}} \left(\frac{w(\mathtt{t})}{w(\mathtt{s}_1, \mathtt{w}_1)} P_L\Big(\mathtt{t}(\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)\Big) \right). \tag{5.6}$$

The restriction to deadlock-free PLCS's ensures that no division by zero occurs in Equation (5.6). Observe that, for all $(\mathtt{s}_1, \mathtt{w}_1) \in S$, Equation (5.6) ensures $\sum_{(\mathtt{s}_2,\mathtt{w}_2) \in S} P((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = 1$, so that (S, P) is indeed a Markov chain.

We now instantiate the probabilistic reachability problems considered in Section 3 and Section 4 to PLCS's.

Below, we assume a PLCS $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$ inducing a Markov chain $M = (S, P)$ with an underlying transition system $T = (S, \longrightarrow)$: observe that $(S, \longrightarrow)$ is the same transition system we associated with the (non-probabilistic) LCS given by $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$.

We shall consider the probabilistic (repeated) reachability problem for PLCS's. We check whether an upward closed set, represented by its minimal elements, is (repeatedly) reached from a given initial state with probability one. We show that the (repeated) reachability problem instantiated in this manner fulfills the three conditions required for effective implementation of the probabilistic (repeated) reachability schemes of Section 4.

5.3 Finite Attractors in Probabilistic Lossy Channel Systems

The following crucial Lemma shows that there always exists a finite attractor in the Markov chain associated with a PLCS.

Lemma 5.3 *For each PLCS* $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$ *with* $\lambda > 0$*, the set* $Q_0 = \{(\mathtt{s}, \varepsilon) \ : \ \mathtt{s} \in \mathtt{S}\}$ *is an attractor.*

The intuition behind this result is simple: in a state $(\mathtt{s}, \mathtt{w})$ with $|\mathtt{w}|$ large enough, the system is more likely to move "down" to a next state with less messages (because of losses), than "up" to a next state with one more message (this requires a send operation and no losses). Thus the system is attracted "down" to small states. Now, whatever finite set A of small states turns out to be an attractor, Q_0 is reachable from any state in A (by message losses) and is thus an attractor (Lemma 3.4).

In the rest of this section, we turn this intuition into a rigorous proof, using only elementary notions. This requires tedious work where one builds adequate upper- and lower-bounds for the probabilities of going "up" or "down". (A possible alternative approach would be to use standard arguments of martingale theory [14].)

Assume $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$ is fixed. For any $n \in \mathbb{N}$, write Q_n for $\{(\mathtt{s}, \mathtt{w}) \ : \ |\mathtt{w}| = n\}$, the set of global states in which the channels currently contain a total of n messages. We want to prove that Q_0 is an attractor.

For any global state $(\mathtt{s}, \mathtt{w})$, let $\mathbb{P}_{\mathtt{s},\mathtt{w}}$ denote $\mathbb{P}_{(\mathtt{s},\mathtt{w})}(\Diamond Q_0)$. We have:

$$\mathbb{P}_{\mathtt{s},\mathtt{w}} = \begin{cases} \sum\limits_{(\mathtt{s}',\mathtt{w}')} P((\mathtt{s},\mathtt{w}),(\mathtt{s}',\mathtt{w}')) \times \mathbb{P}_{\mathtt{s}',\mathtt{w}'} & \text{if } \mathtt{w} \neq \varepsilon, \\ 1 & \text{otherwise.} \end{cases} \tag{5.7}$$

Write $\mathbb{P}_n$ for $\min\{\mathbb{P}_{\mathtt{s},\mathtt{w}} \ : \ (\mathtt{s},\mathtt{w}) \in Q_n\}$ and $\mathbb{Q}_n$ for $\min\{\mathbb{P}_i \ : \ 0 \leq i \leq n\}$: $\mathbb{P}_0 = \mathbb{Q}_0 = 1$, and the sequence $(\mathbb{Q}_n)_{n\in\mathbb{N}}$ is positive and non-increasing.

For any $n > 0$ and $(\mathtt{s},\mathtt{w}) \in Q_n$, we can split the sum in Equation (5.7) by distinguishing whether $(\mathtt{s}',\mathtt{w}')$ is in Q_{n+1}, in Q_n or in $Q_{\leq n-1}$ (shorthand for $\bigcup_{i<n} Q_i$). For this, we introduce the following terms:

$$\begin{aligned} a_{\mathtt{s},\mathtt{w}} &= \sum_{(\mathtt{s}',\mathtt{w}')\in Q_{\leq n-1}} P((\mathtt{s},\mathtt{w}),(\mathtt{s}',\mathtt{w}')), \\ b_{\mathtt{s},\mathtt{w}} &= \sum_{(\mathtt{s}',\mathtt{w}')\in Q_n} P((\mathtt{s},\mathtt{w}),(\mathtt{s}',\mathtt{w}')), \\ c_{\mathtt{s},\mathtt{w}} &= \sum_{(\mathtt{s}',\mathtt{w}')\in Q_{n+1}} P((\mathtt{s},\mathtt{w}),(\mathtt{s}',\mathtt{w}')). \end{aligned}$$

Observe that $a_{\mathtt{s},\mathtt{w}} + b_{\mathtt{s},\mathtt{w}} + c_{\mathtt{s},\mathtt{w}} = 1$.

Using $\mathbb{Q}_{|\mathtt{w}'|} \leq \mathbb{P}_{\mathtt{s}',\mathtt{w}'}$, and observing that $P((\mathtt{s},\mathtt{w}),(\mathtt{s}',\mathtt{w}')) = 0$ when $|\mathtt{w}'| > n+1$, Equation (5.7) entails

$$\mathbb{P}_{\mathtt{s},\mathtt{w}} \geq a_{\mathtt{s},\mathtt{w}}\mathbb{Q}_{n-1} + b_{\mathtt{s},\mathtt{w}}\mathbb{Q}_n + c_{\mathtt{s},\mathtt{w}}\mathbb{Q}_{n+1}. \tag{5.8}$$

Pick one of the $(\mathtt{s},\mathtt{w})$'s in Q_n that make $\mathbb{P}_{\mathtt{s},\mathtt{w}}$ minimal and write a_n, b_n and c_n for $a_{\mathtt{s},\mathtt{w}}$, $b_{\mathtt{s},\mathtt{w}}$, and $c_{\mathtt{s},\mathtt{w}}$ respectively. From Equation (5.8) we derive

$$\mathbb{P}_n \geq a_n\mathbb{Q}_{n-1} + b_n\mathbb{Q}_n + c_n\mathbb{Q}_{n+1}. \tag{5.9}$$

