# Unreliable Data Communication

**Parosh Aziz Abdulla[1], C. Aiswarya[1], and Mohamed Faouzi Atig[1]**

1 **Uppsala Univeristy, Sweden**
parosh,cyriac,faouzi@it.uu.se

## Abstract

We extend the classical model of lossy channel systems by considering systems that operate on a finite set of variables ranging over an infinite data domain. Furthermore, each message inside a channel is equipped with a data item representing its value. Although we restrict the model by allowing the variables to be only tested for equality, we show that the state reachability problem is undecidable. In light of this negative result, we consider bounded-phase reachability, where the processes are restricted to performing either *send* or *receive* operations during each phase. We show decidability of state reachability in this case by computing a symbolic encoding of the set of system configurations that are reachable from a given configuration.

**1998 ACM Subject Classification** F.1.1 Models of Computation, F.3.1 Specifying and Verifying and Reasoning about Programs

**Keywords and phrases** Formal verification, Automata

## 1 Introduction

*Lossy Channel Systems* (LCS) have been studied extensively as a model of communication protocols. Such protocols are designed to work correctly even in the case where the underlying medium is unreliable in the sense that it can lose messages [4, 10]. More recently, LCS have been proposed as a fundamental framework for describing programs running on *weak memories* [6] since they are able to capture the behaviors of classical models such as TSO and PSO. The model of LCS has also been extended to allow quantitative reasoning with respect to time, so that it can encode message passing protocols whose behaviors are constrained by timing conditions [1]. However, all existing non-timed models assume that variables in the protocols and messages inside the channels range over finite domains.

In this paper, we consider an extension, called *data* LCS (or DLCS), that operates on an infinite domain $\mathcal{D}$. DLCS is an extension of LCS to infinite data domains.Concretely, we strengthen LCS in two ways. First, in addition to the channels, a DLCS also uses a finite set of variables ranging over an (arbitrary) infinite domain $\mathcal{D}$. Furthermore, each message inside a channel is equipped with a data item representing its "value". Thus, our model is unbounded in two dimensions, namely we have an unbounded number of messages inside the channels each with an attribute that is fetched from the infinite domain $\mathcal{D}$. The operations we allow on the channels are the standard *send* and *receive* operations. When sending a message to a channel, its value may be defined to be the value of a program variable. When a message is received from a channel, its value may be copied to a variable. A DLCS allows comparing the values of variables, where two variables may be tested for equality or disequality. Also, a variable may be assigned a new arbitrary value, or the value of another variable.

Given the decidability of the state reachability for LCS [4, 10] and for LCS augmented with time constraints [1], one would expect the same problem to be decidable for DLCS. In this paper, we show that this is surprisingly not the case. The undecidability proof relies on a novel encoding that uses the relation (equality and disequality) among the messages inside the channels in order to encode counters. An important property of the encoding is its

"stability" in the presence of messages losses. In fact, any message loss can be detected by the system which will then halt its simulation and thereby not reach the target process state. Thus we can simulate, for instance, a two counter machine using a DLCS with two channels.

A useful policy that has recently been used in order to circumvent undecidability (or to increase efficiency), in the verification of message-passing programs, is that of *phase-bounding* [2]. The idea is to consider only the runs of the system that can be split into a finite number of phases. During each phase, processes are restricted to perform either *send* or *receive* operations on the channels. In [2], it has been shown that many errors are manifested in runs with few phases. In this paper, we show that, when considering only runs with a given bounded number of phases, the reachability problem becomes decidable for DLCS. In fact, we show a stronger result. We prove that the set of configurations reachable under bounded phase assumption can be effectively computed.

**Related Work** The paper [1] introduces a timed extension of LCS. The timed model has an entirely different behavior compared to DLCS. For instance, the state reachability problem is decidable for the former. In particular, the model of [1] does not allow the assignment of data (clock) values carried by messages to the process variables. Such a restriction would reduce our model to the classical model of LCS. Perfect channel systems were also extended in [8] to handle timing constraints in the finite control. However, in their model, the channel contents were assumed to be from finite domain. Bounded phase reachability problem was considered in [2] for LCD. The problem was shown to be in NP by constructing polynomial sized Presburger formula. That technique does not apply in the current setting in any obvious way. Notice that bounded phase reachability of DLCS has a PSPACE lower bound since DLCS have register automata underlying them [9]. Further, rather than the control state reachability, we give a procedure for computing a representation for the symbolic configurations. Finally, several under-approximation techniques have been proposed for message passing programs [12, 7, 5, 11]. All these techniques are orthogonal to the notion of phase considering in [2].

## 2 Lossy Channel system with Data

In this section, we introduce our model, by defining its syntax and operational semantics. We assume an arbitrary infinite data domain $\mathcal{D}$ (e.g., the set of natural numbers $\mathbb{N}$). First we need to fix some notations for the rest of this paper.

For sets $A$ and $B$, we use $f : A \mapsto B$ to denote that $f$ is a (possibly partial) function that maps $A$ to $B$. For $a \in A$ and $b \in B$, we use $f[a \leftarrow b]$ to denote the function $f'$ where $f'(a) = b$ and $f'(a') = f(a')$ for all $a' \neq a$. For $A' \subseteq A$, we use $f\big|_{A'}$ to denote the restriction of $f$ to $A'$. For a relation $\mathcal{R} \subseteq A \times A$, we use $\mathcal{R}\big|_{A'}$ to denote the restriction of $\mathcal{R}$ to $A'$. For functions $f_1 : A_1 \mapsto B$ and $f_2 : A_2 \mapsto B$ with $A_1 \subseteq A_2$, we write $f_1 \sqsubseteq f_2$ to denote that $f_1(a) = f_2(a)$ for all $a \in A_1$. We use $A^*$ to denote the set of words over $A$; and use $\epsilon$ to denote the empty word. For words $w, w' \in A^*$, we use $w_1 \bullet w_2$ to denote the concatenation of $w_1$ and $w_2$. We write $w \preceq w'$ to denote that $w$ is a (not necessarily contiguous) subword of $w'$, and define $w{\downarrow} := \{w' \mid w' \preceq w\}$ to be the downward closure of $w$ wrt. $\preceq$. We use $|w|$ to denote the length of $w$. For $i : 1 \leqslant i \leqslant |w|$ we use $w[i]$ to denote the $i^{th}$ element of $w$.

**Syntax** A *Lossy Channel System with Data* (or simply DLCS) is a tuple $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$ where $Q$ is a finite set of (local) states, $\mathcal{X}$ is a finite set of variables that range over $\mathcal{D}$, $\mathcal{C}$ is a finite set of channels, $\Sigma$ is a finite (message) alphabet, and $\Delta$ is a finite set of transitions. A transition is a triple $\langle q, op, q' \rangle$ where $op$ is one of the following forms (where $x, y \in \mathcal{X}$ are variables, $a \in \Sigma$ is a symbol in the alphabet, and $\mathsf{ch} \in \mathcal{C}$ is a channel):

- $x \leftarrow y$, assigns the value of $y$ to $x$.

- $x \leftarrow \circledast$, assigns a locally fresh value to $x$. That is $x$ is nondeterministically assigned a value that is different from the current values of other registers.
- $x = y$ tests whether the values of $x$ and $y$ are equal.
- $x \neq y$ tests whether the values of $x$ and $y$ are different.
- $\mathsf{ch}!\langle a, x\rangle$ sends $a$ together with the value of the variable $x$ to the channel $\mathsf{ch}$.
- $x := \mathsf{ch}?a$ receives $a$ from $\mathsf{ch}$, and stores the associated value in the variable $x$.

We define $\texttt{SourceOf}\,(t) := q_1$, $\texttt{OpOf}\,(t) := op$, and $\texttt{TargetOf}\,(t) := q_2$.

We define $\Delta^{snd}$ to be the set of transitions in $\Delta$ that perform either (i) *send* operations, or (ii) operations on the variables. That is, it is the set of transitions that do *not* perform *receive* operations. We define $\Delta^{rcv}$ analogously for the *receive* transitions.

**Semantics** A *variable state* is a function $V : \mathcal{X} \mapsto \mathcal{D}$ that defines the values of the variables. A *channel state* is a function $\omega : \mathcal{C} \mapsto (\Sigma \times \mathcal{D})^*$, which gives the content of each channel. This content is a word of pairs each consisting of an alphabet symbol together with a value. A *configuration* is a triple $\gamma = \langle q, V, \omega\rangle$ where $q \in Q$ is a state, $V$ is a variable state, and $\omega$ is a channel state. We use $\texttt{StateOf}\,(\gamma)$, $\texttt{VarStateOf}\,(\gamma)$, $\texttt{ChannelStateOf}\,(\gamma)$ to denote $q$, $V$, and $\omega$ respectively. We say $\gamma$ is *plain* if $\omega(\mathsf{ch}) = \epsilon$ for all channels $\mathsf{ch} \in \mathcal{C}$. We use $\texttt{ConfsOf}\,(\mathcal{L})$ to denote the set of configurations of $\mathcal{L}$.

We extend the ordering $\preceq$ to channel states, such that $\omega \preceq \omega'$ if $\omega(\mathsf{ch}) \preceq \omega'(\mathsf{ch})$ for all channels $\mathsf{ch} \in \mathcal{C}$. We further extend the ordering to configurations such that $\gamma = \langle q, V, \omega\rangle \preceq \langle q', V', \omega'\rangle = \gamma'$ if (i) $q' = q$, (ii) $V' = V$, and (iii) $\omega \preceq \omega'$. In other words, the states and values of variables in $\gamma$ and $\gamma'$ coincide, while the content of each channel in $\gamma$ is a subword of the content of the same channel in $\gamma'$

We define the transition relation $\longrightarrow_{\mathcal{L}}$ on the set of configurations in two steps as follows. In the first step, we define the relation $\hookrightarrow_{\mathcal{L}}$. For configurations $\gamma = \langle q, V, \omega\rangle$ and $\gamma' = \langle q', V', \omega'\rangle$, and a transition $t = \langle q, op, q'\rangle \in \Delta$, we write $\gamma \overset{t}{\hookrightarrow}_{\mathcal{L}} \gamma'$ if one of the following conditions is satisfied:

- $op$ is of the form $x \leftarrow y$, $V' = V[x \leftarrow V(y)]$ and $\omega' = \omega$. The variable $x$ is assigned the value of $y$; the values of other variables and the contents of channels are not changed.
- $op$ is of the form $x \leftarrow \circledast$, $V' = V[x \leftarrow d]$ for some $d \in \mathcal{D}\backslash\{V(x) \mid x \in \mathcal{X}\}$, and $\omega' = \omega$. The variable $x$ is non-deterministically assigned an arbitrary value different from the values of other variables; the other variable values and channel contents remain the same.
- $op$ is of the form $x = y$, $V(x) = V(y)$, $V' = V$, and $\omega' = \omega$. The transition is enabled only if the values of $x$ and $y$ are identical; the control state is only modified.
- $op$ is of the form $x \neq y$, $V(x) \neq V(y)$, $V' = V$, and $\omega' = \omega$. The transition is enabled only if the values of $x$ and $y$ are different; the control state is only modified.
- $op$ is of the form $\mathsf{ch}!\langle a, x\rangle$, $V' = V$, and $\omega' = \omega[\mathsf{ch} \leftarrow \langle a, V(x)\rangle \bullet \omega(\mathsf{ch})]$. The transition appends a message consisting of the symbol $a$ and the value of $x$ to the tail of channel $\mathsf{ch}$.
- $op$ is of the form $x := \mathsf{ch}?a$, $V' = V[x \leftarrow d]$, and $\omega = \omega'[\mathsf{ch} \leftarrow \omega(\mathsf{ch}) \bullet \langle a, d\rangle]$ for some $d \in \mathcal{D}$. The transition receives the message at the head of channel $\mathsf{ch}$ if the alphabet symbol in that message is $a$. The value stored inside the message is assigned to $x$.

In the second step, we define the relation $\longrightarrow_{\mathcal{L}}$. More precisely, we let $\gamma \overset{t}{\longrightarrow}_{\mathcal{L}} \gamma'$ denote that there are configurations $\gamma_1, \gamma_2$ such that $\gamma_1 \preceq \gamma$, $\gamma' \preceq \gamma_2$, and $\gamma_1 \overset{t}{\hookrightarrow}_{\mathcal{L}} \gamma_2$. We write $\gamma \longrightarrow_{\mathcal{L}} \gamma'$ to denote that $\gamma \overset{t}{\longrightarrow}_{\mathcal{L}} \gamma'$ for some $t \in \Delta$. The reflexive transitive closure of the relation $\longrightarrow_{\mathcal{L}}$ is denoted by $\overset{*}{\longrightarrow}_{\mathcal{L}}$.

Observe that a *full* non-deterministic register assignment of the form $x \leftarrow *$, where the variable $x$ is assigned any value in $d \in \mathcal{D}$, can be easily simulated by a combination of variable assignments and locally fresh assignments (i.e., $x \leftarrow * \equiv x \leftarrow \circledast \vee \bigvee_{y \in \mathcal{X}} x \leftarrow y$).

