Decidability of Simulation and Bisimulation between Lossy Channel Systems and Finite State Systems*

(Extended Abstract)

Parosh Aziz Abdulla   Mats Kindahl

Dept. of Computer Systems, P.O. Box 325, 751 05 Uppsala, Sweden

E-mail: {parosh, matkin}@docs.uu.se

Abstract. We consider the verification of a class of infinite-state systems called *lossy channel systems*, which consist of finite-state processes communicating via unbounded but lossy FIFO channels. This class is able to model several interesting protocols, such as HDLC, the Alternating Bit Protocol, and other Sliding Window protocols. In earlier papers we have considered the decidability of various temporal properties for lossy channel systems. In this paper we study simulation and bisimulation relations between lossy channel systems and finite transition systems. More precisely, we show the decidability of (1) whether a state in a finite transition system *simulates* a state in a lossy channel system, and conversely, (2) whether a state in a finite transition system is *bisimilar* to a state in a lossy channel system, and (3) whether a state in a finite transition system *weakly simulates* a state in a lossy channel system. Furthermore, we show the undecidability of the following problems: (1) whether a state in a lossy channel system *weakly simulates* a state in a finite transition system, and (2) Whether a state in a finite transition system is *weakly bisimilar* to a state in a lossy channel system.

1 Introduction

Traditional approaches to automated verification model programs as finite-state systems. A program is described by an explicit representation of its state space. An obvious limitation of such models is that systems with infinitely many states, e.g. programs that operate on data from unbounded domains, fall beyond their capabilities.

We consider a class of infinite-state systems which consist of programs operating on unbounded FIFO channels. These programs have been used to model communicating finite-state processes, such as communication protocols [BZ83, Boc78]. The state space is infinite due to the unboundedness of the channels. It is well-known that such systems possess the same computational power as Turing Machines, and hence most verification problems are undecidable for them.

In this paper, we study a variant of this class where the FIFO channels are unreliable, in that they may nondeterministically lose messages. We call such systems *lossy channel systems*. In spite of the lossiness assumption, we can still model many interesting systems, e.g. link protocols such as the Alternating Bit Protocol [BSW69] and HDLC [ISO79]. These protocols and others are designed to operate correctly even in the case that the FIFO channels are faulty and may lose messages.

* Supported in part by ESPRIT BRA project No. 6021 (REACT), and by the Swedish Research Council for Engineering Sciences (TFR) under contract No. 92-814.

We have previously worked with verification of lossy channel systems. In [AJ93] we showed that several verification problems such as reachability, safety properties, and a simple class of eventuality properties are decidable. The decidability of the eventuality properties was also proven independently by Finkel [Fin94]. In [AJ94] we showed that most of the temporal properties that were not proven decidable in [AJ93] are in fact undecidable. More precisely, we showed that both model checking and verifying eventuality properties under the assumption that the channels are fair are undecidable.

In this work, we compare lossy channel systems with finite-state transition systems. We investigate the decidability of simulation and bisimulation relations between these two classes of systems. More precisely, we show the following problems to be decidable:

- whether a state in a finite transition system simulates a state in a lossy channel system, and conversely,
- whether a state in a finite transition system is bisimilar to a state in a lossy channel system, and
- whether a state in a finite transition system weakly simulates a state in a lossy channel system.

Furthermore, we show undecidability of the following problems:

- whether a state in a lossy channel system weakly simulates a state in a finite transition system, and
- whether a state in a finite transition system is weakly bisimilar to a state in a lossy channel system.

The idea of our verification algorithms is that we find *finite representations* for the simulation and bisimulation relations. Although these relations in general contain infinitely many pairs of states they can be decided by examining only a finite number of states, namely the states belonging to their finite representations. For example, given that a state of a finite transition system does not simulate a state of a lossy channel system, then it is also the case that it does not simulate any other "larger" state of the lossy channel system. A state is "larger" than another if the states differ only in that the content of each channel in the first state is a (not necessarily contiguous) superstring of the content of the same channel in the second state. This means that the "non-simulation" relation is "upward closed" with respect to the "larger than" ordering on states. To decide whether a state of a finite transition system simulates a state of a lossy channel system, we systematically generate all pairs of states of the two systems which belong to the non-simulation relation. Although non-simulation is *a priori* unbounded, two facts make it possible to represent it in a finite manner. The first fact is that (by the above reasoning) we do not have to analyze a state for which we have already analyzed a "smaller" state. The second fact is that by a result in language theory (Higman's theorem) only a finite number of states can be generated if we discard states that have smaller variants.

Related Work Algorithmic verification methods have recently been developed for several classes of infinite-state systems. Examples include certain types of real-time systems that operate on clocks [ACD90, Yi91, Čer92], data-independent systems [JP93, Wol86], systems with many identical processes [CG87, GS92, SG90], context-free processes ([BS92, CHS92, CHM93]), and Petri nets ([Jan90]).

Considerable attention has been paid to the problem of analyzing systems that communicate over perfect unbounded FIFO channels. All interesting verification problems for these systems are in general undecidable, since the channels may be used to simulate the tape of a Turing Machine [BZ83]. Decidability results have been obtained for limited subclasses. Most problems are decidable if the channel alphabets are of size one (in which case the system may be simulated by Petri Nets [KM69, RY86]), or if the language of each channel is bounded (in which case the system becomes finite-state [GGLR87, CF87]).

Algorithms for partial verification, which may or may not succeed in analyzing a given system, have been developed by Purushotaman and Peng [PP91] and by Brand and Joyner [BZ83]. These works do not characterize a class of systems for which their method works. Finkel [Fin88] presents a limited class of systems for which verification is decidable. This class does not cover e.g. the Alternating Bit protocol. Sistla and Zuck [SZ91] present a verification procedure for reasoning about a certain set of temporal properties over systems with FIFO channels. The method is not powerful enough to reason about arbitrary finite transition systems.

Outline In Section 2 we introduce lossy channel systems and finite transition systems. In Section 3 we define simulation and bisimulation between lossy channel systems and finite transition systems. In Section 4 and Section 5 we describe algorithms to decide simulation preorder. In Section 6 we show decidability of strong bisimulation. In Section 7 we consider the decidability of weak bisimulation and weak simulation.

2 Systems with Lossy Channels

In this section we present the basic definition of finite-state systems with unbounded but lossy FIFO channels. Intuitively such a system has two parts: a control part and a channel part. The channel part consists of a set of channels, each of which may contain a sequence of messages taken from a finite alphabet. The control part is a finite-state labeled transition system. Typically, the finite-state part models the total behavior of a number of processes that communicate over the channels. Each transition of the control part represents the performance of both an observable interaction with the environment of the system, and an operation on the channels. This operation may either be empty, remove a message from the head of a channel, or insert a message at the end of a channel. In addition, a channel can nondeterministically lose messages at any time. Message losses are not observed externally.