Since $a_n + b_n + c_n = 1$ and $(\mathbb{Q}_i)_{i\in\mathbb{N}}$ is non-increasing

$$\mathbb{Q}_{n-1} \geq a_n\mathbb{Q}_{n-1} + b_n\mathbb{Q}_n + c_n\mathbb{Q}_{n+1} \tag{5.10}$$

holds obviously. Now, by definition, $\mathbb{Q}_n = \min(\mathbb{P}_n, \mathbb{Q}_{n-1})$. Thus, combining Equations (5.9) and (5.10), we deduce

$$\mathbb{Q}_n \geq a_n\mathbb{Q}_{n-1} + b_n\mathbb{Q}_n + c_n\mathbb{Q}_{n+1}, \tag{5.11}$$

and then (again using $a_n + b_n + c_n = 1$)

$$c_n(\mathbb{Q}_n - \mathbb{Q}_{n+1}) \geq a_n(\mathbb{Q}_{n-1} - \mathbb{Q}_n). \tag{5.12}$$

Write now δ_n for $\mathbb{Q}_n - \mathbb{Q}_{n+1}$: since $(\mathbb{Q}_i)_{i\in\mathbb{N}}$ is non-increasing and stays positive, $(\delta_n)_{n\in\mathbb{N}}$ is positive with $\lim_{n\to\infty}\delta_n = 0$. Equation (5.12) rewrites as $\delta_{n-1} \leq \frac{c_n}{a_n}\delta_n$, entailing, for any n and k

$$\delta_n \leq \frac{c_{n+1}}{a_{n+1}}\frac{c_{n+2}}{a_{n+2}}\cdots\frac{c_{n+k}}{a_{n+k}}\delta_{n+k}. \tag{5.13}$$

We now use the intuition that message losses make the system attracted to small states. Assume $(\mathtt{s},\mathtt{w}) \in Q_n$. Then, using Equations (5.1) and (5.6), one sees that $c_{\mathtt{s},\mathtt{w}} \leq (1-\lambda)^{n+1}$ (equality holds when all operations available in state $(\mathtt{s},\mathtt{w})$ are send operations). Similarly, $b_{\mathtt{s},\mathtt{w}} \leq n(1-\lambda)^n\lambda$. Thus $\lim_{n\to\infty} c_n = \lim_{n\to\infty} b_n = 0$, entailing $\lim_{n\to\infty} a_n = 1$ and

$$\lim_{k\to\infty}\prod_{i=1}^{k}\frac{c_{n+i}}{a_{n+i}} \;=\; 0 \tag{5.14}$$

for all $n \in \mathbb{N}$.

Combining Equations (5.13) and (5.14) shows that $\delta_n = 0$. This holds for all n so that $\mathbb{Q}_n = 1$ for all n, and hence $\mathbb{P}_{\mathtt{s},\mathtt{w}} = 1$ for all $(\mathtt{s},\mathtt{w})$. This exactly means that Q_0 is an attractor.

5.4 Verification of Probabilistic Lossy Channel Systems

From Lemma 5.1, and the fact that the transition system underlying a PLCS $(\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$ is independent of λ we obtain:

Lemma 5.4 *One can build effectively Graph(A) when given a finite set of states A.*

Furthermore, for two PLCS's $\mathcal{L} = (\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$ and $\mathcal{L}' = (\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda',w')$ which differ only by probabilities (we assume that, for all $\mathtt{t} \in \mathtt{T}$, $w(\mathtt{t}) > 0$ iff

$w'(\mathtt{t}) > 0$), A has the same graph in both PLCS's. Now we are ready to solve Probabilistic Reachability and Probabilistic Repeated Reachability problems for PLCS's.

Probabilistic Reachability for PLCS's

Instance: A PLCS $M = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$, a state s, and a set $\mathtt{Q} \subseteq \mathtt{S}$.

Question: Is the set of $\mathtt{Q}$-states reachable from s with probability one?

Probabilistic Repeated Reachability for PLCS's

Instance: A PLCS $M = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$, a state s, and a set $\mathtt{Q} \subseteq \mathtt{S}$.

Question: Is the set of $\mathtt{Q}$-states repeatedly reachable from s with probability one?

Theorem 5.5 *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS's.*

PROOF. Lemmas 5.2, 5.3, and 5.4 provide the effective procedures required to implement the scheme (from Section 4) for probabilistic repeated reachability in Markov chains. For probabilistic reachability, a possible proof is by extending Lemma 5.2 and showing decidability of constrained reachability in LCS's.

However, another proof is possible. For PLCS's, Probabilistic Reachability easily reduces to Probabilistic Repeated Reachability. The probability that the set of $\mathtt{Q}$-states will be reached in some PLCS $\mathcal{L}$ is exactly the probability that this set will be repeatedly reached in the variant PLCS $\mathcal{L}'$ one obtains by removing in $\mathcal{L}$ all transitions of the form $(\mathtt{s}_1, \mathtt{op}, \mathtt{s}_2)$ having $\mathtt{s}_1 \in \mathtt{Q}$, and replacing them by looping transitions $(\mathtt{s}_1, \mathtt{c}!\mathtt{m}, \mathtt{s}_1)$ for some arbitrary $\mathtt{c}$ and $\mathtt{m}$. □

Remark 5.6 *In our definition of LCS's and PLCS's, we assume that messages are lost only* after *performing non-lossy transitions. This choice simplifies the definition of the Markov chain associated with a PLCS. However, our analysis can be modified in a straightforward manner to deal with the case where losses occur before, and the case where losses occur both before and after non-lossy transitions.*

6 Duplication, Corruption, and Insertion

We consider PLCS's with different sources of unreliability such as duplication, corruption, and insertion combined with lossiness.

We analyze a variant of PLCS's, where we add another source of unreliability; namely a message inside a channel may be duplicated [9].