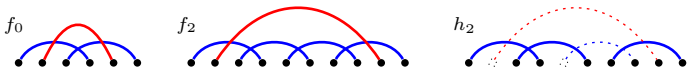


**Figure 1** Illustration of $f_0$, $f_2$, $h_2$ and a chain corresponding to $f_5$.

## 3 Undecidability of Reachability Problem

In this section we define the reachability problem and show that it is undecidable in general.

Let us assume a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$. For a configuration $\gamma$ of $\mathcal{L}$, we define $\texttt{Reach}(\mathcal{L})(\gamma) := \left\{\gamma' \mid \gamma \xrightarrow{*}_{\mathcal{L}} \gamma'\right\}$, i.e., it is the set of configuration reachable from $\gamma'$. For a set $\Gamma \subseteq \texttt{ConfsOf}(\mathcal{L})$, we define $\texttt{Reach}(\mathcal{L})(\Gamma) := \cup_{\gamma \in \Gamma} \texttt{Reach}(\mathcal{L})(\gamma)$. For a state $q \in Q$, we use $\gamma \xrightarrow{*}_{\mathcal{L}} q$ to denote that $\gamma \xrightarrow{*}_{\mathcal{L}} \gamma'$ for some $\gamma'$ with $\texttt{StateOf}(\gamma') = q$. In such a case we say that $q$ is *reachable* from $\gamma$. An instance of the *reachability problem* is defined by a plain configuration $\gamma$ and a state $\mathsf{Target} \in Q$. The question is whether $\mathsf{Target}$ is reachable from $\gamma$.

▸ **Theorem 1.** *The reachability problem for* DLCS *is undecidable in general.*

**Proof sketch.** The proof is done by reduction from the reachability problem for 2-counter Minsky machines (which is well-known to be undecidable). The main idea is to construct a DLCS with two channels (one per counter). Each channel is used to encode the value of a counter. We will use a special set of data-equality relations between messages inside a channel to make this channel robust against message losses. In fact, this encoding has an important feature, namely that we cannot obtain one set of relations from another through the deletion of messages. This makes them good candidates for the exact representation of counter values (in the the presence of losses) in the channels. A counter value cannot be "accidentally" reduced through the loss of messages. Our simulation can recognize that this has happened. In such a case, we make sure that the simulation will never reach the target state. After presenting the encoding, we give a brief description of how to translate an instance of the reachability problem for 2-counter machines to an equivalent instance of the reachability problem for DLCS.

The data-equality relation within the channel is used to enforce a rigid structure as depicted in Figure 1. The rigid structure has an even length $\ell \geqslant 6$. The data value appearing at any odd position $i$, with $i < \ell - 2$, in the channel is equal to one, and only one, other even position, namely position $(i+3)$. Furthermore, we require that the data in the second position of the channel is only equal to the data stored at the position $\ell - 1$. This data-equality relation can be viewed analogously as a closed chain of interlocked links. Each data value that appears in pair corresponds to a link. We need at least 3 links to form a closed chain. Hence an encoding $f_n$ of value $n$ has length $\ell = 6 + 2n$. Notice that, any proper subword $h_n$ of an encoding $f_n$ will open up the closed chain. Thus a channel containing a valid encoding, once becomes invalid by losing messages, will never become valid again.

Let us describe how to construct the DLCS simulating the two counter machine. The message alphabet $\Sigma = \{a, \lhd, \rhd\}$. The occurrences of $a$ together with the associated data values encode the value of a counter. The symbols $\lhd$ and $\rhd$ are the left and right end-markers respectively and their associated data values are irrelevant. The set of variables $\mathcal{X} = \{x, x_0, x_1, x_2, x_3\}$. While simulating increment, decrement and zero test, the DLCS also checks that the contents of the channel is a valid encoding otherwise the simulation is blocked. To simulate a zero test, the DLCS needs to check that the channel associated to this counter is a valid encoding of zero (i.e., the data-equality relations between the messages with alphabet

symbol $a$ form the encoding $f_0$ in Figure 1). To that end, the DLCS will first rotate the message $\langle \triangleright, d \rangle$ at the head of the channel by performing a receive operation followed by a send of the same message. Then the DLCS performs four rotation operations of messages (with an alphabet symbol $a$) while storing their data values in the variables $x_2$, $x_1$, $x_0$ and $x$, respectively. Now, the DLCS is ready to check the first equality relation of the encoding $f_0$ by testing if $x_2 = x$. If it is the case then the DLCS can proceed by rotating the next message at the head of the channel while using the variable $x$ to store its data value. The DLCS can then check the second equality relation of $f_0$ by testing if $x_1 = x$. If the second equality holds, the DLCS can proceed and rotate the message (with $a$ as alphabet symbol) at the head of the channel while storing its data value in the variables $x$. Finally, the DLCS checks the last equality relation of the encoding $f_0$ by testing if $x_0 = x$ and if it is the case, it rotates the message of the form $\langle \triangleleft, d \rangle$.

To simulate a decrement operation, it (i) checks whether $\mathsf{ch}$ encodes $f_n$ for some $n > 0$, and (ii) simultaneously transforms the content of $\mathsf{ch}$ to an encoding of value $n-1$. For this, it erases the last and 4th last elements in $f_n$, and swaps the 2nd last and the 3rd last elements. This is achieved by a cyclic rotation of the channel, during which it also checks that (i) holds. For increment, it first generates a globally fresh value, by means of a locally fresh assignment and a cyclic rotation of channels verifying that it is indeed globally fresh. Then there is another cyclic rotation verifying that $\mathsf{ch}$ contains some $f_n$, while transforming the content of $\mathsf{ch}$ to an encoding of the value $n+1$. For this, it a) swaps 1st and 2nd elements of $f_n$, and b) inserts the generated globally fresh value between 2nd and 3rd elements in $f_n$ and also before the first element of $f_n$. More details of the reduction can be found in Appendix A. ◂

## 4 Bounded-Phase Reachability

Motivated by the undecidability of reachability, we study under-approximate reachability problem. We introduce bounded-phase computations [2]. We view a computation as consisting of a number of phases where, during a given phase, the system either only performs *send* operations, or only performs *receive* operations (in addition to the operations on variables). Consider a computation $\pi = \gamma_0 \xrightarrow{t_1}_{\mathcal{L}} \gamma_1 \xrightarrow{t_2}_{\mathcal{L}} \cdots \xrightarrow{t_n}_{\mathcal{L}} \gamma_n$. We define $\pi^\bullet := t_1 t_2 \cdots t_n$, i.e., it is the sequence of transitions that occur in $\pi$. Given a sequence of transitions $\delta = t_1 t_2 \cdots t_n \in \Delta^*$, we say that $\delta$ is a *phase* if either $t_i \in \Delta^{snd}$ for all $i : 1 \leqslant i \leqslant n$, or $t_i \in \Delta^{rcv}$ for all $i : 1 \leqslant i \leqslant n$. A computation $\pi$ is said to be *$k$-phase bounded* if $\pi^\bullet = \delta_1 \bullet \delta_2 \bullet \cdots \bullet \delta_j$ where $j \leqslant k$ and $\delta_i$ is a phase for all $i : 1 \leqslant i \leqslant j$. In other words, the transitions in $\pi$ form at most $k$ phases. For configurations $\gamma$ and $\gamma'$, we say that $\gamma'$ is *$k$-phase reachable* from $\gamma$ if $\gamma'$ is reachable from $\gamma$ by a $k$-phase bounded computation. For a configuration $\gamma$, we define $\mathtt{Reach}_{\mathtt{P}}(k)(\mathcal{L})(\gamma) := \{\gamma' \mid \gamma' \text{ is } k\text{-phase reachable from } \gamma\}$, i.e., it is the set of configurations that are $k$-phase reachable from $\gamma'$. State $k$-phase reachability is defined in a similar manner to state reachability. In the *bounded-phase reachability problem*, we are also given a natural number $k \in \mathbb{N}$, and we are asked whether $\mathsf{Target}$ is $k$-phase reachable from $\gamma$.

▸ **Theorem 2.** *The bounded-phase reachability problem is decidable.*

The remaining part of the paper is dedicated to the proof of Theorem 2. In fact, we will show a stronger result. We will compute a symbolic encoding of the set of all configurations reachable by a bounded phase computation. Towards that, we will first introduce our symbolic configurations, the representation of a set of symbolic configurations and some operations on them in Section 5. Then, we will show how to characterise the set of bounded-phase reachable configurations using the symbolic encoding in Section 6.

# 5 Symbolic Encoding

The main idea behind the symbolic encoding is that configurations with identical (dis)equality relations on the set of variables and messages inside the channels, will have similar behaviors. For instance, such configurations are able to perform identical sequences of transitions. To use this idea formally, we first set some notions and notations related to equivalence relations.

**Equivalence Relations** Consider an equivalence relation, i.e., a reflexive, symmetric, and transitive relation $\mathcal{R}$ on a set $A$. We use $a\mathcal{R}b$ to denote that $\langle a, b\rangle \in \mathcal{R}$, and use $A/\mathcal{R}$ to denote the set of blocks (equivalence classes) of $\mathcal{R}$. For an element $a \in A$, let $[a]_{\mathcal{R}}$ denote the block of $a$. We have $a\mathcal{R}b$ iff $[a]_{\mathcal{R}} = [b]_{\mathcal{R}}$. We use $\texttt{EqRel}\,(A)$ to denote the set of equivalence relations on the set $A$. We define $\mathcal{R}\,[a \rightsquigarrow b]$ to be the equivalence relation we get by moving $a$ from its original block to the block of $b$. We define $\mathcal{R}\,[a \rightsquigarrow \circledast]$ to be the equivalence relation we get by removing $a$ from its original block, and putting it alone in a new block.

Consider sets $A_1$ and $A_2$, and an equivalence relation $\mathcal{R} \in \texttt{EqRel}\,(A_1)$. For a function $f : A_1 \mapsto A_2$ from $A_1$ to $A_2$, we write $f \models \mathcal{R}$ to denote that, for all $a_1, a_2 \in A_1$, we have that $[a_1]_{\mathcal{R}} = [a_2]_{\mathcal{R}}$ iff $f(a_1) = f(a_2)$, i.e., $f$ assigns identical (and unique) values to elements belonging to the same block in $\mathcal{R}$. Suppose $A_1 \cap A_2 = \varnothing$. Then we define the set $\mathcal{R} \otimes A_2 := \left\{\mathcal{R}_1 \in \texttt{EqRel}\,(A_1 \cup A_2) \mid \mathcal{R} = \mathcal{R}_1\big|_{A_1}\right\}$.

## 5.1 Symbolic Configurations

Assume a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta\rangle$. The symbolic encoding will characterize, in a uniform manner, sets of $k$-phase reachable configurations in terms of the initial values of the variables. To that end, we will use a finite set $I$ whose elements we call *initiators*. The initiators will be used as place holders for the initial values of the variables. For a set $A$ (typically a subset of $\mathcal{X} \cup I$), an *$A$-valuation* is mapping $\theta : A \mapsto \mathcal{D}$. First, we define a symbolic encoding to represent (infinite) sets of channel states. A *symbolic word* $v$ over $I$ is of the form $a_1(\iota_1)a_2(\iota_2)\cdots a_n(\iota_n)$, where $a, a_1, \ldots, a_n \in \Sigma$ and, for all $i : 1 \leqslant i \leqslant n$, either $\iota_i \in I$ or $\iota_i$ is of the form $\neg J$ for some set $J \subseteq I$. Symbolic words encode channel states. If the argument is $\iota \in I$, then the value carried by the message is equal to the value of $\iota$. If the argument is $\neg J$, then the value is different from the values of all initiators in $J$. Formally, for an $I$-valuation $\theta$, we define the denotation $[\![v]\!]_\theta$ to be the set of all words $w \in (\Sigma \times \mathcal{D})^*$ of the form $a_1(d_1)a_2(d_2)\cdots a_n(d_n)$ satisfying the following two conditions for all $i : 1 \leqslant i \leqslant n$: (i) if $\iota_i \in I$ then $d_i = \theta(\iota_i)$; and (ii) if $\iota_i = \neg J$ then $d_i \neq \theta(\iota)$ for all $\iota \in J$.

