Verification of Probabilistic Systems with Faulty Communication

Parosh Aziz Abdulla[1] and Alexander Rabinovich[2]

[1] Uppsala University, Sweden
[2] 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 behaviour of such protocols, *lossy channel systems (LCS)* [AJ96b] have been proposed. In an LCS the communication channels are modelled as FIFO buffers which are unbounded, but also unreliable in the sense that they can nondeterministically lose messages.

Recently, several attempts [BE99, ABIJ00] have been made to study *probabilistic Lossy Channel Systems (PLCS)* in which the probability of losing messages is taken into account. In this paper, we consider a variant of PLCS which is more realistic than those studied in [BE99, ABIJ00]. 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 given property definable by finite state ω-automata 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 through unbounded buffers have been popular in the modelling of communication protocols [BZ83, Boc78]. One disadvantage with such a model is that it has the full computation power of Turing machines [BZ83], 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 behaviour of such protocols, *lossy channel systems (LCS)* [AJ96b] have been proposed as an alternative model. In an LCS the communication channels are modelled as FIFO buffers which are unbounded but also unreliable in the sense that they can nondeterministically lose messages. For LCS it has been shown that the reachability problem is decidable [AJ96b] while progress properties are undecidable [AJ96a].

Since we are dealing with unreliable communication media, it is natural to deal with models where the probability of losing messages is taken into account.

A.D. Gordon (Ed.): FOSSACS 2003, LNCS 2620, pp. 39–53, 2003.

Recently, several attempts [BE99, ABIJ00] have been made to study *probabilistic Lossy Channel Systems (PLCS)* which introduce randomization into the behaviour of LCS. The decidability of model checking for the proposed models depend heavily on the semantics provided. The works in [BE99, ABIJ00] define different semantics for PLCS depending on the manner in which the messages may be lost inside the channels.

Baier and Engelen [BE99] consider a model where it is assumed that at most single message may be lost during each step of the execution of the system. 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 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 [ABIJ00] 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. The idea of the proof is to choose sufficiently low probabilities for message losses to enable the system to simulate the behaviour of (non-probabilistic) systems with perfect channels.

In this paper, we consider a variant of PLCS which are more realistic than that in [BE99, ABIJ00]. 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 [BE99]). Thus, in contrast to [BE99] 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. To do that, we introduce the concept of *attractor sets*: all computations of the system are guaranteed to eventually reach the attractor. The existence of finite attractors imply 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, when interpreted according to our semantics, have finite attractors. More precisely, we prove the existence of an attractor defined by the set of all configuration where the sizes of channels are bound by some natural number. This natural number can be derived from the predefined probability given to the loss of messages. In fact, for the systems considered in [BE99] this bound is equal to 0, and therefore the decidability results in [BE99] can be seen as a consequence of the properties we show for attractors.

We also show that our decidability results extend to PLCS with different sources of unreliability [CFI96], such as duplication, corruption, and insertion combined with lossiness Furthermore, we show how to extend our decidability results to more general properties specified by finite state automata or equivalently by formulas of Monadic Logic of Order.

Remark Bertrand and Schnoebelen [BS03] have independently obtained what essentially amounts to Theorem 1 in this paper.

Outline In the next two Sections we give basics of transition systems and Markov chains respectively. 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. In Section 6 we consider models involving different sources of unreliability combined with lossiness. In Section 7 we generalize our results to verification of the properties definable by ω-behavior of finite state automata (or equivalently 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 (potentially) 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 $\xrightarrow{*}$ to denote the reflexive 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 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 $s_n = s'$. We say that p is *simple* if there are no i, j with $i \neq j$ and $s_i = s_j$. 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 the set of all states s such that there is a path $s_0 \longrightarrow s_1 \longrightarrow \cdots \longrightarrow s_n$ from s 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 transition system $\left(Q, \longrightarrow'\right)$ 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_1 in T with $C \xrightarrow{*} C_1$. In other words, the BSCCs are the leafs 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?

Until
Instance A transition system $T = (S, \longrightarrow)$, a state s, and sets $Q_1, Q_2 \subseteq S$.
Question Is $s \in \mathit{Until}(Q_1, Q_2)$?