An LCS $\mathcal{L}$ with *duplication errors* is of the same form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$ as an LCS. We define the behavior of $\mathcal{L}$ as follows. For $a \in \mathtt{M}$, we use a^n to denote the concatenation of n copies of a. For $x = a_1 a_2 \cdots a_n$ with $x \in \mathtt{M}^*$, we define *Duplicate*(x) to be the set

$$\{b_1 b_2 \cdots b_n \ : \ \text{either } b_i = a_i \text{ or } b_i = a_i^2 \text{ for each } i : 1 \leq i \leq n\}.$$

In other words, we get each member of *Duplicate*(x) by duplicating some of the elements of x. We extend the definition of *Duplicate* to $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$ in a similar manner to Section 5. The transition relation of an LCS $\mathcal{L}$ with duplication errors extends that of the corresponding standard LCS in the sense that:

- If $(\mathtt{s}_1, \mathtt{w}_1) \rightarrow (\mathtt{s}_2, \mathtt{w}_2)$ according to the definition of Section 5 then $(\mathtt{s}_1, \mathtt{w}_1) \rightarrow (\mathtt{s}_2', \mathtt{w}_2')$ for each $(\mathtt{s}_2', \mathtt{w}_2') \in$ *Duplicate*$(\mathtt{s}_2, \mathtt{w}_2)$.

In [9], it is shown that the reachability problem is decidable for LCS's with duplication errors. Hence we have

Lemma 6.1 *Given an LCS with duplication errors.*

(1) For states s_1 and s_2, it is decidable whether s_2 is reachable from s_1 [9]. Hence, Graph(A) is computable for any finite set A of states.
(2) For a state s and a set $\mathtt{Q} \subseteq \mathtt{S}$, it is decidable whether the set of $\mathtt{Q}$-states is reachable from s [9].

A PLCS with *duplication errors* is of the form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_D)$, where $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$ is a PLCS, and $\lambda_D \in [0, 1]$. The value of λ_D represents the probability by which any given message is duplicated inside the channels.

To obtain the Markov chain induced by a PLCS with duplication errors, we compute probabilities of reaching states through duplication of messages. For $x, y \in \mathtt{M}^*$, where $x = a_1 a_2 \cdots a_n$, we define $\#_D(x, y)$ to be the size of the set $\left\{(i_1, \ldots, i_n) \ : \ 1 \leq i_j \leq 2 \text{ and } y = a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n}\right\}$. In other words, $\#_D(x, y)$ is the number of different ways in which we can duplicate symbols in the word x in order to obtain y. In a similar manner to the case of losing messages (Section 5), we define

$$P_D(x, y) = \#_D(x, y) \cdot \lambda_D^{|y| - |x|} \cdot (1 - \lambda_D)^{|x|}, \tag{6.1}$$

and $P_D(\mathtt{w}_1, \mathtt{w}_2) = \prod_{\mathtt{c} \in \mathtt{C}} P_D(\mathtt{w}_1(\mathtt{c}), \mathtt{w}_2(\mathtt{c}))$. The PLCS with duplication errors $\mathcal{L}$

induces a Markov chain (S, P'_D) with $S = (\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*))$ as before, and

$$P'_D\left((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)\right) = \sum_{\mathtt{w}_3 \in (\mathtt{C} \mapsto \mathtt{M}^*)} P\left((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_3)\right) \cdot P_D\left(\mathtt{w}_3, \mathtt{w}_2\right), \tag{6.2}$$

where P has the same definition as in Section 5.

Remark 6.2 *The above choice of a definition for P'_D is partly arbitrary. For example, it considers that duplications occur randomly* after *normal transitions and losses, and that a message is duplicated* at most once *during a single step. Similar remarks apply to our definitions (in the following subsections) for systems with corruptions, insertions, and other unreliability sources.*

All these choices aim at simplicity, and variant definitions are possible. We let the reader convince herself that these variants would lead to decidability results that are essentially identical to the ones we present for our definitions.

Lemma 6.3 *For each PLCS $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_D)$ with $\lambda \geq \lambda_D > 0$, the set $Q_0 = \{(\mathtt{s}, \varepsilon) \ : \ \mathtt{s} \in \mathtt{S}\}$ is an attractor.*

PROOF. (Sketch) Let $s = (\mathtt{s}, \mathtt{w})$ be a state with n messages and consider what happens to each individual message in the corruption phase (i.e., losses and duplications). With probability λ, the message is lost. With probability $(1-\lambda)\lambda_D$ it is duplicated, and with probability $(1-\lambda)(1-\lambda_D)$ it is kept unmodified. Observe that all messages are lost, or duplicated, or kept unmodified, independently of the other messages.

For k between $-n$ and n, write δ_n^k for the probability that the corruption phase ends up with $n+k$ messages. The assumption $\lambda \geq \lambda_D$ entails $\delta_n^{-k} \geq \delta_n^k$ for any positive k. In other words, the number of messages in the channels is more likely to decrease by k than to increase by k through corruption.

If we now take into account the fact that a normal step can at most write one message, the expected number of messages after a step from s is

$$\sum_{s' \in S} P'_D(s, s') \times |s'| \ \leq \ n + 1 + \sum_{k=-n-1}^{n+1} k \delta_{n+1}^k.$$

Using $\lambda > 0$, one can show this expected number is $< n$ for n large enough. Thus, when n is large enough, the system is attracted to small states.

These considerations can be turned into a rigorous proof similar to (but necessarily more tedious than) the proof of Lemma 5.3. Here too a shorter albeit less elementary proof can be obtained with martingale theory. □

As in Section 5, we derive from Lemma 6.1 and Lemma 6.3:

Theorem 6.4 *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS's with duplication errors when* $\lambda \geq \lambda_D > 0$.

6.2 Corruption

We consider LCS's with *corruption errors*, i.e., where a message inside a channel may be changed to any other message. For simplicity, we assume $|\mathtt{M}| > 1$. We extend the semantics of LCS's to include corruption errors in the same manner as we did above for duplication errors. For $x \in \mathtt{M}^*$, we define $Corrupt(x)$ to be the set $\{y \in \mathtt{M}^* \ : \ |y| = |x|\}$, i.e., we get a member of $Corrupt(x)$ by changing any number of symbols in x to another symbol in $\mathtt{M}$. We extend the definition to $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$ in the same manner as before. Furthermore, we enlarge the transition relation of an LCS:

- If $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2)$ according to the definition of Section 5 then $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2', \mathtt{w}_2')$ for each $(\mathtt{s}_2', \mathtt{w}_2') \in Corrupt(\mathtt{s}_2, \mathtt{w}_2)$.