We introduce a class of expressions to generate symbolic words as follows. A *plain atomic expression* over $I$ is of the form $a(\iota) + \epsilon$, where $a \in \Sigma$ and either $\iota \in I$ or $\iota = \neg J$ for some $J \subseteq I$. A *star atomic expression* over $I$ is of the form $(a_1(\iota_1) + \cdots + a_n(\iota_n))^*$, where $a_1, \ldots, a_n \in \Sigma$ and, for all $i : 1 \leqslant i \leqslant n$, either $\iota_i \in I$ or $\iota_i = \neg J$ for some $J \subseteq I$. Sometimes, we write $E^*$ to denote the star expression above where $E = \{a_1(\iota_1), \ldots, a_n(\iota_n)\}$. An *atomic expression* $e$ over $I$ is either a plain or a star atomic expression over $I$. A *product* $p$ over $I$ is of the form: (i) $e_1 \bullet \cdots \bullet e_n$ where $e_1, \ldots, e_n$ are atomic expressions over $I$, or (ii) the empty word $\epsilon$. The denotation $[\![\phi]\!]$ of a product over $I$ is a set of symbolic words defined (in the expected manner) as follows. $[\![a(\iota) + \epsilon]\!] := \{a(\iota), \epsilon\}$, $[\![(a_1(\iota_1) + \ldots + a_n(\iota_n))^*]\!] := \{a_{j_1}(\iota_{j_1}) \bullet \cdots \bullet a_{j_k}(\iota_{j_k}) \mid j_1, \ldots, j_k \in \{1, \ldots, n\}\}$, and $[\![e_1 \bullet \cdots \bullet e_n]\!] := \{v_1 \bullet \cdots \bullet v_n \mid \forall i : 1 \leqslant i \leqslant k.\ v_i \in [\![e_i]\!]\}$. For a product $p$ and an $I$-valuation $\theta$, we define the denotation $[\![p]\!]_\theta := \{w \mid \exists v.\ (v \in [\![p]\!]) \wedge (w \in [\![v]\!]_\theta)\}$. In other words, we first generate the symbolic words in the denotation of $p$ and then instantiate them according to $\theta$.

A *$\mathcal{C}$-indexed Data Simple Regular Expression* (or simply a DSRE) over $I$ is a function $\phi$

that maps each channel $\mathsf{ch} \in \mathcal{C}$ to a product $p$ over $I$. For an $I$-valuation $\theta$, and a DSRE $\phi$, we define a set of channel states $[\![\phi]\!]_\theta$ such that $[\![\phi]\!]_\theta(\mathsf{ch}) := [\![\phi(\mathsf{ch})]\!]_\theta$ for each $\mathsf{ch} \in \mathcal{C}$.

Next, we define a symbolic encoding for (infinite) sets of configurations. A *symbolic configuration* over $I$ is a triple $\beta = \langle q, \mathbb{V}, \phi\rangle$ where $q \in Q$ is a state, $\mathbb{V} \in \texttt{EqRel}(\mathcal{X} \cup I)$ is an equivalence relation over $\mathcal{X} \cup I$, and $\phi$ is a DSRE over $I$. For a set $J \subseteq \mathcal{X} \cup I$, and a $J$-valuation $\theta$, we define the denotation $[\![\beta]\!]_\theta := \{\langle q, V, \omega\rangle \mid \exists \theta' : \mathcal{X} \cup I \mapsto \mathcal{D}.\ (\theta \sqsubseteq \theta') \wedge (\theta' \models \mathbb{V}) \wedge \left(V = \theta'\big|_{\mathcal{X}}\right) \wedge \left(\omega \in [\![\phi]\!]_{\theta'\big|_I}\right)\}$. In other words, it is the set all configurations whose states are $q$, whose variable states respect the equivalence relation $\mathbb{V}$ wrt. the values assigned to the initiators, and where the channel states are defined by the denotation of the given DSRE. Notice that, if $J = \mathcal{X} \cup I$ and $\theta \models \mathbb{V}$ then $[\![\beta]\!]_\theta := \left\{\langle q, V, \omega\rangle \mid \left(V = \theta\big|_{\mathcal{X}}\right) \wedge \left(\omega \in [\![\phi]\!]_{\theta\big|_I}\right)\right\}$. For a set $\mathbb{B}$ of symbolic configurations, we define $[\![\mathbb{B}]\!]_\theta := \cup_{\beta\in\mathbb{B}}[\![\beta]\!]_\theta$. Let $\phi_1 \bullet \phi_2$ be the DSRE $\phi$ where $\phi(\mathsf{ch}) = \phi_1(\mathsf{ch}) \bullet \phi_2(\mathsf{ch})$ for all channels $\mathsf{ch} \in \mathcal{C}$.

## 5.2 The `Post` Operator

The `Post` operator defines the effect of performing transitions on symbolic configurations. It will use two auxiliary operations. First, we give some definitions. We consider an equivalence relation $\mathcal{R} \in \texttt{EqRel}(A)$, and an operation $op$ (that we will specify below). We define $op(\mathcal{R})$ to be a set $B$ that is either empty or a singleton. In the latter case it contains a relation in $\texttt{EqRel}(A)$, which represents the effect of $op$ on $\mathcal{R}$. The set $B$ satisfies one of the following conditions. (i) $op$ is of the form $x \leftarrow y$ and $B = \{\mathcal{R}[x \rightsquigarrow y]\}$, i.e., if $x$ is assigned $y$ then we move $x$ to the block of $y$. (ii) $op$ is of the form $x \leftarrow \circledast$ and $B = \mathcal{R}[x \rightsquigarrow \circledast]$, i.e., $x$ is moved to a newly created block. (iii) $op$ is of the form $x = y$, $[x]_\mathcal{R} = [y]_\mathcal{R}$, and $B = \{\mathcal{R}\}$, i.e., if $x$ and $y$ belong to the same block, then they satisfy the condition of the operation and the relation $\mathcal{R}$ is not changed. (iv) $op$ is of the form $x = y$, $[x]_\mathcal{R} \neq [y]_\mathcal{R}$, and $B = \varnothing$, i.e., since $x$ and $y$ belong to different blocks, they do not satisfy the condition of the operation, and hence the operation is blocked. (v) $op$ is of the form $x \neq y$, $[x]_\mathcal{R} \neq [y]_\mathcal{R}$, and $B = \{\mathcal{R}\}$. (vi) $op$ is of the form $x \neq y$, $[x]_\mathcal{R} = [y]_\mathcal{R}$, and $B = \varnothing$. (vii) $op$ is of the form $\mathsf{ch}!\langle a, x\rangle$ and $B = \{\mathcal{R}\}$, i.e., sending message to a channel does not affect the relation.

Next, we define the effect of *receive* operations on channel states defined by DSREs. Consider a product $p = e_1 \bullet \cdots \bullet e_n$ over $I$. Consider a symbol $a \in \Sigma$ and an initiator $\iota \in I$. We define $p \ominus a(\iota)$ to be the smallest set $A$ containing the following products (with $i : 1 \leqslant i \leqslant n$): (i) if $e_i$ is a plain atomic expression of the form $\{a(\iota) + \epsilon\}$ then $e_1 \bullet \cdots \bullet e_{i-1} \in A$. (ii) if $e_i$ is a plain atomic expression of the form $\{a(\neg J) + \epsilon\}$ where $\iota \notin J$ then $e_1 \bullet \cdots \bullet e_{i-1} \in A$. (iii) if $e_i$ is a star atomic expression of the form $E^*$ where $a(\iota) \in E$ then $e_1 \bullet \cdots \bullet e_i \in A$. (iv) if $e_i$ is a star atomic expression of the form $E^*$ where $a(\neg J) \in E$ and $\iota \notin J$ then $e_1 \bullet \cdots \bullet e_i \in A$. We define $p \ominus a(\neg I)$ to be the smallest set $A$ containing the following products (with $i : 1 \leqslant i \leqslant n$): (i) if $e_i$ is a plain atomic expression of the form $\{a(\neg J) + \epsilon\}$ where $\varnothing \subset J \subseteq I$ then $e_1 \bullet \cdots \bullet e_{i-1} \in A$. (ii) if $e_i$ is a star atomic expression of the form $E^*$ where $a(\neg J) \in E$ and $\varnothing \subset J \subseteq I$ then $e_1 \bullet \cdots \bullet e_i \in A$.

Now, we are ready to define the `Post` operator. For a transition $t = \langle q_1, op, q_2\rangle \in \Delta$, and a symbolic configuration $\beta = \langle q, \mathbb{V}, \phi\rangle$ over $I$, we define $\texttt{Post}(t)(\beta) := \mathbb{B}$ where $\mathbb{B}$ is a set of symbolic configurations. If $q \neq q_1$ then we define $\mathbb{B} := \varnothing$. Otherwise, we define $\mathbb{B}$ as follows. (i) If $op$ is of one of the forms $x \leftarrow y$, $x \leftarrow \circledast$, $x = y$, or $x \neq y$, then $\mathbb{B} = \{\langle q_2, \mathbb{V}', \phi\rangle \mid \mathbb{V}' \in op(\mathbb{V})\}$. (iii) If $op$ is of the form $\mathsf{ch}!\langle a, x\rangle$ and $[x]_\mathbb{V} \cap I \neq \varnothing$ then $\mathbb{B} = \{\langle q_2, \mathbb{V}, \phi[\mathsf{ch} \leftarrow a(\iota) \bullet \phi(\mathsf{ch})]\rangle\}$ where $\iota$ is an arbitrary element of the set $[x]_\mathbb{V} \cap I$. (iv) If $op$ is of the form $\mathsf{ch}!\langle a, x\rangle$ and $[x]_\mathbb{V} \cap I = \varnothing$ then $\mathbb{B} = \{\langle q_2, \mathbb{V}, \phi[\mathsf{ch} \leftarrow a(\neg I) \bullet \phi(\mathsf{ch})]\rangle\}$. (v) If $op$ is of the form $x := \mathsf{ch}?a$

then $\mathbb{B} = \mathbb{B}_1 \cup \mathbb{B}_2 \cup \mathbb{B}_3$, $\mathbb{B}_1 = \cup_{\iota \in I} \{\langle q_2, \mathbb{V}[x \rightsquigarrow \iota], \phi[\mathsf{ch} \leftarrow p]\rangle \mid p \in \phi(\mathsf{ch}) \ominus a(\iota)\}$, $\mathbb{B}_2 = \{\langle q_2, \mathbb{V}[x \rightsquigarrow \circledast], \phi[\mathsf{ch} \leftarrow p]\rangle \mid p \in \phi(\mathsf{ch}) \ominus a(\neg I)\}$, and $\mathbb{B}_3 = \{\langle q_2, \mathbb{V}[x \rightsquigarrow y], \phi[\mathsf{ch} \leftarrow p]\rangle \mid \{(p \in \phi(\mathsf{ch}) \ominus a(\neg I)) \wedge ([y]_{\mathbb{V}} \cap I = \varnothing)\}$. We define $\texttt{Post}(\beta) := \cup_{t \in \Delta} \texttt{Post}(t)(\beta)$. For a set $\mathbb{B}$ of symbolic configurations, we define $\texttt{Post}(\mathbb{B}) := \cup_{\beta \in \mathbb{B}} \texttt{Post}(\beta)$.

## 6 Bounded-Phase Reachability

In this section, we show, for a given symbolic configuration $\beta$, how to characterize the set of bounded-phase reachable configurations from the denotation of $\beta$. For this, we define five types of DLCS, ①, ② . . . , ⑤. We show how to reduce the reachability problem for each type to the reachability problem for a simpler type (one with more restrictions). Furthermore, for the simplest types, we show how to compute the needed sets of symbolic configurations. Finally, we show how to derive the sets of symbolic configurations for the more general types in terms of the sets of symbolic configurations for the simpler types. First, we consider a "graph" view of DLCS that we will use in our construction.

DLCS **as Graphs:** Let $\texttt{GraphOf}(\mathcal{L})$ denote the obvious control-flow graph associated to a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta\rangle$. The set of nodes of $\texttt{GraphOf}(\mathcal{L})$ is exactly the set of states of $\mathcal{L}$ and there is an edge between two nodes $q$ and $q'$ if there is a transition $t \in \Delta$ with $\texttt{SourceOf}(t) = q_1$ and $\texttt{TargetOf}(t) = q_2$ (See Appendix B for more details). For a state $q \in Q$, we use $\texttt{SCCOf}(\mathcal{L}, q)$ to denote the (set of states in the) Strongly Connected Component (SCC) to which $q$ belongs. For a set $Q' \subseteq Q$, we define the DLCS $\mathcal{L}|_{Q'} := \langle Q', \mathcal{X}, \mathcal{C}, \Sigma, \Delta'\rangle$ where $\Delta' := \{\langle q, op, q'\rangle \mid (\langle q, op, q'\rangle \in \Delta) \wedge (q \in Q') \wedge (q' \in Q')\}$.

### 6.1 DLCS Types

In the following, we introduce different types of DLCS.

**Type ① :** A DLCS $\mathcal{L}$ is of type ① if $\Delta = \Delta^{snd}$, i.e., $\mathcal{L}$ has no *receive* transitions.

**Type ② :** Consider an DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta\rangle$. Consider a function $\lambda : Q \mapsto \texttt{EqRel}(\mathcal{X})$ that labels each state of $\mathcal{L}$ with an equivalence relation on the set of variables $\mathcal{X}$. We say that $\mathcal{L}$ is of *type* ② wrt. $\lambda$ if the following conditions are satisfied: (i) $\mathcal{L}$ is of type ①. (ii) Each transition $\langle q_1, op, q_2\rangle \in \Delta$, with $\lambda(q_1) = \mathcal{R}_1$ and $\lambda(q_2) = \mathcal{R}_2$, satisfies $\mathcal{R}_2 \in op(\mathcal{R}_1)$.

