ON TERMINATION FOR FAULTY CHANNEL MACHINES

PATRICIA BOUYER [1], NICOLAS MARKEY [1], JOËL OUAKNINE [2], PHILIPPE SCHNOEBELEN [1], AND JAMES WORRELL [2]

[1] LSV, ENS Cachan, CNRS
61 Av. Pdt. Wilson, F-94230 Cachan, France
`{bouyer,markey,phs}@lsv.ens-cachan.fr`

[2] Oxford University Computing Laboratory
Wolfson Bldg., Parks Road, Oxford OX1 3QD, UK
`{joel,jbw}@comlab.ox.ac.uk`

ABSTRACT. A *channel machine* consists of a finite controller together with several fifo channels; the controller can read messages from the head of a channel and write messages to the tail of a channel. In this paper, we focus on channel machines with *insertion errors*, i.e., machines in whose channels messages can spontaneously appear. Such devices have been previously introduced in the study of Metric Temporal Logic. We consider the *termination* problem: are all the computations of a given insertion channel machine finite? We show that this problem has non-elementary, yet primitive recursive complexity.

1. Introduction

Many of the recent developments in the area of automated verification, both theoretical and practical, have focussed on infinite-state systems. Although such systems are not, in general, amenable to fully algorithmic analysis, a number of important classes of models with decidable problems have been identified. Several of these classes, such as Petri nets, process algebras, process rewrite systems, faulty channel machines, timed automata, and many more, are instances of *well-structured transition systems*, for which various problems are decidable—see [7] for a comprehensive survey.

Well-structured transition systems are predicated on the existence of 'compatible well-quasi orders', which guarantee, for example, that certain fixed-point computations will terminate, etc. Unfortunately, these properties are often non-constructive in nature, so that although convergence is guaranteed, the rate of convergence is not necessarily known. As a result, the computational complexity of problems involving well-structured transition systems often remains open.

In this paper, we are interested in a particular kind of well-structured transition systems, known as faulty channel machines. A channel machine (also known as a queue automaton) consists of a finite-state controller equipped with several unbounded fifo channels (queues, buffers). Transitions of the machine can write messages (letters) to the tail of a channel and read messages from the head of a channel. Channel machines can be used, for example, to model distributed protocols that communicate asynchronously.

Channel machines, unfortunately, are easily seen to be Turing powerful [3], and all non-trivial verification problems concerning them are therefore undecidable. In [1, 6, 4, 2], Abdulla and Jonsson, and Finkel *et al.* independently introduced *lossy channel machines* as channel machines operating over an unreliable medium; more precisely, they made the assumption that messages held in channels could at any point vanish nondeterministically. Not only was this a compelling modelling assumption, more adequately enabling the representation of fault-tolerant protocols, for example, but it also endowed the underlying transition systems of lossy channel machines with a well-structure, thanks to Higman's lemma [8]. As a result, several non-trivial problems, such as control-state reachability, are decidable for lossy channel machines.

Abdulla and Jonsson admitted in [1] that they were unable to determine the complexity of the various problems they had shown to be decidable. Such questions remained open for almost a decade, despite considerable research interest in the subject from the scientific community. Finally, Schnoebelen showed in [16] that virtually all non-trivial decidable problems concerning lossy channel machines have non-primitive recursive complexity. This result, in turn, settled the complexity of a host of other problems, usually via reduction from reachability for lossy channel machines. Recently, the relevance of the lossy channel model was further understood when it was linked to a surprisingly complex variant of Post's correspondence problem [5].

Other models of unreliable media in the context of channel machines have also been studied in the literature. In [4], for example, the effects of various combinations of insertion, duplication, and lossiness errors are systematically examined. Although insertion errors are well-motivated (as former users of modems over telephone lines can attest!), they were surprisingly found in [4] to be theoretically uninteresting: channels become redundant, since read- and write-transitions are continuously enabled (the former because of potential insertion errors, the latter by assumption, as channels are unbounded). Consequently, most verification problems trivially reduce to questions on finite automata.

Recently, however, slightly more powerful models of channel machines with insertion errors have appeared as key tools in the study of Metric Temporal Logic (MTL). In [13, 14], the authors showed that MTL formulas can capture the computations of insertion channel machines *equipped with primitive operations for testing channel emptiness.* This new class of faulty channel machines was in turn shown to have a non-primitive recursive reachability problem and an undecidable recurrent control-state reachability problem. Consequently, MTL satisfiability and model checking were established to be non-primitive recursive over finite words [13], and undecidable over infinite words [14].

