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

Parosh Aziz Abdulla Mats Kindahl†

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

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) trace inclusion and equivalence between lossy channel systems and finite transition systems, (2) whether a state in a finite transition system *simulates* a state in a lossy channel system, and conversely, (3) whether a state in a finite transition system is *bisimilar* to a state in a lossy channel system, (4) whether a state in a finite transition system *weakly simulates* a state in a lossy channel, and (5) whether a state in a finite transition system is *weakly bisimilar* to a state in a lossy channel system system. Furthermore, we show the undecidability of the problem of whether a state in a lossy channel system *weakly simulates* a state in a finite transition system.

*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.

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

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.

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:

- trace inclusion and equivalence,
- 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,

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

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.

Simulation in the other direction, i.e., whether a state of a lossy channel system simulates a state of a finite transition system, is decided using a *tableau method.* The idea of the algorithm is to build a tree which proves that the simulation preorder is satisfied. Due to Higman's theorem, each tableau is finite and there is only a finite number of tableaux; hence simulation preorder is decidable.

For bisimulation, things are not so simple, since the relation is neither upward closed nor downward closed. We use a slightly more complicated representation, where bisimilar states are characterized as differences between pairs of upward closed sets. Furthermore, it turns out that we only need to generate bisimilar states up to a certain predefined limit.

We show decidability of whether a state in finite transition system weakly simulates a state in a lossy channel system, by combining the reachability algorithm described in [AJ93] with the algorithm for computing strong simulation. We use a similar idea to outline an algorithm to decide weak bisimulation equivalence between a finite transition system and a lossy channel system.

For weak simulation preorder in the other direction, we show undecidability by reduction of the *Recurrent State Problem*, which has previously been show undecidable [AJ94].

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 describe some properties of strings. In Section 4 we recall some basic ideas of backward reachability analysis for lossy channel systems. In Section 5 we give algorithms to decide trace inclusion and equivalence. In Section 6 and Section 7 we describe algorithms to decide simulation preorder. In Section 8 we show decidability of strong bisimulation by giving an algorithm to decide it. In Section 9 we consider the decidability of weak bisimulation and weak simulation. Finally in Section 10 we give some conclusions and directions for future work.

2 Lossy Channel Systems

A lossy channel system consists of finite state processes communicating over FIFO channels. The difference from the traditional channel system is that we incorporate the lossiness of the channels into the model. A lossy channel system consists of two parts: a control part and a channel part. The channel part consists of a set of FIFO channels which the control part operates on through insertion and removal of messages. In addition, the channels may at any time lose messages anywhere inside the channels. The control part is a finite-state labelled transition system representing the behaviour of one or more finite-state processes operating on the channels.

With each transition in the control part, an operation on the channels is associated. In addition, each transition has a label taken from a finite set. The labels constitute the externally visible behaviour of the system.

For a finite alphabet M, let M^* denote the set of finite strings of elements from M. For strings $x_1, x_2 \in M^*$, let $x_1 \bullet x_2$ denote the concatenation of x_1 and x_2, and let ε denote the empty string. For any non-empty string x, let $\mathit{first}(x)$ denote the first element of x and $\mathit{last}(x)$ denote the last element of x. For sets C and M, a *string vector from C to M* is a mapping $C \to M^*$. For a string vector w from C to M let $w\,[c := x]$ 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 quintuple $\langle S, L, C, M, \Delta\rangle$ where

S is a finite set of *control states*,

L is a finite set of *labels*,

C is a finite set of *channels*,

M is a finite set of *messages*, and

Δ is a finite set of *transitions*. Each transition is a quadruple of the form $\langle s_1, \theta, l, s_2 \rangle$ where $s_1, s_2 \in S$, $l \in L$, and θ is an operation of the form

$c!m$ where $c \in C$ and $m \in M$, denoting an operation which appends the message m to the end of channel c;

$c?m$ where $c \in C$ and $m \in M$, denoting an operation which removes the message m from the head of channel c; or

e denoting an empty operation which does not alter the contents of the channels. □

The control part of the lossy channel system $\langle S, L, C, M, \Delta \rangle$ is a labelled transition system with states from S and transitions from the set Δ. The channel part is represented using the set C of channel names and the alphabet M of messages; each channel may contain a string of messages taken from M. Each transition in Δ is labelled with an observable event taken from the set L. A transition $\langle s_1, \theta, l, s_2 \rangle$ represents a change of control state from s_1 to s_2, while performing the interaction l with the environment and at the same time modifying the contents of the channels according to the operation θ.

Given a lossy channel system $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$ we define a *global state* σ of $\mathcal{L}$ as a pair $\langle s, w \rangle$ where $s \in S$ and w is a string vector from C to M. To model losses we introduce the silent event τ such that $\tau \notin L$. We formalize the behaviour of a lossy channel system as a labelled transition system with infinitely many states.

Definition 2.2. We will define a relation $\longrightarrow$ as a set of tuples of the form $\langle \sigma, l, \sigma' \rangle$ where σ and σ' are global states and $l \in L \cup \{\tau\}$. Let $\sigma_1 \xrightarrow{l} \sigma_2$ denote $\langle \sigma_1, l, \sigma_2 \rangle \in\longrightarrow$ We define $\longrightarrow$ to be the smallest set such that

1. If $\langle s_1, c!m, l, s_2 \rangle \in \Delta$, then $\langle s_1, w \rangle \xrightarrow{l} \langle s_2, w\,[c := w(c) \bullet m] \rangle$ for any string vector w, i.e., the control state changes from s_1 to s_2 and m is appended at the end of channel c.

2. If $\langle s_1, c?m, l, s_2 \rangle \in \Delta$, then $\langle s_1, w\,[c := m \bullet w(c)] \rangle \xrightarrow{l} \langle s_2, w \rangle$ for any string vector w, i.e., the control state changes from s_1 to s_2 and m is removed from the head of channel c.

3. If $w(c) = x \bullet m \bullet y$ for some $c \in C$, then $\langle s, w\rangle \xrightarrow{\tau} \langle s, w\,[c := x \bullet y]\rangle$ for any string vectors w, x, y, i.e., the message m in channel c is lost without changing the control state. The transition is labelled with τ, which means that losses are not visible to the environment.

4. If $\langle s_1, e, l, s_2\rangle \in \Delta$, then $\langle s_1, w\rangle \xrightarrow{l} \langle s_2, w\rangle$ for any string vector w, i.e., the control state changes from s_1 to s_2 without altering the contents of the channels. $\square$

We use $\sigma_1 \longrightarrow \sigma_2$ to denote that $\sigma_1 \xrightarrow{l} \sigma_2$ for some label l, and use $\xrightarrow{*}$ to denote the reflexive transitive closure of $\longrightarrow$. We extend $\longrightarrow$ to sequences of labels $\rho \in (L \cup \{\tau\})^*$ where $\rho = l_1 \ldots l_n$, and let $\sigma \xrightarrow{\rho} \sigma'$ denote

$$\sigma \xrightarrow{l_1} \sigma_1 \xrightarrow{l_2} \cdots \xrightarrow{l_n} \sigma'$$

for some global states $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$. We let $\sigma \overset{\rho}{\Longrightarrow} \sigma'$ denote $\sigma \xrightarrow{\rho'} \sigma'$ for some $\rho' = \tau^{i_1} l_1 \tau^{i_2} \ldots \tau^{i_n} l_n \tau^{i_{n+1}}$. We use $\sigma_1 \Longrightarrow \sigma_2$ to denote $\sigma_1 \overset{l}{\Longrightarrow} \sigma_2$ for some $l \in L$, and use $\overset{*}{\Longrightarrow}$ to denote the reflexive transitive closure of $\Longrightarrow$. We use $(\xrightarrow{l} \sigma)$ to denote the set $\{\sigma' : \sigma' \xrightarrow{l} \sigma\}$ and $(\sigma \xrightarrow{l})$ to denote the set $\{\sigma' : \sigma \xrightarrow{l} \sigma'\}$. The relations $(\sigma \xrightarrow{*})$, $(\sigma \overset{l}{\Longrightarrow})$, $\overset{*}{\Longrightarrow}$, etc, are defined in a similar manner. We extend $\longrightarrow$ to sets of global states in the obvious manner, so e.g. $(\xrightarrow{l} \Gamma) = \cup_{\sigma\in\Gamma}(\xrightarrow{l} \sigma)$.

We will compare lossy channel systems with finite transition systems. We recall the following standard definitions.

Definition 2.3. A *finite (labelled) transition system* $\mathcal{T}$ is a tuple $\langle Q, L, \longrightarrow\rangle$, where Q is a finite set of *states*, L is a set of *labels*, and $\longrightarrow \subseteq Q \times L \times Q$ is a *transition relation.* $\square$

We let $q_1 \xrightarrow{l} q_2$ denote $(q_1, l, q_2) \in \longrightarrow$. The relations $\longrightarrow$, $\xrightarrow{\rho}$, $\xrightarrow{*}$ $(q \longrightarrow)$, etc, are defined in a similar manner to the previous ones for lossy channel systems. $\square$

3 Properties of Strings

In this section we describe some properties of strings which we will use in our verification algorithms.

Let M be a finite set. For $x_1, x_2 \in M^*$, let $x_1 \preceq x_2$ denote that x_1 is a (not necessarily contiguous) substring of x_2. As an example, the strings `ab`,`acb`,`cabcd` are related as `ab` $\preceq$ `acb` $\preceq$ `cacbd`, while `acb` $\not\preceq$ `adb`. We extend this definition to string vectors in such a manner that if w_1 and w_2 are string vectors from C to M, then $w_1 \preceq w_2$ if $w_1(c) \preceq w_1(c)$, for all $c \in C$. Further, for global states $\langle s_1, w_1 \rangle$ and $\langle s_2, w_2 \rangle$ 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 use the following result to construct finite representations of relations, and proving termination of our algorithms. The result can be found in [Lot83], where it is attributed to Higman [Hig52].

Theorem 3.1 (Higman's Theorem). *For a finite set M, $\preceq$ is a* well quasi-ordering *on M^*. That is, for each infinite sequence $x_1, x_2, \ldots$ of strings over M, there are i, j such that $0 < i < j$ and $x_i \preceq x_j$.*

It is forward to generalize Higman's theorem to 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. We say that W is an ideal if $x \in W$ and $x \preceq x'$ imply $x' \in W$. Consider an ideal I. We say that the set V is a *finite representation* of I if V is finite and $min(I) \subseteq V$. Since $min(I)$ is finite there exits a finite representation V for each ideal I. The set V is a representation of I in the sense that $w \in I$ if and only if there is a $w' \in V$ such that $w' \preceq w$, or equivalently I is exactly the set whose elements are larger than or equal to an element of V. This means that V characterizes a unique ideal, namely I. For a set W of string vectors, we use $\uparrow W$ to be the ideal generated by W, i.e., the set $\{w';\ \exists w \in W.\, w \preceq w'\}$.

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

Proposition 3.2. *For any string vectors $w_1, \ldots, w_n, w$, and v*

$$(w_1 \preceq v) \wedge \cdots \wedge (w_n \preceq v) \quad \supset \quad \exists w \in (w_1 \sqcap \cdots \sqcap w_n) . (w \preceq v)$$

For sets of string vectors $W_1, \ldots, W_n$, we define the *greatest lower bound* $W_1 \sqcup \cdots \sqcup W_n$ to be the minimal elements of the set $W_1 \cup \cdots \cup W_n$.

The above definitions for string vectors can be generalized in the obvious manner to global states.

4 Backward Reachability Analysis

We introduce a "basic backward reachability step" for lossy channel system. We use the relation later in defining our simulation and bisimulation algorithms. The relation was introduced in [AJ93] and used for proving decidability of the reachability problem for lossy channel systems.

Definition 4.1. Let $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$ be a lossy channel system and let σ be a global state of $\mathcal{L}$. For a label $l \in L$ define $pre_l(\sigma)$ to be the smallest set of global states such that:

1. if $\langle s_2, c!m, l, s_1 \rangle \in \Delta$ and $\sigma = \langle s_1, w\,[c := w(c) \bullet m] \rangle$ for some string vector w and message m, then $\langle s_2, w \rangle \in pre_l(\sigma)$;
2. if $\langle s_2, c!m, l, s_1 \rangle \in \Delta$, $w(c) \neq \varepsilon$, $\sigma = \langle s_1, w \rangle$, and $last(w(c)) \neq m$, then $\langle s_2, w \rangle \in pre_l(\sigma)$,
3. if $\langle s_2, c!m, l, s_1 \rangle \in \Delta$, $\sigma = \langle s_1, w \rangle$, and $w(c) = \varepsilon$, then $\langle s_2, w \rangle \in pre_l(\sigma)$,
4. if $\langle s_2, c?m, l, s_1 \rangle \in \Delta$ and $\sigma = \langle s_1, w \rangle$, then $\langle s_2, w\,[c := m \bullet w(c)] \rangle \in pre_l(\sigma)$,
5. if $\langle s_2, e, l, s_1 \rangle \in \Delta$ and $\sigma = \langle s_1, w \rangle$, then $\langle s_2, w \rangle \in pre_l(\sigma)$. □

For global state σ we let $pre(\sigma)$ denote the set $\bigcup_{l \in L} pre_l(\sigma)$. The relation between $\Longrightarrow$ and *pre* is captured in the following two lemmas, the proof of which can be found in [AJ93].