Decidability of the reachability problem for LCS's with corruption errors follows from the fact $(\mathtt{s}_1, \mathtt{w}_1) \xrightarrow{+} (\mathtt{s}_2, \mathtt{w}_2)$ implies $(\mathtt{s}_1, \mathtt{w}_1) \xrightarrow{+} (\mathtt{s}_2, \mathtt{w}_3)$ for each $\mathtt{w}_3$ with $|\mathtt{w}_3(\mathtt{c})| = |\mathtt{w}_2(\mathtt{c})|$ for all $\mathtt{c} \in \mathtt{C}$. This implies that the only relevant information to consider about the channels in the reachability algorithm is the length of their contents. In other words, the problem is reduced to a special case of LCS's where the set $\mathtt{M}$ can be considered to be a singleton. The constrained reachability problem can be solved in a similar manner. Hence,

Lemma 6.5 *Given an LCS with corruption errors.*

(1) For states s_1 *and* s_2*, it is decidable whether* s_2 *is reachable from* s_1*. Hence,* $Graph(A)$ *is computable for any finite set* A *of states.*
(2) For a state s *and a set* $\mathtt{Q} \subseteq \mathtt{S}$*, it is decidable whether the set of* $\mathtt{Q}$*-states is reachable from* s*.*

A PLCS with *corruption errors* is of the form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_C)$, where $\lambda_C \in [0,1]$ represents the probability by which any given message is corrupted to some other message. For $x, y \in \mathtt{M}^*$, we define $\#_C\,(x,y)$ to be the size of the set $\{i \ : \ x(i) \neq y(i)\}$. In other words, $\#_C\,(x,y)$ is the number of elements which must change in order to obtain y from x. We define

$$P_C(x,y) = \begin{cases} \left(\frac{\lambda_C}{|\mathtt{M}|-1}\right)^{\#_C(x,y)} (1-\lambda_C)^{|x|-\#_C(x,y)} & \text{if } |x| = |y|, \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

Thus $P_C(x, y)$ is the probability that x will become y when its $|x|$ letters are independently corrupted with probability λ_C. Observe that for any $x \in \mathtt{M}^*$ we have $\sum_{y \in \mathtt{M}^*} P_C(x, y) = 1$: this is seen by noting that, for $k \leq |x|$, there are exactly $(|\mathtt{M}| - 1)^k \binom{|x|}{k}$ words y for which $\#_C(x, y) = k$.

We extend P_C from $\mathtt{M}^*$ to $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$ as before. In a manner similar to the previous case with duplication, the PLCS with corruption errors $\mathcal{L}$ induces a Markov chain (S, P'_C) with

$$P'_C\left((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)\right) = \sum_{\mathtt{w}_3 \in (\mathtt{C} \mapsto \mathtt{M}^*)} P\left((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_3)\right) \cdot P_C(\mathtt{w}_3, \mathtt{w}_2). \qquad (6.4)$$

Lemma 6.6 *For each PLCS* $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_C)$ *with* $\lambda > 0$, *the set* $Q_0 = \{(\mathtt{s}, \varepsilon) \;:\; \mathtt{s} \in \mathtt{S}\}$ *is an attractor.*

From Lemma 6.5 and Lemma 6.6 we can derive in a similar manner to Section 5.

Theorem 6.7 *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS's with corruption errors.*

6.3 Insertion

We consider LCS's with *insertion errors*, i.e., where arbitrary messages can be inserted spuriously inside a channel [9]. As before, we extend the semantics of LCS's to include insertion errors: for $x \in \mathtt{M}^*$, we define *Insert*(x) to be the set $\{y \in \mathtt{M}^* \;:\; x \preceq y\}$. We extend this in the usual way to obtain a definition of *Insert*(s) where s is a state in $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$. Then we enlarge the transition relation on an LCS:

- If $(\mathtt{s}_1, \mathtt{w}_1) \to (\mathtt{s}_2, \mathtt{w}_2)$ according to the definition of Section 5 then $(\mathtt{s}_1, \mathtt{w}_1) \to (\mathtt{s}'_2, \mathtt{w}'_2)$ for each $(\mathtt{s}'_2, \mathtt{w}'_2) \in$ *Insert*$(\mathtt{s}_2, \mathtt{w}_2)$.

Decidability of the reachability problem for LCS's with insertion errors is easy [9] since, for one-step moves, it holds that

$$(\mathtt{s}_1, \mathtt{w}_1) \to (\mathtt{s}_2, \mathtt{w}_2) \text{ iff } (\mathtt{s}_1, \mathtt{w}_1) \to (\mathtt{s}_2, \varepsilon).$$

A PLCS with *insertion errors* is of the form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_I)$ where $\lambda_I \in [0, 1)$ commands the probability that some message is inserted. We assume a geometric distribution, where there is a probability $\lambda_I^k(1 - \lambda_I)$ that k messages will be inserted during one step, but other choices are possible.

The definition of $P_I(x, y)$, the probability that x is transformed into y by insertion errors, considers several cases. We let $P_I(x, x) = (1 - \lambda_I)$ and, when $|y| > |x|$, we define $P_I(x, y)$ by induction on the number $|y| - |x|$ of inserted messages:

$$P_I(x, y) = \lambda_I \sum_{z \in \mathtt{M}^{|x|+1}} \frac{\#(x, z)}{|\mathtt{M}| \cdot (1 + |x|)} P_I(z, y). \tag{6.5}$$

In all other cases, we let $P_I(x, y) = 0$.

Using the following combinatorial equality:

$$\sum_{z \in \mathtt{M}^{|x|+1}} \#(x, z) = |\mathtt{M}| \cdot (1 + |x|), \tag{6.6}$$

and induction on k, one easily shows that

$$\sum_{y \in \mathtt{M}^{|x|+k}} P_I(x, y) = \lambda_I^k (1 - \lambda_I) \tag{6.7}$$

as intended. Note that, as a consequence, $\sum_{y \in \mathtt{M}^*} P_I(x, y) = 1$ for all $x \in \mathtt{M}^*$.