3 Markov Chains

In this section, we introduce (potentially infinite state) *Markov chains.*

A *Markov chain* M is a pair (S, P) where S is a (potentially infinite) set of states and P is a mapping from $S \times S$ to the set $[0, 1]$, 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 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 a SCC in the underlying transition system.

A Markov chain (S, P) induces a natural measure on the set of computations from every state s.

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 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 Prob* : $\Delta \rightarrow [0, 1]$ such that $Prob(\Omega) = 1$ and that is countably additive, we get a *probability space* $(\Omega, \Delta, Prob)$.

Consider a state s of a Markov chain (S, P). On the sets of computations that start at s, the probabilistic space is defined as follows:

Probabilistic space $(\Omega, \Delta, Prob)$(see [KSK66]) : $\Omega = sS^\omega$ 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 $u \in sS^*$, and the probability measure *Prob* is defined by $Prob(D_u) = \prod_{i=0,\ldots,n-1} P(s_i, s_{i+1})$ where $u = s_0 s_1 \ldots s_n$; 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.

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.

In the rest of this section, we assume a Markov chain $M = (S, P)$ with an underlying transition system $T = (S, \longrightarrow)$.

Consider a set $Q \subseteq S$ of states and a computation π. We say that π *reaches* Q if there is an $i \geq 0$ with $\pi(i) \in Q$. We say that π *repeatedly reaches* Q if there are infinitely many i with $\pi(i) \in Q$. Let s be a state in S. We define the probability of Q being (repeatedly) reachable from s by

$Prob\,\{\pi |\ \pi$ is a computation from s and π (repeatedly) reaches $Q\}$.

We consider the following two problems for Markov chains:

Probabilistic Reachability
Instance A Markov chain $M = (S, P)$, a state $s \in S$, and a set $Q \subseteq S$.
Question Is Q reachable from s with probability one?

Probabilistic Repeated Reachability
Instance A Markov chain $M = (S, P)$, a state $s \in S$, and a set $Q \subseteq S$.
Question Is Q repeatedly reachable from s with probability one?

In the above problems, we do not assume that Markov chains are finite. Hence these are not instances of algorithmic problems. 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.

We introduce a central concept which we use in our solution for the probabilistic (repeated) reachability problem, namely that of *attractors.*
Definition [attractors] A set $A \subseteq S$ is said to be an *attractor*, if for each $s \in S$, the set A is reachable from s with probability one.

In other words, regardless of the state in which we start, we are guaranteed that we will eventually enter the attractor.

We consider two preliminary lemmas which are derived from the standard properties of recurrent classes. The Lemma below describes a property of BSCCs of the graph of a finite attractor A, which will make use of in our algorithms (to prove Lemma 2 and Lemma 3).

Lemma 1. *Consider a finite attractor A, a BSCC C in Graph(A), and a state $s \in C$. Then, for every $s' \in C$, the probability that s' is repeatedly reachable from s is one.*

The following Lemma enables us to construct an algorithm for solving the probabilistic reachability problem.

Lemma 2. *Consider a finite attractor A, a state $s \in S$, and a set $Q \subseteq S$. Then, Q is reachable from s with probability one iff for each BSCC C in Graph(A), if C is reachable from s then either*

- *Q is reachable from C; or*
- *For every finite simple path in T from s, if p reaches C then p also reaches Q.*

From Lemma 2 we conclude that we can define a scheme for solving the reachability problem as follows.

```
Scheme – Probabilistic Reachability

Input Markov chain $M = (S, P)$ with an underlying transition system $T = (S, \longrightarrow)$, a state $s \in S$, and a set $Q \subseteq S$.
Output Is $Q$ reachable from $s$ with probability one?
begin
    1. construct a finite attractor $A$
    2. construct $\mathit{Graph}(A)$
    3. for each BSCC $C$ in $\mathit{Graph}(A)$ which is reachable from $s$
        3a. if $\neg\left(C \xrightarrow{*} Q\right)$ and $s \in \mathit{Until}(\neg Q, C)$ then return(false)
    4. return(true)
end
```

The following Lemma enables us to construct an algorithm for solving the probabilistic repeated reachability problem.