Lemma 4.2. *For any global states σ_1, σ_2, and label l*

$$\sigma_1 \in pre_l(\sigma_2) \text{ implies } \sigma_1 \overset{l}{\Longrightarrow} \sigma_2$$

Lemma 4.3. *For any global states σ_1, σ_2, and σ_3, and label l*

$$(\sigma_1 \overset{l}{\Longrightarrow} \sigma_2) \wedge (\sigma_3 \preceq \sigma_2) \text{ implies } \exists \sigma_4 \in pre_l(\sigma_3).\, \sigma_4 \preceq \sigma_1$$

We extend the definition of pre to cover sets of global states by defining $pre_l(\Gamma)$ to be $min \bigcup_{\sigma\in\Gamma} pre_l(\sigma)$. We let pre_l^* (pre^*) to denote the reflexive transitive closure of pre_l (pre).

In [AJ93] we show that following lemma.

Lemma 4.4. *Given a global state σ of a lossy channel system and a label l, it is possible to compute a finite representation of $pre^*(\sigma)$ (and consequently for $pre_l^*(\sigma)$).*

5 Trace Inclusion and Trace Equivalence

In this section we show decidability of trace inclusion and equivalence between lossy channels systems and finite transition systems.

Given a global state σ of a lossy channel system $\mathcal{L}$, we define $traces(\sigma) = \{\rho : \exists\sigma'.\, \sigma \stackrel{\rho}{\Longrightarrow} \sigma'\}$. We extend the definition to cover sets of global states by defining $traces(\Gamma)$ as the set $\bigcup_{\sigma\in\Gamma} traces(\sigma)$. We observe that $\sigma \preceq \sigma'$ implies $traces(\sigma) \subseteq traces(\sigma')$. In a similar manner, we define the set $traces(q)$, where q is a state of a finite transition system.

The following result was shown in [AJ93].

Theorem 5.1. *Trace inclusion in the order $traces(\sigma) \subseteq traces(q)$ is decidable.*

Now, we concentrate on the problem of deciding trace inclusion in the other direction. The idea is to let the lossy channel system "follow" the finite transition system and object whenever the finite transition system makes an illegal move. By putting a bound on how far the lossy channel system has to follow the finite transition system we are able to construct a procedure for deciding trace inclusion.

We extend the relation $\preceq$ to sets of global states, so that $\Gamma_1 \preceq \Gamma_2$ if there for every $\sigma_1 \in \Gamma_1$ exists a $\sigma_2 \in \Gamma_2$ such that $\sigma_1 \preceq \sigma_2$. The new relation can easily be shown to be a well quasi-ordering.

Let $\mathcal{L} = \langle S, L, C, M, \Delta\rangle$ and let $\mathcal{T} = \langle Q, L, \longrightarrow\rangle$. To decide trace inclusion we use a tree-searching method. Each node in the tree consists of a state q and a *tracing set* G of global states. We denote a node $\mathbf{n}$ with state q and tracing set G by $(\mathbf{n}\colon q \subseteq_T G)$. The topmost node of the search tree is called the *root node*. We denote a search tree with root node $(\mathbf{n}: q \subseteq_T G)$ by $\textsc{Tree}[q \subseteq_T G]$.

Given the node $(\mathbf{n}: q \subseteq_T G)$, we let the *descendents of* $(\mathbf{n}: q \subseteq_T G)$ be the nodes $(\mathbf{n}_1: q_1 \subseteq_T G_1), \ldots, (\mathbf{n}_n: q_n \subseteq_T G_n)$, where $q \xrightarrow{l_i} q_i$ and

$$G_i = \{\sigma' : \exists \sigma \in G.\, \sigma \stackrel{l_i}{\Longrightarrow} \sigma'\}. \tag{1}$$

Observe that we actually only need to consider the *maximal* elements (w.r.t. $\preceq$), since $\Longrightarrow$ is monotone.

Each branch of the search tree terminates in a *leaf node*, which may be either *successful* or *unsuccessful*. A node $(\mathbf{n}: q \subseteq_T G)$ is unsuccessful if the tracing set G is empty. A node $(\mathbf{n}: q \subseteq_T G)$ is successful if: (1) q have no successors or, (2) there is $(\mathbf{n}': q' \subseteq_T G')$ above $\mathbf{n}$ in the tree such that $q = q'$ and $G' \preceq G$. The search tree is successful if all leaves are successful, otherwise the search tree is unsuccessful.

We show the correctenss of the algorithm in the following lemmas.

Lemma 5.2 (Soundness). *If* $\textsc{Tree}[q \subseteq_T G]$ *is a successful search tree, then* $traces(q) \subseteq traces(G)$.

Lemma 5.3 (Completeness). *If* $traces(q) \subseteq traces(\sigma)$*, then the search tree* $\textsc{Tree}[q \subseteq_T \{\sigma\}]$ *is successful.*

Lemma 5.4 (Finiteness). $\textsc{Tree}[q \subseteq_T G]$ *is finite for any finite set* G.

From Lemma 5.2, Lemma 5.3, and Lemma 5.4, we get the following.

Theorem 5.5. *Trace inclusion in the order* $traces(q) \subseteq traces(\sigma)$ *is decidable.*

From Theorem 5.1 and Theorem 5.5 it follows that

Theorem 5.6. *Trace equivalence between lossy channel systems and finite transitions systems is decidable.*

6 Simulation Preorder: $\mathcal{L} \sqsubseteq \mathcal{T}$

In this section we show it is decidable to check whether a global state σ in a lossy channel system is simulated by a state q in a finite transition system.

Definition 6.1 (Simulation Preorder). Let $\mathcal{L}$ be a lossy channel system, where the set of global states is Σ, and $\mathcal{T}$ a finite transition system, where the set of states is Q. A set $\mathcal{S} \subseteq \Sigma \times Q$ is a *simulation* if and only if for every pair $(\sigma, q) \in \mathcal{S}$, whenever $\sigma \stackrel{l}{\Longrightarrow} \sigma'$ for some global state state σ' and label l, there is a state q' such that $q \stackrel{l}{\longrightarrow} q'$ and $(\sigma', q') \in \mathcal{S}$. $\square$

We say that σ *is simulated by* q, written $\sigma \sqsubseteq q$, if there is a simulation $\mathcal{S}$ such that $(\sigma, q) \in \mathcal{S}$.

An alternative characterization of simulation preorder is the following.

Definition 6.2 (k-limited simulation preorder). We define σ to be a k-limited simulation of q ($\sigma \mathrel{\hat{\sqsubseteq}_k} q$) if and only if

- $k = 0$, or
- $k > 0$ and for every $l \in L$, whenever $\sigma \stackrel{l}{\Longrightarrow} \sigma'$ for some state σ', there is a state q' such that $q \stackrel{l}{\longrightarrow} q'$ and $\sigma' \mathrel{\hat{\sqsubseteq}_{k-1}} q'$. $\square$

We observe that $\hat{\sqsubseteq}_k$ is downward closed for any $k \geq 0$, i.e., if $\sigma \preceq \sigma'$ and $\sigma' \mathrel{\hat{\sqsubseteq}_k} q$, then $\sigma \mathrel{\hat{\sqsubseteq}_k} q$. It follows that $\hat{\not\sqsubseteq}_k$ is an ideal for any $k \geq 0$. A standard result for finite-image systems (and therefore also for lossy channel systems) is that $\sqsubseteq = \bigcap_k \hat{\sqsubseteq}_k$.

The idea of the algorithm is to compute $\hat{\not\sqsubseteq}_1, \hat{\not\sqsubseteq}_2, \ldots$ until we reach a fixpoint k such that $\hat{\not\sqsubseteq}_k = \hat{\not\sqsubseteq}_{k+1}$. Since $\hat{\not\sqsubseteq}_k$ is an ideal, we can use the finite representation suggested in Section 3 to represent each $\hat{\not\sqsubseteq}_k$. We start by computing an initial representation of $\hat{\not\sqsubseteq}_1$ and then iteratively compute the "next" $\hat{\not\sqsubseteq}_k$ by adding new state-pairs until we reach a fixpoint. Termination follows from the fact that if we can add new states infinitely, we have an infinite sequence of incomparable states, which contradicts Higman's theorem. To check $\sigma \sqsubseteq q$, we simply examine the finite representation of $\not\sqsubseteq$ which is identical to the finite representation of $\hat{\not\sqsubseteq}_k$.

Computing the initial set $min(\hat{\not\sqsubseteq}_1)$ From the definition of $\hat{\sqsubseteq}_k$ we know that $\hat{\not\sqsubseteq}_1$ is the set of all pairs $\langle \sigma, q\rangle$ such that $(\sigma \stackrel{l}{\Longrightarrow}) \neq \emptyset$ and $(q \stackrel{l}{\longrightarrow}) = \emptyset$, for some label l.

It is easy to see that the minimal elements of this set have empty channels or one channel with one message and the other channels empty. Hence

$$min(\hat{\not\sqsubseteq}_1) = \{\langle\langle s, w\rangle, q\rangle : \exists l \in L.\ (\sigma \stackrel{l}{\Longrightarrow}) \neq \emptyset \wedge (q \stackrel{l}{\longrightarrow}) = \emptyset \wedge \textstyle\sum_{c\in C} |w(c)| \leq 1\},$$

which is easily computable since we only have to consider a finite number of global states, namely those with channels containing a maximum of one message in total.

Computing $min(\hat{\not\sqsubseteq}_{k+1})$ from $min(\hat{\not\sqsubseteq}_k)$ The algorithm to compute $min(\hat{\not\sqsubseteq}_{k+1})$ from $min(\hat{\not\sqsubseteq}_k)$ is given in Figure 1. Informally, the algorithm starts with a set O being a finite representation of $\hat{\not\sqsubseteq}_k$ and then computes a new set N being a finite representation of $\hat{\not\sqsubseteq}_{k+1}$. To simplify the presentation of the algorithm we partition the set O into a collection of disjoint sets $O_{s,q}$ where $s \in S$ and $q \in Q$. Each set $O_{s,q}$ contains the minimal string vectors of the set $\{w : \langle s, w\rangle \hat{\not\sqsubseteq}_k q\}$. We also assume that the output set N is partitioned in the same manner.

The idea is to start with pairs of states in $\hat{\not\sqsubseteq}_k$ and go backwards by computing pairs of states in $\hat{\not\sqsubseteq}_{k+1}$. From the definition of $\hat{\not\sqsubseteq}_k$ we know that $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$ if and only if there is a string vector w' and label $l \in L$ such that

$$\langle s, w\rangle \stackrel{l}{\Longrightarrow} \langle s', w'\rangle \tag{2}$$
$$\langle s', w'\rangle \hat{\not\sqsubseteq}_k q_1, \ \ldots\ , \langle s', w'\rangle \hat{\not\sqsubseteq}_k q_n\,, \tag{3}$$

where $\{q_1, \ldots, q_n\} = (q \stackrel{l}{\longrightarrow})$.

For each i ($1 \leq i \leq n$), pick a string vector $w_i \in O_{s,q_i}$. To find a string vector w' satisfying (3) we observe that $O = min(\hat{\not\sqsubseteq}_k)$, hence $\langle s, w_i\rangle \hat{\not\sqsubseteq}_k q_i$. Since $\hat{\not\sqsubseteq}_k$ is upward closed we conclude that for any $t \in w_1 \sqcap \cdots \sqcap w_n$ it holds that $\langle s', t\rangle \hat{\not\sqsubseteq}_k q_i$; hence t satisfies (3). From Lemma 4.2 it follows that (2) will be satisfied by any w such that $\langle s, w\rangle \in pre_l\langle s', t\rangle$.

We use the following lemmas to prove the correctness of the algorithm.

Lemma 6.3 (Soundness). *If w is added to $N_{s,q}$, then $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$.*

Lemma 6.4 (Completeness). *If $k \geq 0$ and $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$, then some string vector w' such that $w' \preceq w$ will be added to $N_{s,q}$.*

Algorithm 6.1 (NEXT)
Input: An LCS $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$,
an FTS $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$, and
a set $O = min(\hat{\not\sqsubseteq}_k)$.

Output: A set $N = min(\hat{\not\sqsubseteq}_{k+1})$.