For a set M we use M^* to denote the set of finite strings of elements in M. For $x, y \in M^*$ we use $x \bullet y$ to denote the string resulting from the concatenation of the strings x and y. The empty string is denoted by ε. If $x \neq \varepsilon$, then *first*(x) (*last*(x)) denotes the first (last) element of x. For sets C and M, a ***string vector*** w from C to M is a function $C \mapsto M^*$. For a string vector w from C to M we use $w\,[c := x]$ to denote a string vector w' such that $w'(c) = x$ and $w'(c') = w(c')$ for all $c' \neq c$. The string vector that maps all elements in C to the empty string is denoted by ε.

Definition 2.1. A *Lossy Channel System* $\mathcal{L}$ is a tuple $\langle S, A, C, M, \delta \rangle$, where

S is a finite set of *control states*,

A is a finite set of *labels*,

C is a finite set of *channels*,

M is a finite set of *messages*,

δ is a finite set of *transitions*. Each transition is a tuple of the form $\langle s_1, \alpha, \lambda, s_2 \rangle$, where s_1 and s_2 are control states, $\lambda \in \Lambda$ is a *label*, and α is an operation of one of the forms:

- $c!m$, where $c \in C$ and $m \in M$,
- $c?m$, where $c \in C$ and $m \in M$, or
- the empty operation e. □

Intuitively, the control part of the lossy channel system $\langle S, \Lambda, C, M, \delta \rangle$ is an ordinary labeled transition system with states S and transitions δ. The channel part is represented by the set C of channels, each of which may contain a string of messages in M. The set Λ denotes a set of observable interactions with the environment. A transition $\langle s_1, \alpha, \lambda, s_2 \rangle$ represents a change of control state from s_1 to s_2, while performing the observable interaction λ, and at the same time modifying the contents of the channels according to α, where

- $c!m$ appends the message m to the end of channel c,
- $c?m$ removes the message m from the head of channel c, and
- e does not change the contents of the channels.

The operational behavior of a lossy channel system is defined by formalizing the intuitive behavior of the system as a labeled transition system with infinitely many states. Let $\mathcal{L}$ be the lossy channel system $\langle S, \Lambda, C, M, \delta \rangle$. A *global state* γ of $\mathcal{L}$ is a pair $\langle s, w \rangle$, where $s \in S$ and w is a string vector from C to M. To model the fact that message losses are considered as unobservable events we introduce the silent event τ such that $\tau \notin \Lambda$, and define $\hat{\Lambda}$ to be the set $\Lambda \cup \{\tau\}$.

Definition 2.2. We shall define a relation $\longrightarrow$ as a set of tuples of the form $\langle \gamma, \lambda, \gamma' \rangle$ where γ and γ' are global states and $\lambda \in \hat{\Lambda}$. Let $\gamma \xrightarrow{\lambda} \gamma'$ denote $\langle \gamma, \lambda, \gamma' \rangle \in \longrightarrow$. We define $\longrightarrow$ to be the smallest set such that

1. if $\langle s, c!m, \lambda, s' \rangle \in \delta$, then $\langle s, w \rangle \xrightarrow{\lambda} \langle s', w[c := w(c) \bullet m] \rangle$, i.e. the control state changes from s to s', the observable action λ is performed, and m is appended to the end of channel c.
2. If $\langle s, c?m, \lambda, s' \rangle \in \delta$, then $\langle s, w[c := m \bullet w(c)] \rangle \xrightarrow{\lambda} \langle s', w \rangle$, i.e. the control state changes from s to s', the observable action λ is performed, and m is removed from the head of channel c. Notice that if $w(c) = \varepsilon$, or if $first(w(c)) \neq m$, then the transition $\langle s, c?m, \lambda, s' \rangle$ cannot be performed from the global state $\langle s, w \rangle$.
3. If $w(c) = x \bullet m \bullet y$, then $\langle s, w \rangle \xrightarrow{\tau} \langle s, w[c := x \bullet y] \rangle$, i.e. the message m in channel c is lost without changing the control state. The τ indicates that the event is not observable to the environment.
4. If $\langle s, e, \lambda, s' \rangle \in \delta$, then $\langle s, w \rangle \xrightarrow{\lambda} \langle s', w \rangle$, i.e. the control state changes from s to s' and the observable action λ is performed. The contents of the channels are not affected. □

For global states γ and γ', and sequence $\sigma \in \Lambda^*$, where $\sigma = \lambda_1 \lambda_2 \ldots \lambda_n$, we let $\gamma \stackrel{\sigma}{\Longrightarrow} \gamma'$ denote

$$\gamma \xrightarrow{\tau}{}^{*} \gamma_1 \xrightarrow{\lambda_1} \gamma_1' \xrightarrow{\tau}{}^{*} \gamma_2 \xrightarrow{\lambda_2} \gamma_2' \cdots \gamma_n \xrightarrow{\lambda_n} \gamma_n' \xrightarrow{\tau}{}^{*} \gamma'$$

for some global states $\gamma_1, \gamma_1', \gamma_2, \gamma_2', \ldots, \gamma_n, \gamma_n'$.

We say that a control state s_2 is *potentially reachable* from a control state s_1 through a label λ (written $s_1 \xrightarrow{\lambda}_P s_2$) if $\langle s_1, \alpha, \lambda, s_2 \rangle \in \delta$ for some α. We use $(s_1 \xrightarrow{\lambda}_P)$ to denote the set $\{ s_2 \mid s_1 \xrightarrow{\lambda}_P s_2 \}$.

In this paper we study simulation and bisimulation relations between lossy channel systems and finite labeled transition systems.

Definition 2.3 (Finite Labeled Transition Systems). A *finite labeled transition system* (or simply a finite transition system) $\mathcal{T}$ is a tuple $\langle Q, \Lambda, T \rangle$, where

Q is a finite set of *states*,

Λ is a finite set of *labels*, and

T is a finite set of *transitions* of the form $\langle q_1, \lambda, q_2 \rangle$, where $q_1, q_2 \in S$ and $\lambda \in \Lambda$. □

We use $q \xrightarrow{\lambda} q'$ to denote $\langle q_1, \lambda, q_2 \rangle \in T$. Notice that $\tau \notin \Lambda$. Let $(q \xrightarrow{\lambda})$ denote the set $\{ q' \mid q \xrightarrow{\lambda} q' \}$.

3 Simulation and Bisimulation

In this section we introduce the notions of simulation and bisimulation between lossy channel systems and finite transition systems. The simulation relation will be studied in both directions, i.e. whether a state of a finite transitions simulates a global state of a lossy channel system and conversely.