Lemma 3. *Consider a finite attractor A, a state $s \in S$, and a set $Q \subseteq S$. Then, Q is repeatedly reachable form s with probability one iff the reachability of C from s implies the reachability of Q from C, for each BSCC C in $Graph(A)$.*

From Lemma 3 we conclude that we can define a scheme for solving the repeated reachability problem as follows.

```
        3a. if $\neg\left(C \xrightarrow{*} Q\right)$ then return(false)
```

The correctness of the two schemes follows immediately from Lemma 2 and Lemma 3. Furthermore, we observe that, in order to obtain algorithms for checking the reachability and repeated reachability problems, we need the following three effectiveness properties for the operations involved:

1. Existence and computability of a finite attractor. This condition is necessary for computing the set A.
2. Decidability of the reachability problem for the underlying class of transition systems T. This condition is necessary for computing $\mathit{Graph}(A)$ and for checking the relation $C \xrightarrow{*} Q$.
3. Decidability of the until problem for the underlying class of transition systems. This condition is only needed 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 scheme 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.

Lossy Channel Systems A *lossy channel system* consists of a finite state process operating on a finite set of channels each of which behaves as a FIFO buffer which is unbounded and 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 $\mathtt{M}^*$.

For words $x, y \in \mathtt{M}^*$, we use $x \bullet y$ to denote the concatenation of x and y. We write $x \preceq y$ to denote that x is a (not necessarily contiguous) substring of y. By Higman's Lemma [Hig52] it follows that $\preceq$ is a well quasi-ordering, i.e., for each infinite sequence $x_0, x_1, x_2, \ldots$ there are i and j with $i < j$ and $x_i \preceq x_j$. We use $|x|$ to denote the length of x, and use $x(i)$ to denote the i^{th} element of x where $i : 1 \leq i \leq |x|$. For $\mathtt{w}_1, \mathtt{w}_2 \in (\mathtt{C} \mapsto \mathtt{M}^*)$, we use $\mathtt{w}_1 \preceq \mathtt{w}_2$ to denote that $\mathtt{w}_1(\mathtt{c}) \preceq \mathtt{w}_2(\mathtt{c})$ for each $\mathtt{c} \in \mathtt{C}$, and define $|\mathtt{w}| = \sum_{\mathtt{c} \in \mathtt{C}} |\mathtt{w}(\mathtt{c})|$. We also extend $\preceq$ to a relation on $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$, where $(\mathtt{s}_1, \mathtt{w}_1) \preceq (\mathtt{s}_2, \mathtt{w}_2)$ iff $\mathtt{s}_1 = \mathtt{s}_2$ and $\mathtt{w}_1 \preceq \mathtt{w}_2$.

The LCS $\mathcal{L}$ induces a transition system $(S, \longrightarrow)$, where S is the set of global states, i.e., $S = (\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*))$, and $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2)$ iff 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 $\mathtt{w}_1$ is the result of removing $\mathtt{m}$ from the head of $\mathtt{w}_2(\mathtt{c})$.
- Furthermore, if $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2, \mathtt{w}_2)$ according to one of the previous two rules then $(\mathtt{s}_1, \mathtt{w}_1) \longrightarrow (\mathtt{s}_2', \mathtt{w}_2')$ for each $(\mathtt{s}_2', \mathtt{w}_2') \preceq (\mathtt{s}_2, \mathtt{w}_2)$.

In the first two cases we define $\mathtt{t}(\mathtt{s}_1, \mathtt{w}_1) = (\mathtt{s}_2, \mathtt{w}_2)$.

A transition $(\mathtt{s}_1, \mathtt{op}, \mathtt{s}_2)$ is said to be *enabled* at $(\mathtt{s}, \mathtt{w})$ if $s = s_1$ and either

- $\mathtt{op}$ is of the form $\mathtt{c!m}$; or
- $\mathtt{op}$ is of the form $\mathtt{c?m}$ and $\mathtt{w}(\mathtt{c}) = \mathtt{m} \bullet x$, for some $x \in \mathtt{M}^*$.