Independently of Metric Temporal Logic, the notion of emptiness testing, broadly construed, is a rather old and natural one. Counter machines, for instance, are usually assumed to incorporate primitive zero-testing operations on counters, and likewise pushdown automata are able to detect empty stacks. Variants of Petri nets have also explored emptiness testing for places, usually resulting in a great leap in computational power. In the context of channel machines, a slight refinement of emptiness testing is *occurrence testing*, checking

that a given channel contains no occurrence of a particular message, as defined and studied in [14]. Emptiness and occurrence testing provide some measure of control over insertion errors, since once a message has been inserted into a channel, it remains there until it is read off it.

Our main focus in this paper is the complexity of the *termination* problem for insertion channel machines: given such a machine, are all of its computations finite? We show that termination is non-elementary, yet primitive recursive. This result is quite surprising, as the closely related problems of reachability and recurrent reachability are respectively non-primitive recursive and undecidable. Moreover, the mere *decidability* of termination for insertion channel machines follows from the theory of well-structured transition systems, in a manner quite similar to that for lossy channel machines. In the latter case, however, termination is non-primitive recursive, as shown in [16]. Obtaining a primitive recursive upper bound for insertion channel machines has therefore required us to abandon the well-structure and pursue an entirely new approach.

On the practical side, one of the main motivations for studying termination of insertion channel machines arises from the safety fragment of Metric Temporal Logic. Safety MTL was shown to be decidable in [15], although no non-trivial bounds on the complexity could be established at the time. It is not difficult, however, to show that (non-)termination for insertion channel machines reduces (in polynomial time) to satisfiability for Safety MTL; the latter, therefore, is also non-elementary. We note that in a similar vein, a lower bound for the complexity of satisfiability of an extension of Linear Temporal Logic was given in [10], via a reduction from the termination problem for counter machines with incrementation errors.

2. Decision Problems for Faulty Channel Machines: A Brief Survey

In this section, we briefly review some key decision problems for lossy and insertion channel machines (the latter equipped with either emptiness or occurrence testing). Apart from the results on termination and structural termination for insertion channel machines, which are presented in the following sections, all results that appear here are either known or follow easily from known facts. Our presentation is therefore breezy and terse. Background material on well-structured transition systems can be found in [7].

The *reachability* problem asks whether a given distinguished control state of a channel machine is reachable. This problem was shown to be non-primitive recursive for lossy channel machines in [16]; it is likewise non-primitive recursive for insertion channel machines via a straightforward reduction from the latter [13].

The *termination* problem asks whether all computations of a channel machine are finite, starting from the initial control state and empty channel contents. This problem was shown to be non-primitive recursive for lossy channel machines in [16]. For insertion channel machines, we prove that termination is non-elementary in Section 4 and primitive recursive in Section 5.

The *structural termination* problem asks whether all computations of a channel machine are finite, starting from the initial control state but regardless of the initial channel contents. This problem was shown to be undecidable for lossy channel machines in [12]. For insertion channel machines, it is easy to see that termination and structural termination coincide, so that the latter is also non-elementary primitive-recursive decidable.

	Lossy Channel Machines	**Insertion Channel Machines**
Reachability	non-primitive recursive	non-primitive recursive
Termination	non-primitive recursive	non-elementary / primitive recursive
Struct. term.	undecidable	non-elementary / primitive recursive
Response	undecidable	non-primitive recursive
Recurrence	undecidable	undecidable
CTL / LTL	undecidable	undecidable

Figure 1: Complexity of decision problems for faulty channel machines.

Given a channel machine $\mathcal{S}$ and two distinguished control states p and q of $\mathcal{S}$, a *response* property is an assertion that every p state is always eventually followed by a q state in any infinite computation of $\mathcal{S}$. Note that a counterexample to a response property is a computation that eventually visits p and forever avoids q afterwards. The undecidability of response properties for lossy channel machines follows easily from that of structural termination, as the reader may wish to verify.

In the case of insertion channel machines, response properties are decidable, albeit at non-primitive recursive cost (by reduction from reachability). For decidability one first shows using the theory of well-structured transition systems that the set of all reachable configurations, the set of p-configurations, and the set of configurations that have infinite q-avoiding computations are all effectively computable. It then suffices to check whether their mutual intersection is empty.

The *recurrence* problem asks, given a channel machine and a distinguished control state, whether the machine has a computation that visits the distinguished state infinitely often. It is undecidable for lossy channel machines by reduction from response, and was shown to be undecidable for insertion channel machines in [14].

Finally, *CTL and LTL model checking* for both lossy and insertion channel machines are undecidable, which can be established along the same lines as the undecidability of recurrence.

These results are summarised in Figure 1.