We extend P_I from $\mathtt{M}^*$ to $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$ as before. In a manner similar to the previous cases, the PLCS with insertion errors $\mathcal{L}$ induces a Markov chain (S, P'_I) with

$$P'_I((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = \sum_{\mathtt{w}_3 \in (\mathtt{C} \mapsto \mathtt{M}^*)} P((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_3)) \cdot P_I(\mathtt{w}_3, \mathtt{w}_2). \tag{6.8}$$

Since the probability that k messages will be inserted in one step does not depend on the size of the current state (Eq. (6.7)), the system is attracted to small states.

Lemma 6.8 *For each PLCS* $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_I)$ *with* $\lambda > 0$*, the set* $Q_0 = \{(\mathtt{s}, \varepsilon) \ : \ \mathtt{s} \in \mathtt{S}\}$ *is an attractor.*

Since the necessary reachability properties are decidable for PLCS's with insertion errors, Lemma 6.8 allows us to proceed as in Section 5.

Theorem 6.9 *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS's with insertion errors.*

6.4 Other Unreliability Sources

The approach we just developed for duplication, corruption and insertion errors can be adapted to deal with variant, or restricted, versions of these three main kinds of errors. Furthermore, we can combine different sources of unreliability. For instance, we can consider models where we have both duplication and corruption together with lossiness. In all these cases, our methods will carry over when reachability remains decidable and when a finite attractor exists. In general, this requires that unreliability sources which may increase the number of messages inside the channels (such as insertion and duplication but not corruption) have sufficiently low probabilities (compared to lossiness).

7 Automata-Definable Properties

In this section we consider more general properties than reachability and repeated reachability for PLCS's. Let φ be a property of computations (also called a *linear-time property*). We are interested in whether $\mathbb{P}_s(\varphi) = 1$ for s a state of a PLCS, i.e., whether a run starting from s almost surely satisfies φ. We show that if the properties of computations are specified by (the ω-behavior of) finite-state automata, or equivalently by formulas of the monadic logic of order, called "*MSO formulas*", then the above problem is decidable. Similar results hold for the other families of faulty probabilistic systems we considered in Section 6. Since the proofs for these systems follow the same pattern as for PLCS's, we will confine ourselves to PLCS's here.

7.1 State-Labeled Systems and ω-Automata

In order to check a property defined by a deterministic finite-state automaton $\mathcal{A}$, we shall build a product of $\mathcal{A}$ with the given PLCS. This approach assumes that we extend LCS's with a labeling function: a *state-labeled LCS* is an LCS together with a finite alphabet Σ and a labeling function *lab* from the local states to Σ. Throughout this section we assume that LCS's are state-labeled and will often use "LCS" for "state-labeled LCS". We lift the labeling from an LCS $\mathcal{L}$ to the *state-labeled transition system* $T = (S, \to, \Sigma, \mathit{lab})$ it induces: the label of every state $(\mathtt{s}, \mathtt{w})$ in T is the label $\mathit{lab}(\mathtt{s})$ of its local state component. When we deal with probabilistic lossy channel systems we also assume that the underlying LCS is labeled, and this labeling is lifted to the labeling of the corresponding Markov chain. In this manner we obtain *state-labeled PLCS's* inducing *state-labeled Markov chains.*

A path $s_0, s_1, \ldots$ in a state-labeled transition system gives rise to its *trace*, the ω-string $\mathit{lab}(s_0).\mathit{lab}(s_1)\ldots$ over the alphabet Σ. We consider properties of paths that are defined using automata: the trace of the path must be accepted by the automaton. Recall that a *finite (Muller) automaton* $\mathcal{A}$ is a tuple $(\mathcal{Q}, \Sigma, \rightarrow, q_0, \mathcal{F})$, consisting of a finite set $\mathcal{Q}$ of *states*, a finite *alphabet* Σ, a *transition relation* $\rightarrow$ which is a subset of $\mathcal{Q} \times \Sigma \times \mathcal{Q}$, an *initial state* $q_0 \in \mathcal{Q}$, and a collection $\mathcal{F} \subseteq 2^{\mathcal{Q}}$ of *fairness conditions.* We write $q \xrightarrow{a} q'$ if $\langle q,\ a,\ q' \rangle \in \rightarrow$. We say that $\mathcal{A}$ is *deterministic* if for every state $q \in \mathcal{Q}$ and every letter $a \in \Sigma$ there is one and only one $q' \in \mathcal{Q}$ such that $q \xrightarrow{a} q'$.

A *run* of $\mathcal{A}$ is an ω-sequence $q_0 a_0 q_1 a_1 \ldots$ such that $q_i \xrightarrow{a_i} q_{i+1}$ for all i. With such a run we associate the set `Inf` of all $q \in \mathcal{Q}$ that appear infinitely many times. A run meets the fairness conditions $\mathcal{F}$ if its `Inf` set belongs to $\mathcal{F}$ (Muller acceptance). An ω-string $a_0 a_1 \ldots$ over Σ is accepted by $\mathcal{A}$ if there is a run $q_0 a_0 q_1 a_1 \ldots$ that meets the fairness conditions of $\mathcal{A}$. The ω-language *accepted* by $\mathcal{A}$ is the set of all ω-strings accepted by $\mathcal{A}$.

We recall the following classical theorem (see [20]) stating that automata have the same expressive power as the monadic logic of order:

Theorem 7.1 *For an ω-language L, the following conditions are equivalent:*
1. L is accepted by a finite-state automaton,
2. L is accepted by a deterministic Muller automaton,
3. L is definable by a MSO formula.

7.2 Products With Automata

Consider an automaton $\mathcal{A} = (\mathcal{Q}, \Sigma, \rightarrow, q_0, \mathcal{F})$, and a state-labeled transition system $T = (S, \rightarrow, \Sigma, lab)$. The *product* $\mathcal{A} \times T$ of $\mathcal{A}$ and T is a state-labeled transition system $T' = (S', \rightarrow', \Sigma, lab')$ defined as follows:

States: $S' = \mathcal{Q} \times S$ is the Cartesian product of the states of $\mathcal{A}$ and of T.
Labeling: A state (q, s) is labeled by $lab(s)$, i.e., it has the same label as s in T.
Transition relation: There is a transition $(q, s) \rightarrow' (q', s')$ iff there are a transition $s \rightarrow s'$ in T and a transition $q \xrightarrow{lab(s)} q'$ in $\mathcal{A}$.