We defined *enabled*$(\mathtt{s}, \mathtt{w}) = \{\mathtt{t} |\ \mathtt{t}$ is enabled at $(\mathtt{s}, \mathtt{w})\}$. In the sequel, we assume that for all $(\mathtt{s}, \mathtt{w})$, the set *enabled*$(\mathtt{s}, \mathtt{w})$ is not empty. This is guaranteed for instance, by requiring that for any local state $\mathtt{s}_1$ there are $\mathtt{c}$, $\mathtt{m}$, and $\mathtt{s}_2$ with $(\mathtt{s}_1, \mathtt{c!m}, \mathtt{s}_2) \in \mathtt{T}$

Remark on notation We use $\mathtt{s}$ and $\mathtt{S}$ to range over local states and sets of local states respectively. On the other hand, we 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)

For the rest of this section we assume an LCS $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T})$.

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.

In [AJ96b], algorithms are given which shows the following decidability results for LCS:

Lemma 4. *For states s_1 and s_2, it is decidable whether s_2 is reachable from s_1.*

Lemma 5. *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.*

Decidability of the corresponding until problem follows from a straightforward modification of the reachability algorithm of [AJ96b]. This gives the following.

Lemma 6. *For a state s, a set $\mathtt{Q}_1 \subseteq \mathtt{S}$, and a finite set Q_2 of states, it is decidable whether $s \in Until(\neg Q_1, Q_2)$, where Q_1 is the set of $\mathtt{Q}_1$-states.*

Probabilistic Lossy Channel Systems 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 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 $(s_3, \mathtt{w}_3)$ of) the probability of reaching some $(s_3, \mathtt{w}_3)$ 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 derive these probabilities from the definition of $\mathcal{L}$.

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

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

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 $P_L(x, y) = \#(x, y) \cdot \lambda^{|y|-|x|} \cdot (1 - \lambda)^{|x|}$. For $\mathtt{w}_1, \mathtt{w}_2 \in (\mathtt{C} \mapsto \mathtt{M}^*)$, we define $P_L(\mathtt{w}_1, \mathtt{w}_2) = \prod_{\mathtt{c} \in \mathtt{C}} P_L(\mathtt{w}_1(\mathtt{c}), \mathtt{w}_2(\mathtt{c}))$. Intuitively, $P_L(\mathtt{w}_1, \mathtt{w}_2)$ defines the probability by which $\mathtt{w}_2$ can change to $\mathtt{w}_1$ through loss of messages during a single step of the execution of the system. Notice that $P_L(\mathtt{w}_1, \mathtt{w}_2) = 0$ in case $\mathtt{w}_1 \not\preceq \mathtt{w}_2$. We take $P_L((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = P_L(\mathtt{w}_1, \mathtt{w}_2)$ if $\mathtt{s}_1 = \mathtt{s}_2$, and $P_L((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = 0$ otherwise. We define $w(\mathtt{s}, \mathtt{w}) = \sum_{\mathtt{t} \in enabled(\mathtt{s}, \mathtt{w})} w(\mathtt{t})$.

The PLCS $\mathcal{L}$ induces a Markov chain (S, P), where $S = (\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*))$ and $P((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = \sum_{\mathtt{t} \in \mathtt{T}} ((w(\mathtt{t})/w(\mathtt{s}_1, \mathtt{w}_1)) \cdot P_L(\mathtt{t}(\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)))$. Notice that this is well-defined by the assumption that there are no deadlock states.

We instantiate the reachability problems considered in Section 3 and Section 4 to PLCS.

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 $T = (S, \longrightarrow)$.

We shall consider the probabilistic (repeated) reachability problem for PLCS. We check whether an upward closed set, represented by its minimal elements, is (repeatedly) reachable 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.

The following Lemma shows that we can always construct a finite attractor in a PLCS.