3. Definitions

A *channel machine* is a tuple $\mathcal{S} = (Q, init, \Sigma, C, \Delta)$, where Q is a finite set of control states, $init \in Q$ is the initial control state, Σ is a finite channel alphabet, C is a finite set of channel names, and $\Delta \subseteq Q \times L \times Q$ is the transition relation, where $L = \{c!a, c?a, c=\emptyset, a\notin c : c \in C, a \in \Sigma\}$ is the set of transition labels. Intuitively, label $c!a$ denotes the writing of message a to tail of channel c, label $c?a$ denotes the reading of message a from the head of channel c, label $c=\emptyset$ tests channel c for emptiness, and label $a\notin c$ tests channel c for the absence (non-occurrence) of message a.

We first define an *error-free* operational semantics for channel machines. Given $\mathcal{S}$ as above, a *configuration* of $\mathcal{S}$ is a pair (q, U), where $q \in Q$ is the control state and $U \in (\Sigma^*)^C$ gives the contents of each channel. Let us write *Conf* for the set of possible configurations of $\mathcal{S}$. The rules in Δ induce an L-labelled transition relation on *Conf*, as follows:

(1) $(q, c!a, q') \in \Delta$ yields a transition $(q, U) \xrightarrow{c!a} (q', U')$, where $U'(c) = U(c){\cdot}a$ and $U'(d) = U(d)$ for $d \neq c$. *In other words, the channel machine moves from control*

  state q to control state q', writing message a to the tail of channel c and leaving all other channels unchanged.

(2) $(q, c?a, q') \in \Delta$ yields a transition $(q, U) \xrightarrow{c?a} (q', U')$, where $U(c) = a \cdot U'(c)$ and $U'(d) = U(d)$ for $d \neq c$. *In other words, the channel machine reads message a from the head of channel c while moving from control state q to control state q', leaving all other channels unchanged.*

(3) $(q, c{=}\emptyset, q') \in \Delta$ yields a transition $(q, U) \xrightarrow{c=\emptyset} (q', U)$, provided $U(c)$ is the empty word. *In other words, the transition is only enabled if channel c is empty; all channel contents remain the same.*

(4) $(q, a{\notin}c, q') \in \Delta$ yields a transition $(q, U) \xrightarrow{a\notin c} (q', U)$, provided a does not occur in $U(c)$. *In other words, the transition is only enabled if channel c contains no occurrence of message a; all channels remain unchanged.*

If the only transitions allowed are those listed above, then we call $\mathcal{S}$ an *error-free* channel machine. This machine model is easily seen to be Turing powerful [3]. As discussed earlier, however, we are interested in channel machines with (potential) *insertion errors*; intuitively, such errors are modelled by postulating that channels may at any time acquire additional messages interspersed throughout their current contents.

For our purposes, it is convenient to adopt the *lazy* model of insertion errors, given next. Slightly different models, such as those of [4, 14], have also appeared in the literature. As the reader may easily check, all these models are equivalent insofar as reachability and termination properties are concerned.

The lazy operational semantics for channel machines with insertion errors simply augments the transition relation on *Conf* with the following rule:

(5) $(q, c?a, q') \in \Delta$ yields a transition $(q, U) \xrightarrow{c?a} (q', U)$. *In other words, insertion errors occur 'just in time', immediately prior to a read operation; all channel contents remain unchanged.*

The channel machines defined above are called *insertion channel machines with occurrence testing*, or *ICMOT*s. We will also consider *insertion channel machines with emptiness testing*, or *ICMET*s. The latter are simply ICMOTs without any occurrence-testing transitions (i.e., transitions labelled with $a{\notin}c$).

A *run* of an insertion channel machine is a finite or infinite sequence of transitions of the form $\sigma_0 \xrightarrow{l_0} \sigma_1 \xrightarrow{l_1} \ldots$ that is consistent with the lazy operational semantics. The run is said to start from the initial configuration if the first control state is *init* and all channels are initially empty.

Our main focus in this paper is the study of the complexity of the *termination* problem: given an insertion channel machine $\mathcal{S}$, are all runs of $\mathcal{S}$ starting from the initial configuration finite?

4. Termination is Non-Elementary

In this section, we show that the termination problem for insertion channel machines—ICMETs and ICMOTs—is non-elementary. More precisely, we show that the termination problem for ICMETs of size n in the worst case requires time at least $2 {\Uparrow} \Omega(\log n)$.[1] Note that the same immediately follows for ICMOTs.

---

[1]The expression $2{\Uparrow}m$, known as tetration, denotes an exponential tower of 2s of height m.

Our proof proceeds by reduction from the termination problem for two-counter machines in which the counters are tetrationally bounded; the result then follows from standard facts in complexity theory (see, e.g., [9]).