Local: A set T of string vectors.

```
for each s ∈ S and q ∈ Q do
    N_{s,q} := O_{s,q}
    for each l ∈ L and s_1 ∈ {s' : ⟨s, l, _, s'⟩ ∈ Δ} do
        let {q_1, ..., q_n} = (q --l-->)
        T := O_{s',q_1} ⊓ ··· ⊓ O_{s',q_n}
        for each t ∈ T do
            N_{s,q} := N_{s,q} ∪ {w : ⟨s, w⟩ ∈ pre_l⟨s', t⟩}
        done
    done
done
N := min(N)
```

Table 1: Computing $min(\hat{\not\sqsubseteq}_{k+1})$ from $min(\hat{\not\sqsubseteq}_k)$

From Lemma 6.3 and Lemma 6.4 it follows that

Lemma 6.5. *Algorithm 1 correctly computes* $N = min(\hat{\not\sqsubseteq}_{k+1})$.

Computing $min(\not\sqsubseteq)$ To compute $min(\not\sqsubseteq)$ we have to compute a fixpoint such that $min(\hat{\not\sqsubseteq}_k) = min(\hat{\not\sqsubseteq}_{k+1})$.

Lemma 6.6. *Algorithm 6.2 correctly computes a set* $W = min(\not\sqsubseteq)$.

Proof. It is clear that in the k:th iteration, $O = min(\hat{\not\sqsubseteq}_k)$ and $N = min(\hat{\not\sqsubseteq}_{k+1}$). Therefore, upon termination of the loop $min(\hat{\not\sqsubseteq}_k) = min(\hat{\not\sqsubseteq}_{k+1})$. Hence $\hat{\not\sqsubseteq}_k = \hat{\not\sqsubseteq}_{k+1} (= \hat{\not\sqsubseteq}_{k+2} = \cdots)$. This implies that $\not\sqsubseteq = \hat{\not\sqsubseteq}_k$, which in turn implies that W is a finite representation of $\not\sqsubseteq$, i.e., $W = min(\not\sqsubseteq)$. To prove termination it is sufficient to observe that if NEXT is executed infinitely many times, we add infinitely many new state-pairs to N. Since N is minimal (in the sense that it does not contain comparable elements), this implies that we add infinitely many incomparable state-pairs, which contradicts Higman's Theorem. It follows that the algorithm will terminate. $\square$

Algorithm 6.2 (Computing $min(\not\sqsubseteq)$)
Input: An LCS $\mathcal{L} = \langle S, L, C, M, \Delta\rangle$ and an FTS $\mathcal{T} = \langle Q, L, \longrightarrow\rangle$.
Output: A set $W = min(\not\sqsubseteq)$
Local Variables: A set T of string vectors

$N := min(\hat{\not\sqsubseteq}_1)$
repeat
    $O := N$
    $N := \mathrm{NEXT}(O)$
until $N = O$
$W := N$

Table 2: Algorithm for computing $\sqsubseteq$

This implies the result.

Theorem 6.7. *$\sigma \sqsubseteq q$ is decidable.*

7 Simulation Preorder: $\mathcal{T} \sqsubseteq \mathcal{L}$

We will consider the other direction of the simulation preorder, i.e., whether a global state σ simulates a state q in a finite transition system. The definition of (k-limited) simulation can be stated in similar manner to Section 6. Simulation preorder in this direction is upward closed, but in contrast to the algorithm in the previous section it is difficult to compute a simulation by computing successively smaller and smaller simulations. A simple example consists of the following automata:

For this example we may find a simulation for an arbitrary number of steps, but since the two automata are *not* related by any simulation preorder, the sequence

$$\uparrow(1, \langle 1, \mathtt{m}\rangle) \supset \uparrow(\mathtt{1}, \langle \mathtt{1}, \mathtt{mm}\rangle) \supset \uparrow(\mathtt{1}, \langle \mathtt{1}, \mathtt{mmm}\rangle) \supset \cdots$$

is an infinitely decreasing sequence of simulations.

Instead of trying to compute a set containing *all* simulations, we perform a *local search* [And92] to decide if a state q is simulated by a global state σ. The idea is to start with the state q and the global state σ and build a *tableau*, which is a finite tree of pairs of states.

For our purpose a tableau method is a goal directed proof system. To prove a goal we replace it with a set of sufficient subgoals to establish. The replacement of goals with subgoals is driven by a set of *tableau rules*, in our case we have only one:

$$\frac{q \mathrel{\dot{\sqsubseteq}} \sigma}{q_1 \mathrel{\dot{\sqsubseteq}} \sigma_1 \ \cdots \ q_n \mathrel{\dot{\sqsubseteq}} \sigma_n} \{q_1, \ldots, q_n\} = \{q' : \exists l \in L.\, q \xrightarrow{l} q'\}$$

where $\sigma_1, \ldots, \sigma_n$ are global states such that $\sigma \stackrel{l}{\Longrightarrow} \sigma_i$ if $q \xrightarrow{l} q_i$. Intuitively the inference rule says "whatever q does, σ should be able to do the same".

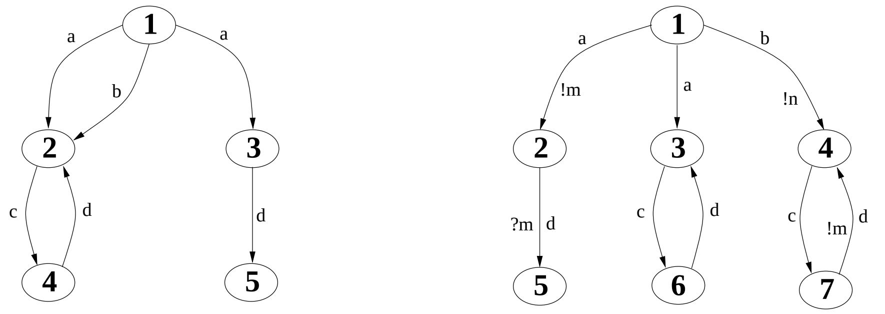

Figure 1: Example FTS and LCS

Example 7.1. For the finite transition system and lossy channel system in Figure 1, the following is a correct application of the rule.

$$\frac{1 \mathrel{\dot{\sqsubseteq}} \langle 1, \varepsilon\rangle}{2 \mathrel{\dot{\sqsubseteq}} \langle 4, \mathtt{n}\rangle \ \ 2 \mathrel{\dot{\sqsubseteq}} \langle 3, \varepsilon\rangle \ \ 3 \mathrel{\dot{\sqsubseteq}} \langle 2, \mathtt{m}\rangle}$$

□

A tableau is a tree of labelled *nodes*. The topmost node of a tableau is called the *root node*. If a node $\mathbf{n}$ is labelled $q \mathrel{\dot\sqsubseteq} \sigma$ we write it as $(\mathbf{n}\colon q \mathrel{\dot\sqsubseteq} \sigma)$. We denote a tableau with the root node labelled $q \mathrel{\dot\sqsubseteq} \sigma$ by $\mathrm{Sim}[q \mathrel{\dot\sqsubseteq} \sigma]$. A sequence of nodes starting at the root node of the tableau is called a *path*.

Given nodes $(\mathbf{n}_1\colon q_1 \mathrel{\dot\sqsubseteq} \sigma_1)$ and $(\mathbf{n}_2\colon q_2 \mathrel{\dot\sqsubseteq} \sigma_2)$ we say that $\mathbf{n}_1$ *is dominating* $\mathbf{n}_2$ if and only if $q_1 = q_2$ and $\sigma_1 \preceq \sigma_2$. In the tableau in Example 7.2, the node $(\mathbf{n}\colon 2 \mathrel{\dot\sqsubseteq} \langle 4, \mathtt{n}\rangle)$ is dominating the node $(\mathbf{n}'\colon 2 \mathrel{\dot\sqsubseteq} \langle 4, \mathtt{nm}\rangle)$

The leaf nodes of a tableau are called *terminal nodes*. Terminal nodes may be either *successful* or *unsuccessful*. A node $(\mathbf{n}\colon q \mathrel{\dot\sqsubseteq} \sigma)$ may be *successful* in one of two ways: (1) if there are no successors to q for any label, or (2) if $\mathbf{n}$ is dominated by some node above it in the tableau. A node $(\mathbf{n}\colon q \mathrel{\dot\sqsubseteq} \sigma)$ is *unsuccessful* if there exists label l such that $(q \xrightarrow{l}) \neq \emptyset$ and $(\sigma \xRightarrow{l}) = \emptyset$.

A tableau is successful if every terminal node is successful, otherwise the tableau is unsuccessful.

Example 7.2. A short example of a proof by using the tableau method is:

$$\dfrac{1 \mathrel{\dot\sqsubseteq} \langle 1,\varepsilon\rangle}{\dfrac{\dfrac{2 \mathrel{\dot\sqsubseteq} \langle 4,n\rangle}{4 \mathrel{\dot\sqsubseteq} \langle 7,n\rangle}}{2 \mathrel{\dot\sqsubseteq} \langle 4,nm\rangle} \quad \dfrac{\dfrac{2 \mathrel{\dot\sqsubseteq} \langle 3,\varepsilon\rangle}{4 \mathrel{\dot\sqsubseteq} \langle 6,\varepsilon\rangle}}{2 \mathrel{\dot\sqsubseteq} \langle 3,\varepsilon\rangle} \quad \dfrac{3 \mathrel{\dot\sqsubseteq} \langle 2,m\rangle}{5 \mathrel{\dot\sqsubseteq} \langle 5,\varepsilon\rangle}}$$

□

Intuitively, if we can find a successful tableau with root $(\mathbf{n}\colon q \mathrel{\dot\sqsubseteq} \sigma)$ then we have found a finite representation of a simulation $\mathcal{S}$ such that $(q, \sigma) \in \mathcal{S}$. We may stop searching when we reach a node that is dominated by a node above it in the tableau.

We use the following lemmas to prove correctness of our algorithm.

Lemma 7.1 (Soundness). *Let q be a state and σ a global state. If* $\mathrm{Sim}[q \mathrel{\dot\sqsubseteq} \sigma]$ *is a successful tableau, then $q \sqsubseteq \sigma$.*

Lemma 7.2 (Finiteness). *For any state q and global state σ, there is only a finite number of tableaux and each tableau is finite.*

Lemma 7.3 (Completeness). *If $q \sqsubseteq \sigma$, then there exists a successful tableau* $\mathrm{Sim}[q \mathrel{\dot\sqsubseteq} \sigma]$.

From Lemma 7.1, Lemma 7.2, and Lemma 7.3 we get the following

Theorem 7.4. *$q \sqsubseteq \sigma$ is decidable.*

Boolean Graphs The method for finding a tableau that proves that q is simulated by σ can be quite cumbersome. Basically we have to try all alternative successors to σ in turn, seeing that one leads to a result. Instead of exhaustively searching for a tableau one might use *boolean graphs* to construct the tableau. In our case we limit ourself to use a boolean *tree*. Boolean graphs (trees) can be used to describe and search for solutions in a variety of different systems, e.g., searching for fixpoints to formulae in the modal μ-calculus [And92].

Definition 7.5 (Boolean Graph). A boolean graph $\mathcal{G}$ is a tuple $\langle V, E, F\rangle$ where V is the set of *nodes* (or *vertices*), $E \subseteq V \times V$ is a set of *edges*, i.e. pairs of vertices representing the connection of the graph (tree), and $F = V \to \{\wedge, \vee\}$ is a labelling of each vertex. □

In our case we want to search for a solution to the boolean equation system

$$\begin{aligned} A(q,\sigma) &= \bigwedge_{q \xrightarrow{l} q'} O(q',\sigma,l) \\ O(q',\sigma,l) &= \bigvee_{\sigma \overset{l}{\Longrightarrow} \sigma'} A(q',\sigma') \end{aligned}$$

We terminate our search whenever we reach

- an arbitrary node that have no successors, or
- we reach an $\wedge$-node $A(q,\sigma')$ such that there is an $\wedge$-node $A(q,\sigma)$ earlier in the path for which $\sigma \preceq \sigma'$.

An $\vee$-node without successors is trivially false and a $\wedge$-node without successors is trivially true.

Algorithm 7.1 and 7.2 implements this search method as a pair of mutually recursive functions. To decide if $q \sqsubseteq \sigma$ we call *all-ok?*$(\emptyset, q, \sigma)$ and if the result is true then $q \sqsubseteq \sigma$, otherwise $q \not\sqsubseteq \sigma$.

Algorithm 7.1 *all-ok?* (P, q, σ)

Input: An LCS $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$,
an FTS $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$,
a state q in the finite transition system,
a global state σ in the lossy channel system, and
a set P containing all the previously visited $\wedge$-nodes in this path.