Lemma 7. *For each* λ, w, *and PLCS* $(\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$, *thse set* $\{(\mathtt{s},\mathtt{w}) \mid |\mathtt{w}| = 0\}$ *is an attractor.*

From Lemma 4, and the fact that the transition system underlying a PLCS $(\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$ is independent on λ we obtain:

Lemma 8. *For each PLCS* $(\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$, *we can compute the graph* $Graph(A)$ *of a finite set* A.

Furthermore, for two PLCS $\mathcal{L} = (\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda,w)$ and $\mathcal{L}' = (\mathtt{S},\mathtt{C},\mathtt{M},\mathtt{T},\lambda',w')$ which differ only by probabilities, If $\lambda, \lambda' > 0$ and $w(\mathtt{t}) > 0$ iff $w'(\mathtt{t}) > 0$ then A has the same graph in both PLCS. Now we are ready to solve Probabilistic Reachability and Probabilistic Repeated Reachability problems for PLCS.

Probabilistic Reachability for PLCS
Instance An 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 is reachable from s with probability one?

Probabilistic Repeated Reachability
Instance An 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 is repeatedly reachable from s with probability one?

From the results of Section 4 and Lemma 8, Lemma 5, Lemma 7, and Lemma 6 we get the following.

Theorem 1. *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS. .*

Remark In our definition of LCS and PLCS, we assume that messages are lost only after performing non-lossy transitions. 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 with different sources of unreliability such as duplication, corruption, and insertion combined with lossiness.

Duplication We analyze a variant of PLCS, where we add another source of unreliability; namely a message inside a channel may be duplicated [CFI96].

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 behaviour 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

$\left\{ b_1 b_2 \cdots b_n \middle| \text{ either } b_i = a_i \text{ or } b_i = a_i^2 \text{ for each } i : 1 \le i \le n \right\}$

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 is enlargement of that of the corresponding standard LCS in the sense that:

- 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 $(\mathtt{s}'_2, \mathtt{w}'_2) \in$ *duplicate*$(\mathtt{s}_2, \mathtt{w}_2)$.

In [CFI96], it is shown that the reachability problem is decidable for LCS with duplication errors. The reachability algorithm can be modified in a similar manner to Section 5 to solve the until problem. Hence we have

Lemma 9. *Given LCS with duplication errors.*

1. *For states s_1 and s_2, it is decidable whether s_2 is reachable from s_1 [CFI96]. 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 [CFI96].*
3. *For a state s, a set $\mathtt{Q}_1 \subseteq \mathtt{S}$, and a finite set Q_2 of states, it is decidable whether $s \in$ Until$(\neg Q_1, Q_2)$, where Q_1 is the set of $\mathtt{Q}_1$-states.*

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 $\left\{ (i_1, \ldots, i_n) \,|\, 1 \le i_j \le 2 \text{ and } y = a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n} \right\}$. In other words, $\#_D\,(x, y)$ is the number of the 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 $P_D(x, y) = \#_D\,(x, y) \cdot \lambda_D^{|y|-|x|} \cdot (1 - \lambda_D)^{|x|}$, 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 $\mathcal{L}$ with duplication errors induces a Markov chain (S, P'_D), where $S = (\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*))$ and

$P'_D\,((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_2, \mathtt{w}_2)) = \sum_{(\mathtt{s}_3, \mathtt{w}_3)} P\,((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_3, \mathtt{w}_3)) \cdot P_D\,((\mathtt{s}_3, \mathtt{w}_3), (\mathtt{s}_2, \mathtt{w}_2))$

where P has the same definition as in Section 5. Notice that the sum is computable since the set $\{(\mathtt{s}_3, \mathtt{w}_3) \,|\, P\,((\mathtt{s}_1, \mathtt{w}_1), (\mathtt{s}_3, \mathtt{w}_3)) \ne 0\}$ is finite and computable.