Without insertion errors, it is clear that a channel machine can directly simulate a two-counter machine simply by storing the values of the counters on one of its channels. To simulate a counter machine in the presence of insertion errors, however, we require periodic integrity checks to ensure that the representation of the counter values has not been corrupted. Below we give a simulation that follows the 'yardstick' construction of Meyer and Stockmeyer [17, 11]: roughly speaking, we use an m-bounded counter to check the integrity of a 2^m-bounded counter.

Theorem 4.1. *The termination problem for ICMETs and ICMOTs is non-elementary.*

Proof. Let us say that a counter is m-bounded if it can take values in $\{0, 1, \ldots, m-1\}$. We assume that such a counter u comes equipped with procedures $\textsc{Inc}(u)$, $\textsc{Dec}(u)$, $\textsc{Reset}(u)$, and $\textsc{IsZero}(u)$, where $\textsc{Inc}$ and $\textsc{Dec}$ operate modulo m, and increment, resp. decrement, the counter. We show how to simulate a deterministic counter machine $\mathcal{M}$ of size n equipped with two $2{\Uparrow}n$-bounded counters by an ICMET $\mathcal{S}$ of size $2^{O(n)}$. We use this simulation to reduce the termination problem for $\mathcal{M}$ to the termination problem for $\mathcal{S}$.

By induction, assume that we have constructed an ICMET $\mathcal{S}_k$ that can simulate the operations of a $2{\Uparrow}k$-bounded counter u_k. We assume that $\mathcal{S}_k$ correctly implements the operations $\textsc{Inc}(u_k)$, $\textsc{Dec}(u_k)$, $\textsc{Reset}(u_k)$, and $\textsc{IsZero}(u_k)$ (in particular, we assume that the simulation of these operations by $\mathcal{S}_k$ is guaranteed to terminate). We describe an ICMET $\mathcal{S}_{k+1}$ that implements a $2{\Uparrow}(k+1)$-bounded counter u_{k+1}. $\mathcal{S}_{k+1}$ incorporates $\mathcal{S}_k$, and thus can use the above-mentioned operations on the counter u_k as subroutines. In addition, $\mathcal{S}_{k+1}$ has two extra channels c and d on which the value of counter u_{k+1} is stored in binary. We give a high-level description.

We say that a configuration of $\mathcal{S}_{k+1}$ is *clean* if channel c has size $2{\Uparrow}k$ and channel d is empty. We ensure that all procedures on counter u_{k+1} operate correctly when they are invoked in clean configurations of $\mathcal{S}_{k+1}$, and that they also yield clean configurations upon completion. In fact, we only give details for the procedure $\textsc{Inc}(u_{k+1})$—see Figure 2; the others should be clear from this example.

Since the counter u_k is assumed to work correctly, the above procedure is guaranteed to terminate, having produced the correct result, in the absence of any insertion errors on channels c or d. On the other hand, insertion errors on either of these channels will be detected by one of the two emptiness tests, either immediately or in the next procedure to act on them.

The initialisation of the induction is handled using an ICMET $\mathcal{S}_1$ with no channel (in other words, a finite automaton) of size 2, which can simulate a 2-bounded counter (i.e., a single bit). The finite control of the counter machine, likewise, is duplicated using a further channel-less ICMET.

Using a product construction, it is straightforward to conflate these various ICMETs into a single one, $\mathcal{S}$, of size exponential in n (more precisely: of size $2^{O(n)}$). As the reader can easily check, $\mathcal{M}$ has an infinite computation iff $\mathcal{S}$ has an infinite run. The result follows immediately. ∎

Procedure $\text{INC}(u_{k+1})$
  $\text{RESET}(u_k)$
  repeat
    $c?x$; $d!(1-x)$ /* Increment counter u_{k+1} while transferring c to d */
    $\text{INC}(u_k)$
  until $\text{ISZERO}(u_k)$ or $x = 0$
  while not $\text{ISZERO}(u_k)$ **do**
    $c?x$; $d!x$ /* Transfer remainder of c to d */
    $\text{INC}(u_k)$
  endwhile
  test$(c{=}\emptyset)$ /* Check that there were no insertion errors on c, otherwise halt */
  repeat
    $d?x$; $c!x$ /* Transfer d back to c */
    $\text{INC}(u_k)$
  until $\text{ISZERO}(u_k)$
  test$(d{=}\emptyset)$ /* Check that there were no insertion errors on d, otherwise halt */
  return

Figure 2: Procedure to increment counter u_{k+1}. Initially, this procedure assumes that counter u_{k+1} is encoded in binary on channel c, with least significant bit at the head of the channel; moreover, c is assumed to comprise exactly $2{\Uparrow}k$ bits (using padding 0s if need be). In addition, channel d is assumed to be initially empty. Upon exiting, channel c will contain the incremented value of counter u_{k+1} (modulo $2{\Uparrow}(k+1)$) in binary, again using $2{\Uparrow}k$ bits, and channel d will be empty. We regularly check that no insertion errors have occurred on channels c or d by making sure that they contain precisely the right number of bits. This is achieved using counter u_k (which can count up to $2{\Uparrow}k$ and is assumed to work correctly) together with emptiness tests on c and d. If an insertion error does occur during execution, the procedure will either halt, or the next procedure to handle channels c and d (i.e., any command related to counter u_{k+1}) will halt.