We show how to reduce the reachability problem for DLCS of type ① to the reachability problem for DLCS of type ②. To that end, we will define three functions that (i) convert a DLCS $\mathcal{L}$ of type ① to a DLCS $\mathcal{L}'$ of type ②, (ii) convert a configuration of $\mathcal{L}$ to a configuration of $\mathcal{L}'$, (iii) convert a set of configurations of $\mathcal{L}'$ to a set of configurations of $\mathcal{L}$.

We define $\texttt{Convert}^{①.②}(\mathcal{L})$ to be the DLCS $\mathcal{L}' := \left\langle Q', \mathcal{X}', \mathcal{C}', \Sigma', \Delta' \right\rangle$, where $Q' := \{\langle q, \mathcal{R}\rangle \mid (q \in Q) \wedge (\mathcal{R} \in \texttt{EqRel}(\mathcal{X}))\}$, $\mathcal{X}' := \mathcal{X}$, $\mathcal{C}' := \mathcal{C}$, and $\Sigma' := \Sigma$. We let $\Delta' := \{\langle\langle q_1, \mathcal{R}_1\rangle, op, \langle q_2, \mathcal{R}_2\rangle\rangle \mid (\langle q_1, op, q_2\rangle \in \Delta) \wedge (\mathcal{R}_2 \in op(\mathcal{R}_1))\}$. Define the labeling $\lambda$ where $\lambda(\langle q, \mathcal{R}\rangle) := \mathcal{R}$. Notice that $\mathcal{L}'$ is of type ② wrt. $\lambda$ by construction.

For a configuration $\gamma = \langle q, V, \omega\rangle$, we define $\texttt{Convert}^{①.②}(\gamma) := \langle\langle q, \mathcal{R}\rangle, V, \omega\rangle$ where $\mathcal{R}$ is the (unique) equivalence relation $\mathcal{R} \in \texttt{EqRel}(\mathcal{X})$ such that $V \models \mathcal{R}$. In other words, $\mathcal{R}$ is an abstraction of $V$ relating elements that are assigned identical values by $V$. For a set of configurations $\Gamma \subseteq \texttt{ConfsOf}(\mathcal{L}')$, we define $\texttt{Convert}^{②.①}(\Gamma) := \{\langle q, V, \omega\rangle \mid \langle\langle q, \mathcal{R}\rangle, V, \omega\rangle \in \Gamma\}$, i.e., we abstract away the equivalence relation in the definition of a state in $\mathcal{L}'$.

▸ **Lemma 3.** *Suppose $\mathcal{L}$ is of type ①. Then*
$\texttt{Reach}(\mathcal{L})(\gamma) = \texttt{Convert}^{②.①}\left(\texttt{Reach}\left(\texttt{Convert}^{①.②}(\mathcal{L})\right)\left(\texttt{Convert}^{①.②}(\gamma)\right)\right)$.

**Type ③ :** We say that $\mathcal{L}$ is of type ③ wrt. a labeling function $\lambda : Q \mapsto \texttt{EqRel}(\mathcal{X})$ if (i) $\mathcal{L}$ is of type ② wrt. $\lambda$. (ii) $\mathcal{L}$ is an SCC.

**Type ④ :** First we define when is a DLCS $\mathcal{L}$ stable wrt. a subset of variables $\mathcal{Y}$. We say that $\mathcal{L}$ is *stable* wrt. $\mathcal{Y}$ if each variable $y \in \mathcal{Y}$ is only used in operations of the form $x \leftarrow y$, where $x \notin \mathcal{Y}$. In other words, the value of a variable in $\mathcal{Y}$ may be assigned to another variable (outside $\mathcal{Y}$). However, its value is never modified, or compared to other variables. Furthermore, such a variable is not used in *send* operations.

Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X} \cup \mathcal{X}_0, \mathcal{C}, \Sigma, \Delta \rangle$, where $\mathcal{X}_0 = \{x_0 \mid x \in \mathcal{X}\}$, i.e., the set of variables can be partitioned in two subsets $\mathcal{X}$ and $\mathcal{X}_0$ where each variable $x \in \mathcal{X}$ has a corresponding element $x_0 \in \mathcal{X}_0$ (and vice versa). Consider a function $\lambda : Q \mapsto \texttt{EqRel}(\mathcal{X} \cup \mathcal{X}_0)$. We say that $\mathcal{L}$ is of *type* ④ wrt. $\lambda$ if the following conditions are satisfied: (i) $\mathcal{L}$ is of type ③ wrt. $\lambda$, and (ii) $\mathcal{L}$ is stable wrt. $\mathcal{X}_0$. Intuitively, each variable $x_0 \in \mathcal{X}_0$ is used to record the initial value of the corresponding variable $x \in \mathcal{X}$, and hence the former is never used in transitions except when its value is assigned to some other variable.

A state $q \in Q$ is called *proper* wrt. $\lambda$, if $[x]_{\lambda(q)} = [x_0]_{\lambda(q)}$, for all $x \in \mathcal{X}$. Since $\mathcal{L}$ is stable wrt. $\mathcal{X}_0$, it follows that $\lambda(q_1) = \lambda(q_2)$ for all $q_1, q_2 \in Q$ that are *proper* wrt. $\lambda$ and $q_1 \overset{*}{\rightsquigarrow}_{\mathcal{L}} q_2$.

We will define three functions to convert the reachability problem for type ③ DLCS to that of type ④ DLCS, in a similar manner to converting from type ① to type ②.

Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$ of type ③ wrt. a labeling $\lambda : Q \mapsto \texttt{EqRel}(\mathcal{X})$. We will derive a type ④ DLCS from $\mathcal{L}$ in two steps. In the first step, we define $\texttt{Convert}_1^{③.④}(\mathcal{L}, \lambda)$ to be the DLCS $\mathcal{L}' := \left\langle Q', \mathcal{X}', \mathcal{C}', \Sigma', \Delta' \right\rangle$. We define $\mathcal{X}' = \mathcal{X} \cup \mathcal{X}_0$ where $\mathcal{X}_0$ is defined as described above. We define $Q' := \{\langle q, \mathcal{R} \rangle \mid (q \in Q) \wedge (\mathcal{R} \in (\lambda(q) \otimes \mathcal{X}_0)\}$, i.e., a state augments the equivalence relation $\lambda(q)$ defined on the set $\mathcal{X}$ by adding the members of $\mathcal{X}_0$. The members of $\mathcal{X}_0$ may have arbitrary relations with the members of $\mathcal{X}$. We define $\mathcal{C}' := \mathcal{C}$, $\Sigma' := \Sigma$, and $\Delta' := \Delta_1 \cup \Delta_2$, i.e., we include two sets of transitions. We define the set $\Delta_1 := \{\langle\langle q_1, \mathcal{R}_1 \rangle, op, \langle q_2, \mathcal{R}_2 \rangle\rangle \mid \langle q_1, op, q_2 \rangle \in \Delta \wedge \mathcal{R}_2 \in op(\mathcal{R}_1)\}$, i.e, for each transition in $\Delta$, we add a transition performing an identical operation in $t'$ such that the latter respects the equivalence relations on the variables (the ones in both $\mathcal{X}$ and $\mathcal{X}_0$). Notice such a transition will only use the variables in $\mathcal{X}$. Also, we observe that $\Delta_1$ contains operations of the form $x \leftarrow \circledast$ that will assign to $x$ values that are different from the values of all the variables in $\mathcal{X}_0$. Therefore, we need also to add the set $\Delta_2 := \{\langle\langle q_1, \mathcal{R}_1 \rangle, x \leftarrow y, \langle q_2, \mathcal{R}_2 \rangle\rangle \mid (\langle q_1, x \leftarrow \circledast, q_2 \rangle \in \Delta) \wedge (y \in \mathcal{X}_0) \wedge ([y]_{\mathcal{R}_1} \cap \mathcal{X} = \varnothing) \wedge (\mathcal{R}_2 = \mathcal{R}_1[x \rightharpoonup y])\}$.

In the second step of our construction, we only consider a part of the graph of $\mathcal{L}'$, namely those that reachable (in the graph) from a proper state. Consider a state $q \in Q$. We define $\mathcal{R}$ to be the unique relation $\mathcal{R} \in \texttt{EqRel}(\mathcal{X} \cup \mathcal{X}_0)$ such that $\mathcal{R}\big|_{\mathcal{X}} = \lambda(q)$ and $[x]_{\mathcal{R}} = [x_0]_{\mathcal{R}}$ for all $x \in \mathcal{X}$. We define $\texttt{Convert}^{③.④}(\mathcal{L}, \lambda, q) := \texttt{Convert}_1^{③.④}(\mathcal{L}, \lambda)\Big|_{(\langle q, \mathcal{R} \rangle \overset{*}{\rightsquigarrow}_{\mathcal{L}'})}$, i.e., we cut those parts of $\texttt{GraphOf}(\mathcal{L})$ that are not reachable from $\langle q_0, \mathcal{R}_0 \rangle$. Let $\mathcal{L}'' := \left\langle Q'', \mathcal{X}'', \mathcal{C}'', \Sigma'', \Delta'' \right\rangle$ be the resulting DLCS. Define the labeling $\lambda'$ where $\lambda'(\langle q', \mathcal{R}' \rangle) := \mathcal{R}'$. We observe that $\mathcal{L}''$ is an SCC, and hence is of type ④ wrt. $\lambda'$.

Consider a plain configuration $\gamma \in \texttt{ConfsOf}(\mathcal{L})$ such that $\texttt{StateOf}(\gamma) = q$ and $V \models \lambda(q)$. Define $\texttt{Convert}^{③.④}(\gamma) := \gamma' = \langle \langle q, \mathcal{R} \rangle, V', \omega \rangle$ where $V' : (\mathcal{X} \cup \mathcal{X}_0) \mapsto \mathcal{D}$ is the unique variable state such that $V'(x) = V(x)$ and $V'(x_0) = V(x)$, for all $x \in \mathcal{X}$.

We define $\texttt{Convert}^{④.③}(\Gamma) := \left\{\left\langle q, V\big|_{\mathcal{X}}, \omega \right\rangle \mid \langle q, V, \omega \rangle \in \Gamma\right\}$ where $\Gamma \subseteq \texttt{ConfsOf}(\mathcal{L}'')$ is a set of configurations. I.e., we abstract away the values of the variables in $\mathcal{X}_0$.

▸ **Lemma 4.** *Suppose $\mathcal{L}$ is of type ③. Then*
$\texttt{Reach}(\mathcal{L})(\gamma) = \texttt{Convert}^{④.③}\left(\texttt{Reach}\left(\texttt{Convert}^{③.④}(\mathcal{L}, \lambda, q)\right)\left(\texttt{Convert}^{③.④}(\gamma)\right)\right)$.

**Type ⑤ :** A DLCS $\mathcal{L}$ is of type ⑤ if $\Delta = \Delta^{rcv}$, i.e., $\mathcal{L}$ has no *send* transitions.

## 6.2 Computing $k$-Phase Reachability

In this section, we show how to compute a finite set of symbolic configurations that characterize the reachability set. We start by the simplest types of DLCS, and then derive the set of symbolic configurations for one type from those computed for the simpler types. In defining the symbolic configurations, we will use particular sets of initiators. More precisely, for a DLCS with a set of variables $\mathcal{X}$, we will use the set $I_{\mathcal{X}} := \{\iota_x \mid x \in \mathcal{X}\}$, that contains one initiator for each variable in $\mathcal{X}$. Furthermore, for each $i, j \in \mathbb{N}$, we consider the initiator set $I^i_{\mathcal{X}} := \{\iota^i_x \mid x \in \mathcal{X}\}$, and define $I^{[i..j]}_{\mathcal{X}} := \cup_{i \leqslant k \leqslant j} I^k_{\mathcal{X}}$.

**Type ④** Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X} \cup \mathcal{X}_0, \mathcal{C}, \Sigma, \Delta \rangle$ of type ④ wrt. a labeling $\lambda$. Consider a configuration $\gamma \in \texttt{ConfsOf}(\mathcal{L})$. We say that $\gamma$ is *proper* wrt. $\lambda$ if (i) $\gamma$ is plain, (ii) $\texttt{StateOf}(\gamma)$ is proper wrt. $\lambda$, and (iii) $\texttt{VarStateOf}(\gamma) \models \lambda(\texttt{StateOf}(\gamma))$. Proper configurations represent "initial configurations" from which we will start the computations of the system. More precisely, the channels are empty, and each variable $x$ is in the same block as the variable $x_0$ representing its initial value. We will define a set of symbolic configurations over the set $I_{\mathcal{X}}$ that uniformly characterizes the reachable sets from all proper configurations that have a given local state and whose variables satisfy a given equivalence relation. For each $x \in \mathcal{X}$, we use $\iota_x$ to carry the initial value of $x$ in the channels (this value is also stored in $x_0$.)