Lemma 10. *For each λ, w, λ_D, and PLCS $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_D)$ with $\lambda_D < \lambda$, the set $\{(\mathtt{s}, \mathtt{w}) \,|\, |\mathtt{w}| = 0\}$ is an attractor.*

Using a similar reasoning to Section 5, we derive from Lemma 9 and Lemma 10

Theorem 2. *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS with duplication errors when $\lambda_D < \lambda$.*

Corruption We consider LCS with *corruption errors*, i.e., a message inside a channel may be changed to any other message. We extend the semantics of LCS to include corruption errors in the same manner as we did above for duplication errors. For $x \in \mathtt{M}^*$, we define $\mathit{Corrupt}(x)$ to be the set $\{y| \ |y| = |x|\}$, i.e., we get a member of $\mathit{Corrupt}(x)$ 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 transition 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 $(\mathtt{s}_2', \mathtt{w}_2') \in \mathit{Corrupt}(\mathtt{s}_2, \mathtt{w}_2)$.

Decidability of the reachability problem for LCS 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 the channels. In other words, the problem is reduced to ¡a special case of LCS systems where the set $\mathtt{M}$ can be considered to be a singleton. The until problem can be solved in a similar manner. Hence,

Lemma 11. *Given LCS with corruption errors.*

1. *For states s_1 and s_2, it is decidable whether s_2 is reachable from s_1. Hence, $\mathit{Graph}(A)$ 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.*
3. *For a state s, a set $\mathtt{Q}_1 \subseteq \mathtt{S}$, and a finite set Q_2 of states, it is decidable whether $s \in \mathit{Until}(\neg Q_1, Q_2)$, where Q_1 is the set of $\mathtt{Q}_1$-states.*

A PLCS with *corruption errors* is of the form $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_C)$, where $\lambda_D \in [0, 1]$ 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 $\{i| \ x(i) = y(i)\}$. In other words, $\#_C\,(x, y)$ is the number of elements which must change in order to obtain y from x. We define $P_C(x, y) = \left(\frac{\lambda_C}{|\mathtt{M}|-1}\right)^{\#_C(x,y)} \cdot (1 - \lambda_C)^{|x| - \#_C(x,y)}$ if $|x| = |y|$, and $P_C(x, y) = 0$ otherwise. We extend $P_C(x, y)$ to $\mathtt{S} \times (\mathtt{C} \mapsto \mathtt{M}^*)$ as before. This induces a Markov chain in a similar manner to the case with duplication.

Lemma 12. *For each λ, w, λ_C, and PLCS $(\mathtt{S}, \mathtt{C}, \mathtt{M}, \mathtt{T}, \lambda, w, \lambda_C)$, the set $\{(\mathtt{s}, \mathtt{w}) \mid |\mathtt{w}| = 0\}$ is an attractor.*

From Lemma 11 and Lemma 12 we can derive in a similar manner to Section 5.

Theorem 3. *Probabilistic Reachability and Probabilistic Repeated Reachability are decidable for PLCS with corruption errors.*

Other Unreliability Sources In a similar manner to the cases with duplication and corruption, we can obtain decidability results for models involving other sources of unreliability such as insertion of messages [CFI96]. Furthermore, we can combine different sources of unreliability. For instance, we can consider models where we have both duplication and corruption together with lossiness. The crucial aspect of the model is that unreliability sources which may increase the number of messages inside the channels (such as insertion and duplication but not corruption) should have sufficiently low probabilities (compared to lossiness) to guarantee existence of a finite attractor.

7 Automata Definable Properties

In this section we consider more general properties than reachability and repeated reachability for PLCS. Let φ be a property of computations. We will be interested in whether

$$\mathit{Prob}\,\{\pi |\ \pi \text{ is a computation from } s \text{ in PLCS } M \text{ and } \pi \text{ satisfies } \varphi\ \} = 1.$$

We show that if the properties of computations are specified by (the ω-behavior of) finite state automata or equivalently by formulas of Monadic Logic of Order then the above problem is decidable

In order to check a property defined by a finite state automaton, we take its product with the given PLCS. The acceptance conditions are reduced to the reachability problem for the non-probabilistic system underlying the product. Similar results hold for the faulty probabilistic systems considered in section 6. The proofs for these systems follow the same pattern as for PLCS, therefore here we will confine ourself only with PLCS.