5. Termination is Primitive Recursive

The central result of our paper is the following:

Theorem 5.1. *The termination problem for ICMOTs and ICMETs is primitive recursive. More precisely, when restricting to the class of ICMOTs or ICMETs that have at most k channels, the termination problem is in $(k+1)$-EXPSPACE.*

Proof. In what follows, we sketch the proof for ICMOTs, ICMETs being a special case of ICMOTs. Let us also assume that our ICMOTs do not make use of any emptiness tests; this restriction is harmless since any emptiness test can always be replaced by a sequence of occurrence tests, one for each letter of the alphabet, while preserving termination.

Let $\mathcal{S} = (Q, init, \Sigma, C, \Delta)$ be a fixed ICMOT without emptiness tests; in other words, $\mathcal{S}$'s set of transition labels is $L = \{c!a, c?a, a \notin c : c \in C, a \in \Sigma\}$. Our strategy is as follows: we suppose that $\mathcal{S}$ has no infinite runs, and then derive an upper bound on the length of the longest possible finite run. The result follows by noting that the total number of possible runs is exponentially bounded by this maximal length.

For a subset $D \subseteq C$ of channels, we define an equivalence $\equiv_D$ over the set $Conf$ of configurations of $\mathcal{S}$ as follows:

$$(q, U) \equiv_D (q', U') \text{ iff } q = q' \text{ and } U(d) = U'(d) \text{ for every } d \in D.$$

Let us write $Conf_D$ to denote the set $Conf/\equiv_D$ of equivalence classes of $Conf$ with respect to $\equiv_D$. Furthermore, given $f : D \to \mathbb{N}$ a 'bounding function' for the channels in D, let

$$Conf_D^f = \{[(q, U)]_D \in Conf_D : |U(d)| \le f(d) \text{ for every } d \in D\}$$

be the subset of $Conf_D$ consisting of those equivalence classes of configurations whose D-channels are bounded by f. As the reader can easily verify, we have the following bound on the cardinality γ_D^f of $Conf_D^f$:

$$\gamma_D^f \le |Q| \prod_{d \in D} (|\Sigma| + 1)^{f(d)}. \tag{5.1}$$

Consider a finite run $\sigma_0 \xrightarrow{l_0} \sigma_1 \xrightarrow{l_1} \ldots \xrightarrow{l_{n-1}} \sigma_n$ of $\mathcal{S}$ (with $n \ge 1$), where each $\sigma_i \in Conf$ is a configuration and each $l_i \in L$ is a transition label. We will occasionally write $\sigma_0 \xRightarrow{\lambda} \sigma_n$ to denote such a run, where $\lambda = l_0 l_1 \ldots l_{n-1} \in L^+$.

We first state a pumping lemma of sorts, whose straightforward proof is left to the reader:

Lemma 5.2. *Let $D \subseteq C$ be given, and assume that $\sigma \xRightarrow{\lambda} \sigma'$ (with $\lambda \in L^+$) is a run of $\mathcal{S}$ such that $\sigma \equiv_D \sigma'$. Suppose further that, for every label $a\not\in c$ occurring in λ, either $c \in D$, or the label $c!a$ does not occur in λ. Then λ is repeatedly firable from σ, i.e., there exists an infinite run $\sigma \xRightarrow{\lambda} \sigma' \xRightarrow{\lambda} \sigma'' \xRightarrow{\lambda} \ldots$.*

Note that the validity of Lemma 5.2 rests crucially on (the potential for) insertion errors.

Let $\langle w_i \rangle_{1 \le i \le n}$ be a finite sequence, and let $0 < \alpha \le 1$ be a real number. A set S is said to be α-frequent in the sequence $\langle w_i \rangle$ if the set $\{i : w_i \in S\}$ has cardinality at least αn.