Output: A truth value.

if $\exists \sigma' \in P. \sigma' \preceq \sigma$ **then**
  return true
else
  for each $l \in L$ **and** $q' \in (q \xrightarrow{l})$ **do**
    if $\neg$*some-ok?*$(P \cup \{\langle q, \sigma \rangle\}, q', \sigma, l)$ **then**
      return false
  done
  return true

Figure 2: Algorithm for checking $\wedge$-nodes of a boolean graph

8 Bisimulation

In this section we show decidability of checking bisimulation between lossy channel systems and finite transition systems. In contrast to the case with simulation, neither the relations $\simeq_k$ nor the relations $\not\simeq_k$ are ideals. Therefore we cannot use the finite representation we use for describing the simulation relation in Section 6. Instead, we show that the relations $\simeq_k$ can be represented as differences between pairs of ideals. Furthermore, it turns out that we can define an upper bound on the number of $\simeq_k$ relations which we have to generate.

The definition of (k-limited bisimulation) can be given in a similar manner to that of simulation (Section 6).

8.1 Properties of Bisimulation

We show some properties of bisimulation which imply that we only need to generate a bounded number of $\simeq_k$.

Algorithm 7.2 *some-ok?*(P, q', σ, l)
Input: An LCS $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$,
an FTS $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$,
a state q' in the finite transition system,
a global state σ in the lossy channel system,
a label $l \in L$, and
a set P containing all the previously visited $\wedge$-nodes in this path.
Output: A truth value.

for each $\sigma' \in \{\sigma' : \sigma \stackrel{l}{\Longrightarrow} \sigma'\}$ **do**
  if *all-ok?*(P, q', σ') **then**
    return true
done
return false

Figure 3: Algorithm for checking $\vee$-nodes of a boolean graph

From the theory of bisimulation for finite state systems, we know that a finite transition system can be minimized with respect to bisimulation. We therefore consider, without loss of generality, only minimal finite transition systems. This means in particular that for any states q_1 and q_2 of such a system, $q_1 \sim q_2$ if and only if $q_1 = q_2$. We define the *size* of a finite transition system to be the number of states in it.

Lemma 8.1. *For a minimal finite transition system $\mathcal{T}$ with size m, and states q_1 and q_2 in $\mathcal{T}$, $q_1 \sim q_2$ if and only if $q_1 \simeq_m q_2$.*

Proof. A trivial application of the pigeonhole principle, since we can not partition the m states of $\mathcal{T}$ into more than m equivalence classes—there has to be at least one state in each equivalence class. □

For a lossy channel system $\mathcal{L}$ we use $\sigma_1 \stackrel{*}{\Longrightarrow} \sigma_2$ to denote that there exits $\rho \in L^*$ such that $\sigma \stackrel{\rho}{\Longrightarrow} \sigma'$. For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$ with size m, we define the set of *critical global states* (denoted $\textsc{Crit}_{\mathcal{L},\mathcal{T}}$) to be $\{\sigma : \exists q \in Q. \sigma \simeq_{m+1} q\}$, i.e., the set of global states that are $m+1$-limited bisimilar to some state of $\mathcal{T}$. We will omit the subscripts $\mathcal{L}$ and $\mathcal{T}$ on $\textsc{Crit}$ when they are clear from the context.

Lemma 8.2. *Let $\mathcal{L}$ be a lossy channel system and $\mathcal{T}$ a finite transition system of size m. Let σ be a global state of $\mathcal{L}$ and q a state of $\mathcal{T}$. If $\sigma \sim q$ then* $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq \textsc{Crit}$.

Proof. We prove this lemma using a contrapositive proof, i.e., we assume that $\sigma \stackrel{*}{\Longrightarrow} \sigma'$ and $\sigma' \notin \textsc{Crit}$ and show that $\sigma \not\sim q$. If $\sigma' = \sigma$ then the result follows immediately. If $\sigma \neq \sigma'$ then we know that there are labels $l_1, \ldots, l_n$ such that $\sigma \stackrel{l_1 l_2 \ldots l_n}{\Longrightarrow} \sigma'$. We use induction on n to prove that $\sigma \not\sim q$.

For the base case ($n = 1$): let $\sigma \stackrel{l_1}{\Longrightarrow} \sigma'$. Suppose that $\sigma \sim q$. It follows that there is an q' such that $q \stackrel{l_1}{\longrightarrow} q'$ and $\sigma' \sim q'$. This is impossible since $\sigma' \notin \textsc{Crit}$.

For the induction case, when $n > 1$: suppose that $\sigma \stackrel{l_1 l_2 \ldots l_{n+1}}{\Longrightarrow} \sigma'$ and that $\sigma' \notin \textsc{Crit}$. Further assume that $\sigma \sim q$; hence there is q'' and σ'' such that $q \stackrel{l_1}{\longrightarrow} q''$, $\sigma \stackrel{l_1}{\Longrightarrow} \sigma''$, and $\sigma'' \sim q''$. But since $\sigma'' \stackrel{l_2 \ldots l_{n+1}}{\Longrightarrow} \sigma'$ and $\sigma' \notin \textsc{Crit}$ it follows from the induction hypothesis that $\sigma'' \not\sim q''$, which leads to a contradiction. □

Lemma 8.3. *Let $\mathcal{L}$ be a lossy channel system, $\mathcal{T}$ a finite transition system of size m, and σ a global state of $\mathcal{L}$. If $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq \textsc{Crit}$ then the set $\{(\sigma', q') : \sigma' \simeq_{m+1} q' \wedge \sigma \stackrel{*}{\Longrightarrow} \sigma'\}$ is a bisimulation.*

Proof. Let $\mathcal{B} = \{(\sigma', q') : \sigma' \simeq_{m+1} q' \wedge \sigma \stackrel{*}{\Longrightarrow} \sigma'\}$ and let (σ_1, q_1) be a pair in $\mathcal{B}$, i.e., $\sigma_1 \simeq_{m+1} q_1$ and $\sigma \stackrel{*}{\Longrightarrow} \sigma_1$. We now have two cases to check, according to the two cases in the definition of bisimulation.

Suppose that $\sigma_1 \stackrel{l}{\Longrightarrow} \sigma_2$. It is immediately clear that $\sigma \stackrel{*}{\Longrightarrow} \sigma_2$. From the definition of bisimulation, there is q_2 such that $q_1 \stackrel{l}{\longrightarrow} q_2$ and $\sigma_2 \simeq_m q_2$. On the other hand, since $\sigma_2 \in \textsc{Crit}$, we know there is q_3 such that $\sigma_2 \simeq_{m+1} q_3$. Since $\mathcal{T}$ is minimal we know that q_2 and q_3 must be identical (by Lemma 8.1); hence $\sigma_2 \simeq_{m+1} q_2$.

For the other direction, suppose that $q_1 \stackrel{l}{\longrightarrow} q_2$. Then there is σ_2 such that $\sigma_1 \stackrel{l}{\Longrightarrow} \sigma_2$ and $\sigma_2 \simeq_m q_2$. It is immediately clear that $\sigma \stackrel{*}{\Longrightarrow} \sigma_2$. Similarly to above, since $\sigma_2 \in \textsc{Crit}$ we know that there is state q_3 such that $\sigma_2 \simeq_{m+1} q_3$. Since $\mathcal{T}$ is minimal we know that q_2 and q_3 must be identical (again by Lemma 8.1); hence $\sigma_2 \simeq_{m+1} q_2$.

It follows that $(\sigma_2, q_2) \in \mathcal{B}$: hence $\mathcal{B}$ is a bisimulation. □

Lemma 8.4. *Let $\mathcal{L}$ be a lossy channel system and $\mathcal{T}$ a finite transition system of size m. Let σ be a global state of $\mathcal{L}$ and let q be a state of $\mathcal{T}$. Then $\sigma \sim q$ if and only if $\sigma \simeq_m q$ and $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq$ CRIT.*

Proof. We prove both directions of the equivalence.

(only if) Assume that $\sigma \sim q$. It is clear that $\sigma \simeq_m q$. From Lemma 8.3 we get $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq$ CRIT.

(if) Assume that $\sigma \simeq_m q$ and $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq$ CRIT. From Lemma 8.3 we know that the set $\{(\sigma_1, q_1) : \sigma_1 \simeq_m q_1 \wedge \sigma \stackrel{*}{\Longrightarrow} \sigma_1\}$ is a bisimulation. The result follows immediately from the fact that $\sigma \simeq_m q$ and $\sigma \stackrel{*}{\Longrightarrow} \sigma$. □

8.2 Ideal Differences

We give a representation for describing the relations $\simeq_k$. More precisely, we introduce the domain $\mathcal{G}$, such that each element $G \in \mathcal{G}$ consists of sets of pairs (P, N) where P and N are canonical (finite) sets of global states. We define the *interpretation* of a pair (P, N) to be the set $\{\sigma : \sigma \in \uparrow P \wedge \sigma \notin \uparrow N\}$ and denote it *intrp* (P, N). We observe that we can assume, without loss of generality, the following invariant for any pair (P, N).

$$\forall w \in P. \forall w' \in N. w' \not\preceq w. \tag{4}$$

Given a set G of pairs of the above form, we define *intrp* (G) to be the set $\bigcup_{(P,N) \in G}$ *intrp* (P, N). In Figure 4 a pair (P, N) and its associated expansion and ideals are illustrated.

Given non-empty sets G_1 and G_2 we let $G_1 \otimes G_2$ denote the set $\{(P_1 \sqcap P_2, N_1 \sqcup N_2) : (P_1, N_1) \in G_1 \wedge (P_2, N_2) \in G_2\}$. For the special case that G_1 (or G_2) is empty, we let $G_1 \otimes G_2 = G_2$ (or $G_1 \otimes G_2 = G_1$). We further let $G_1 \oplus G_2$ denote the set $G_1 \cup G_2$. We use $\pi_s G$ for $s \in S$, to denote $\{(\pi_s P, \pi_s N) : (P, N) \in G\}$, i.e., the set of all pairs of global states such that the control state of each global state is s.

Example 8.1. Let $G_1 = \{(\langle s, m\rangle, \langle s, n\rangle)\}$ and let $G_2 = \{(\langle s, n\rangle, \langle s, nn\rangle)\}$. In this example we omit the control state since there is only one.

$$G_1 \otimes G_2 = \{(\{nm, mn\}, \{n, nn\})\} = \{(\{nm, mn\}, \{n\})\}$$
$$G_1 \oplus G_2 = \{(\{n, m\}, \{n, nn\})\} = \{(\{n, m\}, \{n\})\}$$

The simplification above is due to 4 above. □

It is easy to see that the interpretation of a pair $(P, N) \in G$ is *convex* in the following sense.

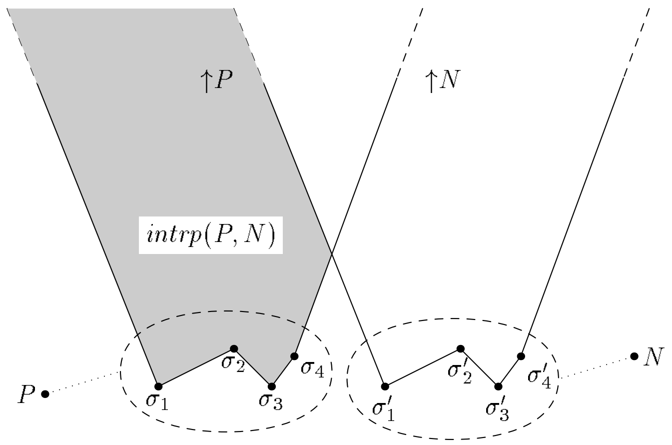

Figure 4: A pair of of canonical sets of global states

Lemma 8.5. *If* $\sigma_1, \sigma_2 \in intrp\,(P, N)$ *and* $\sigma_1 \preceq \sigma \preceq \sigma_2$*, then* $\sigma \in intrp\,(P, N)$*.*

Lemma 8.6. *Given* $G \in \mathcal{G}$ *we can compute the complement of* G*.*

Proof. For each pair $(P_i, N_i) \in G$ we generate $G_i = \{(N_i, \emptyset), (\{\varepsilon\}, P_i)\}$. Intuitively $(N_i, \emptyset)$ represents all global states that is larger than a global state in N_i, while $(\{\varepsilon\}, P_i)$ represents all global states except those larger than a global state of P_i. It is easy to see that G_i is a finite representation of the complement of (P_i, N_i). To compute the complement of G we now construct the intersection of the finite representations by computing $G = \bigotimes_i G_i$. $\square$

It is straightforward to show the following properties.