We consider an extension of LCS by adding 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 always assume that LCS are state labeled and will often use "LCS" for "state labeled LCS". We lift the labeling from LCS to the *state labeled transition system* $T = (S, \longrightarrow, \Sigma, \mathit{lab})$ induced by an LCS $\mathcal{L}$: the label of every state in T is the same as the label of its local state component. Similarly, with a path $s_0, s_1, \ldots$ we associate an ω-string $\mathit{lab}(s_1), \mathit{lab}(s_2), \ldots$ over the alphabet Σ. When we deal here 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* inducing *state labeled Markov chains.*

Next, we recall basic definitions and notations about finite state automata and cite a classical theorem (Theorem 4 [Tho90]) that automata have the same expressive power as monadic logic of order. A *finite 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 of *actions*, a *transition relation* $\rightarrow$ which is a subset of $\mathcal{Q} \times \Sigma \times \mathcal{Q}$, $q_0 \in \mathcal{Q}$ is the *initial state* of $\mathcal{A}$, and $\mathcal{F} \subseteq 2^{\mathcal{Q}}$ is a collection of *fairness conditions.* We write $q \xrightarrow{a} q'$ if $\langle q,\ a,\ q' \rangle \in \rightarrow$.

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. Such a run meets the the fairness conditions if the set of states that occur in the run infinitely many times is a member of $\mathcal{F}$. 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 acceptable by $\mathcal{A}$. We say that $\mathcal{A}$ is *deterministic* if for every state q and every letter b there is a unique q' such that $q \xrightarrow{b} q'$.

Theorem 4. *The following conditions are equivalent for ω-language L:*

1. *L is acceptable by a finite state automaton.*
2. *L is acceptable by a deterministic finite state automaton.*
3. *L is definable by a monadic formula .*

Products We define products of automata and state labeled transition systems. We also define products of automata and state labeled Markov chains. We investigate the reachability problem for these products and provide reduction of verification of automata definable properties of computations to the reachability problem. Consider an automaton $\mathcal{A} = (\mathcal{Q},\ \Sigma,\ \rightarrow,\ q_0, \mathcal{F})$, and a state labeled transition system $T = (S, \longrightarrow, \Sigma, lab)$. The *product* of $\mathcal{A}$ and M is a state labeled transition system defined as follows:

States: $Q \times S$ - 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 transition from (q, s) to (q', s') iff there is a transition $q \overset{lab(s)}{\rightarrow} q'$ in $\mathcal{A}$ and there is a transition from s to s' in T.

Problem 1
Instance A state labeled LCS which defines a state labeled transition system $T = (S, \rightarrow,\ lab, \Sigma)$, an automaton $\mathcal{A}$, states s_1 and s_2 in the product of T and $\mathcal{A}$.
Question Is s_2 reachable from s_1?

Problem 2
Instance A state labeled LCS which defines a state labeled transition system $T = (S, \rightarrow,\ lab, \Sigma)$, an automaton $\mathcal{A}$, states s_1 and a finite set of states S_2 in the product of T and $\mathcal{A}$.
Question Is the upward closure of S_2 reachable from s_1?

Lemma 13. *Problem 1 and Problem 2 are decidable.*

Next, we consider products of automata and state labeled Markov chains. Consider a deterministic automaton $\mathcal{A} = \langle Q,\ \Sigma,\ \rightarrow,\ q_0,\ \mathcal{F}\rangle$ and a state labeled Markov chain $M = (S, P,\ lab,\ \Sigma)$. The product of A and M is a state labeled Markov chain defined as follows:

States: $Q \times S$ - the Cartesian product of the states of $\mathcal{A}$ and of M.
Labeling: A state (q, s) is labeled by $lab(s)$, i.e., it has the same label as s in M.
Transition relation: The probability of transition from (q, s) to (q', s') is p iff there is a transition $q \overset{lab(s)}{\rightarrow} q'$ in $\mathcal{A}$ and the probability of transition from s to s' in M is p.

Observe that the requirement that $\mathcal{A}$ is deterministic ensures that the sum of probabilities of the transitions from the state (q, s) 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.