The next result we need is a technical lemma guaranteeing a certain density of repeated elements in an α-frequent sequence:

Lemma 5.3. *Let $\langle w_i \rangle_{1 \le i \le n}$ be a finite sequence, and assume that S is a finite α-frequent set in $\langle w_i \rangle$. Then there exists a sequence of pairs of indices $\langle (i_j, i_j') \rangle_{1 \le j \le \frac{\alpha n}{2(|S|+1)}}$ such that, for all $j < \frac{\alpha n}{2(|S|+1)}$, we have $i_j < i_j' < i_{j+1}$, $i_j' - i_j \le \frac{2(|S|+1)}{\alpha}$, and $w_{i_j} = w_{i_j'} \in S$.*

Proof. By assumption, $\langle w_i \rangle$ has a subsequence of length at least αn consisting exclusively of elements of S. This subsequence, in turn, contains at least $\frac{\alpha n}{|S|+1}$ disjoint 'blocks' of length $|S| + 1$. By the pigeonhole principle, each of these blocks contains at least two identical elements from S, yielding a sequence of pairs of indices $\langle (i_j, i_j') \rangle_{1 \le j \le \frac{\alpha n}{|S|+1}}$ having all the required properties apart, possibly, from the requirement that $i_j' - i_j \le \frac{2(|S|+1)}{\alpha}$. Note also that there are, for now, twice as many pairs as required.

Consider therefore the half of those pairs whose difference is smallest, and let p be the largest such difference. Since the other half of pairs in the sequence $\langle (i_j, i_j') \rangle$ have difference at least p, and since there is no overlap between indices, we have $\frac{1}{2} \cdot \frac{\alpha n}{|S|+1} \cdot p < n$, from which we immediately derive that p is bounded by $\frac{2(|S|+1)}{\alpha}$, as required. This concludes the proof of Lemma 5.3. $\blacksquare$

Recall our assumption that $\mathcal{S}$ has no infinite run, and let $\pi = \sigma_0 \xrightarrow{l_0} \sigma_1 \xrightarrow{l_1} \ldots \xrightarrow{l_{n-1}} \sigma_n$ be any finite run of $\mathcal{S}$, starting from the initial configuration; we seek to obtain an upper bound on n.

Given a set $D \subseteq C$ of channels, it will be convenient to consider the sequence $[\pi]_D = \langle [\sigma_i]_D \rangle_{0 \leq i \leq n}$ of equivalence classes of configurations in π modulo $\equiv_D$ (ignoring the interspersed labelled transitions for now).

Let $f : C \to \mathbb{N}$ and $0 < \alpha \leq 1$ be given, and suppose that $Conf_C^f$ is α-frequent in $[\pi]_C$, so that there are at least αn occurrences of configuration equivalence classes in $Conf_C^f$ along $[\pi]_C$. Recall that $Conf_C^f$ contains γ_C^f elements. Observe, by Lemma 5.2, that no member of $Conf_C^f$ can occur twice along $[\pi]_D$, otherwise $\mathcal{S}$ would have an infinite run. Consequently,

$$n \leq \frac{\gamma_C^f}{\alpha}. \tag{5.2}$$

We will now inductively build an increasing sequence $\emptyset = D_0 \subset D_1 \subset \ldots \subset D_{|C|} = C$, as well as functions $f_i : D_i \to \mathbb{N}$ and real numbers $0 < \alpha_i \leq 1$, for $0 \leq i \leq |C|$, such that $Conf_{D_i}^{f_i}$ is α_i-frequent in $[\pi]_{D_i}$ for every $i \leq |C|$.

The base case is straightforward: the set $Conf_\emptyset^{f_0} = Conf_\emptyset$ is clearly 1-frequent in $[\pi]_\emptyset$.

Let us therefore assume that $Conf_D^f$ is α-frequent in $[\pi]_D$ for some strict subset D of C and some $f : D \to \mathbb{N}$ and $\alpha > 0$. We now compute $D' \subseteq C$ strictly containing D, $f' : D' \to \mathbb{N}$, and $\alpha' > 0$ such that $Conf_{D'}^{f'}$ is α'-frequent in $[\pi]_{D'}$.

Thanks to our induction hypothesis and Lemma 5.3, we obtain a sequence of pairs of configurations $\langle (\theta_j, \theta_j') \rangle_{1 \leq j \leq h}$, where $h = \frac{\alpha n}{2(\gamma_D^f + 1)}$, $[\theta_j]_D = [\theta_j']_D \in Conf_D^f$, and such that

$$\pi = \sigma_0 \Longrightarrow \theta_1 \xrightarrow{\lambda_1} \theta_1' \Longrightarrow \theta_2 \xrightarrow{\lambda_2} \theta_2' \Longrightarrow \ldots \Longrightarrow \theta_h \xrightarrow{\lambda_h} \theta_h' \Longrightarrow \sigma_n$$

with each $\lambda_j \in L^+$ having length no greater than $\frac{2(\gamma_D^f + 1)}{\alpha}$, for $1 \leq j \leq h$.

For each λ_j, let OT_j be the set of occurrence-test labels that occur at least once in λ_j. Among these sets, let OT denote the one that appears most often. Note that there are $2^{|\Sigma| \cdot |C|}$ different possible sets of occurrence-test labels, and therefore at least $\frac{h}{2^{|\Sigma| \cdot |C|}}$ of the OT_j are equal to OT.