Definition 3.1 (Simulation Preorder). For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T}$, a *simulation preorder* $\mathcal{S}$ between $\mathcal{L}$ and $\mathcal{T}$ is a set of pairs of the form $\langle \gamma, q \rangle$, where γ is a global state of $\mathcal{L}$ and q is a state of $\mathcal{T}$, such that if $\langle \gamma, q \rangle \in \mathcal{S}$ then, for all $\lambda \in \Lambda$:

- whenever $\gamma \stackrel{\lambda}{\Longrightarrow} \gamma'$, for some state γ', then there exists a q' such that $q \xrightarrow{\lambda} q'$ and $\langle \gamma', q' \rangle \in \mathcal{S}$. □

We say that γ *is simulated by* q, written $\gamma \sqsubseteq q$, if there exists a simulation $\mathcal{S}$ between $\mathcal{L}$ and $\mathcal{T}$ such that $\langle \gamma, q \rangle \in \mathcal{S}$.

In a similar manner we can define what it means for a state q of a finite transitions system to be *simulated by* a global state γ of a lossy channel system (written $q \sqsubseteq \gamma$), and what it means for γ to be *bisimilar* to q (written $\gamma \sim q$).

An alternative characterization of simulation preorder is given by the relation $\sqsubseteq_k$:

Definition 3.2. The relation $\sqsubseteq_k$ is defined for any natural number k as follows.

- $\gamma \sqsubseteq_0 q$, for all pairs γ and q.
- $\gamma \sqsubseteq_{k+1} q$, if for all $\lambda \in \Lambda$, whenever $\gamma \stackrel{\lambda}{\Longrightarrow} \gamma'$, for some state γ', then there exists a state q' such that $q \xrightarrow{\lambda} q'$ and $\gamma' \sqsubseteq_k q'$. □

In [Mil89] it is proved that for any finite-branching transition system $\sqsubseteq \, = \bigcap_k \sqsubseteq_k$. It is obvious that lossy channel systems and finite transition systems are finite-branching. This means that $\gamma \sqsubseteq q$ iff $\gamma \sqsubseteq_k q$ for each natural number k.

The relation $\sim_k$, for a natural number k, can be defined in a similar manner.

4 Deciding Simulation Preorder: LCS - FTS

In this section we describe and prove the correctness of an algorithm to decide whether a global state of a lossy channel system is simulated by a state of a finite transition system. The idea of the algorithm is that we compute the complement of $\sqsubseteq_k$, denoted $\not\sqsubseteq_k$, for $k = 1, 2, \ldots$, until we reach a point where $\not\sqsubseteq_k = \not\sqsubseteq_{k+1}$. As we will show later such a k always exists. We show that each $\not\sqsubseteq_k$ can be characterized by a finite set of pairs of states which we call the *finite representation* of $\not\sqsubseteq_k$. To check whether a pair of states belongs to $\not\sqsubseteq_k$, it suffices to inspect the finite representation of $\not\sqsubseteq_k$. We start by computing $\not\sqsubseteq_1$ and use Algorithm 1 to generate $\not\sqsubseteq_{k+1}$ from $\not\sqsubseteq_k$. When we reach the point where $\not\sqsubseteq_k = \not\sqsubseteq_{k+1}$, we know that $\not\sqsubseteq = \not\sqsubseteq_k$. To check $\gamma \sqsubseteq q$, we simply examine the finite representation of $\not\sqsubseteq$ which is identical to the finite representation of $\not\sqsubseteq_k$.

First we study some properties of strings and the $\sqsubseteq$ relation which will help us to construct our algorithms.

For a finite set M and $x_1, x_2 \in M^*$, let $x_1 \preceq x_2$ denote that x_1 is a (not necessarily contiguous) substring of x_2. If w_1 and w_2 are string vectors from C to M, then $w_1 \preceq w_2$ denotes that $w_1(c) \preceq w_2(c)$ for any $c \in C$. Let $\langle s_1, w_1 \rangle \preceq \langle s_2, w_2 \rangle$ denote that $s_1 = s_2$ and $w_1 \preceq w_2$.

We will use the following theorem to construct finite representations of relations and to prove termination of our algorithms.

Theorem 4.1 (Higman's theorem). *Let M be a finite set. There is no infinite sequence $w_1, w_2, \ldots$ of strings in M^* such that $w_i \not\preceq w_j$, for any $i < j$.*

It is straightforward to generalize Higman's theorem to sequences of string vectors and global states.

Let W be a set of string vectors. We say that $w \in W$ is a *minimal element* of W if for any $w' \in W$, we have $w' \not\preceq w$. We denote the set of minimal elements of W by $\min(W)$. By Higman's Theorem we know that $\min(W)$ is finite. Suppose that W is upward closed, i.e. if $w \in W$ and $w \preceq w'$ then $w' \in W$. We say that the set V is a *finite representation* of W if V is finite and $\min(W) \subseteq V$. Since $\min(W)$ is finite there exits a finite representation V for each upward closed set W. The set V is a representation of W in the sense that $w \in W$ if and only if there is a $w' \in V$ such that $w' \preceq w$, or equivalently W is exactly the set whose elements are larger than or equal to an element of V. This means that V characterizes a unique upward closed set, namely W. The notions of minimality and finite representation can easily be generalized to global states.

For string vectors $w_1, \ldots, w_n$ we define the set of *minimal upper bounds* to be the minimal elements of the set $\{ v \mid (w_1 \preceq v) \wedge \cdots \wedge (w_n \preceq v) \}$, and denote it $\sqcup\langle w_1, \ldots, w_n \rangle$. Observe that $\sqcup\langle w_1, \ldots, w_n \rangle$ is always finite and non-empty. It is easy to see that $\sqcup\langle w_1, \ldots, w_n \rangle$ is effectively computable for any $w_1, \ldots, w_n$. For sets of string vectors $W_1, \ldots, W_n$, let $\sqcup(W_1 \times \cdots \times W_n)$ denote the union of all $\sqcup\langle w_1, \ldots, w_n \rangle$ such that $\langle w_1, \ldots, w_n \rangle \in W_1 \times \cdots \times W_n$.

Proposition 4.2. *For any string vectors $w_1, \ldots, w_n, w$, and v*

$$(w_1 \preceq v) \wedge \cdots \wedge (w_n \preceq v) \supset \exists w \in \sqcup\langle w_1, \ldots, w_n \rangle . (w \preceq v)$$

In the following lemmas we show that $\sqsubseteq_k$ is downward closed, and consequently $\not\sqsubseteq_k$ is upward closed.

Lemma 4.3. *For any state q in a finite transition system and global states γ_1 and γ_2 in a lossy channel system we have* $(\gamma_1 \sqsubseteq_k q) \wedge (\gamma_2 \preceq \gamma_1) \supset (\gamma_2 \sqsubseteq_k q)$.

Corollary 4.4. $(\gamma_1 \not\sqsubseteq_k q) \wedge (\gamma_1 \preceq \gamma_2) \supset (\gamma_2 \not\sqsubseteq_k q)$.

In fact the above lemma and corollary can be shown for $\sqsubseteq$ and $\not\sqsubseteq$ in a similar manner.

From Corollary 4.4 we conclude that $\not\sqsubseteq$ and $\not\sqsubseteq_k$ are upward closed and consequently have finite representations.