Proposition 8.7. *Let* G_1 *and* G_2 *be pairs of sets of global states as defined above.*

1. $\otimes$ *and* $\oplus$ *are associative and commutative.*

2. $\sigma \in intrp(G_1 \otimes G_2)$ *if and only if* $\sigma \in intrp(G_1)$ *and* $\sigma \in intrp(G_2)$*.*

3. $\sigma \in intrp(G_1 \oplus G_2)$ *if and only if* $\sigma \in intrp(G_1)$ *or* $\sigma \in intrp(G_1)$*.*

8.3 The Algorithm

In this section we describe an algorithm to compute the k-limited bisimulation relations $\simeq_k$ for a lossy channel system $\mathcal{L} = \langle S, L, C, M, \Delta\rangle$ and a finite transition system $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$. To simplify the notation, we assume in the rest of the section that we have only channel in the system. The presentation can easily be generalized to the case with several channels. To decide k-bisimilarity we need to go "backwards" in the transition relation. Since the strict inverse $\Longrightarrow^{-1}$ generates an infinite set of states, we introduce two functions operating on elements of $\mathcal{G}$. The function $w\text{-}pre_l(G)$ corresponds the weakest precondition to G, i.e., the set of states such that *all* transitions, w.r.t. $\stackrel{l}{\Longrightarrow}$, go into G. The function $pre_l(G)$ correspond to the set of states from which *some* transition go into G.[1]

Definition 8.8. Let G' be an elements of $\mathcal{G}$. We now construct elements G_1, G_2, and G_3 of $\mathcal{G}$ in such a manner that $w\text{-}pre_l(G') = G_1 \otimes G_2 \otimes G_3$.

For each pair $(P', N') \in G'$ there is a corresponding pair $(P_i, N_i) \in G_i$ ($1 \leq i \leq 3$), where P_i and N_i are the smallest sets such that:

1. if there is transition $\langle s_1, l, c!m, s_2\rangle \in \Delta$ such that $\langle s_2, \varepsilon\rangle \in P'$ and $\langle s_2, m\rangle \notin \uparrow N'$ then $\langle s_1, \varepsilon\rangle \in P_1$;
2. if there is a transition $\langle s_1, l, c!m, s_2\rangle \in \Delta$ such that $\langle s_2, \varepsilon\rangle \in P'$ and $\langle s_2, w'\rangle \in N'$ for some string vector w', then for every global state $\langle s', w'\rangle \in N'$ such that $s' = s_2$ and $m = last(w')$, there is global state $\langle s_1, w''\rangle \in N_1$, where $w' = w'' \bullet m$; and
3. if there is a transition $\langle s_1, l, c!m, s_2\rangle \in \Delta$ such that $\langle s_2, \varepsilon\rangle \in P'$ and $\langle s_2, w'\rangle \in N'$, then for every global state $\langle s', w'\rangle \in N'$ such that $s' = s_2$ and $last(w') \neq m$, there is global state $\langle s, w'\rangle \in N_1$.
4. If there is a transition $\langle s_1, l, c?m, s_2\rangle \in \Delta$ such that $\langle s_2, \varepsilon\rangle \in P'$, then $\langle s_1, m\rangle \in P_2$ and $\langle s_1, m \bullet w\rangle \in N_2$, for every global state $\langle s', w\rangle \in N'$ such that $s' = s_2$.
5. If there is a transition $\langle s_1, l, e, s_2\rangle \in \Delta$ such that $\langle s_2, \varepsilon\rangle \in P'$, then $\langle s_1, \varepsilon\rangle \in P_3$ and $\langle s_1, w'\rangle \in N_3$, for every $\langle s', w'\rangle \in N'$ such that $s' = s_2$. □

Example 8.2. Given lossy channel system $\mathcal{L} = \langle S, L, C, M, \Delta\rangle$ with $\Delta = \{\langle 1, a, c!m, 2\rangle, \langle 1, a, c?n, 2\rangle\}$ and $G = \{(\{\langle 2, \varepsilon\rangle\}, \{\langle 2, mm\rangle\})\}$ then $w\text{-}pre_a(G) =$

[1] In the modal μ-calculus [Sti92] $pre_l(G)$ corresponds to $\langle\langle l\rangle\rangle G$ and $w\text{-}pre_l(G)$ corresponds to $[\![l]\!]G$.

$G_1 \otimes G_2 \otimes G_3$, where

$$\begin{aligned} G_1 &= \{(\langle 1, \varepsilon\rangle, \langle 1, m\rangle)\} \\ G_2 &= \{(\langle 1, n\rangle, \langle 1, nmm\rangle)\} \\ G_3 &= \emptyset \end{aligned}$$

Hence $w\text{-}pre_a(G) = \{(\langle 1, n\rangle, \langle 1, m\rangle)\}$. □

The connection between *w-pre* and $\Longrightarrow$ is illustrated by the following lemmas.

Lemma 8.9. *Let G' be an element of $\mathcal{G}$. If $\sigma \in intrp(w\text{-}pre_l(G'))$ and $\sigma \stackrel{l}{\Longrightarrow} \sigma'$, then $\sigma' \in intrp(G')$.*

Lemma 8.10. *Let G' be an element of $\mathcal{G}$ and σ a global state. If $\sigma \stackrel{l}{\Longrightarrow} \sigma'$ and $\sigma' \in interp(G')$ for any global state σ', then $\sigma \in interp(w\text{-}pre_l(G'))$.*

For elements of $\mathcal{G}$ we also have a function *pre* corresponding to the function *pre* in previous sections. The definition is similar, but the function operates on elements of $\mathcal{G}$ instead of on elements of Σ.

Definition 8.11. Let G' be an element of $\mathcal{G}$. The set $pre_l(G')$ consists of the set of pairs $(pre_l(P'), \emptyset)$ such that $(P', N') \in G'$, i.e. $\{(pre_l(P'), \emptyset) : (P', N') \in G'\}$. □

The connection between *pre* and $\Longrightarrow$ is illustrated in the following lemma (which follows immediately from Lemma 4.2).

Lemma 8.12. *For $G \in \mathcal{G}$ and $l \in L$ and global states σ, σ'; if $\sigma \in intrp(pre_l(G))$, then there is $\sigma' \in intrp(G)$ such that $\sigma \stackrel{l}{\Longrightarrow} \sigma'$.*

Computing a $\simeq_{n+1}$ from $\simeq_n$ We are now ready to show how to compute $\simeq_{n+1}$ from $\simeq_n$. To simplify the presentation we assume that $O \in \mathcal{G}$ is partitioned into a collection of disjoint pairs of sets $O_{s,q}$ where $s \in S$ and $q \in Q$. We let $O_{s,q}$ be the set of pairs

$$\{(\{\langle s, w\rangle : (\langle s, w\rangle, q) \in P\}, \{\langle s, w\rangle : (\langle s, w\rangle, q) \in N\}) : (P, N) \in O\}.$$

In Figure 5 we see the algorithm to compute $\simeq_{n+1}$ given $\simeq_n$. The idea is to start with a representation of $\simeq_n$ and go backwards by computing $\simeq_{n+1}$. The algorithm takes a finite representation of $\simeq_n$ denoted O, and computes a finite representation of $\simeq_{n+1}$ denoted N.

We use the following lemmas to show correctness of the algorithm.

Algorithm 8.1 (Computing $\simeq_{n+1}$ from $\simeq_n$)
Input: A lossy channel system $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$,
a finite transition system $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$, and
O being a finite representation of $\simeq_n$.
Output: N being a finite representation of $\simeq_{n+1}$.

for each $s \in S$ and $q \in Q$ **do**
  let $\{l_1, \ldots, l_m\} = L$
  for each l_k such that $1 \leq k \leq m$ **do**
    let $\{q_1, \ldots, q_n\} = (q \xrightarrow{l_k})$
    let $\{s_1, \ldots, s_l\} = \{s' : \langle s, l_k, \_, s' \rangle \in \Delta\}$
    for each q_i $(1 \leq i \leq n)$ **do**
      $V_{l_k, q_i} := \pi_s \, pre_{l_k} (O_{s_1, q_i}) \oplus \cdots \oplus \pi_s \, pre_{l_k} (O_{s_l, q_i})$
    for each s_j $(1 \leq j \leq l)$ **do**
      $H_{l_k, s_j} := \pi_s \, w\text{-}pre_{l_k} \left(O_{s_j, q_1} \oplus \cdots \oplus O_{s_j, q_n} \right)$
    $T_{l_k} := H_{l_k, s_1} \otimes \cdots \otimes H_{l_k, s_l} \otimes V_{l_k, q_1} \otimes \cdots \otimes V_{l_k, q_n}$
  done
  $N_{s,q} := T_{l_1} \otimes \cdots \otimes T_{l_m}$
done

Figure 5: Procedure to compute $\simeq_{n+1}$ from $\simeq_n$

Lemma 8.13 (Soundness). *If $w \in intrp(N_{s,q})$, then $\langle s, w \rangle \simeq_{n+1} q$.*

Lemma 8.14 (Completeness). *If $\langle s, w \rangle \simeq_{n+1} q$ then $w \in intrp(N_{s,q})$.*

Lemma 8.15. *For a lossy channel system $\mathcal{L}$ and a finite transition system $\mathcal{T}$ we can compute a finite representation of $\simeq_k$ for any natural number k.*

Proof. Starting with a finite representation of $\simeq_0$ we can repeatedly compute a finite representation of $\simeq_{k+1}$ from a finite representation of $\simeq_k$ using Algorithm 8.1. The correctness of the algorithm follows from Lemma 8.13 and Lemma 8.14. □

Lemma 8.16. *Let $\mathcal{L}$ be a lossy channel system, σ a global state of $\mathcal{L}$, and* Crit *a finite representation of the critical states. We can then decide $\{\sigma' : \sigma \overset{*}{\Longrightarrow} \sigma'\} \subseteq$* Crit.

Proof. Deciding $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq \textsc{Crit}$ is done in the following steps.

1. Compute the complement $\overline{\textsc{Crit}}$ of $\textsc{Crit}$. We know, by Lemma 8.6, that $\textsc{Crit}$ is computable. We denote the complement by $\overline{\textsc{Crit}}$.

2. Compute a finite representation Γ of the ideal generated by $\overline{\textsc{Crit}}$. We do this since any state larger than a state in $\overline{\textsc{Crit}}$ trivially can reach a state in $\overline{\textsc{Crit}}$ simply by losing the appropriate messages. This is done by computing the union $\Gamma = min(\bigcup_{1 \leq i \leq n} P_i)$ where $\overline{\textsc{Crit}} = \{(P_1, N_1), \ldots, (P_n, N_n)\}$.

3. Use Lemma 4.4 to compute the reachability set $pre^*(\Gamma)$.

4. We now see if there is a global state $\sigma' \in pre^*(\Gamma)$ such that $\sigma' \preceq \sigma$. If that is the case, then $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \not\subseteq \textsc{Crit}$, otherwise $\{\sigma' : \sigma \stackrel{*}{\Longrightarrow} \sigma'\} \subseteq \textsc{Crit}$. □

From Lemma 8.4, Lemma 8.15, and Lemma 8.16 we get the following

Theorem 8.17. *$\sigma \sim q$ is decidable.*

9 Weak Process Relations

In this section, we will investigate the weak versions of the simulation and bisimulation relations. In these weak relations, we allow the transitions to be labelled with the silent action τ, with the effect that transitions can be hidden from the environment.

We will show that simulation preorder in the order $q \lessapprox \sigma$ becomes undecidable. Interestingly enough, simulation preorder in the order $\sigma \lessapprox q$ remains decidable, and so does bisimulation equivalence. This stands in contrast with the result in [AK95], where the undecidability of the weak bisimulation was conjectured by reduction of the *Global Reachability Problem.*[2] The conjecture was falsely given as a theorem.

[2]The problem of whether every computation of a lossy channel system can reach a control state.

9.1 Preliminaries

Let $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$ be a lossy channel system. We assume that L contains a special element, namely the silent event τ. Intuitively, a transition labelled with the silent event can not be observed by the environment.

Let $\mathcal{L}$ be a lossy channel system and σ a global state of $\mathcal{L}$. We define a *computation* π_σ *of* $\mathcal{L}$ as an infinite sequence $\sigma_0\sigma_1\sigma_2\ldots$ of global states in $\mathcal{L}$, where $\sigma_0 = \sigma$ and for each $i = 0, 1, \ldots$ either (1) $\sigma_i \longrightarrow \sigma_{i+1}$, or (2) if $\sigma_i \not\longrightarrow$ and $\sigma_i = \sigma_{i+1}$. In a similar manner we define *computation* π_q $\mathcal{T}$, for a finite transition system $\mathcal{T}$. as an infinite sequence of states.

Given a global state σ and a label l, we define $pre_l^\tau(\sigma)$ as $pre_\tau^*(pre_l(pre_\tau^*(\sigma)))$, i.e., the set of global states which can reach σ through performing one transition labelled with l, preceded and followed by an arbitrary number of τ-transitions. According to Lemma 4.4, this set is computable.

9.2 Decidability Results

In this section we will show that weak simulation preorder (in one direction) and weak bisimulation equivalence between a lossy channel system and a finite transition system is decidable by giving algorithm for their computations.

We use $\sigma \lessapprox q$ to denote weak simulation preorder and use $\sigma \approx q$ to denote weak bisimulation equivalence. The definitions of weak simulation preorder and weak bisimulation equivalence are similar to the definitions of the strong versions of the relations given in the previous sections, except that we allow transitions in the lossy channel system and the finite transition system to be labelled with the silent event τ.