We say that a computation $s_1, s_2, \ldots$ is accepted by an automaton iff the corresponding ω-string $lab(s_1), lab(s_2), \ldots$ is accepted

Lemma 14. *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 of $\mathcal{A}$ and M, and let B be an attractor of R. Then the following are equivalent:*

1. *The probability of the set computations of M that start at s and are accepted by $\mathcal{A}$ is one.*
2. *For each BSCC C in Graph(B), if C is reachable from 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.

Probabilistic Model Checking The next problem deals with probabilistic LCS.
Problem: Probabilistic Model-checking.
Instance A stated labeled PLCS which defines a state labeled Markov chain M, a state s in M, and an automaton $\mathcal{A}$.
Question Is the probability that a computation of M that starts at s is accepted by $\mathcal{A}$ equal to one?

Theorem 5. *Probabilistic Model-checking. Problem is decidable.*

Proof. Let R be the product of $\mathcal{A}$ and M. It is easy to see, by the same arguments as in Lemma 7, that the set B of states with empty channels in R is a finite attractor for R. By Lemma 13, we obtain that *Graph*(B) is computable. Now, applying reachability algorithm of Lemma 13, we can verify the conditions of Lemma 14(2). By Lemma 14 these conditions are satisfied if and only if the probability that a computation of M that starts at s is accepted by $\mathcal{A}$ equal to one.

8 Conclusions and Discussion

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.

In Section 5 we assume that our LCS are deadlock-free. In case of existence of deadlock states, Lemma 7 does not hold. However, it is straightforward to modify our algorithm to deal with deadlock. This follows from the fact that, we can use the reachability algorithm in [AJ96b] in order to check reachability of deadlock states.

A work closely related to this is [BE99]. In fact, our work can been as a generalization of the ideas presented in [BE99]. More precisely, in [BE99], a model of PLCS is considered where at each state either one message lost or an non-lossy transition is performed. The probability λ of losing messages is assumed to be at least 0.5. Under this semantics, it is proved that for an PLCS the set $\{(\mathtt{s}, \mathtt{w}) \mid |\mathtt{w}| = 0\}$ is an attractor. The decidability of reachability follows then in a similar manner to Section 5. Also, in [BE99], in contrast to the model of LCS presented in this paper, loss transitions are explicit. Therefore, the product of the transition system generated by an LCS with an automaton (Section 7), might not be equivalent to the transition system generated by any other LCS. In fact, under this semantics, it is undecidable whether the set of computations of a PLCS is accepted by a finite state automaton with probability one. To overcome this difficulty, it is assumed in [BE99] that the given automaton accepts an ω-language which is closed under stuttering.

Acknowledgement The work of the first author was partially supported by the European project ADVANCE, Contract No. IST – 1999-29082.

References

[ABIJ00] Parosh Aziz Abdulla, Christel Baier, Purushothaman Iyer, and Bengt Jonsson. Reasoning about probabilistic lossy channel systems. In C. Palamidessi, editor, *Proc. CONCUR 2000,* 11^{th} *Int. Conf. on Concurrency Theory*, volume 1877 of *Lecture Notes in Computer Science*, 2000.

[AJ96a] Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996.

[AJ96b] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996.

[BE99] C. Baier and B. Engelen. Establishing qualitative properties for probabilistic lossy channel systems. In Katoen, editor, *ARTS'99, Formal Methods for Real-Time and Probabilistic Systems, 5th Int. AMAST Workshop*, volume 1601 of *Lecture Notes in Computer Science*, pages 34–52. Springer Verlag, 1999.

[Boc78] G. V. Bochman. Finite state description of communicating protocols. *Computer Networks*, 2:361–371, 1978.

[BS03] N. Bertrand and Ph. Schnoebelen. Model checking lossy channels systems is probably decidable. In *Proc. FOSSACS03, Conf. on Foundations of Software Science and Computation Structures*, 2003.

[BZ83] D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 2(5):323–342, April 1983.

[CFI96] Gérard Cécé, Alain Finkel, and S. Purushothaman Iyer. Unreliable channels are easier to verify than perfect channels. *Information and Computation*, 124(1):20–31, 10 January 1996.

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

[KSK66] J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains.* D Van Nostad Co., 1966.

[Tho90] W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Methods and Semantics*, pages 133–192, 1990.