When computing the finite representation of $\not\sqsubseteq_{k+1}$ from the finite representation of $\not\sqsubseteq_k$, we go backwards from pairs of states which belong to $\not\sqsubseteq_k$ to find pairs of states which belong to $\not\sqsubseteq_{k+1}$. We need to define a notion of "backward reachability" for global states of lossy channels systems. More precisely, we need to find a finite representation of the infinite set $\{\gamma' \mid \gamma' \stackrel{\lambda}{\Longrightarrow} \gamma\}$. To achieve that we define a new transition relation $\leadsto$ on global states (Definition 4.5), such that (Lemma 4.6 and Lemma 4.7) the set $\{\gamma' \mid \gamma \stackrel{\lambda}{\leadsto} \gamma'\}$ is a finite representation of $\{\gamma' \mid \gamma' \stackrel{\lambda}{\Longrightarrow} \gamma\}$.

Definition 4.5. Let $\mathcal{L} = \langle S, A, C, M, \delta\rangle$ be a lossy channel system. Define $\leadsto$ to be the smallest relation on global states such that:

1. if $\langle s_2, c!m, \lambda, s_1\rangle \in \delta$ then $\langle s_1, w[c := w(c) \bullet m]\rangle \stackrel{\lambda}{\leadsto} \langle s_2, w\rangle$,
2. if $\langle s_2, c!m, \lambda, s_1\rangle \in \delta$, $w(c) \neq \varepsilon$, and $last(w(c)) \neq m$, then $\langle s_1, w\rangle \stackrel{\lambda}{\leadsto} \langle s_2, w\rangle$,
3. if $\langle s_2, c!m, \lambda, s_1\rangle \in \delta$, and $w(c) = \varepsilon$, then $\langle s_1, w\rangle \stackrel{\lambda}{\leadsto} \langle s_2, w\rangle$,
4. if $\langle s_2, c?m, \lambda, s_1\rangle \in \delta$ then $\langle s_1, w\rangle \stackrel{\lambda}{\leadsto} \langle s_2, w[c := m \bullet w(c)]\rangle$,
5. if $\langle s_2, e, \lambda, s_1\rangle \in \delta$, then $\langle s_1, w\rangle \stackrel{\lambda}{\leadsto} \langle s_2, w\rangle$. □

We can capture the relation between $\leadsto$ and $\Longrightarrow$ in the following two lemmas:

Lemma 4.6. *For any global states γ_1, γ_2, and label λ,* $\gamma_1 \stackrel{\lambda}{\leadsto} \gamma_2 \supset \gamma_2 \stackrel{\lambda}{\Longrightarrow} \gamma_1$.

Lemma 4.7. *For any global states γ_1, γ_2, and γ_3, and label λ*

$$(\gamma_1 \stackrel{\lambda}{\Longrightarrow} \gamma_2) \wedge (\gamma_3 \preceq \gamma_2) \quad \supset \quad \exists \gamma_4. (\gamma_4 \preceq \gamma_1) \wedge (\gamma_3 \stackrel{\lambda}{\leadsto} \gamma_4)$$

We now proceed by describing how to compute a finite representation of the initial set $\not\sqsubseteq_1$. From the definition of $\sqsubseteq_k$ we know that

$$\not\sqsubseteq_1 = \{ \langle\langle s, w\rangle, q\rangle \mid \exists\lambda. (((\langle s, w\rangle \stackrel{\lambda}{\Longrightarrow}) \neq \emptyset) \wedge ((q \stackrel{\lambda}{\longrightarrow}) = \emptyset) \}$$

It can easily be seen that minimal elements of this set have either all channels empty, or all but one channel is empty and the remaining channel contains exactly one message. This means that the effectively computable set

$$\{ \langle\langle s, w\rangle, q\rangle \mid \exists\lambda. (((\langle s, w\rangle \stackrel{\lambda}{\Longrightarrow}) \neq \emptyset) \wedge ((q \stackrel{\lambda}{\longrightarrow}) = \emptyset) \wedge (\sum_{c\in C} |w(c)| \leq 1) \}$$

is a finite representation of $\not\sqsubseteq_1$.

To compute $\not\sqsubseteq_{k+1}$ given $\not\sqsubseteq_k$ we use Algorithm 1. It inputs a lossy channel system $\mathcal{L} = \langle S, \Lambda, C, M, \delta \rangle$, a finite transition system $\mathcal{T} = \langle Q, \Lambda, T \rangle$, and a finite representation W of $\not\sqsubseteq_k$, and produces a finite representation N of $\not\sqsubseteq_{k+1}$. To simplify the presentation of the algorithm, we assume that W is given as a collection of disjoint sets $W_{s,q}$, where $s \in S$ and $q \in Q$. A set $W_{s,q}$ contains a finite representation of the set of string vectors $\{ w \mid \langle s, w \rangle \not\sqsubseteq_k q \}$. The output set N is also given as a collection of disjoint sets $N_{s,q}$, where $N_{s,q}$ contains a finite representation of the set of string vectors $\{ w \mid \langle s, w \rangle \not\sqsubseteq_{k+1} q \}$.

Name: NEXTSIM$_1$
Input: An LCS $\mathcal{L} = \langle S, \Lambda, C, M, \delta \rangle$,
an FTS $\mathcal{T} = \langle Q, \Lambda, T \rangle$, and
a finite representation W of $\not\sqsubseteq_k$.
Output: A finite representation N of $\not\sqsubseteq_{k+1}$.
Algorithm:

```
for each s ∈ S and q ∈ Q do
    N_{s,q} ← W_{s,q}
    for each λ ∈ Λ and s' ∈ (s —λ→_P) do
        let {q_1,...,q_n} denote the set (q —λ→)
        V ← ⊔(W_{s',q_1} × ··· × W_{s',q_n})
        for each v ∈ V do
            N_{s,q} ← N_{s,q} ∪ { w | ⟨s',v⟩ ~λ~> ⟨s,w⟩ }
        done
    done
done
```

Algorithm 1 Computing $\not\sqsubseteq_{k+1}$ from $\not\sqsubseteq_k$

The idea of the algorithm is that we start with pairs of states in $\not\sqsubseteq_k$, and go "backwards" to find pairs of states in $\not\sqsubseteq_{k+1}$. We know that $\langle s, w \rangle \not\sqsubseteq_{k+1} q$ if and only if there is a $\lambda \in \Lambda$ and w' such that

$$\langle s, w \rangle \stackrel{\lambda}{\Longrightarrow} \langle s', w' \rangle \tag{1}$$

$$\langle s', w' \rangle \not\sqsubseteq_k q_1, \ldots, \langle s', w' \rangle \not\sqsubseteq_k q_n \tag{2}$$

where $\{q_1, \ldots, q_n\}$ is the set $(q \stackrel{\lambda}{\longrightarrow})$. To find a string vector w' satisfying (2) we consider the sets W_{s',q_i} for $1 \leq i \leq n$. Suppose that $w_i \in W_{s',q_i}$. We know that $\langle s', w_i \rangle \not\sqsubseteq_k q_i$. Since $\not\sqsubseteq_k$ is upward closed, we know that for any $v \in \sqcup\langle w_1, \ldots, w_n \rangle$ we have $\langle s', v \rangle \not\sqsubseteq_k q_i$ and hence v satisfies (2). From the definition of $\rightsquigarrow$, it follows that (1) will be satisfied by any w such that $\langle s', v \rangle \stackrel{\lambda}{\rightsquigarrow} \langle s, w \rangle$.