Theorem 9.1. *$\sigma \lessapprox q$ is decidable.*

Proof. The idea of the algorithm is similar to the algorithm in Chapter 6, with the difference that we replace the pre_l function with the pre_l^τ function. □

We now show that weak bisimulation equivalence for lossy channel systems is decidable by giving an algorithm for it. The approach is similar to the method used in Section 8 but we have to take into account the fact that the lossy channel system may, in addition to losing messages, change control state through τ transitions. To simplify the explanation we will use a constraint-based description of the sets instead of the set-based representation from Section 8. We use the following language to describe sets of global states.

$$\Phi \ ::= \ \langle s, w\rangle \preceq \sigma \mid \langle s, w\rangle \not\preceq \sigma \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2$$

where s is a control state and w is a string vector. The constraint $\langle s, w\rangle \preceq \sigma$ denotes the set $\{\sigma' : \langle s, w\rangle \preceq \sigma'\}$, i.e., the set of all global states that are greater than or equal to $\langle s, w\rangle$ (w.r.t $\preceq$), while the constraint $\langle s, w\rangle \not\preceq \sigma$ denotes the set $\{\sigma' : \langle s, w\rangle \not\preceq \sigma'\}$, i.e., the set of all global states that is *not* greater than or equal to $\langle s, w\rangle$. Further note that there is no negation in the language since we can use De Morgan's law to drive the negations "inward", e.g.

$$\neg((\langle 1, ab\rangle \preceq \sigma) \wedge (\langle 1, ac\rangle \not\preceq \sigma)) \ = \ (\langle 1, ab\rangle \not\preceq \sigma) \vee (\langle 1, ac\rangle \preceq \sigma).$$

We define the constants **true** $= \bigvee_{s\in S\, c\in C} \langle s, \varepsilon\rangle \preceq c$ and **false** $= \neg$**true**. It is easy to see that we can represent each pair (P, N) from Section 8 by the formula

$$(\bigvee_{\sigma\in P} \sigma \preceq c) \wedge (\bigwedge_{\sigma\in N} \sigma \not\preceq c).$$

Hence each element G corresponds to the formula

$$\bigvee_{(P,N)\in G} (\bigvee_{\langle s,w\rangle\in P} \langle s, w\rangle \preceq \sigma) \wedge (\bigwedge_{\langle s,w\rangle\in N} \langle s, w\rangle \not\preceq \sigma).$$

Given a constraint $\phi \in \Phi$ we use $\|\phi\|$ to denote the finite representation—i.e., the set-of-pairs representation—of the constraint ϕ. By rewriting a constraint to the form above, we can represent an arbitrary constraint using the set-of-pairs representation from Section 8. It follows that the two notations are equivalent with respect to their expressive power. It is easy to see that we can compute a set-of-pairs representation of any given constraint. We can compute the complement of a set $\|\phi\|$ by computing the set $\|\neg\phi\|$, after suitable simplification of $\neg\phi$ using De Morgan's law. Since the two notations are equivalent with respect to their expressive power, we will henceforth use the constraint-based and set-of-pairs-based representation interchangeably.

We continue by extending the language given above with modalities. The resulting language is very similar to the well known Hennessey-Milner logic [Mil89, HM85].

$$\Psi ::= [\![l]\!]\Psi \mid \langle\!\langle l\rangle\!\rangle\Psi \mid \Phi.$$

Intuitively, $\langle\!\langle l\rangle\!\rangle\psi$ for a constraint $\psi \in \Psi$ denotes the set of global states such that it is possible to do an l transition and reach a state satisfying the constraint ψ, while $[\![l]\!]\psi$ denotes the set of global states such that whenever we do an l transition we reach a global state satisfying the constraint ψ. Given a constraint $\psi \in \Psi$ we can compute $\langle\!\langle l \rangle\!\rangle\psi$ by computing $pre_l^{\tau}(\|\psi\|)$ using Lemma 4.4. The resulting set can then be described in a constraint-based fashion. A basic equivalence in Hennessey-Milner logic is that $[\![l]\!]\psi$ is equivalent to $\neg\langle\!\langle l \rangle\!\rangle\neg\psi$. Hence we can compute $[\![l]\!]\psi$ by computing a finite representation of the constraint $\neg\langle\!\langle l \rangle\!\rangle\neg\psi$.

Given a lossy channel system $\mathcal{L} = \langle S, L, C, M, \Delta \rangle$ and a finite transition system $\mathcal{T} = \langle Q, L, \longrightarrow \rangle$ we can compute a finite representation of $\hat{\approx}_n$ for any $n \geq 0$. For each $q \in Q$ we let ψ_q^n denote (a constraint for) the set $\{\sigma' : \sigma' \hat{\approx}_n q\}$, i.e., the set of global states that are weakly n-limited bisimilar to q. Starting with $\psi_q^0 = \mathbf{true}$ we can then iteratively compute a sequence $\psi_q^0, \psi_q^1, \ldots, \psi_q^n$ using the following formula.

$$\psi_q^{n+1} = \bigwedge_{l \in L} \left(\bigwedge_{q \xrightarrow{l} q'} \langle\!\langle l \rangle\!\rangle \psi_{q'}^n \right) \wedge \left([\![l]\!] \bigvee_{q \xrightarrow{l} q'} \psi_{q'}^n \right).$$

Since every step is computable and we only compute ψ_q^n for a finite n, it follows that weak n-limited bisimulation is decidable.

Theorem 9.2. *$\sigma \approx q$ is decidable.*

Proof. The proof is similar to the proof of Theorem 8.17. The result follows from Lemma 8.4, Lemma 8.16 and the algorithm to compute weak n-limited bisimulations described above. $\square$

9.3 Undecidability Results

In this section we will show the undecidability of weak simulation preorder $q \lessapprox \sigma$. The undecidability is shown through reduction from the following problem, which has previously been shown undecidable [AJ94].

Definition 9.3 (Recurrent State Problem). The *Recurrent State Problem*[3] is defined as follows.

Instance $(\mathcal{L}, \sigma, s)$ where $\mathcal{L}$ is a lossy channel system, σ a global state of $\mathcal{L}$, and s a control state of $\mathcal{L}$.

[3]This problem is referred to as the *Recurrent Path Problem* in [AJ94]

Question: Is there a computation π_σ of $\mathcal{L}$ visiting s infinitely often? □

Theorem 9.4. *RSP is undecidable.*

Proof. In [AJ94], the problem is shown undecidable through a reduction from the *Cyclic Post's Correspondence Problem* [Ruo83]. □

We formulate the weak simulation problem by giving a precise definition of the instance we are working with. The definition is as follows.

Definition 9.5 (WSP). The *(Weak) Simulation Problem* between a finite transitions system and a lossy channel system is defined as follows.

Instance $(\mathcal{L}, \mathcal{T}, \sigma, q)$**:** $\mathcal{L}$ is a lossy channel system, $\mathcal{T}$ is a finite transitions system, σ is a global state of $\mathcal{L}$, and q is a state of $\mathcal{T}$.

Question: Is $q \lessapprox \sigma$? □

Lemma 9.6. *RSP is reducible to WSP.*

Proof. Given an instance $(\mathcal{L}, \sigma, s)$ of RSP we construct an instance $(\mathcal{L}', \mathcal{T}, \sigma, q)$ of WSP. $\mathcal{T}$ contains only one state q, one label l, and one transition $q \xrightarrow{l} q$. $\mathcal{L}'$ is built from $\mathcal{L}$ by separating the control state s of $\mathcal{L}$ into two control states s_1 and s_2 in $\mathcal{L}'$. The control state s_1 has the same "in" transitions in $\mathcal{L}'$ as s has in $\mathcal{L}$ and s_2 has the same "out" transitions in $\mathcal{L}'$ as s has in $\mathcal{L}$. We add a transition $s_1 \xrightarrow{l} s_2$ to $\mathcal{L}'$, and label all other transitions in $\mathcal{L}'$ with the silent action τ. This will have the effect that all transition except the single one labelled l are hidden from the environment. The reduction is illustrated in Figure 6.

Observe that for every computation in RSP π_σ, there is a corresponding computation with the difference that each global state $\langle s, w\rangle$ in π_σ is replaced with the two global states $\langle s_1, w\rangle\langle s_2, w\rangle$.

We proceed by proving that $q \lessapprox \sigma$ if and only if there is a computation in $\mathcal{L}$ that visits s infinitely often. We prove each direction separately.

(if) Suppose that there is a computation π_σ such that s is visited an infinite number of times. It is easy to see that for every global state σ' on π_σ, there is a computation $\pi_{\sigma'}$ that visits s an infinite number of times. It follows that there is a simulation $\mathcal{S} = \{(q, \sigma') : \sigma' \text{ is on } \pi_\sigma\}$.

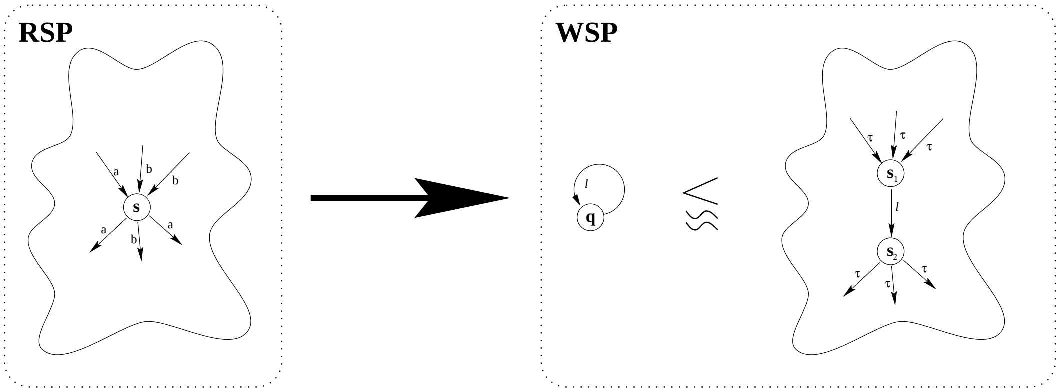

Figure 6: Reduction of RSP to WSP

(only if) Suppose that $q \lessapprox \sigma$. Since q can do an infinite number of l transitions—the only way $\mathcal{L}'$ can do a l transition is by passing through s_1 and s_2—it follows that there is a computation π_σ that visits s_1 an infinite number of times. It follows that there is a computation in $\mathcal{L}$ such that s is visited an infinite number of times.

□

From Lemma 9.6 and Theorem 9.4 we get the following

Theorem 9.7. *$q \lessapprox \sigma$ is undecidable.*

10 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.

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*, pages 160–170, 1993.

[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.

[AK95] Parosh Aziz Abdulla and Mats Kindahl. Decidability of simulation and bisimulation between lossy channel systems and finite state systems. In Lee and Smolka, editors, *Proc. CONCUR '95, 6^{th} Int. Conf. on Concurrency Theory*, volume 962 of *Lecture Notes in Computer Science*, pages 333 – 347. Springer Verlag, 1995.

[And92] H. R. Andersen. Model checking and boolean graphs. In *ESOP '92*, number 582 in Lecture Notes in Computer Science, pages 1–17, 1992.

[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.

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

[HM85] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.

[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. In *Proc. 7^{th} Int. Conf. on Computer Aided Verification*, volume 939 of *Lecture Notes in Computer Science*, pages 42–53. Springer Verlag, 1995.

[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.

[Lot83] M. Lothaire. *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and its Applications.* Addison-Wesley, 1983.

[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.

[Ruo83] K. Ruohonen. On some variants of Post's correspondence problem. *Acta Informatica*, (19):357–367, 1983.

[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.

[Sti92] Colin Stirling. Modal and temporal logics for processes. Technical Report LFCS-92-221, Laboratory for Foundation of Computer Science, 1992.

[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.

Proofs of Some Lemmas

Lemma 5.2

We prove this using induction on the length k of the traces. For the base case, the result holds trivially. For the induction case, let $\rho \in \mathit{traces}(q)$ be a trace of length $k+1$. As induction hypothesis, we assume that for every node $(\mathbf{n}'\colon q' \subseteq_T G')$, any trace $\rho \in \mathit{traces}(q')$ of length k is also in $\mathit{traces}(G')$, i.e., there is $\sigma' \in G'$ such that $\rho \in \mathit{traces}(\sigma')$ is a trace of length k. Assume that $(\mathbf{n}'\colon q' \subseteq_T G')$ is a node in the tree and that there is a trace $\rho \in \mathit{traces}(q')$ of length $k+1$, where $\rho = l_1 \ldots l_{k+1}$. We now have three cases to consider, depending on the three possible nodes in a successful tree: (1) $\mathbf{n}'$ is an internal node, (2) $\mathbf{n}'$ is a terminal node such that $(q' \longrightarrow) = \emptyset$, or (3) $\mathbf{n}'$ is a terminal node such that there is a node $(\mathbf{n}''\colon q'' \subseteq_T G'')$ above $\mathbf{n}'$ in the tree for which $q'' = q'$ and $G'' \preceq G$.