The characterization is based on several properties of DLCS of type ④. First we observe that if we start from a proper configuration $\gamma$, then for any reachable configuration $\gamma'$ (i.e., $\gamma \xrightarrow{*}_{\mathcal{L}} \gamma'$), it is the case that $\texttt{VarStateOf}(\gamma') \models \lambda(\texttt{StateOf}(\gamma'))$. This implies that all transitions $t$ with $\texttt{SourceOf}(t) = \texttt{StateOf}(\gamma')$ are enabled from $\gamma'$, and hence, from $\gamma'$ we can traverse any sequence of transitions corresponding to a path inside $\texttt{GraphOf}(\mathcal{L})$. Consequently all states in $Q$ are actually reachable from $\gamma$. Furthermore, for all states, we reach exactly the same set of channel states. The reason is that whenever we have a given word $w$ in a channel $\mathsf{ch}$ in a state $q_1$, we can traverse a sequence of transitions leading from $q_1$ to another state $q_2$ while losing all the extra messages we send to $\mathsf{ch}$ along the path, and hence obtaining $w$ in $\mathsf{ch}$ in $q_2$. Finally, the set of channel states can be characterized by a star atomic expression. The reason is that different messages can be sent to the channels aribtrary numbers of times, by traversing the graph of $\mathcal{L}$, and the order among messages is irrelevant (we can obtain any order trough re-sending and losing of messages).

First, we characterize the channel states as a DSRE over $I_{\mathcal{X}}$. For each channel $\mathsf{ch} \in \mathcal{C}$, we define $\texttt{DSREOf}(\mathsf{ch}, \mathcal{L}, \lambda) := E^*$ where $E$ is the smallest star atomic expression $E^*$ over $I_{\mathcal{X}}$, with $E$ containing the following elements: (i) For each $q_1, q_2 \in Q$, $\mathsf{ch} \in \mathcal{C}$, $x \in \mathcal{X}$, $y_0 \in \mathcal{X}_0$, $a \in \Sigma$, such that $[x]_{\lambda(q_1)} = [y_0]_{\lambda(q_1)}$, $\langle q_1, \mathsf{ch}!\langle a, x\rangle, q_2\rangle \in \Delta$, we have that $a(\iota_y) \in E$. Intuitively, the current value of $x$ is equal to the initial value of $y$, and hence the value of the message sent to the channel is encoded by the corresponding initiator $\iota_y$. (ii) For each $q_1, q_2 \in Q$, $\mathsf{ch} \in \mathcal{C}$, $x \in \mathcal{X}$, $a \in \Sigma$, such that $[x]_{\lambda(q_1)} \cap \mathcal{X}_0 = \varnothing$, $\langle q_1, \mathsf{ch}!\langle a, x\rangle, q_2\rangle \in \Delta$, we have that $a(\neg I_{\mathcal{X}}) \in E$. The current value of $x$ is different from the initial values of all the variables, and hence the value of the message is encoded accordingly.

Define the DSRE $\texttt{DSREOf}(\mathcal{L}, \lambda) := \phi$ where $\phi(\mathsf{ch}) := \texttt{DSREOf}(\mathsf{ch}, \mathcal{L}, \lambda)$ for all $\mathsf{ch} \in \mathcal{C}$. Define the set of symbolic configurations $\texttt{SymConfOf}(\mathcal{L}, \lambda) := \{\langle q, \lambda'(q), \texttt{DSREOf}(\mathcal{L}, \lambda)\rangle \mid q \in Q\}$, where $\lambda'$ is defined a follows. Consider a state $q \in Q$. We get $\lambda'(q)$ from $\lambda(q)$ by replacing all occurrences of $x_0$ by the corresponding initiator $\iota_x$, for all $x \in \mathcal{X}$. Notice that $\texttt{SymConfOf}(\mathcal{L}, \lambda)$ is defined over $I_{\mathcal{X}}$.

Consider a configuration $\gamma \in \texttt{ConfsOf}(\mathcal{L})$ that is proper wrt. $\lambda$. Define the valuation

$\theta : I_{\mathcal{X}} \mapsto \mathcal{D}$ such that $\theta(\iota_x) := \texttt{VarStateOf}\,(\gamma)\,(x)$.

▸ **Lemma 5.** $\left\{\left\langle q, V\big|_{\mathcal{X}}, \omega\right\rangle \mid \langle q, V, \omega\rangle \in \texttt{Reach}\,(\mathcal{L})\,(\gamma)\right\} = [\![\texttt{SymConfOf}\,(\mathcal{L}, \lambda)]\!]_\theta$.

**Type** ③ We derive the set of symbolic configurations for DLCS of type ③ from the set of symbolic configurations DLCS of type ④. Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta\rangle$ of type ③ wrt. a labeling $\lambda : Q \mapsto \texttt{EqRel}\,(\mathcal{X})$, and a state $q \in Q$. Define $\mathcal{L}' := \texttt{Convert}^{③.④}\,(\mathcal{L}, \lambda, q)$, $\lambda' := \texttt{Convert}^{③.④}\,(\lambda)$. Define the set of symbolic configurations $\texttt{SymConfOf}\,(\mathcal{L}, \lambda, q) := \texttt{SymConfOf}\,(\mathcal{L}', \lambda')$. Consider a plain configuration $\gamma \in \texttt{ConfsOf}\,(\mathcal{L})$ such that $\texttt{StateOf}\,(\gamma) = q$ and $\texttt{VarStateOf}\,(\gamma) \models \lambda\,(\texttt{StateOf}\,(\gamma))$. Define the valuation $\theta : I_{\mathcal{X}} \mapsto \mathcal{D}$ such that $\theta(\iota_x) := \texttt{VarStateOf}\,(\gamma)\,(x)$. By Lemma 5 and Lemma 4 we get the following lemma.

▸ **Lemma 6.** $\texttt{Reach}\,(\mathcal{L})\,(\gamma) = [\![\texttt{SymConfOf}\,(\mathcal{L}, \lambda, q)]\!]_\theta$.

**Type** ② Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta\rangle$ that is of type ② wrt. a labeling $\lambda : Q \mapsto \texttt{EqRel}\,(\mathcal{X})$. The idea is to consider the SCC graph of $\texttt{GraphOf}\,(\mathcal{L})$, derive the symbolic configurations for each SCC separately, and then combine the results for the different SCCs. We define $\texttt{Depth}\,(\mathcal{L})$ to be the length of the longest simple path in the SCC graph of $\mathcal{L}$. Let $d := \texttt{Depth}\,(\mathcal{L})$. We will use induction on $d$ to compute the set of symbolic configurations using the set of $I_{\mathcal{X}}^{[0..d]}$ of initiators. Intuitively, we first define a set of symbolic configurations over $I_{\mathcal{X}}$ for the current SCC $C$ using the construction for type ③ described above. Then, using induction, we derive a set of symbolic configurations over $I_{\mathcal{X}}^{[0..d-1]}$ corresponding to the paths from $C$ in the SCC graph of $\mathcal{L}$. Finally, we compose these sets to obtain a set over $I_{\mathcal{X}}^{[0..d]}$. First we define some operations on the sets. For $x \in \mathcal{X}$, and $i \in \mathbb{N}$, we define $x^{++} := x$, $(\iota_x)^{++} := \iota_x^0$, and $\left(\iota_x^i\right)^{++} := \iota_x^{i+1}$. In other words, the operation leaves a variable in $\mathcal{X}$ as it is, transforms an initiator in $I_{\mathcal{X}}$ to the corresponding initiator in $I_{\mathcal{X}}^0$, and increases the index of the initiator otherwise. We will now define operations to compose symbolic configurations for a given SCC $C$ with the ones derived for the SCCs below $C$. Consider a relation $\mathcal{R} \in \texttt{EqRel}\left(\mathcal{X} \cup I_{\mathcal{X}}^{[i..j]}\right)$. We define the relation $\mathcal{R}^{++} := \{\langle x^{++}, y^{++}\rangle \mid \langle x, y\rangle \in \mathcal{R}\}$. Notice that $\mathcal{R}^{++} \in \texttt{EqRel}\left(\mathcal{X} \cup I_{\mathcal{X}}^{[i+1..j+1]}\right)$. For relations $\mathcal{R}_1 \in \texttt{EqRel}\,(\mathcal{X} \cup I_{\mathcal{X}})$ and $\mathcal{R}_2 \in \texttt{EqRel}\left(\mathcal{X} \cup I_{\mathcal{X}}^{[0..i]}\right)$, we define $\mathcal{R}_1 \odot \mathcal{R}_2$ to be the set of equivalence relations $\mathcal{R}_3 \in \texttt{EqRel}\left(\mathcal{X} \cup I_{\mathcal{X}}^{[0..i+1]}\right)$ such that ${\mathcal{R}_2}^{++} = \mathcal{R}_3\big|_{\mathcal{X} \cup I_{\mathcal{X}}^{[1..i+1]}}$ and $\left(\mathcal{R}_3 \cap (I_{\mathcal{X}} \times I_{\mathcal{X}}^1)\right) = \left\{\left\langle \iota_x^0, \iota_y^1\right\rangle \mid \langle \iota_x, y\rangle \in \mathcal{R}_1\right\}$. For a DSRE $\phi$ over $I_{\mathcal{X}}$ or over $I_{\mathcal{X}}^{[0..i]}$, we define $\phi^{++}$ to be the DSRE $\phi'$ we get by replacing each occurrence of an initiator $\iota$ in $\phi$ with $\iota^{++}$. Notice that $\phi'$ is defined over $I_{\mathcal{X}}^0$ in the first case, and over $I_{\mathcal{X}}^{[1..i+1]}$ in the second case. Consider symbolic configurations $\beta_1 = \langle q_1, \mathbb{V}_1, \phi_1\rangle$ over $I_{\mathcal{X}}$, and $\beta_2 = \langle q_2, \mathbb{V}_2, \phi_2\rangle$ over $I_{\mathcal{X}}^{[0..i]}$. We define $\beta_1 \odot \beta_2$ to be the set of symbolic configurations $\beta_3 = \langle q_3, V_3, \phi_3\rangle$ where $q_3 = q_2$, $V_3 \in V_1 \odot V_2$, and $\phi_3 = \phi_1^{++} \bullet \phi_2^{++}$. Notice that $\beta_3$ is defined over $I_{\mathcal{X}}^{[0..i+1]}$. Intuitively, $\beta_1$ charaterizes a set of reachable configurations, defined over $I_{\mathcal{X}}$, for a given SCC $C$, while $\beta_2$ charaterizes a set of reachable configurations, defined over $I_{\mathcal{X}}^{[0..d]}$, for the SCCs below $C$.

Consider a state $q \in Q$. We define a set of symbolic configurations $\mathbb{B} := \texttt{SymConfOf}\,(\mathcal{L}, \lambda, q)$ where $\mathbb{B}$ over $I_{\mathcal{X}}^{[0..d]}$. We define $\mathbb{B}$ using induction on $d$ as follows. Define $\mathcal{L}_1 := \mathcal{L}\big|_{\texttt{SCCOf}(\mathcal{L},q)}$, and $\lambda_1 := \lambda\big|_{\texttt{SCCOf}(\mathcal{L},q)}$. Since $\mathcal{L}_1$ is of type ③ wrt. $\lambda_1$, we can, as described above, compute the set of symbolic configurations $\mathbb{B}_1 = \texttt{SymConfOf}\,(\mathcal{L}_1, \lambda_1, q)$ over $I_{\mathcal{X}}$. Define $\mathbb{B}_2 := \bigcup\{\texttt{Post}\,(t)\,(\mathbb{B}_1) \mid (t \in \Delta) \wedge (\texttt{SourceOf}\,(t) = q) \wedge (\texttt{TargetOf}\,(t) \notin \texttt{SCCOf}\,(\mathcal{L}, q))\}$. Intuitively, the set $\mathbb{B}_2$ characterizes the set of configurations we get after performing transitions that connect the SCC corresponding to $\mathcal{L}_1$ to the next SCCs in $\texttt{GraphOf}\,(\mathcal{L})$. We define $\mathbb{B}$ to be

the smallest set containing the following elements. Take any $\beta_2 = \langle q_2, V_2, \phi_2 \rangle \in \mathbb{B}_2$. Define the DLCS $\mathcal{L}_2 := \mathcal{L}\big|_{q_2 \overset{*}{\rightsquigarrow} \mathcal{L}}$. Notice that $\mathscr{d}_2 := \texttt{Depth}\,(\mathcal{L}_2) \leqslant \mathscr{d} - 1$. Define $\lambda_2 := \lambda\big|_{q_2 \overset{*}{\rightsquigarrow} \mathcal{L}}$. By the induction hypothesis, we can compute finite the set $\mathbb{B}_3 := \texttt{SymConfOf}\,(\mathcal{L}_2, \lambda_2, q_2)$ of symbolic configurations over $I_{\mathcal{X}}^{[0..\mathscr{d}_2]}$. For each $\beta_3 \in \mathbb{B}_3$, the set $\mathbb{B}$ contains the set $\beta_2 \odot \beta_3$. Notice that we are taking only finite unions in this definition.