The formal proof of correctness of the algorithm is given by the following lemmas.

Lemma 4.8. *If w is added to $N_{s,q}$, then $\langle s, w \rangle \not\sqsubseteq_{k+1} q$.*

Proof. Assume that w is added to $N_{s,q}$. If $w \in W_{s,q}$, we know that $\langle s, w \rangle \not\sqsubseteq_k q$ and hence $\langle s, w \rangle \not\sqsubseteq_{k+1} q$. Otherwise, there exists s' and v such that $\langle s', v \rangle \stackrel{\lambda}{\rightsquigarrow} \langle s, w \rangle$. Let $(q \stackrel{\lambda}{\longrightarrow})$ be of the form $\{q_1, \ldots, q_n\}$. From the algorithm we observe that there are $w_1, \ldots, w_n$ such that $v \in \sqcup\langle w_1, \ldots, w_n\rangle$, and $\langle s', w_1\rangle \not\sqsubseteq_k q_1, \ldots, \langle s', w_n\rangle \not\sqsubseteq_k q_n$. From Corollary 4.4 it follows that $\langle s', v\rangle \not\sqsubseteq_k q_1, \ldots, \langle s', v\rangle \not\sqsubseteq_k q_n$. From Lemma 4.6 we know that $\langle s, w\rangle \stackrel{\lambda}{\Longrightarrow} \langle s', v\rangle$. It follows that $\langle s, w\rangle \not\sqsubseteq_{k+1} q$. □

Lemma 4.9. *If $k \geq 1$ and $\langle s, w\rangle \not\sqsubseteq_{k+1} q$, then some w' such that $w' \preceq w$ will be added to $N_{s,q}$.*

Proof. Assume that $\langle s, w\rangle \not\sqsubseteq_{k+1} q$. If $\langle s, w\rangle \not\sqsubseteq_1 q$ then $w \in W_{s,q}$ and some $w' \preceq w$ will be added to $N_{s,q}$. Otherwise, there exists λ, w', and a non-empty set $\{q_1, \ldots, q_n\}$, such that $(q \stackrel{\lambda}{\longrightarrow}) = \{q_1, \ldots, q_n\}$, $\langle s, w\rangle \stackrel{\lambda}{\Longrightarrow} \langle s', w'\rangle$, and $\langle s', w'\rangle \not\sqsubseteq_k q_1, \ldots, \langle s', w'\rangle \not\sqsubseteq_k q_n$. As W is a finite representation of $\not\sqsubseteq_k$, then there are $w_1, \ldots, w_n \preceq w'$, such that $w_1 \in W_{s',q_1}, \ldots, w_n \in W_{s',q_n}$. From Proposition 4.2 we know that there exists $v \in \sqcup\langle w_1, \ldots, w_n\rangle$ such that $v \preceq w'$. From Lemma 4.7 we know that there is a w'' such that $w'' \preceq w$ and $\langle s', v\rangle \stackrel{\lambda}{\leadsto} \langle s, w''\rangle$. From the algorithm we observe that w'' will be added to $N_{s,q}$. □

Lemma 4.10. *If $k \geq 1$, then Algorithm 1 computes a finite representation N of $\not\sqsubseteq_{k+1}$.*

Proof. Follows from Lemma 4.8 and Lemma 4.9. □

> **Input:** An LCS $\mathcal{L} = \langle S, A, C, M, \delta\rangle$ and
> an FTS $\mathcal{T} = \langle Q, A, T\rangle$.
> **Output:** A finite representation W of $\not\sqsubseteq$
> **Algorithm:**
>
> $W \leftarrow \min(\not\sqsubseteq_1)$
> **repeat**
> $W_{\text{old}} \leftarrow W$
> $W \leftarrow \min(\text{NEXTSIM}_1(W))$
> **until** $W_{\text{old}} = W$

Algorithm 2 Main procedure

Theorem 4.11. *Given a finite transition system $\mathcal{T}$ and a lossy channel system $\mathcal{L}$, Algorithm 2 computes a finite representation W of $\not\sqsubseteq$.*

Proof. We observe that in the k:th iteration of the algorithm (where $k > 0$), W_{old} and W are finite representations of $\not\sqsubseteq_k$ and $\not\sqsubseteq_{k+1}$ respectively (easily proved using Lemma 4.10). This means that when the algorithm terminates, we have $\not\sqsubseteq_k = \not\sqsubseteq_{k+1}$, and consequently $\not\sqsubseteq_k = \not\sqsubseteq$. To prove termination we observe that the set W is minimized at each iteration. This implies that if the loop is executed infinitely many times then infinitely many minimal elements will be added to the set W, which is a contradiction according to Higman's theorem. □

Theorem 4.12. *$\gamma \sqsubseteq q$ is decidable.*

Proof. We use Algorithm 2 to generate a finite representation W of $\not\sqsubseteq$. We know that $\gamma \sqsubseteq s$ if and only if there is no γ' such that $\gamma' \preceq \gamma$ and $\langle\gamma', s\rangle \in W$. □

5 Deciding Simulation Preorder: FTS - LCS

In this section we describe an algorithm to check whether a state of a finite transition system is simulated by a global state of a lossy channel system. The idea of the algorithm is similar to the one in Section 4. The difference is that we compute $\sqsubseteq_k$ instead of $\not\sqsubseteq_k$. In contrast to the simulation preorder in Section 4, where we decide simulation preorder in the direction $\gamma \sqsubseteq q$, we here decide simulation preorder in the opposite direction, i.e. if $q \sqsubseteq \gamma$. This results in that $\sqsubseteq_k$ is upward closed instead of, as in Section 4, downward closed. We make use of the upward closedness of $\sqsubseteq_k$ to provide a finite representation of $\sqsubseteq_k$ in the same manner as we did with $\not\sqsubseteq_k$ in Section 4. In Algorithm 3, given a lossy channel system $\mathcal{L} = \langle S, A, C, M, \delta \rangle$, a finite transition system $\mathcal{T} = \langle Q, A, T \rangle$, and a finite representation of $\sqsubseteq_k$, we provide a finite representation of $\sqsubseteq_{k+1}$ as follows. For $s \in S$ and $q \in Q$, we give a finite representation $W_{s,q}$ of the set of string vectors $\{ w \mid q \sqsubseteq_k \langle s, w \rangle \}$. We say that a string vector w is an *answer* to a transition $q \xrightarrow{\lambda} q'$ if there exists a string vector v such that $\langle s, w \rangle \xRightarrow{\lambda} \langle s', v \rangle$ and $q' \sqsubseteq_k \langle s', v \rangle$. In the algorithm, we compute a finite representation of the answers to each transition $q \xrightarrow{\lambda} q'$ and store it in $V_{\lambda,q'}$. The relation $q \sqsubseteq_{k+1} \langle s, w \rangle$ holds if and only if w is an answer to all transitions from q, or equivalently if w is a minimal upper bound to a set containing at least one element from each $V_{\lambda,q'}$.