We also define the product $R = \mathcal{A} \times M$ of a *deterministic* automaton and a state-labeled Markov chain $M = (S, P, \Sigma, lab)$. Here the states and labels are as in $\mathcal{A} \times T$. The probability P' in R is given by

$$P'((q, s), (q', s')) = \begin{cases} P(s, s') & \text{if } q \xrightarrow{lab(s)} q' \text{ in } \mathcal{A}, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that the requirement that $\mathcal{A}$ is deterministic ensures that the sum of probabilities of the transitions from the state (q, s) in R is the same as the sum of probabilities of the transitions from the state s in M, i.e., the sum is one. Hence the product is indeed a labeled Markov chain. Observe further that if T is the transition system underlying M, then $\mathcal{A} \times T$ is exactly the transition system underlying $\mathcal{A} \times M$.

Finally, the product $\mathcal{L}' = \mathcal{A} \times \mathcal{L}$ of an automaton with an LCS is defined along the same lines: the local states are pairs $(q, \mathtt{s})$ of a state of $\mathcal{A}$ and a local state of $\mathcal{L}$. The transitions $\mathtt{T}'$ of $\mathcal{L}'$ are all $((q, \mathtt{s}), \mathtt{op}, (q', \mathtt{s}'))$ such that $(\mathtt{s}, \mathtt{op}, \mathtt{s}')$ is a transition of $\mathcal{L}$ and $q \xrightarrow{lab(\mathtt{s})} q'$ is a transition in $\mathcal{A}$. We define the product of a *deterministic* $\mathcal{A}$ and a PLCS $\mathcal{L}$ along the same lines.

A crucial property of these constructions is the following:

Lemma 7.2
1. If T is the transition system induced by an LCS $\mathcal{L}$ then $\mathcal{A} \times T$ is (isomorphic to) the transition system induced by the LCS $\mathcal{A} \times \mathcal{L}$.
2. If M is the Markov chain induced by a PLCS $\mathcal{L}$ then $\mathcal{A} \times M$ is (isomorphic to) the Markov chain induced by the PLCS $\mathcal{A} \times \mathcal{L}$.

Here the isomorphism associates $(q, (\mathtt{s}, \mathtt{w}))$, a state of $\mathcal{A} \times T$ (resp. $\mathcal{A} \times M$), with $((q, \mathtt{s}), \mathtt{w})$, a state of the transition system (resp. Markov chain) induced by $\mathcal{A} \times \mathcal{L}$.

We extend the notion of $\mathtt{Inf}$ sets to computations $(q_0, s_0)(q_1, s_1)\ldots$ in some $\mathcal{A} \times T$ or some $\mathcal{A} \times M$: it is the set of states (from $\mathcal{Q}$) that appear infinitely many times in the sequence $q_0 q_1 \ldots$.

Lemma 7.3 *Let $\mathcal{A}$ be a deterministic automaton with a set $\mathcal{F}$ of fairness conditions, let M be a labeled Markov chain, let R be the product $\mathcal{A} \times M$. Then the probability that a computation of M starting from s is accepted by $\mathcal{A}$ is the probability that a computation of R starting from (q_0, s) has $\mathtt{Inf} \in \mathcal{F}$.*

PROOF. (Idea) With a path $\pi = s_0 s_1 \ldots$ in M we associate $\pi^{\mathcal{A}}$, the only path in R of the form $(q_0, s_0)(q_1, s_1)\ldots$ with $q_i \xrightarrow{lab(s_i)} q_{i+1}$ for all i. For any π, $\pi^{\mathcal{A}}$ exists and is unique because $\mathcal{A}$ is deterministic. Furthermore, any path in R is $\pi^{\mathcal{A}}$ for some path π in M. The measure of a set L of paths in M is exactly the measure in R of $L^{\mathcal{A}}$, the set $\left\{\pi^{\mathcal{A}} \ : \ \pi \in L\right\}$. It remains to observe that π is accepted by $\mathcal{A}$ iff $\pi^{\mathcal{A}}$ has $\mathtt{Inf} \in \mathcal{F}$. See [10, § 4.1] for more details. $\square$

7.3 Probabilistic Model Checking

We can now verify whether a probabilistic LCS satisfies an automata-definable (or MSO) property almost surely. We consider the following problem:

Probabilistic Model Checking for PLCS's

Instance: A state-labeled PLCS $\mathcal{L}$ which defines a state-labeled Markov chain M, a state s in M, and an automaton $\mathcal{A}$.

Question: Are the computations of M starting from s accepted by $\mathcal{A}$ with probability one?

In the rest of this section we prove the following:

Theorem 7.4 *Probabilistic Model Checking for PLCS's is decidable.*

Assume $\mathcal{A} = (\mathcal{Q}, \Sigma, \rightarrow, q_0, \mathcal{F})$ is deterministic (or replace it with an equivalent deterministic automaton, using Theorem 7.1). Let R be the product of $\mathcal{A}$ with the labeled Markov chain M induced by $\mathcal{L}$. It is enough to check whether $\mathbb{P}_{(q_0,s)}(\mathtt{Inf} \in \mathcal{F}) = 1$ in R (Lemma 7.3).

Lemma 7.5 *Assume B is a finite attractor of R. Then the following are equivalent:*

(1) $\mathbb{P}_{(q_0,s)}(\mathtt{Inf} \in \mathcal{F}) = 1$.
(2) For each BSCC C in Graph(B), if C is reachable from (q_0, s) then there is F in $\mathcal{F}$ such that:
(a) if (q, u) is reachable from C in R then $q \in F$ and
(b) for each $q \in F$ there is $u \in M$ such that (q, u) is reachable from C in R.

PROOF. *(1) ⇒ (2)*: Assume that C is a BSSC in *Graph*(B) reachable from (q_0, s): then $\mathbb{P}(\Diamond C) > 0$, and $\mathbb{P}(\Box\Diamond C) > 0$ by Lemma 3.5. If $\mathbb{P}(\mathtt{Inf} \in \mathcal{F}) = 1$ then $\mathbb{P}(\Box\Diamond C \wedge \mathtt{Inf} \in \mathcal{F}) > 0$, so that there must exist some $F \in \mathcal{F}$ with

$$\mathbb{P}(\Box\Diamond C \ \wedge \ \mathtt{Inf} = F) > 0. \tag{7.1}$$

This requires that, for every $q \in F$, some (q, u) is reachable from C. Furthermore, if some (q, u) is reachable from C then, by Lemma 3.1, $\mathbb{P}(\Box\Diamond C \ \Rightarrow \ \Box\Diamond\,(q, u)) = 1$. Therefore (7.1) entails $q \in F$.