Consider a plain configuration $\gamma \in \texttt{ConfsOf}\,(\mathcal{L})$ such that $\texttt{StateOf}\,(\gamma) = q$ and $\texttt{VarStateOf}\,(\gamma) \models \lambda\,(\texttt{StateOf}\,(\gamma))$. Define the valuation $\theta : I_{\mathcal{X}}^0 \mapsto \mathcal{D}$ such that $\theta(\iota_x^0) := \texttt{VarStateOf}\,(\gamma)\,(x)$. By Lemma 6 we get the following lemma:

▸ **Lemma 7.** $\texttt{Reach}\,(\mathcal{L})\,(\gamma) = [\![\texttt{SymConfOf}\,(\mathcal{L}, \lambda, q)]\!]_\theta$.

**Type** ① Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$ of type ①, a state $q \in Q$, and a relation $\mathcal{R} \in \texttt{EqRel}\,(\mathcal{X})$. Define $\mathcal{L}' := \texttt{Convert}^{①.②}\,(\mathcal{L})$. Define $\texttt{SymConfOf}\,(\mathcal{L}, q, \mathcal{R}) := \texttt{SymConfOf}\,(\mathcal{L}', \lambda, \langle q, \mathcal{R} \rangle)$ where $\lambda\,(\langle q, \mathcal{R} \rangle) = \mathcal{R}$. Consider a plain configuration $\gamma \in \texttt{ConfsOf}\,(\mathcal{L})$ such that $\texttt{StateOf}\,(\gamma) = q$ and $\texttt{VarStateOf}\,(\gamma) \models \mathcal{R}$. Define the valuation $\theta : I_{\mathcal{X}}^0 \mapsto \mathcal{D}$ such that $\theta(\iota_x^0) := \texttt{VarStateOf}\,(\gamma)\,(x)$. By Lemma 7 and Lemma 3 we get the following lemma.

▸ **Lemma 8.** $\texttt{Reach}\,(\mathcal{L})\,(\gamma) = [\![\texttt{SymConfOf}\,(\mathcal{L}, q, \mathcal{R})]\!]_\theta$.

**Type** ⑤ Consider a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$ of type ⑤ Consider a set of symbolic configurations $\mathbb{B}$ defined over an initiator set $I$. We define $\texttt{measure}\,(\mathbb{B}) \in \mathbb{N}$ as follows. For a product $p = e_1 \bullet \cdots \bullet e_n$, we define $\texttt{measure}\,(p) := n$, i.e., it is the length of $p$. For a DSRE $\phi$, we define $\texttt{measure}\,(\phi) := \max_{\mathsf{ch} \in \mathcal{C}} \texttt{measure}\,(\phi(\mathsf{ch}))$. For a symbolic configuration $\beta = \langle q, \mathbb{V}, \phi \rangle$, we define $\texttt{measure}\,(\beta) := \texttt{measure}\,(\phi)$. Finally we define $\texttt{measure}\,(\mathbb{B}) := \max_{\beta \in \mathbb{B}} \texttt{measure}\,(\beta)$. Notice that, for symbolic configurations $\beta_1, \beta_2$, and a transition $t \in \Delta^{rcv}$, if $\beta_2 \in \texttt{Post}\,(t)\,(\beta_1)$ then $\texttt{measure}\,(\beta_2) \leqslant \texttt{measure}\,(\beta_1)$. Notice also that for each $k \in \mathbb{N}$, there are only finitely many symbolic configurations over $I$ with measure $k$.

Consider a finite set $\mathbb{B}$ of symbolic configurations over $I$. Let $\mathbb{B}'$ be the smallest set such that (i) $\mathbb{B} \subseteq \mathbb{B}'$, and (ii) $\mathbb{B}' = \texttt{Post}\,(\mathbb{B}')$. By the two properties of the function $\texttt{measure}$ stated above, it follows that $\mathbb{B}'$ is finite. Consider an $(\mathcal{X} \cup I)$-valuation.

▸ **Lemma 9.** $\texttt{Reach}\,(\mathcal{L})\,([\![\mathbb{B}]\!]_\theta) = [\![\mathbb{B}']\!]_{\theta\big|_I}$

Finally, we get a symbolic representation for the $k$-phase reachable configurations of DLCS.

▸ **Theorem 10.** *Consider a* DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$*, a state* $q \in Q$*, and a relation* $\mathcal{R} \in \texttt{EqRel}\,(\mathcal{X})$*. For any* $k \in \mathbb{N}$*, we can derive an* $\ell \in \mathbb{N}$ *and a finite set* $\mathbb{B}$ *of symbolic configurations over* $I_{\mathcal{X}}^{[0..\ell]}$ *such that for any plain configuration* $\gamma = \langle q, V, \omega \rangle \in \texttt{ConfsOf}\,(\mathcal{L})$*, we have* $\texttt{Reach}_{\mathtt{P}}\,(k)\,(\gamma)\,(q) = [\![\mathbb{B}]\!]_\theta$*, where* $\theta : I_{\mathcal{X}}^0 \mapsto \mathcal{D}$ *is such that* $\theta(\iota_x^0) = V(x)$ *for all* $x \in \mathcal{X}$*.*

**Proof.** Suppose that we have already derived $\mathbb{B}$ for $k$ phases. We will show by induction that we can, for the next phase, derive a new set $\mathbb{B}'$ and $\ell' \in \mathbb{N}$ such that the statement of Theorem 10 is satisfied. If the next phase is a *receive* then we apply the construction of Lemma 9 to derive $\mathbb{B}'$ and define $\ell' := \ell$ (the set of initiators does not change in this case.)

Now, we consider the case where the next phase is a *send*. The next phase will then correspond to a DLCS We define the set $\mathbb{B}'$ to contain all symbolic configurations of the form of $\beta'$ derived below. Let $\beta_1 = \langle q_1, \mathbb{V}_1, \phi_1 \rangle \in \mathbb{B}$. Let $\mathcal{R}_1 := \mathbb{V}_1\big|_{\mathcal{X}}$. Use the construction of Lemma 8 to derive $\mathbb{B}_2 := \texttt{SymConfOf}\,(\mathcal{L}, q_1, \mathcal{R}_1)$. Suppose that $\mathbb{B}_2$ is defined over the set $I_{\mathcal{X}}^{[0..\ell_2]}$. Select any symbolic configuration $\beta_3 = \langle q_3, \mathbb{V}_3, \phi_3 \rangle \in \mathbb{B}_2$. Let $\beta_4 = \langle q_4, \mathbb{V}_4, \phi_4 \rangle$ be the symbolic configuration we get from $\beta_3$ by replacing each initiator $\iota_x^i$ by the initiator $\iota_x^{i+\ell+1}$. Notice that $q_4 = q_3$ and that $\beta_4$ is defined over $I_{\mathcal{X}}^{[\ell+1..\ell+\ell_2+1]}$. Define $\beta' := \langle q_5, \mathbb{V}_5, \omega_5 \rangle$,

where $q_5 = q_4$ and $\omega_5 = \omega_4 \bullet \omega_1$. Furthermore, we select $\mathbb{V}_5 \in \texttt{EqRel}\left(\mathcal{X} \cup I_{\mathcal{X}}^{[0..+\ell+\ell_2+1]}\right)$ to be a relation such that $\mathbb{V}_5\big|_{I_{\mathcal{X}}^{[0..\ell]}} = \mathbb{V}_1$, $\mathbb{V}_5\big|_{I_{\mathcal{X}}^{[\ell+1..\ell+\ell_2+1]}} = \mathbb{V}_4$, and $\mathbb{V}_5 \cap (I_{\mathcal{X}}^{[0..\ell]} \times I_{\mathcal{X}}^{[\ell+1]}) = \left\{\left\langle \iota_x^j, \iota_{\ell+1}^y \right\rangle \mid \left\langle \iota_x^j, y \right\rangle \in \mathbb{V}_1\right\}$. ◂

From Theorem 10 we get decidability of the $k$-bounded-phase reachability problem. More precisely, given a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$, a plain configuration $\gamma = \langle q, V, \omega \rangle$, and a state $\mathsf{Target} \in Q$ we proceed as follows. Let $\mathcal{R} \in \texttt{EqRel}\,(\mathcal{X})$ be the unique relation such that $V \models \mathcal{R}$. Derive a set $\mathbb{B}$ of symbolic configurations as described in Theorem 10, and check whether the state of any member of $\mathbb{B}$ is equal to $\mathsf{Target}$.

## 7 Conclusions and Future Work

We have presented a model, called DLCS, that generalizes lossy channel systems by augmenting the model with a finite set of variables and channel messages that carry values from an infinite data domain. We have shown undecidability of the reachability problem, and decidability of the bounded phase reachability problem.

A direction for future work is to bridge the gap between the lower and upper bounds on complexity. Another direction for future work is to consider bounded-phase reachability under a richer set of relations on the data domain, e.g., by allowing gap-order constraints [3] on the set of variables instead of only equalities and disequalities. Finally we intend to study context-bounded analysis [12, 5] as another way to achieve decidability results for DLCS.

## References


1. P. A. Abdulla, M. F. Atig, and J. Cederberg. Timed lossy channel systems. In *FSTTCS*, volume 18 of *LIPIcs*, pages 374–386, 2012.
2. P. A. Abdulla, M. F. Atig, and J. Cederberg. Analysis of message passing programs using SMT-solvers. In *ATVA*, volume 8172 of *LNCS*, pages 272–286, 2013.
3. P. A. Abdulla, M. F. Atig, G. Delzanno, and A. Podelski. Push-down automata with gap-order constraints. In *FSEN*, volume 8161 of *LNCS*, pages 199–216, 2013.
4. P. A. Abdulla and B. Jonsson. Verifying programs with unreliable channels. In *LICS*, pages 160–170. IEEE Computer Society, 1993.
5. C. Aiswarya, P Gastin, and K. Narayan Kumar. Controllers for the verification of communicating multi-pushdown systems. In *CONCUR*, volume 8704 of *LNCS*, pages 297–311, 2014.
6. M. F. Atig, A. Bouajjani, S. Burckhardt, and M. Musuvathi. On the verification problem for weak memory models. In *POPL*, pages 7–18. ACM, 2010.
7. A. Bouajjani and M. Emmi. Bounded phase analysis of message-passing programs. In *TACAS*, volume 7214 of *LNCS*, pages 451–465, 2012.
8. L. Clemente, F. Herbreteau, A. Stainer, and G. Sutre. Reachability of communicating timed processes. In *FOSSACS 2013*, volume 7794 of *LNCS*, pages 81–96. Springer, 2013.
9. S. Demri and R. Lazić. Ltl with the freeze quantifier and register automata. *ACM Trans. Comput. Logic*, 10(3):16:1–16:30, 2009.
10. A. Finkel. Decidability of the termination problem for completely specified protocols. *Distributed Computing*, 7(3):129–135, 1994.
11. A. Heußner, J. Leroux, A. Muscholl, and G. Sutre. Reachability analysis of communicating pushdown systems. *Logical Methods in Computer Science*, 8(3), 2012.
12. S. La Torre, P. Madhusudan, and G. Parlato. Context-bounded analysis of concurrent queue systems. In *TACAS*, volume 4963 of *LNCS*, pages 299–314. Springer, 2008.

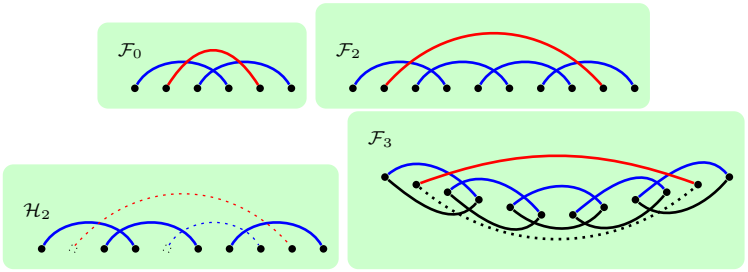


**Figure 2** Illustrations of $\mathcal{F}_n$ and $\mathcal{H}_n$.

## A Undecidability of Reachability

In this section, we reduce the reachability problem for 2-counter machines to the reachability problems for DLCS. The main idea of the reduction is to encode the value of each counter using a family of sets $\mathcal{F}_n$ (described below). These sets have an important feature (captured by Lemma 11), namely that we cannot obtain one set from another through the deletion of messages. This makes them good candidates for the exact representation of counter values (in the the presence of losses) in the channels. A counter value cannot be "accidentally" reduced through the loss of messages. This would give an element of a different set $\mathcal{H}_n$. As we shall see below, our simulation can recognize that this has happened. In such a case, we make sure that the simulation will never reach the target state. After presenting the encoding, we describe how to translate an instance of the reachability problem for 2-counter machines to an equivalent instance of the reachability problem for DLCS. We will give three gadgets that simulate the operations of decrementing, incrementing, and testing the value of a counter respectively. Using this, we describe how a run of the counter machine is simulated by a run of the DLCS.