Following a line of reasoning entirely similar to that used in Lemma 5.3[2], we can deduce that π contains at least $\frac{h}{4 \cdot 2^{|\Sigma| \cdot |C|}} = \frac{\alpha n}{8(\gamma_D^f + 1) 2^{|\Sigma| \cdot |C|}}$ non-overlapping patterns of the form

$$\theta \Longrightarrow \theta' \xrightarrow{\delta} \bar{\theta} \xrightarrow{\bar{\lambda}} \bar{\theta}',$$

where:

- $[\theta]_D = [\theta']_D \in Conf_D^f$ and $[\bar{\theta}]_D = [\bar{\theta}']_D \in Conf_D^f$,
- $\lambda, \bar{\lambda} \in L^+$ each have length no greater than $\frac{2(\gamma_D^f + 1)}{\alpha}$,
- $\delta \in L^+$ has length no greater than $\frac{8(\gamma_D^f + 1) 2^{|\Sigma| \cdot |C|}}{\alpha}$, and
- the set of occurrence-test labels occurring in λ and $\bar{\lambda}$ in both cases is OT.

---

[2]Formally, we could directly invoke Lemma 5.3, as follows. Write the sequence of transition labels of π as $\delta_0 \lambda_1 \delta_1 \lambda_2 \cdots \lambda_h \delta_h$, with the λ_i as above. Next, formally replace each instance of λ_i whose set of occurrence-test labels is OT by a new symbol $\mathbf{O}$; if needed, add dummy non-$\mathbf{O}$ symbols to the end of the sequence to bring its length up to n, and call the resulting sequence $\langle w_i \rangle$. Finally, note that the singleton set $\{\mathbf{O}\}$ is $\frac{h}{2^{|\Sigma| \cdot |C|} \cdot n}$-frequent in $\langle w_i \rangle$.

Consider such a pattern. Observe that λ must contain at least one occurrence-test label $a \notin c$ with $c \notin D$ and such that the label $c!a$ occurs in λ, otherwise $\mathcal{S}$ would have an infinite run according to Lemma 5.2. Pick any such occurrence-test label and let us denote it $a \notin c$.

We now aim to bound the size of channel c in the $\bar{\theta}$ configuration of our patterns. Note that since λ and $\bar{\lambda}$ contain the same set of occurrence-test labels, the label $a \notin c$ occurs in $\bar{\lambda}$. That is to say, somewhere between configurations $\bar{\theta}$ and $\bar{\theta}'$, we know that channel c did not contain any occurrence of a. On the other hand, an a was written to the tail of channel c at some point between configurations θ and θ', since λ contains the label $c!a$. For that a to be subsequently read off the channel, the whole contents of channel c must have been read from the time of the $c!a$ transition in λ to the time of the $a \notin c$ transition in $\bar{\lambda}$. Finally, note that, according to our lazy operational semantics, the size of a channel changes by at most 1 with each transition. It follows that the size of channel c in configuration $\bar{\theta}$ is at most $|\lambda| + |\delta| + |\lambda'| \leq \frac{(\gamma_D^f + 1)(4 + 8 \cdot 2^{|\Sigma| \cdot |C|})}{\alpha}$.

Let $D' = D \cup \{c\}$, and define the bounding function $f' : D' \to \mathbb{N}$ such that $f'(d) = f(d)$ for all $d \in D$, and $f'(c) = \frac{(\gamma_D^f + 1)(4 + 8 \cdot 2^{|\Sigma| \cdot |C|})}{\alpha}$. From our lower bound on the number of special patterns, we conclude that the set $Conf_{D'}^{f'}$ is α'-frequent in $[\pi]_{D'}$, where $\alpha' = \frac{\alpha}{8(\gamma_D^f + 1)2^{|\Sigma| \cdot |C|}}$.

We now string everything together to obtain a bound on n, the length of our original arbitrary run π. For convenience, let $c_1, c_2, \ldots, c_{|C|}$ be an enumeration of the channel names in C in the order in which they are picked in the course of our proof; thus $D_i = D_{i-1} \cup \{c_i\}$ for $1 \leq i \leq |C|$. Correspondingly, let $M_i = f_i(c_i)$, for $0 \leq i \leq |C|$, with the convention that $M_0 = 1$; it is easy to see that M_i is the maximum value of f_i over D_i, since the sequences $\langle \gamma_{D_i}^{f_i} \rangle$ and $\langle \alpha_i \rangle$ are monotonically increasing and decreasing respectively.