Name: NEXTSIM$_2$
Input: An LCS $\mathcal{L} = \langle S, A, C, M, \delta \rangle$,
an FSM $\mathcal{T} = \langle Q, A, T \rangle$, and
a finite representation W of $\sqsubseteq_k$.
Output: A finite representation N of $\sqsubseteq_{k+1}$.

for each $s \in S$ **and** $q \in Q$ **do**
  $N_{s,q} \leftarrow \{\varepsilon\}$
  for each $\lambda \in A$ **and** $q' \in (q \xrightarrow{\lambda})$ **do**
    $V_{\lambda,q'} \leftarrow \{ w \mid \exists v, s'. v \in W_{s',q'} \wedge \langle s', v \rangle \overset{\lambda}{\rightsquigarrow} \langle s, w \rangle \}$
  for each $\lambda \in A$ **and** $q' \in (q \xrightarrow{\lambda})$ **do**
    $N_{s,q} \leftarrow \sqcup(N_{s,q} \times V_{\lambda,q'})$

Algorithm 3 Function to calculate $\sqsubseteq_{k+1}$ from $\sqsubseteq_k$

The formal proof of correctness of the algorithm can be carried out in a similar manner to that of Algorithm 1.

6 Bisimulation

In this section we show the decidability of bisimulation between a global state of a lossy channel system and a state of a finite transition system. It is easy to see that the relation $\sim_k$ between a finite transition system and a lossy channel system is downward closed and hence that the relation $\not\sim_k$ is upward closed.

To decide $\sim$ we could use an approach similar to the one in Section 4, i.e. to generate $\not\sim_1, \not\sim_2, \ldots$, until we get $\not\sim_k = \not\sim_{k+1}$. However, we introduce a different algorithm, where we generate $\sim_1, \sim_2, \ldots$, and in fact prove that there is an upper bound on the number of iterations we need to perform. As we shall see, it suffices to compute $\sim_1, \sim_2, \ldots, \sim_{m^2+1}$, where m is the size of the finite transition systems, and then use a reachability algorithm described in [AJ93].

For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$, a global state γ of $\mathcal{L}$, and a state q of $\mathcal{T}$, we show (Theorem 6.4) that the problem of checking $\gamma \sim q$ is equivalent to the following two problems: (1) that $\gamma \sim_{m^2+1} q$, where m is the number of states of $\mathcal{T}$, and (2) that it is not possible to reach "bad" global states from γ. A global state is "bad" if it is not bisimilar to any state of $\mathcal{T}$. We show (Theorem 6.5 and Theorem 6.6) that both these problems are decidable.

From the theory of bisimulation for finite state systems, we know that a finite transition system can be minimized with respect to bisimulation. We consider therefore (without loss of generality) only minimal finite transition systems. This means in particular that for any different states q_1 and q_2 of such a system, we have $q_1 \not\sim q_2$. We define the *size* of a finite transition system to be the number of states in it. We use the following result.

Lemma 6.1. *For a minimal finite transition system $\mathcal{T}$ with size m, and states s_1 and s_2 in $\mathcal{T}$, $q_1 \sim q_2 \equiv q_1 \sim_{m^2} q_2$.*

For a lossy channel system $\mathcal{L}$ we use $\gamma_1 \stackrel{*}{\Longrightarrow} \gamma_2$ to denote that there exits $\sigma \in \Lambda^*$ such that $\gamma \stackrel{\sigma}{\Longrightarrow} \gamma'$. By $(\gamma_1 \stackrel{*}{\Longrightarrow})$ we mean the set $\{\, \gamma_2 \mid \gamma_1 \stackrel{*}{\Longrightarrow} \gamma_2 \,\}$. For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T} = \langle Q, \Lambda, T \rangle$ with size m, we define the set of *critical global states* (denoted $\mathcal{C}(\mathcal{L}, \mathcal{T})$) to be $\{\, \gamma \mid \exists q \in Q.\, \gamma \sim_{m^2+1} q \,\}$.

Lemma 6.2. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$ with size m, a global state γ of $\mathcal{L}$, and a state q of $\mathcal{T}$, we have $\gamma \sim q \supset (\gamma \stackrel{*}{\Longrightarrow}) \subseteq \mathcal{C}(\mathcal{L}, \mathcal{T})$.*

Lemma 6.3. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$ with size m, and a global state γ in $\mathcal{L}$, if $(\gamma \stackrel{*}{\Longrightarrow}) \subseteq \mathcal{C}(\mathcal{L}, \mathcal{T})$, then the set $\{\, \langle \gamma_1, q_1 \rangle \mid \gamma_1 \sim_{m^2+1} q_1 \wedge \gamma \stackrel{*}{\Longrightarrow} \gamma_1 \,\}$ is a bisimulation.*

Theorem 6.4. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$ with size m, a global state γ in $\mathcal{L}$, a state q in $\mathcal{T}$*

$$(\gamma \sim q) \equiv (\gamma \sim_{m^2+1} q) \wedge ((\gamma \stackrel{*}{\Longrightarrow}) \subseteq \mathcal{C}(\mathcal{L}, \mathcal{T}))$$

Proof. We prove both directions of the equivalence.

($\Longrightarrow$): Assume that $\gamma \sim q$. It is clear that $\gamma \sim_{m^2+1} q$. From Lemma 6.2 we get $(\gamma \stackrel{*}{\Longrightarrow}) \subseteq \mathcal{C}(\mathcal{L}, \mathcal{T})$.

($\Longleftarrow$): Assume that $\gamma \sim_{m^2+1} q$ and $(\gamma \stackrel{*}{\Longrightarrow}) \subseteq \mathcal{C}(\mathcal{L}, \mathcal{T})$. From Lemma 6.3 we know that the set $\{\, \langle \gamma_1, q_1 \rangle \mid \gamma_1 \sim_{m^2+1} q_1 \wedge \gamma \stackrel{*}{\Longrightarrow} \gamma_1 \,\}$ is a bisimulation. The result follows immediately from the fact that $\gamma \sim_{m^2+1} q$ and $\gamma \stackrel{*}{\Longrightarrow} \gamma$. □

Theorem 6.5. *For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T}$, we can effectively compute a finite representation of $\sim_k$ for any natural number k.*

Proof. The idea of the algorithm is similar to Algorithm 1. □

Theorem 6.6. *For a lossy channel system $\mathcal{L}$, and a finite representation of a set Γ of global states of $\mathcal{L}$, we can effectively compute a finite representation of the set* $\{ \gamma' \mid (\gamma' \overset{*}{\Longrightarrow} \gamma) \wedge (\gamma \in \Gamma) \}$.