(2) ⇒ (1): Assume that for a BSCC C there is some $F \in \mathcal{F}$ satisfying *(2a)* and *(2b)*. Then *(2a)* entails $\mathbb{P}(\Box\Diamond C) = \mathbb{P}(\Box\Diamond C \ \wedge \ \mathtt{Inf} \subseteq F)$. On the other hand, for each $q \in F$, *(2b)* and Lemma 3.1 entail that $\mathbb{P}(\Box\Diamond C) = \mathbb{P}(\Box\Diamond C \ \wedge \ \Box\Diamond\,(q, u)) = \mathbb{P}(\Box\Diamond C \wedge q \in \mathtt{Inf})$. Thus $\mathbb{P}(\Box\Diamond C) = \mathbb{P}(\Box\Diamond C \wedge \mathtt{Inf} \in \mathcal{F})$. Since this holds for every BSCC reachable from (q_0, s), we obtain $\mathbb{P}(\mathtt{Inf} \in \mathcal{F}) = 1$ by Lemma 3.6. □

Now, since R is also the Markov chain induced by the PLCS $\mathcal{L}' = \mathcal{A} \times \mathcal{L}$ (Lemma 7.2), the set B of states with empty channels in R is a finite attractor for R (Lemma 5.3). Thus Lemma 7.5 applies and provides necessary and sufficient conditions for the computations of M that start at s to be accepted by $\mathcal{A}$ with probability one.

It remains to show that these conditions can be checked effectively. First *Graph*(B) is computable using reachability algorithms on $\mathcal{L}'$. Then the conditions of Lemma 7.5(2) can be checked with algorithms for reachability of upward-closed sets (again on $\mathcal{L}'$): condition (a) requires that $((\mathcal{Q} \setminus F) \times \mathtt{S}) \uparrow$ is not reachable, and condition (b) requires that each $(\{q\} \times \mathtt{S}) \uparrow$ is reachable.

8 Concluding Remarks

We have shown decidability of model checking for a realistic class of probabilistic lossy channel systems, where during each step of the runs of the systems, any message inside the channels may be lost with a certain predefined probability.

8.1 Comparison With Other Work

A work closely related to this article is [5]. In fact, our work can be seen as a generalization of the ideas presented in [5]. More precisely, in [5], a formal model of PLCS's is considered where at most one message can be lost during each step of the execution of the system. Thus a PLCS $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$ induces a Markov chain $(\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*), P)$ where P is defined [3] as follows:

$$P\left((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)\right) = \begin{cases} \dfrac{(1-\lambda)w(\mathtt{t})}{w(\mathtt{s}_1, \mathtt{w}_1)} & \text{if } (\mathtt{s}_2, \mathtt{w}_2) = \mathtt{t}(\mathtt{s}_1, \mathtt{w}_1) \text{ and } |\mathtt{w}_1| > 0, \\ \dfrac{w(\mathtt{t})}{w(\mathtt{s}_1, \mathtt{w}_1)} & \text{if } (\mathtt{s}_2, \mathtt{w}_2) = \mathtt{t}(\mathtt{s}_1, \mathtt{w}_1) \text{ and } |\mathtt{w}_1| = 0, \\ \lambda \dfrac{\#\left(\mathtt{w}_1, \mathtt{w}_2\right)}{|\mathtt{w}_1|} & \text{if } |\mathtt{w}_1| > 0 \text{ and } |\mathtt{w}_2| = |\mathtt{w}_1| - 1. \end{cases}$$

Then [5] restricts to the case where the probability λ of losing messages is assumed to be at least 0.5 and shows decidability of model checking $LTL_{\setminus \mathtt{X}}$ formulas (i.e., LTL formulas that are insensitive to stuttering [16]). Decidability of model checking is shown by proving that, under the assumption that $\lambda \geq 0.5$, one gets a *probabilistic input-enabled PLCS*: a system where in any state there is probability at least 0.5 that the size of the channel

[3] We give a simplified version of the definition, where we do not account for the fact that *several* different transitions may allow reaching $(\mathtt{s}_2, \mathtt{w}_2)$ from $(\mathtt{s}_1, \mathtt{w}_1)$.

contents decreases. In fact, for probabilistic input enabled PLCS's, the set $\{(\mathtt{s}, \mathtt{w}) \ : \ |\mathtt{w}| = 0\}$ is an attractor, and decidability of model checking follows then in a similar manner to Section 5 and Section 7.

There are, however, some problems with the definition in [5] (disregarding the issues of whether it is more natural to have the probability λ applying independently to all messages in a channel, or globally to its whole contents, or whether $\lambda \geq 0.5$ is commonplace). Their operational semantics keeps message losses apart from perfect steps (while we amalgamate them). As a consequence the product of an automaton $\mathcal{A}$ with the Markov chain M associated with some PLCS $\mathcal{L}$ is *not* (isomorphic to) the Markov chain of some $\mathcal{L}'$. In other words, their definition does not support a result like our Lemma 7.2. Furthermore, their definition makes it possible for an automaton to *observe* losses: in their framework, it is undecidable whether the traces of a state-labeled PLCS almost surely belong to $L.\Sigma^\omega$ for a *finitary* regular language L.

8.2 Complexity Analysis

Our method for checking that a state, or a set of states, will be reached almost surely in a PLCS $\mathcal{L}$ reduces the problem to polynomially-many reachability questions on the underlying (non-probabilistic) LCS. In the other direction, it is easy to reduce reachability in an LCS to probabilistic reachability in some associated PLCS: we conclude that the verification of qualitative probabilistic properties on PLCS's and the verification of reachability properties on LCS's are equally hard. Recall that reachability in LCS's cannot be solved in primitive recursive time [19].

When verifying a linear-time property given by a deterministic Muller automaton $\mathcal{A}$, our method reduces the problem to reachability problems in the product $\mathcal{A} \times \mathcal{L}$. When the property is given via a nondeterministic automaton, our method requires a determinization step that can cause an exponential blowup. In that case, we require time bounded by some $F(|\mathcal{L}| \times \exp(|\mathcal{A}|))$ for a nonprimitive recursive F. An interesting question is whether one can avoid the price of analyzing the product $\mathcal{A} \times \mathcal{L}$, i.e., whether there exist methods that only require time of the form $F(|\mathcal{L}|) \times G(|\mathcal{A}|)$ where G is simpler than F (e.g., G is elementary).