From Equation 5.1, we easily get that $\gamma_{D_i}^{f_i} \in O(|\mathcal{S}|^{|\mathcal{S}|M_i})$, where $|\mathcal{S}|$ is any reasonable measure of the size of our ICMOT $\mathcal{S}$. Combining this with our expressions for f' and α' above, we obtain that $M_{i+1}, \frac{1}{\alpha_{i+1}} \in O\left(\frac{|\mathcal{S}|^{|\mathcal{S}|^2 M_i}}{\alpha_i}\right)$ for $0 \leq i \leq |C| - 1$. This, in turns, lets us derive bounds for $\gamma_C^{f_{|C|}}$ and $\alpha_{|C|}$, which imply, together with Equation 5.2, that

$$n \leq 2^{2^{\cdot^{\cdot^{2^{P(|\mathcal{S}|)}}}}},$$

where P is some polynomial (independent of $\mathcal{S}$), and the total height of the tower of exponentials is $|C| + 2$.

The ICMOT $\mathcal{S}$ therefore has an infinite run iff it has a run whose length exceeds the above bound. Since the lazy operational semantics is finitely branching (bounded, in fact, by the size of the transition relation), this can clearly be determined in $(|C|+1)$-EXPSPACE, which concludes the proof of Theorem 5.1.   ∎

Theorems 4.1 and 5.1 immediately entail the following:

Corollary 5.4. *The structural termination problem—are all computations of the machine finite, starting from the initial control state but regardless of the initial channel contents?—is decidable for ICMETs and ICMOTs, with non-elementary but primitive-recursive complexity.*

6. Conclusion

The main result of this paper is that termination for insertion channel machines with emptiness or occurrence testing has non-elementary, yet primitive recursive complexity. This result is in sharp contrast with the equivalent problem for lossy channel machines, which has non-primitive recursive complexity.

We remark that the set of configurations from which a given insertion channel machine has at least one infinite computation is finitely representable (thanks to the theory of well-structured transition systems), and is in fact computable as the greatest fixed point of the pre-image operator. The proof of Theorem 5.1, moreover, shows that this fixed point will be reached in primitive-recursively many steps. The set of configurations from which there is an infinite computation is therefore primitive-recursively computable, in contrast with lossy channel machines for which it is not even recursive (as can be seen from the undecidability of structural termination).

Finally, another interesting difference with lossy channel machines can be highlighted by quoting a slogan from [16]: *"Lossy systems with k channels can be [polynomially] encoded into lossy systems with one channel."* We can deduce from Theorems 4.1 and 5.1 that any such encoding, in the case of insertion channels machines, would require non-elementary resources to compute, if it were to preserve termination properties.

References

[1] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. In *Proc. 8th Annual Symposium on Logic in Computer Science (LICS'93)*, pages 160–170. IEEE Computer Society Press, 1993.

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

[3] Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, 1983.

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

[5] Pierre Chambart and Philippe Schnoebelen. Post embedding problem is not primitive recursive, with applications to channel systems. In *Proc. 27th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'07)*, volume 4855 of *Lecture Notes in Computer Science*, pages 265–276. Springer, 2007.

[6] Alain Finkel. Decidability of the termination problem for completely specificied protocols. *Distributed Computing*, 7(3):129–135, 1994.

[7] Alain Finkel and Philippe Schnoebelen. Well structured transition systems everywhere! *Theoretical Computer Science*, 256(1–2):63–92, 2001.

[8] Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 2:326–336, 1952.

[9] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[10] Ranko Lazić. Safely freezing LTL. In *Proc. 26th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'06)*, volume 4337 of *Lecture Notes in Computer Science*, pages 381–392. Springer, 2006.

[11] Ranko Lazić, Thomas C. Newcomb, Joël Ouaknine, A. W. Roscoe, and James Worrell. Nets with tokens which carry data. In *Proc. 28th International Conference on Application and Theory of Petri Nets (ICATPN'07)*, volume 4546 of *Lecture Notes in Computer Science*, pages 301–320. Springer, 2007.

[12] Richard Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 297(1):35–65, 2003.

[13] Joël Ouaknine and James Worrell. On the decidability of Metric Temporal Logic. In *Proc. 19th Annual Symposium on Logic in Computer Science (LICS'05)*, pages 188–197. IEEE Computer Society Press, 2005.

[14] Joël Ouaknine and James Worrell. On metric temporal logic and faulty Turing machines. In *Proc. 9th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'06)*, volume 3921 of *Lecture Notes in Computer Science*, pages 217–230. Springer, 2006.

[15] Joël Ouaknine and James Worrell. Safety metric temporal logic is fully decidable. In *Proc. 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, volume 3920 of *Lecture Notes in Computer Science*, pages 411–425. Springer, 2006.

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

[17] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In *Proc. 5th AMS Symposium on Theory of Computing*, 1973.