Proof. The proof can be found in [AJ93]. □

Theorem 6.7. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$, a global state γ of $\mathcal{L}$, and a state q of $\mathcal{T}$, it is decidable to check $\gamma \sim q$.*

Proof. The proof follows from Theorem 6.4, Theorem 6.5, and Theorem 6.6. □

7 Weak Simulation and Bisimulation

In this section we consider the decidability of a variant of the simulation and bisimulation problems which we call weak simulation/bisimulation.

We consider a more general form of lossy channel systems and finite transition systems. For a lossy channel system $\mathcal{L} = \langle S, \Lambda, C, M, \delta \rangle$, we allow the transitions in δ to have the form $\langle s_1, \alpha, \lambda, s_2 \rangle$ where $\lambda \in \hat{\Lambda}$. This means that the transitions may be labeled by the silent event τ. The same holds for finite transition systems.

For $\mathcal{T} = \langle Q, \Lambda, T \rangle$, $q_1, q_2 \in Q$, and sequence $\sigma \in \Lambda^*$, where $\sigma = \lambda_1 \lambda_2 \ldots \lambda_n$, we let $q \overset{\sigma}{\Longrightarrow} q'$ denote

$$q \xrightarrow{\tau}{}^{*} q_1 \xrightarrow{\lambda_1} q_1' \xrightarrow{\tau}{}^{*} q_2 \xrightarrow{\lambda_2} q_2' \cdots q_n \xrightarrow{\lambda_n} q_n' \xrightarrow{\tau}{}^{*} q'$$

for some states $q_1, q_1', q_2, q_2', \ldots, q_n, q_n' \in Q$.

Definition 7.1 (Weak Simulation). For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T}$, a *weak simulation preorder* $\mathcal{S}$ between $\mathcal{L}$ and $\mathcal{T}$ is a set of pairs of the form $\langle \gamma, q \rangle$, where γ is a global state of $\mathcal{L}$ and q is a state of $\mathcal{T}$, such that if $\langle \gamma, q \rangle \in \mathcal{S}$ then, for all $\sigma \in \Lambda^*$:

- whenever $\gamma \overset{\sigma}{\Longrightarrow} \gamma'$, for some state γ', then there exists a q' such that $q \overset{\sigma}{\Longrightarrow} q'$ and $\langle \gamma', q' \rangle \in \mathcal{S}$. □

We say that γ *is weakly simulated by* q, written $\gamma \underset{\sim}{\sqsubset} q$, if there exists a weak simulation $\mathcal{S}$ between $\mathcal{L}$ and $\mathcal{T}$ such that $\langle \gamma, q \rangle \in \mathcal{S}$.

The relations $\underset{\sim}{\sqsubset}$ (in the reverse direction), $\underset{\sim_k}{\sqsubset}$ (in both directions), $\approx$, and $\approx_k$ are defined by making similar modifications to the definitions in Section 3.

To decide the relation $\underset{\sim}{\sqsubset}$ we generate $\underset{\sim_k}{\sqsubset}$, for $k = 0, 1, 2, \ldots$, until we obtain a k such that $\underset{\sim_k}{\sqsubset} = \underset{\sim_{k+1}}{\sqsubset}$. The idea of generating $\underset{\sim_{k+1}}{\not\sqsubset}$ from $\underset{\sim_k}{\not\sqsubset}$ is similar to that of generating $\not\sqsubset_{k+1}$ from $\not\sqsubset_k$ (Algorithm 1). We start from pairs of states which belong to $\underset{\sim_k}{\not\sqsubset}$ and go backwards to try to find pairs of states belonging to $\underset{\sim_{k+1}}{\not\sqsubset}$. The main difference between the two algorithms is that in the case of Algorithm 1, we need only to consider one step of the $\rightsquigarrow$ relation. However, to compute $\not\sqsubseteq_{\sim_{k+1}}$, we need to take the silent transitions into consideration, and therefore have to replace the relation $\overset{\lambda}{\rightsquigarrow}$, for $\lambda \in \Lambda$, by the relation $\underset{*}{\overset{\lambda}{\rightsquigarrow}}$, where $\gamma \underset{*}{\overset{\lambda}{\rightsquigarrow}} \gamma'$ is defined by $\gamma \overset{\tau}{\rightsquigarrow}^{*} \gamma_1 \overset{\lambda}{\rightsquigarrow} \gamma'$. Although the $\underset{*}{\rightsquigarrow}$ is *a priori* unbounded, we get decidability through the following result which we have shown in [AJ93] for lossy channel systems.

Theorem 7.2. *For a lossy channel system $\mathcal{L} = \langle S, \Lambda, C, M, \delta \rangle$ and a global state γ of $\mathcal{L}$, we can effectively compute a finite representation of $(\gamma \underset{*}{\overset{\lambda}{\rightsquigarrow}})$.*

We prove the undecidability of the problem of checking whether a state of a finite transition system is weakly simulated by a global state of a lossy channel system. The result is shown through a reduction from the following problem which we have previously shown to be undecidable for lossy channel systems.

For a lossy channel system $\mathcal{L}$, we define a *computation π of $\mathcal{L}$* as an infinite sequence $\gamma_1 \gamma_2 \ldots$ of global states in $\mathcal{L}$ such that for each $i = 1, 2, \ldots$ either (1) $\gamma_i \longrightarrow \gamma_{i+1}$ or (2) $\gamma_i = \gamma_{i+1}$, if there are no γ such that $\gamma_i \longrightarrow \gamma$.

Definition 7.3 (Recurrent State Problem). The *Recurrent State Problem* is defined as follows. Given a lossy channel system $\mathcal{L}$, a global state γ, and a control state s of $\mathcal{L}$, check whether $\mathcal{L}$ has a computation starting at γ and visiting s infinitely often. □

Theorem 7.4. *RSP is undecidable.*

The definition of RSP and the proof of its undecidability can be found in [AJ94].

Theorem 7.5. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$, a global state γ of $\mathcal{L}$, and a state q of $\mathcal{T}$, it is undecidable to check $q \sqsubseteq_{\sim} \gamma$.*

Proof. The proof is achieved through a reduction from RSP. □

We now proceed to prove undecidability of weak bisimulation between a lossy channel system and a finite transition system. The result is shown through a reduction from the following problem which we have previously shown to be undecidable for lossy channel systems.