In the same direction, it would be interesting to see whether we could apply the recent results by Couvreur *et al.* [11] to our problems. For finite Markov chains, these authors show how to verify properties described with unambiguous separated ω-automata instead of just deterministic ones, thereby saving one exponential blowup when translating from LTL formulas. Applying their method requires extending it from finite chains to countable chains with a finite attractor, and further proving that it provides a reduction to decidable questions on the underlying LCS.

8.3 Dealing With Deadlock States

PLCS's with deadlock states raise some difficulties. First one has to fix the definition of the Markov chain induced by the PLCS, since Equation (5.6) assumes that the PLCS is deadlock-free. In order to retain Markov chain models, we introduce a dummy *sink* state and define for any state s:

$$P(s, \mathit{sink}) = \begin{cases} 1 & \text{if } s \text{ is a deadlock state, or } s = \mathit{sink}, \\ 0 & \text{otherwise.} \end{cases}$$

When s is not a deadlock state and s' is not *sink*, $P(s, s')$ is as before.

Now, if we write Q_0 for the set of states where the channels are empty, then $Q_0 \cup \{\mathit{sink}\}$ is a finite attractor. We can thus reuse our approach based on finite attractors to prove that Probabilistic Reachability, Probabilistic Repeated Reachability and Probabilistic Model Checking are decidable for PLCS's with deadlock states.

For this, we only need to prove the necessary effectiveness conditions: that $\mathit{Graph}(Q_0 \cup \{\mathit{sink}\})$ is constructible, etc. Everything boils down to the following Lemma, that we prove in the rest of this section.

Lemma 8.1 *It is decidable whether sink is reachable from a given state u.*

Reachability of *sink* amounts to reachability of a deadlock state in the underlying LCS. Let us write D for the set of deadlock states. D is usually infinite, and is not an upward-closed set of states. It is not downward-closed either [4]. However, a closure property satisfied by D is

$$(\mathtt{s}, \mathtt{w}) \in D \text{ implies } (\mathtt{s}, \varepsilon) \in D.$$

Say a local state $\mathtt{s}$ in $\mathtt{S}$ is *deadlockable* if all transitions $(\mathtt{s}, \mathtt{op}, \mathtt{s}')$ in $\mathtt{T}$ are receiving transitions, i.e., with $\mathtt{op}$ the form $\mathtt{c?m}$.

Lemma 8.2 *Assume $u \notin D$. Then D is reachable from u iff there exists a deadlockable $\mathtt{s}$ s.t. $(\mathtt{s}, \varepsilon)$ is reachable from u.*

[4] This is because our definition in Section 5 groups normal steps and message losses in a way where losses occur *after* normal steps.

PROOF. If $(\mathtt{s}, \mathtt{w}) \neq u$ is reachable from u then $(\mathtt{s}, \varepsilon)$ is also reachable from u: it is enough to lose all messages from $\mathtt{w}$ in the last step of the path from u to $(\mathtt{s}, \mathtt{w})$. Furthermore, if $(\mathtt{s}, \mathtt{w}) \in D$ then $\mathtt{s}$ is deadlockable. □

This reduces reachability of *sink* to a finite number of (decidable) reachability questions, establishing Lemma 8.1.

8.4 Perspectives

There are several problems left open in this article, which we believe are worth considering in future research.

In view of our positive results, it is natural to investigate more elaborate models of PLCS's, allowing the combination of nondeterministic and probabilistic behavior. We refer to [6] for a preliminary study of such extensions.

Along another direction, it is also natural to ask whether *quantitative verification* is possible for PLCS's. A typical quantitative problem is the following:

Computing Probability of Reachability

Instance: A PLCS $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$, a state s, and a set $\mathtt{Q} \subseteq \mathtt{S}$.

Question: What is the probability r that, starting from s, one reaches the set of $\mathtt{Q}$-states?

A related problem is to decide whether $r \geq h$ for some given h.

These problems are still open. It is not even known that r is an algebraic number, or is expressible by standard mathematical functions.

However, the third author recently showed how to solve the following *approximate* quantitative verification problem [18]:

Approximating Probability of MSO Properties for PLCS's

Instance: A PLCS $\mathcal{L} = (\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w)$, a starting state s, an MSO property $\mathcal{A}$, and a rational $\tau > 0$ (called the *tolerance*).

Question: Compute a rational r such that the probability that the runs of $\mathcal{L}$ are accepted by $\mathcal{A}$ is in the interval $[r - \tau, r + \tau]$.

References

[1] P. A. Abdulla, C. Baier, S. Purushothaman Iyer, and B. Jonsson. Simulating perfect channels with probabilistic lossy channels. *Information and Computation*, 197(1–2):22–40, 2005.

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

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

[4] 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)*, volume 2620 of *Lecture Notes in Computer Science*, pages 39–53. Springer, 2003.

[5] 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)*, volume 1601 of *Lecture Notes in Computer Science*, pages 34–52. Springer, 1999.

[6] 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)*, volume 2620 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2003.

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

[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] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, 1995.

[11] J.-M. Couvreur, N. Saheb, and G. Sutre. An optimal automata approach to LTL model checking of probabilistic systems. In *Proc. 10th Int. Conf. Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2003)*, volume 2850 of *Lecture Notes in Artificial Intelligence*, pages 361–375. Springer, 2003.

[12] J. Esparza, A. Kucera, and R. Mayr. Model checking probabilistic pushdown automata. In *Proc. 19th IEEE Symp. Logic in Computer Science (LICS 2004)*, pages 12–21. IEEE Comp. Soc. Press, 2004.

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

[14] J. G. Kemeny, J. L. Snell, and A. W. Knapp. *Denumerable Markov Chains.* D. Van Nostrand Co., Princeton, NJ, USA, 1966.

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

[16] D. Peled and T. Wilke. Stutter-invariant temporal properties are expressible without the next-time operator. *Information Processing Letters*, 63(5):243–246, 1997.

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

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

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

[20] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 4, pages 133–191. Elsevier Science, 1990.

[21] M. Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proc. 26th IEEE Symp. Foundations of Computer Science (FOCS '85)*, pages 327–338, 1985.