### Encoding

We present a family of sets that we use to encode counter values. Define the set $\mathcal{E} := \mathcal{D}^*$, i.e., $\mathcal{E}$ is the set of words over $\mathcal{D}$. We define a family $\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \ldots \subset \mathcal{E}$ such that $w \in \mathcal{F}_n$ if the following two conditions are satisfied: (i) $|w| = 2n + 6$, and (ii) $w[i_1] = w[i_2]$ iff either (2a) $i_1 = 2$ and $i_2 = |w| - 1$, or (2b) $1 \leqslant i_1 \leqslant |w| - 3$, $i_1 \mod 2 = 1$, and $i_2 = i_1 + 3$. In other words, two elements are equal if either (i) the first element is in an odd position and the second element is three positions away, or (ii) they are the second and next last elements respectively. As an example, if we take $\mathcal{D}$ to be $\mathbb{N}$, then an element of $\mathcal{F}_2$ would be 4 2 5 4 3 5 8 3 2 8. We can imagine the sets $\mathcal{F}_n$ as words with edges that connect the partner elements. Illustrations of elements in the sets $\mathcal{F}_0$ and $\mathcal{F}_2$ are given in the Fig. 2, where a line connecting two positions indicates the values stored in these positions are equal. We notice that each word in $\mathcal{F}_n$ consists of pairs of elements with identical values. We call each element of a pair, a (*left* or *right*) *partner* of the other. For instance, in a word in $\mathcal{F}_3$, element 5 is the left partner of 8, and 10 is the right partner of 7. For each $\mathcal{F}_n$ we distinguish six elements that we call the *pivot elements*. The three left-most symbols are called the first, second, and third left pivot elements; and the three right-most symbols are called the first, second, and third right pivot elements. We call the rest of the elements in the word *intermediate elements*. For instance, in a word in $\mathcal{F}_2$, the pivot elements are the ones in the first, second, third, $8^{th}$, $9^{th}$, and $10^{th}$ elements respectively; while the $4^{th}$, $5^{th}$, $6^{th}$ and $7^{th}$ elements are intermediate. Notice that the second left pivot and the right second pivot are

always partners. Also, notice that a word in $\mathcal{F}_0$ only contains pivot elements, and that each left pivot element is the left partner of the corresponding right pivot element.

For $n \in \mathbb{N}$, we define $\mathcal{G}_n := \mathcal{F}_n \downarrow$ and define $\mathcal{H}_n := \mathcal{G}_n - \mathcal{F}_n$. In other words, we get a member of $\mathcal{H}_n$ by deleting some elements in $\mathcal{F}_n$. Fig. 2 shows a member of $\mathcal{H}_2$ derived by deleting the second and fifth elements. We define $\mathcal{F} := \cup_{n \geqslant 0} \mathcal{F}_n$, $\mathcal{G} := \cup_{n \geqslant 0} \mathcal{G}_n$, and $\mathcal{H} := \cup_{n \geqslant 0} \mathcal{H}_n$. The main property of $\mathcal{F}$ is captured by the following lemma, namely, for $m \neq n$, we cannot derive $\mathcal{F}_m$ from $\mathcal{F}_n$ though the deletion of elements.

▸ **Lemma 11.** $\mathcal{F} \cap \mathcal{H} = \varnothing$.

**Proof.** Notice that the any data word from the set $\mathcal{F}$ forms a closed inter-linked chain. (The red pivot closes the chain at the back.) If some message (or letter data pair ) is lost, then a link is broken and the chain opens up. Losing more messages will only result in breaking more links, and hence the chain once open can never be closed again. ◂

## 2-Counter Machines

We recall the standard model of 2-counter machines operating on two counters $\mathsf{c}_1$ and $\mathsf{c}_2$. Such a machine is a triple $\mathcal{M} = \langle Q, q_{init}, \Delta \rangle$, where $Q$ is a finite set of (local) states, $q_{init} \in Q$ is the initial state, and $\Delta$ is a finite set of transitions. A transition is a triple $\langle q, op, q' \rangle$ where $q, q' \in Q$ are states, and $op$ is an operation of one of the three forms (where $\mathsf{c} \in \{\mathsf{c}_1, \mathsf{c}_2\}$): (i) $\mathtt{inc}(\mathsf{c})$ increases the value of $\mathsf{c}$ by one; (ii) $\mathtt{dec}(\mathsf{c})$ decreases the value of $\mathsf{c}$ by one; (iii) $\mathsf{c} = 0?$ checks whether the value of $\mathsf{c}$ is equal to zero. The machine $\mathcal{M}$ induces a transition system as follows. A configuration $\gamma$ is a triple $\langle q, n_1, n_2 \rangle$ where $q \in Q$ gives the state of $\mathcal{M}$, and $n_1, n_2 \in \mathbb{N}$ are the values of the counters. The transition relation $\longrightarrow_{\mathcal{M}}$ is the standard one for Minsky machines. An instance of the reachability problem consists of a state $\mathsf{Target} \in Q$, and the task is to check whether $\langle q_{init}, 0, 0 \rangle \xrightarrow{*}_{\mathcal{M}} \langle \mathsf{Target}, n_1, n_2 \rangle$ for some $n_1, n_2 \in \mathbb{N}$. It is well-known that this problem is undecidable.

## Translation

Consider an instance of the reachability problem for 2-counter machines defined by a 2-counter machine, $\mathcal{M} = \langle Q^{\mathcal{M}}, q^{\mathcal{M}}_{init}, \Delta^{\mathcal{M}} \rangle$ and a state $\mathsf{Target} \in Q$. We will construct an equivalent instance of the reachability problem for DLCS. More precisely, we will derive a DLCS $\mathcal{L} = \langle Q^{\mathcal{L}}, \mathcal{X}^{\mathcal{L}}, \mathcal{C}^{\mathcal{L}}, \Sigma^{\mathcal{L}}, \Delta^{\mathcal{L}} \rangle$ with $\mathsf{Target} \in Q^{\mathcal{L}}$, and a plain configuration $\gamma^{\mathcal{L}}_{init} \in \mathtt{ConfsOf}\,(\mathcal{L})$, such that $\gamma^{\mathcal{L}}_{init} \xrightarrow{*}_{\mathcal{L}} \mathsf{Target}$ iff $\gamma^{\mathcal{M}}_{init} = \langle q^{\mathcal{M}}_{init}, 0, 0 \rangle \xrightarrow{*}_{\mathcal{M}} \langle \mathsf{Target}, n_1, n_2 \rangle$ for some $n_1, n_2 \in \mathbb{N}$. The DLCS $\mathcal{L}$ operates on two channels $\mathsf{ch}_1$ and $\mathsf{ch}_2$ which are used to encode the values of the counters $\mathsf{c}_1$ and $\mathsf{c}_2$ respectively. The message alphabet is defined by $\Sigma = \{a, \lhd, \rhd\}$. The occurrences of $a$ together with the associated data values encode some set $\mathcal{F}_n$ corresponding to the value $n$ of a counter. The symbols $\lhd$ and $\rhd$ are end-markers: they mark the left and right ends of the word of messages inside the channel. A word $w \in (\{a, \lhd, \rhd\} \times \mathcal{D})^*$ is said to be an *encoding* of *value* $n$ if $w$ is of the form $\langle \lhd, d \rangle \langle a, d_1 \rangle \langle a, d_2 \rangle \cdots \langle a, d_k \rangle \langle \rhd, d' \rangle$, and $d_1 d_2 \cdots d_k \in \mathcal{F}_n$ (the values $d$ and $d'$ can be chosen arbitrarily and have no effect on the encoding.) We say that $w$ a *semi-encoding* of *value* $n$ if $w$ is of one of the forms (i) $\langle \lhd, d \rangle \langle a, d_1 \rangle \langle a, d_2 \rangle \cdots \langle a, d_k \rangle \langle \rhd, d' \rangle$, (ii) $\langle a, d_1 \rangle \langle a, d_2 \rangle \cdots \langle a, d_k \rangle \langle \rhd, d' \rangle$, (iii) $\langle \lhd, d \rangle \langle a, d_1 \rangle \langle a, d_2 \rangle \cdots \langle a, d_k \rangle$ or (iv) $\langle a, d_1 \rangle \langle a, d_2 \rangle \cdots \langle a, d_k \rangle$ where $d_1 d_2 \cdots d_k \in \mathcal{G}_n$. By Lemma 11 it follows that if $\mathsf{ch}$ is a semi-encoding of value $n$ then it cannot be an encoding of value $m$ for any $m \neq n$. A channel state $\omega$ of $\mathcal{L}$ is said to be an *encoding* (of *values* $n_1$ and $n_2$) if $\omega(\mathsf{ch}_1)$ and $\omega(\mathsf{ch}_2)$ are encodings (of values $n_1$ and $n_2$ respectively). For each state $q \in Q^{\mathcal{M}}$ there is a copy $q \in Q^{\mathcal{L}}$. Furthermore, $Q^{\mathcal{L}}$ contains a number of additional

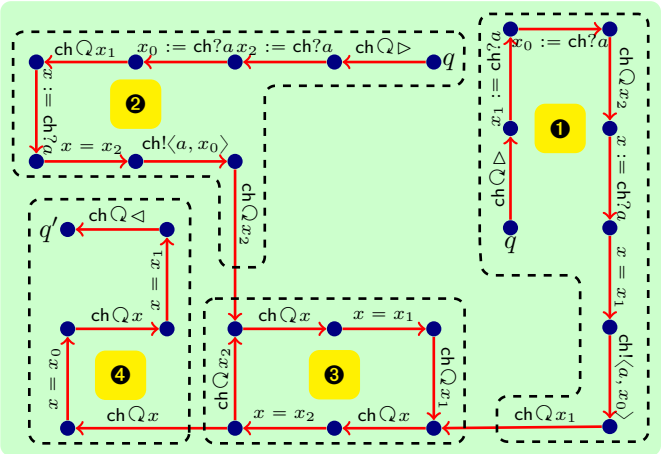


**Figure 3** Decrementing the value of a counter

"temporary" states that are used to perform "intermediate steps" in the simulation. These states will be given names like $\mathsf{tmp}$, $\mathsf{tmp}_t^1$, $\mathsf{tmp}_t^{12}$, etc. Whenever we introduce such a new state in the simulation we assume that it is unique (different from all other states, whether temporary or not). A configuration $\langle q, n_1, n_2 \rangle$ of $\mathcal{M}$ is encoded by configurations of $\mathcal{L}$ of the form $\langle q, V, \omega \rangle$ where $\omega$ is an encoding of values $n_1$ and $n_2$. Finally, the set of variables is given by $\mathcal{X} = \{x, x_0, x_1, x_2, x_3\}$.

We introduce some syntactic sugar that we will use in the rest of the section. Consider a channel $\mathsf{ch}$ and states $q_1, q_2$. We use $\langle q_1, \mathsf{ch}❍◁, q_2 \rangle$ to denote the sequence of transitions $\langle q_1, x := \mathsf{ch}?◁, \mathsf{tmp} \rangle \langle \mathsf{tmp}, \mathsf{ch}!\langle x, ◁ \rangle, q_2 \rangle$. In other words, we rotate the left end-marker by first receiving it, storing its value in $x$, and then sending it back to the channel (in fact, the value of $◁$ will not be relevant in our simulation). We define the operation $\mathsf{ch}❍▷$ analogously. For $y \in \{x, x_0, x_1, x_2, x_3\}$, we use $\langle q_1, \mathsf{ch}❍y, q_2 \rangle$ to denote the sequence of transitions $\langle q_1, y := \mathsf{ch}?a, \mathsf{tmp} \rangle \langle \mathsf{tmp}, \mathsf{ch}!\langle y, a \rangle, q_2 \rangle$. In other words, we rotate the next $a$-symbol in the channel storing its value in $y$.