Definition 7.6 (Global Reachability Problem). The *Global Reachability Problem (GRP)* is defined as follows. Given a lossy channel system $\mathcal{L}$, a global state γ, and a control state s of $\mathcal{L}$, check whether all computations of $\mathcal{L}$ starting at γ have the property that it is possible to reach s from any global state in the computation. □

Observe that the GRP property can be expressed as the temporal formula $\forall\Box\exists\Diamond\, s$. From [AJ94] we have the following theorem.

Theorem 7.7. *GRP is undecidable.*

Theorem 7.8. *For a lossy channel system $\mathcal{L}$, a finite transition system $\mathcal{T}$, a global state γ of $\mathcal{L}$, and a state q of $\mathcal{T}$, it is undecidable to check $\gamma \approx q$.*

Proof. The proof is achieved through a reduction from GRP in a similar manner to the proof of Theorem 7.5.

8 Conclusion and Future work

We have considered the decidability of simulation and bisimulation problems between lossy channel systems and finite transition systems. We intend to consider the corresponding problems for pairs of lossy channel systems. Early results indicate that simulation and bisimulation are decidable at least when one of the lossy channels systems is deterministic.

Although lossy channel systems are interesting in their own, we believe that many of the ideas presented here can be used to design verification algorithms for other classes of infinite-state systems such as Petri Nets, Relational Automata [Čer94], systems operating on unbounded graphs [JK95], etc. We hope this will lead us towards a general theory of verification for infinite-state systems.

Acknowledgements

We are grateful to Karlis Čerāns, Bengt Jonsson, and Yih-Kuen Tsay for comments and discussions.

References

[ACD90] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proc. 5^{th} IEEE Int. Symp. on Logic in Computer Science*, pages 414–425, Philadelphia, 1990.

[AJ93] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. In *Proc. 8^{th} IEEE Int. Symp. on Logic in Computer Science*, 1993. Accepted for Publication in Information and Computation.

[AJ94] Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. In Abiteboul and Shamir, editors, *Proc. ICALP '94*, volume 820 of *Lecture Notes in Computer Science*, pages 316–327. Springer Verlag, 1994.

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

[BS92] O. Burkart and B. Steffen. Model checking for context-free processes. In Cleaveland, editor, *Proc. CONCUR '92, Theories of Concurrency: Unification and Extension*, number 630 in Lecture Notes in Computer Science, pages 123–137. Springer Verlag, 1992.

[BSW69] K. Bartlett, R. Scantlebury, and P. Wilkinson. A note on reliable full-duplex transmissions over half duplex lines. *Communications of the ACM*, 2(5):260–261, 1969.

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

[Čer92] K. Čerāns. Decidability of bisimulation equivalence for parallel timer processes. In *Proc. Workshop on Computer Aided Verification*, volume 663 of *Lecture Notes in Computer Science*, pages 302–315, 1992.

[Čer94] K. Čerāns. Deciding properties of integral relational automata. In Abiteboul and Shamir, editors, *Proc. ICALP '94*, volume 820 of *Lecture Notes in Computer Science*, pages 35–46. Springer Verlag, 1994.

[CF87] A. Choquet and A. Finkel. Simulation of linear FIFO nets having a structured set of terminal markings. In *Proc. 8^{th} European Workshop on Applications and Theory of Petri Nets*, 1987.

[CG87] E. M. Clarke and O. Grumberg. Avoiding the state explosion problem in temporal logic model checking algorithms. In *Proc. 6^{th} ACM Symp. on Principles of Distributed Computing, Vancouver, Canada*, pages 294–303, 1987.

[CHM93] S. Christensen, Y. Hirshfeld, and F. Moller. Bisimulation equivalence is decidable for basic parallel processes. In *Proc. CONCUR '93, Theories of Concurrency: Unification and Extension*, pages 143–157, 1993.

[CHS92] S. Christensen, H. Hüttel, and C. Stirling. Bisimulation equivalence is decidable for all context-free processes. In W. R. Cleaveland, editor, *Proc. CONCUR '92, Theories of Concurrency: Unification and Extension*, pages 138–147, 1992.

[Fin88] A. Finkel. A new class of analyzable CFSMs with unbounded FIFO channels. In *Protocol Specification, Testing, and Verification VIII*, pages 1–12, Atlantic City, USA, 1988. IFIP WG 6.1, North-Holland.

[Fin94] A. Finkel. Decidability of the termination problem for completely specified protocols. *Distributed Computing*, 7(3), 1994.

[GGLR87] M.G. Gouda, E.M. Gurari, T.-H. Lai, and L.E. Rosier. On deadlock detection in systems of communicating finite state machines. *Computers and Artificial Intelligence*, 6(3):209–228, 1987.

[GS92] S. M. German and A. P. Sistla. Reasoning about systems with many processes. *Journal of the ACM*, 39(3):675–735, 1992.

[ISO79] ISO. Data communications – HDLC procedures – elements of procedures. Technical Report ISO 4335, International Standards Organization, Geneva, Switzerland, 1979.

[Jan90] P. Jančar. Decidability of a temporal logic problem for petri nets. *Theoretical Computer Science*, 74:71–93, 1990.

[JK95] Bengt Jonsson and Lars Kempe. Verifying safety properties of a class of infinite-state distributed algorithms, 1995. To appear in: Proc. CAV '95.

[JP93] B. Jonsson and J. Parrow. Deciding bisimulation equivalences for a class of non-finite-state programs. *Information and Computation*, 107(2):272–302, Dec. 1993.

[KM69] R.M. Karp and R.E. Miller. Parallel program schemata. *Journal of Computer and Systems Sciences*, 3(2):147–195, May 1969.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[PP91] W. Peng and S. Purushothaman. Data flow analysis of communicating finite state machines. *ACM Trans. on Programming Languages and Systems*, 13(3):399–442, July 1991.

[RY86] L.E. Rosier and H-C. Yen. Boundedness, empty channel detection and synchronization for communicating finite automata. *Theoretical Computer Science*, 44:69–105, 1986.

[SG90] Z. Shtadler and O. Grumberg. Network grammars, communication behaviours and automatic verification. In Sifakis, editor, *Proc. Workshop on Computer Aided Verification*, volume 407 of *Lecture Notes in Computer Science*, pages 151–165. Springer Verlag, 1990.

[SZ91] A.P. Sistla and L.D. Zuck. Automatic temporal verification of buffer systems. In Larsen and Skou, editors, *Proc. Workshop on Computer Aided Verification*, volume 575 of *Lecture Notes in Computer Science*. Springer Verlag, 1991.

[Wol86] Pierre Wolper. Expressing interesting properties of programs in propositional temporal logic (extended abstract). In *Proc. 13^{th} ACM Symp. on Principles of Programming Languages*, pages 184–193, Jan. 1986.

[Yi91] Wang Yi. CCS + Time = an interleaving model for real time systems. In Leach Albert, Monien, and Rodriguez Artalejo, editors, *Proc. ICALP '91*, volume 510 of *Lecture Notes in Computer Science*. Springer Verlag, 1991.