For the first case, we know that there is q'_1 such that $q' \xrightarrow{l_1} q'_1$ and $l_2 \ldots l_{k+1} \in \mathit{traces}(q'_1)$. Since $\mathbf{n}'$ is an internal node, we know that there is σ'_1 such that $\sigma' \stackrel{l_1}{\Longrightarrow} \sigma'_1$ and $(\mathbf{n}'_1\colon q'_1 \subseteq_T G'_1)$ is a node in the tree. By the induction hypothesis, we know that $l_2 \ldots l_{k+1} \in \mathit{traces}(G'_1)$ and it follows immediately that $l_1 l_2 \ldots l_{k+1} \in \mathit{traces}(G')$.

For the second case, where $(q' \longrightarrow) = \emptyset$, every trace of length $k+1$ in $\mathit{traces}(q')$ is trivially also in $\mathit{traces}(G')$.

For the third case, we observe that $G'' \preceq G'$ and $q'' = q'$. It follows that $\mathit{traces}(G'') \subseteq \mathit{traces}(G')$. Since $\mathbf{n}''$ is an internal node, $\mathit{traces}(q') = \mathit{traces}(q'') \subseteq \mathit{traces}(G'') \subseteq \mathit{traces}(G')$ for the same reasons as in the first case. $\square$

Lemma 5.3

We prove this lemma using a contrapositive proof, i.e., assume $\textsc{Tree}[q \subseteq_T \{\sigma\}]$ is unsuccessful and prove that $\mathit{traces}(q) \not\subseteq \mathit{traces}(\sigma)$. Since $\textsc{Tree}[q \subseteq_T \{\sigma\}]$ is unsuccessful it follows that there is an unsuccessful leaf $\mathbf{n}'$. From the definition of descendant nodes, we know that there are nodes

$$(\mathbf{n}_1\colon q_1 \quad \subseteq_T \quad G_1), \ldots, (\mathbf{n}_n\colon q_n \quad \subseteq_T \quad G_n)$$

and label $l_1, \ldots, l_{n-1}$ such that $\mathbf{n}_n = \mathbf{n}'$, $\mathbf{n}_1$ is the root node, $q_i \xrightarrow{l_i} q_{i+1}$, and $G_i \stackrel{l_i}{\Longrightarrow} G_{i+1}$ (where $1 \leq i \leq n-1$). Since $\mathbf{n}'$ is unsuccessful, $\{G'_n : G_n \Longrightarrow G'_n\} = \emptyset$, and $(q_n \xrightarrow{l}) \neq \emptyset$ for some label l. It follows that $l_1 \ldots l_n l \in \mathit{traces}(q)$, but $l_1 \ldots l_n l \notin \mathit{traces}(\{\sigma\})$. Hence $\mathit{traces}(q) \not\subseteq \mathit{traces}(\{\sigma\})$ $\square$

Lemma 5.4

Assume that the tree is infinite. According to Königs lemma, it then follows that there is an infinite sequence $\mathbf{n}_1, \mathbf{n}_2, \ldots$ of nodes. Since the number of states of the finite transition system is finite, it follows that there has to be an infinite sequence of sets of global states $G_1, G_2, \ldots$ such that $G_i \not\preceq G_j$ for any $i < j$. This contradicts the fact that $\preceq$ was a well quasi-ordering. $\square$

Lemma 6.3

Assume that w is added to $N_{s,q}$. We then have two cases to consider: either $w \in O_{s,q}$ or $w \notin O_{s,q}$. If $w \in O_{s,q}$ we know that $\langle s, w\rangle \hat{\not\sqsubseteq}_k q$, hence $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$. Otherwise, there exists s', t, l, $q_1, \ldots, q_n$, and $w_1, \ldots, w_n$ such that $\langle s, w\rangle \in pre_l\langle s', t\rangle$, $(q \xrightarrow{l}) = \{q_1, \ldots, q_n\}$, $t \in (w_1 \sqcap \cdots \sqcap w_n)$, and $\langle s', w_1\rangle \hat{\not\sqsubseteq}_k q_1, \ldots, \langle s', w_n\rangle \hat{\not\sqsubseteq}_k q_n$. By the definition of $\sqcap$ and the fact that $\hat{\not\sqsubseteq}_k$ is an ideal, it follows that

$$\langle s', t\rangle \hat{\not\sqsubseteq}_k q_1,\ \ldots,\ \langle s', t\rangle \hat{\not\sqsubseteq}_k q_n,$$

and by Lemma 4.2 we know that $\langle s, w\rangle \overset{l}{\Longrightarrow} \langle s', v\rangle$. It follows immediately that $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$. $\square$

Lemma 6.4

Assume that $\langle s, w\rangle \hat{\not\sqsubseteq}_{k+1} q$. If $k = 0$ then $w \in O_{s,q}$ and some $w' \preceq w$ will be added to $N_{s,q}$. Otherwise, from the definition of $\hat{\not\sqsubseteq}_{k+1}$, there exists l, w' and $q_1, \ldots, q_n$ such that $\{q_1, \ldots, q_n\} = (q \xrightarrow{l})$, $\langle s, w\rangle \overset{l}{\Longrightarrow} \langle s', w'\rangle$, and $\langle s', w'\rangle \hat{\not\sqsubseteq}_k q_1, \ldots, \langle s', w'\rangle \hat{\not\sqsubseteq}_k q_n$. Since $O = min(\hat{\not\sqsubseteq}_k)$ there exists $w_1, \ldots, w_n \preceq w'$ such that $w_i \in O_{s',q_i}$ $(1 \leq i \leq n)$. We conclude that there exists $t \in (w_1 \sqcap \cdots \sqcap w_n)$ such that $t \preceq w'$. From Lemma 4.3 and the monotonicity of *pre*, we know that there exists a w'' such that $w'' \preceq w$ and $\langle s, w''\rangle \in pre_l\langle s', t\rangle$, which will be added to $N_{s,q}$. $\square$

Lemma 7.1

Let Sim$[q \dot{\sqsubseteq} \sigma]$ be a successful tableau. Using induction on k, we show that $q' \hat{\sqsubseteq}_k \sigma'$ for any node $(\mathbf{n}'\colon q' \dot{\sqsubseteq} \sigma')$ in the tree.

For the base case, where $k = 0$, the lemma holds trivially, since $q' \hat{\sqsubseteq}_0 \sigma'$ for any q' and σ'.

For the induction case we assume that for every node $(\mathbf{n}'\colon q' \dot{\sqsubseteq} \sigma')$ it is the case that $q' \hat{\sqsubseteq}_k \sigma'$. We now have three cases to consider, corresponding to the three types of nodes possible in a successful tree. The node is (1) an internal node, (2) a terminal node $(\mathbf{n}'\colon q' \mathrel{\dot\sqsubseteq} \sigma')$ where $(q' \longrightarrow) = \emptyset$, or (3) a terminal node $\mathbf{n}'$ dominated by a node $\mathbf{n}''$ higher in the tree.

For the first case: let $(\mathbf{n}'\colon q' \mathrel{\dot\sqsubseteq} \sigma')$ be an internal node, l any label, and q'' an arbitrary state such that $q' \longrightarrow q''$. By the definition of the tableau rule we know that there is global state σ'' such that $\sigma' \overset{l}{\Longrightarrow} \sigma''$ and $(\mathbf{n}''\colon q'' \mathrel{\dot\sqsubseteq} \sigma'')$ is a decedent of $\mathbf{n}'$. By the induction hypothesis it then follows that $q'' \mathrel{\hat\sqsubseteq_k} \sigma''$ and, by the definition of limited simulation preorder, it follows that $q' \mathrel{\hat\sqsubseteq_{k+1}} \sigma'$.

For the second case, where $(q' \longrightarrow) = \emptyset$, it trivially holds that $q' \mathrel{\hat\sqsubseteq_{k+1}} \sigma'$.

For the third case, where $(\mathbf{n}'\colon q' \mathrel{\dot\sqsubseteq} \sigma')$ is dominated by a node $(\mathbf{n}''\colon q'' \mathrel{\dot\sqsubseteq} \sigma'')$, we observe the following. From the definition of dominating nodes, we know that $q'' = q'$ and $\sigma'' \preceq \sigma'$. We have already in case (1) shown that $q'' \mathrel{\hat\sqsubseteq_{k+1}} \sigma''$, since $\mathbf{n}''$ is an internal node. By the fact that $\hat\sqsubseteq_{k+1}$ is upward closed, it immediately follows that $q' \mathrel{\hat\sqsubseteq_{k+1}} \sigma'$. □

Lemma 7.2

Since finite transition systems and lossy channel systems are image-finite it follows that if there were an infinite sequence of different tableaux there has to be a path that grows infinitely. This contradicts Higman's Theorem.

By König's Lemma it follows that for an infinite tableau there has to be an infinite path. Since the number of states is finite, there has to be an infinite sequence of incomparable global states. This contradicts Higman's Theorem. □

Lemma 7.3

To prove completeness, we need to introduce an alternative definition of successful trees. We say that a tableau is h-successful, denoted $\mathrm{Sim}^h[q \mathrel{\dot\sqsubseteq} \sigma]$, if it can not be shown unsuccessful in h steps or less. An immediate consequence of this is that the tableau $\mathrm{Sim}^0[q \mathrel{\dot\sqsubseteq} \sigma]$ is successful for any state q and global state σ.

We use induction on k to show that there for any k such that $q \mathrel{\hat\sqsubseteq_k} \sigma$ exists a h-successful tableau $\mathrm{Sim}^h[q \sqsubseteq \sigma]$, where $h \leq k$.

For the base case, when $k = 0$, the theorem holds trivially.

For the induction case, when $k > 0$, we assume that $q \mathrel{\hat\sqsubseteq_k} \sigma$ implies that there exists a h-successful tableau $\mathrm{Sim}^h[q \mathrel{\dot\sqsubseteq} \sigma]$ for some $h \leq k$. Suppose that $q \mathrel{\hat\sqsubseteq_{k+1}} \sigma$. We now have two cases to consider, depending on whether $(q \longrightarrow)$ is empty or not. If $(q \longrightarrow)$ is empty, then the tree $\mathrm{Sim}^1[q \mathrel{\dot\sqsubseteq} \sigma]$ is trivially successful, and $1 \leq k+1$. If $(q \longrightarrow)$ is not empty, there are labels $l_1, \ldots, l_n$, states $q_1, \ldots, q_n$ and global states $\sigma_1, \ldots, \sigma_n$ such that $q \xrightarrow{l_i} q_i$, $\sigma \xrightarrow{l_i} \sigma_i$, and $q_i \mathrel{\hat\sqsubseteq_k} \sigma_i$ (where $1 \leq i \leq n$).

By the induction hypothesis, there are tableaux $\text{Sim}_i^{h_i}[q_i \dot{\sqsubseteq} \sigma_i]$ such that $h_i \leq k$ (where $1 \leq i \leq n$). By the definition of the inference rule, $\text{Sim}^{\max\{h_i\}_i+1}[q \dot{\sqsubseteq} \sigma]$ is successful and $\max\{h_i\}_{1\leq i\leq n} + 1 \leq k+1$ since $h_i \leq k$ for each i. By Theorem 7.2, this process will terminate and build a tableau. □

Lemma 8.9

Let G' be an element of $\mathcal{G}$ and $G = w\text{-}pre_l(G')$ for some label $l \in L$.

Assume that $\sigma \in intrp(G)$ and let $S = \{\sigma' : \sigma \stackrel{l}{\Longrightarrow} \sigma'\}$ be the set of successors. It is easy to see that S is finite. Using Definition 8.8 we separate the set S into the sets S_1, S_2, S_3 depending on the operation done in the transition.

Using Lemma 8.5 we observe that we only need to consider the largest (w.r.t. $\preceq$) global state in S_i and the smallest (w.r.t $\preceq$) global state in S_i for $1 \leq i \leq 3$.

Since $\sigma \in intrp(G) = intrp(G_1 \otimes G_2 \otimes G_3)$ it follows from Proposition 8.7 that $\sigma \in intrp(G_1), \sigma \in intrp(G_2)$, and $\sigma \in intrp(G_2)$. Hence, to prove that $\sigma' \in intrp(G')$ (for every $\sigma' \in S$) we have 3 cases to consider, corresponding to the three different elements G_1, G_2, and G_3 in Definition 8.8.

S_1 Let $\langle s', w'\rangle \in S_1$ be an arbitrary global state and let $\sigma = \langle s, w\rangle$. It follows immediately that there is $\langle s_1, l, c!m, s_2\rangle \in \Delta$ such that $s = s_1$ and $s' = s_2$. Since $\sigma \in intrp(G_1)$ there is a pair $(P_1, N_1) \in G_1$ such that $\sigma \in \uparrow P_1$ and $\sigma \notin \uparrow N_1$. From Definition 8.8, Lemma 8.5, and the fact that $\varepsilon \preceq w' \preceq w \bullet m$, we see that to prove $\sigma' \in intrp(G')$ it suffices to prove that there is pair $(P', N') \in G'$ such that $\langle s', \varepsilon\rangle, \langle s', w \bullet m\rangle \in intrp(P', N')$.