## Decrementing

Consider a transition $\langle q, \mathsf{dec}(\mathsf{c}), q' \rangle \in \Delta^{\mathcal{M}}$. Let $\mathsf{ch}$ be the channel used for encoding the value of $\mathsf{c}$ (i.e., $\mathsf{ch} = \mathsf{ch}_1$ if $\mathsf{c} = \mathsf{c}_1$ and $\mathsf{ch} = \mathsf{ch}_2$ if $\mathsf{c} = \mathsf{c}_2$). The gadget for decrementing the value of $\mathsf{c}$ is shown in Fig. 3. It performs two tasks (i) it checks whether the content of $\mathsf{ch}$ is an encoding of value $n$ for some $n > 0$, and (ii) at the same time transforms the content of $\mathsf{ch}$ to an arbitrary semi-encoding of value $n - 1$ (in particular it may transform the content of $\mathsf{ch}$ to an encoding of value $n - 1$). If the test in task (i) fails at any point of the simulation the system is immediately blocked. To obtain a (semi-) encoding of value $n - 1$, we do the following: (i) Remove the third right pivot together with its partner. (ii) Keep the role of the second right pivot. (iii) Transform the first right pivot to the third right pivot. (iv) Transform the fifth right-most element into the first right pivot. This is carried out by four segments (sequences) of transitions, called ❶, ❷, ❸, and ❹, described below. **Segments ❶ and ❷** These segments are the initial phase of the gadget. They correspond to the case where $n$ is odd and even respectively. We will consider the case where $n$ is odd (segment ❶). The case where $n$ is even (segment ❷) is similar. In segment ❶, we start from $q$ and perform the following steps: (i) Rotate the right end-marker. (ii) Remove the third right pivot and

store its value in $x_1$. (iii) Receive the second pivot and store its value in $x_0$. (iv) Rotate the first right pivot, which means that it will now become the third right pivot. (v) Remove the partner of the third right pivot and store its value in $x$ (together with step (ii), this means that we have now removed both the third right pivot and its partner). (vi) check whether the values of the third pivot and its partner are equal. This is done by comparing the values of $x$ and $x_1$. This step is needed since we want to ensure that the content of the channel is an encoding. (vii) Put back the second right pivot (Recall that the value of the second right pivot was stored in $x_0$ in step (iii)). (viii) Rotate the new third right pivot. (ix) Enter segment ❸. Observe that if the simulation starts from the segment ❶ and an odd value of $n$, then it will get blocked during the simulation of the segment ❸ and therefore will never succeed to enter into the segment ❹.

**Segment ❸.** In this segment, the intermediate elements are checked and rotated in the channel. We read the values of the right partners and compare them with the values of their left partners. Recall that such partners are three positions apart in the channels, and that their values should be equal. During this segment, the variables $x_1$ and $x_2$ hold the values of the two most recently read right partners. The value of the most recently read left partner is stored in the variable $x$. **Segment ❹** We perform the following steps: (i) Rotate the next message which we guess to be the second left pivot and store its value in $x$ (if the guess fails then the simulation is halted). (ii) Compare the value of this message for equality with the value $x_0$ (which still holds the value of the second right pivot). (iii) Rotate the next message which we expect to be the first left pivot and store its value in $x$. (iv) Compare the value of this message for equality with the value $x_1$ (which holds the value of most recently read right partner, i.e., the supposed partner of the first left pivot). (v) Rotate the left end-marker and reach $q'$.

## Incrementing

Consider a transition $\langle q, \mathtt{inc}(\mathsf{c}), q' \rangle \in \Delta^{\mathcal{M}}$. Let $\mathsf{ch}$ be the channel used for encoding the value of $\mathsf{c}$. The gadget for incrementing the value of $\mathsf{c}$ is shown in Fig. 5. In a similar manner to the case of decrementing, the gadget for incrementing the value of $\mathsf{c}$ performs two tasks (i) it checks whether the content of $\mathsf{ch}$ is an encoding of value $n$ for some $n \in \mathbb{N}$, and (ii) transforms the content of $\mathsf{ch}$ to an arbitrary semi-encoding of value $n+1$ (also here, it may transform the content of $\mathsf{ch}$ to an encoding of value $n+1$). Even in this case, if the test in step (i) fails at any point, the system will be blocked. To obtain a (semi-) encoding of value $n+1$, we do the following: (i) Keep the right pivots as they are. (ii) Keep the intermediate elements. (iii) Transform the third left pivot to an intermediate element. (iv) Transform the first left pivot into the third left pivot. (v) Keep the second left pivot. (vi) Add a new first left pivot with a fresh value (together with its right partner). The gadget consists of four segments of transitions, namely, ❶, ❷, ❸, and ❹. We first introduce a gadget that produces a value that is *fresh* wrt. a channel, i.e., a value that is different from all values that are currently inside the channel.

**Generating Fresh Values** To get a value that is fresh wrt. channel $\mathsf{ch}$, we generate a locally fresh value and then rotate the contents of $\mathsf{ch}$ (see Fig. 4.) In each rotation step, we check that the generated value is different from the value that is being rotated. Concretely, we perform the following steps: (i) Assign an arbitrary value to $x_1$. (ii) Rotate the right end-marker. (iii) Rotate each message in the channel storing its value in $x_2$ and then compare (to check disequality) with the newly generated value (stored in $x_1$). (iv) Rotate the left end-marker. We use $x \leftarrow \mathit{fresh}(\mathsf{ch})$ to denote that we generate a value that is fresh wrt.

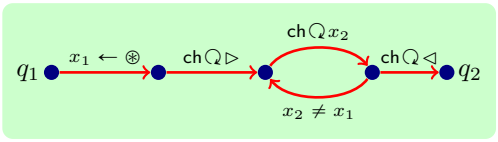


**Figure 4** Generating a fresh value.

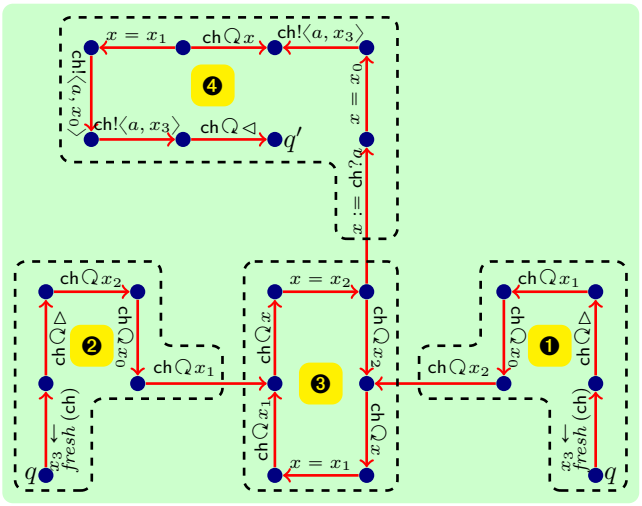


**Figure 5** Incrementing the value of a counter

channel ch, and store the value in the variable $x$.

**Segments ❶ and ❷.** These segments are the initial phase of the gadget, and they handle changes to the right pivots. They correspond to the case where $n$ is odd and even respectively. We will consider the case where $n$ is odd (segment ❶). In segment ❶, we start from $q$ and perform the following steps: (i) Use the freshness gadget to generate a new value that we store in the variable $x_3$. This value will be later used to define the new left pivot together with its right partner. (ii) Rotate the right end-marker. (iii) Rotate the third, second, and first right pivots and store them in the variables $x_1$, $x_0$, and $x_2$ respectively. These messages are rotated (and not removed) since they will keep their places in the new channel content.

**Segment ❸.** This segment is similar to the case of decrementing. The intermediate elements are checked and rotated to the channel in an identical manner.

**Segment ❹.** We perform the following steps: (i) Remove the next message which we guess to be the old second left pivot and store its value in $x$ (if the guess fails then the simulation is halted). This message will keep its status as the second left pivot, and will therefore be added back to the channel in a later step. (ii) Compare the value of this message for equality with the value $x_0$ (which still holds the value of the second right pivot). (iii) Insert the right partner of the new first left pivot. The value of this message is still stored in $x_3$. (iv) Rotate the next message which expect to be the old first left pivot and store its value in $x$. By this rotation, the message will become the third left pivot element. (v) Compare the value of this message for equality with the value $x_1$ which holds the value of the current third left pivot (i.e., the value of the old first left pivot). (vi) Insert the new second left pivot (whose value is stored in $x_0$). (vii) Insert the new first left pivot (whose value is stored in $x_3$). (viii) Rotate the left end-marker.

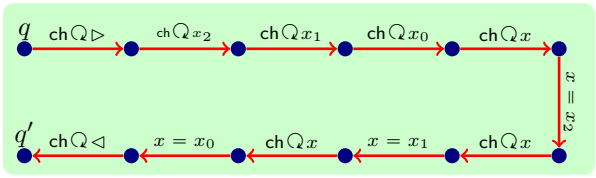


**Figure 6** Testing whether the value of a counter is equal to 0.

## Zero Testing

Consider a transition $\langle q, \mathsf{c} = 0?, q' \rangle \in \Delta^{\mathcal{M}}$. Let $\mathsf{ch}$ be the channel used for encoding the value of $\mathsf{c}$. The gadget for checking whether the value of $\mathsf{c}$ is equal to zero is shown in Fig. 6. It checks whether the content of $\mathsf{ch}$ is an encoding of value 0. As a side effect of the testing the content of the channel may be changed to an arbitrary semi-encoding of value 0. In particular, it may keep the content of $\mathsf{ch}$ as an encoding of value 0 (if no messages are lost in the channel during the simulation). If the content of the channel is not an encoding of value 0 then the simulation is blocked. We start from $q$ and perform the following steps: (i) Check that the right end-marker is the last message in the channel, in which case we rotate it. (ii) Rotate the next three messages. These messages are supposed to contain the values of the third, second, and first right pivots. We store these values in the variables $x_2$, $x_1$, and $x_0$ respectively. (iii) Rotate the next message. This message is supposed to contain the value of the third left pivot. We store this value in the variable $x$. Since the values of the third left and right pivots should be equal, we compare the values of $x$ and $x_2$ for equality. We repeat the procedure for the second and first pivots. (iv) We rotate the left end-marker, and enter state $q'$.

## Simulation

$\mathcal{L}$ initializes the contents of the channels $\mathsf{ch}_1$ and $\mathsf{ch}_2$. More precisely, $\mathcal{L}$ starts from $q^{\mathcal{L}}_{init}$ (with empty channels) and sends two arbitrary sequences of messages to the channels $\mathsf{ch}_1$ and $\mathsf{ch}_2$. Let $\gamma^{\mathcal{L}}_{init} = \langle q^{\mathcal{L}}_{init}, V_{init}, \omega_{init} \rangle$ be the initial configuration of $\mathcal{L}$ with $\omega_{init}(\mathsf{ch}_1) = \omega_{init}(\mathsf{ch}_2) = \epsilon$. Then, it checks whether the sent sequences form encodings of size 0 (using the gadget for zero testing). If the test is successful, $\mathcal{L}$ enters the state $q^{\mathcal{M}}_{init}$ from which it starts the proper simulation. In such a way, for each computation $\pi$ of $\mathcal{M}$ from $\gamma^{\mathcal{M}}_{init} = \langle q^{\mathcal{M}}_{init}, 0, 0 \rangle$ to a configuration of the form $\langle \mathsf{Target}, n_1, n_2 \rangle$, there is a computation $\pi'$ of $\mathcal{L}$ from $\gamma^{\mathcal{L}}_{init}$ to a configuration of the form $\langle \mathsf{Target}, V, \omega \rangle$ where $\omega$ is an encoding of values $n_1$ and $n_2$. On the other hand, since each gadget represents one transition in $\mathcal{M}$, and the simulation halts immediately if it detects a semi-encoding (if any messages have been lost inside the channels) it follows that if there is a computation $\pi$ of $\mathcal{L}$ from $\gamma^{\mathcal{L}}_{init}$ to a configuration of the form $\langle \mathsf{Target}, V, \omega \rangle$ then there is a computation $\pi'$ in $\mathcal{M}$ from $\gamma^{\mathcal{M}}_{init}$ to a configuration of the form $\langle \mathsf{Target}, n_1, n_2 \rangle$. This gives the following theorem.

▸ **Theorem 12.** *The reachability problem for 2-counter machines is reducible to the reachability problem for* DLCS.

Theorem 1 follows.

## Discussions

The undecidability result can be strengthened in several ways. For instance, we can show undecidability for DLCS that are restricted to have a single channel, by concatenating the

encoding of the two counters in the given channel. We can also remove the endmarkers used in the simulation thus obtaining undecidability for the case where the channel alphabet is a singleton. We use five variables in our undecidability proof, and leave open the case where we restrict the DLCS to have a fewer number of variables (the case of a single variable can be easily reduced to the case of LCS without data.)

## B The graph associated to a DLCS

Sometimes, we view a DLCS $\mathcal{L} = \langle Q, \mathcal{X}, \mathcal{C}, \Sigma, \Delta \rangle$ as a graph $\texttt{GraphOf}(\mathcal{L})$ where each node in the graph is a state $q \in Q$, and where there is an edge between two nodes $q_1$ and $q_2$ if there is a transition $t \in \Delta$ with $\texttt{SourceOf}(t) = q_1$ and $\texttt{TargetOf}(t) = q_2$. We write $q_1 \rightsquigarrow_{\mathcal{L}} q_2$ to denote the existence of such an edge, and use $\overset{*}{\rightsquigarrow}_{\mathcal{L}}$ to denote the reflexive transitive closure of $\rightsquigarrow_{\mathcal{L}}$. We define $(q\overset{*}{\rightsquigarrow}_{\mathcal{L}}) := \left\{ q' \mid q\overset{*}{\rightsquigarrow}_{\mathcal{L}} q' \right\}$. Using the graph view, we will use graph terms to describe properties of $\mathcal{L}$. For instance, by "$\mathcal{L}$ is a Strongly Connect Component (SCC)" we mean that $\texttt{GraphOf}(\mathcal{L})$ is an SCC, and by "the SCC graph of $\mathcal{L}$" we mean the SCC graph of $\texttt{GraphOf}(\mathcal{L})$, etc.