We first prove that $\langle s', \varepsilon\rangle \in intrp(P', N')$. By Definition 8.8 case 1 and the fact that P_1 is non-empty and canonical, we conclude that $\langle s, \varepsilon\rangle \in P_1$. By equation (4) we conclude that $\langle s', \varepsilon\rangle \notin \uparrow N'$; hence $\langle s', \varepsilon\rangle \in intrp\,(P', N')$.

We now prove that $\langle s', w \bullet m\rangle \in intrp(P', N')$. It is clear that $\langle s', w \bullet m\rangle \in \uparrow P'$. Suppose that $\langle s', w \bullet m\rangle \in \uparrow N'$, i.e., there is $\langle s', w_1\rangle \in N'$ such that $\langle s', w_1\rangle \preceq \langle s', w \bullet m\rangle$. From Definition 8.8 we see that either (1) $w_1 = w_2 \bullet m$ for some $w_2 \in N_1$, or (2) $last(w_1) \neq m$ and $w_1 \in N_1$. Since $\langle s, w\rangle \in intrp(G_1)$, we have a contradiction in (1) due to the fact that $w_2 \preceq w$, $w_2 \in N_1$, and $w \notin \uparrow N_1$. We further have a contradiction in (2) due to the fact that $w_1 \preceq w$, $w_1 \in N_1$ and $w \notin \uparrow N_1$. Since $w \preceq w \bullet m$, it follows that $\langle s', w \bullet m\rangle \in intrp\,(P', N')$.

Using Lemma 8.5 we conclude $S_1 \subseteq intrp(P', N')$ and further that $S_1 \subseteq intrp(G')$.

S_2 Let $\langle s', w'\rangle \in S_2$ be an arbitrary global state and let $\sigma = \langle s, w\rangle$. It follows immediately that there is $\langle s_1, l, c?m, s_2\rangle \in \Delta$ such that $s = s_1$ and $s' = s_2$.

Since $\sigma \in intrp(G_2)$ there is a pair $(P_2, N_2) \in G_2$ such that $\sigma \in \uparrow P_2$ and $\sigma \notin \uparrow N_2$. From the definition of $\Longrightarrow$, we see that there are $x, y \in M^*$ such that $w = x \bullet m \bullet y$. Without loss of generality we can assume that x does not contain m. It is immediately clear that $\varepsilon \preceq w' \preceq y$. By Definition 8.8 and Lemma 8.5, we conclude that to prove $\sigma' \in intrp(G')$, it is sufficient to prove that there is pair $(P', N') \in G'$ such that $\langle s', \varepsilon\rangle, \langle s', y\rangle \in\in intrp\,(P', N')$.

We first prove that $\langle s', \varepsilon\rangle \in intrp\,(P', N')$. By Definition 8.8 case 4 and the fact that P_2 is non-empty and canonical, we conclude that $\langle s', \varepsilon\rangle \in P_2$. We can further conclude that $\langle s', \varepsilon\rangle \in \uparrow P_2$ and, by equation (4), that $\langle s', \varepsilon\rangle \notin \uparrow N_2$; hence $\langle s', \varepsilon\rangle \in intrp\,(P', N')$.

We now prove that $\langle s', y\rangle \in intrp\,(P', N')$. It is clear that $\langle s', y\rangle \in \uparrow P'$. Suppose that $\langle s', y\rangle \in \uparrow N'$, i.e., there is $y_1 \in N'$ such that $y_1 \preceq y$. By Definition 8.8, we see that $m \bullet y_1 \in N_2$. On the other hand, since $\sigma \in intrp(G_2)$, we conclude that $x \bullet m \bullet y \notin \uparrow N_2$, which is a contradiction, since $m \bullet y_1 \preceq x \bullet m \bullet y$.

Using Lemma 8.5 we conclude that $S_2 \subseteq intrp\,(P', N')$ and further that $S_2 \subseteq intrp(G')$.

S_3 Let $\langle s', w'\rangle \in S_3$ be an arbitrary global state and let $\sigma = \langle s, w\rangle$. It follows immediately that there is $\langle s_1, l, e, s_2\rangle \in \Delta$ such that $s = s_1$ and $s' = s_2$. Since $\sigma \in intrp(G_3)$, there is a pair $(P_3, N_3) \in G_3$ such that $\sigma \in \uparrow P_3$ and $\sigma \notin \uparrow N_3$. From Definition 8.8, Lemma 8.5, and the fact that $\varepsilon \preceq w' \preceq w$, we see that to prove $\sigma' \in intrp(G')$ it is sufficient to prove that $\langle s', \varepsilon\rangle, \langle s', w\rangle \in intrp\,(P', N')$.

We first prove that $\langle s', \varepsilon\rangle \in intrp\,(P', N')$. By Definition 8.8 case 5 and the fact that P_3 is non-empty and canonical, we conclude that $\langle s', \varepsilon\rangle \in \uparrow P'$. By equation (4) we conclude that $\langle s', \varepsilon\rangle \notin \uparrow N'$.

We now prove that $\langle s', w\rangle \in intrp\,(P', N')$. It is clear that $\langle s', w\rangle \in \uparrow P'$. Suppose that $w \in \uparrow N'$, i.e., there is $w_1 \in N'$ such that $w_1 \preceq w$. By Definition 8.8, we see that $w_1 \in N_1$. On the other hand, since $\sigma \in intrp(G_3)$, we conclude that $w \notin \uparrow N_3$, which is a contradiction since $w_1 \preceq w$.

Using Lemma 8.5 we conclude that $S_3 \subseteq intrp\,(P', N')$ and further that $S_3 \subseteq intrp(G')$. □

Lemma 8.10

Let $\sigma = \langle s, w\rangle$ and let $\sigma'\langle s', w'\rangle$ be an arbitrary global state such that $\sigma \stackrel{l}{\Longrightarrow} \sigma'$. We separate $(\sigma \stackrel{l}{\Longrightarrow})$ into three different sets S_1, S_2, and S_3, depending on the operation done in the transition. Observe that the sets does not have to be disjoint. We now have three cases to consider, depending on the three different operations possible in the transition, each operation contributing to G_1, G_2, and G_3 respectively (as defined in Definition 7.3.1). For each case, we need not consider the alternative that G_i is empty ($1 \leq i \leq 3$), e.g., if G_1 is empty, there are no global states σ' such that $\sigma \overset{l}{\Longrightarrow} \sigma'$ by doing a send action. By definition, G_1 will then not contribute to $w\text{-}pre(G')$, since $\emptyset \otimes G = G$ for any G.

G_1 Suppose that $\langle s, l, c!m, s' \rangle \in \Delta$. It follows that $\langle s', w' \rangle \in interp(G')$. It is easy to see that $\langle s', \varepsilon \rangle \in$ it interp(G') and it is also easy to see that $\langle s', w \bullet m \rangle \in$ it interp(G'). It follows that there is $(P', N') \in G'$ such that $\langle s', \varepsilon \rangle \in P'$, $\langle s', \varepsilon \rangle \notin N'$, and $\langle s', w \bullet m \rangle \notin \uparrow N'$. Hence, by case 2 of Definition 7.3.1, $\sigma \in interp(G_1)$.

G_2 Suppose that $\langle s, l, c?m, s' \rangle \in \Delta$ and that S_2 is non-empty. It follows that $\langle s', w' \rangle \in interp(G')$. It is easy to see that $w = x \bullet m \bullet y$ for some sequence x not containing m, hence $w' \preceq y$. It follows that there is $(P', N') \in G'$ such that $\langle s', \varepsilon \rangle \in P'$, $\langle s', \varepsilon \rangle \notin N'$, and $\langle s', y \rangle \notin \uparrow N'$. Hence, by case 4 of Definition 7.3.1, $\langle s, w \rangle \in interp(G_2)$.

G_3 Suppose that $\langle s, l, e, s' \rangle \in \Delta$. It follows that $\langle s', w' \rangle \in interp(G')$. It is easy to see that $\langle s', \varepsilon \rangle \in interp(G')$ and that it is also easy to see that $\langle s', w \rangle \in interp(G')$. It follows that there is $(P', N') \in G'$ such that $\langle s', \varepsilon \rangle \in P'$ and $\langle s', \varepsilon \rangle \notin N'$ and further that $\langle s', w \rangle \notin \uparrow N'$. Hence, by case 5 of Definition 7.3.1, $\sigma \in interp(G_3)$.

It follows that $\sigma \in interp(w\text{-}pre(G'))$. □

Lemma 8.13

Assume that $\langle s, w \rangle \in intrp(N_{s,q})$. We have two symmetric cases to consider.

- We know by the assumption and Proposition 8.7 $\langle s, w \rangle \in intrp(T_{l_k})$ for each l_k and further that $\langle s_j, w \rangle \in intrp(H_{l_k,s_j})$ for each s_j. From the algorithm we observe that for every q_i ($1 \leq i \leq m$) there is a set O_{s_j,q_i}. Since $H_{l_k,s_j} = \pi_s w\text{-}pre_{l_k}\left(O_{s_j,q_1} \oplus \cdots \oplus O_{s_j,q_n}\right)$ we use Lemma 8.9 and Proposition 8.7 to conclude that if $\langle s, w \rangle \in intrp(H_{l_k,s_j})$ and $\langle s, w \rangle \overset{l_k}{\Longrightarrow} \langle s_j, w' \rangle$, then certainly $w' \in intrp(O_{s_j,q_i})$ for some q_i. Since O is a finite representation of $\simeq_n$ we conclude that $\langle s_j, w' \rangle \simeq_n q_i$. This corresponds to one of the cases in the definition of bisimulation.
- In analogy with the previous case, we know by the assumption and Proposition 8.7 that $\langle s, w \rangle \in intrp(T_{l_k})$ for each l_k and further that $\langle s, w \rangle \in intrp(V_{l_k,q_i})$ for each q_i. Using Proposition 8.7 we conclude that $\langle s, w \rangle \in intrp(\pi_s\, pre_l\left(O_{s_j,q_i}\right))$ for some s_j and from Lemma 4.2 we conclude that

there is $\langle s_j, w\rangle' \in \mathit{intrp}(O_{s_j,q_i})$ such that $\langle s, w\rangle \stackrel{l_k}{\Longrightarrow} \langle s_j, w'\rangle$. Since O is a finite representation of $\simeq_n$ we conclude that $\langle s_j, w'\rangle \simeq_n q_i$. This corresponds to the second case in the definition of bisimulation. $\square$

Lemma 8.14

Assume that $\langle s, w\rangle \simeq_{n+1} q$ and let $\{l_1, \ldots, l_m\} = L$. For each label l_k we then have two cases to consider, corresponding to the two cases in the definition of bisimulation. Let $\{\sigma_1, \ldots, \sigma_h\} = \{\sigma' : \sigma \stackrel{l_k}{\Longrightarrow} \sigma'\}$ and let $\{q_1, \ldots, q_n\} = (q \xrightarrow{l_k})$.

- Suppose that $\sigma \stackrel{l_k}{\Longrightarrow} \sigma_j$ for some $1 \leq j \leq h$. It then exists state q_i $(1 \leq i \leq n)$ such that $q \xrightarrow{l_k} q_i$ and $\sigma_j \simeq_n q_i$. Since O is a finite representation of $\simeq_n$ there is $w_j \in \mathit{intrp}(O_{s_j,q_i})$ for some q_i, where $\langle s_j, w_j\rangle = \sigma_j$. From Proposition 8.7 we conclude that $w_j \in \mathit{intrp}(O_{s_1,q_1} \oplus \cdots \oplus O_{s_k,q_h})$ and from Lemma 8.10 we conclude that $w \in \mathit{intrp}(H_{l_k,s_j})$ and further from Proposition 8.7 on $\otimes$ that $w \in \mathit{intrp}(T_{l_k})$.

- Suppose that $q \xrightarrow{l_k} q_i$ for some $1 \leq i \leq n$. It then exists global state σ_j $(1 \leq j \leq h)$ such that $\sigma \stackrel{l_k}{\Longrightarrow} \sigma_j$ and $\sigma_j \simeq_n q_i$. Since O is a finite representation of $\simeq_n$ there is $w_j \in \mathit{intrp}(O_{s_j,q_i})$, where $\langle s_j, w_j\rangle = \sigma_j$. From Lemma 4.2 and Proposition 8.7 we conclude that $w \in \mathit{intrp}(V_{l_k,q_i})$ and further that $w \in \mathit{intrp}(T_{l_k})$.

Since $w \in \mathit{intrp}(T_{l_1}), \ldots, w \in \mathit{intrp}(T_{l_m})$ we conclude that $w \in \mathit{intrp}(N_{s,q})$. $\square$