4th Asian Computing Science Conference, ASIAN '98
Proceedings: Jieh Hsiang and Atsushi Ohori (eds.)
Springer Lecture Notes in Computer Science 1538 (1998), 282–299.

Towards a characterisation of finite-state message-passing systems

Madhavan Mukund[1⋆], K Narayan Kumar[1⋆⋆],
Jaikumar Radhakrishnan[2], and Milind Sohoni[3]

[1] SPIC Mathematical Institute, 92 G.N. Chetty Road, Madras 600 017, India.
E-mail: {madhavan,kumar}@smi.ernet.in
[2] Computer Science Group, Tata Institute of Fundamental Research, Homi Bhabha Road, Bombay 400 005, India. E-mail: jaikumar@tcs.tifr.res.in
[3] Department of Computer Science and Engineering, Indian Institute of Technology, Bombay 400 076, India. E-mail: sohoni@cse.iitb.ernet.in

Abstract. We investigate an automata-theoretic model of distributed systems which communicate via message-passing. Each node in the system is a finite-state device. Channels are assumed to be reliable but may deliver messages out of order. Hence, each channel is modelled as a set of counters, one for each type of message. These counters may *not* be tested for zero.

Though each node in the network is finite-state, the overall system is potentially infinite-state because the counters are unbounded. We work in an interleaved setting where the interactions of the system with the environment are described as sequences. The behaviour of a system is described in terms of the language which it accepts—that is, the set of valid interactions with the environment that are permitted by the system.

Our aim is to characterise the class of message-passing systems whose behaviour is finite-state. Our main result is that the language accepted by a message-passing system is regular if and only if both the language and its complement are accepted by message-passing systems. We also exhibit an alternative characterisation of regular message-passing languages in terms of deterministic automata.

1 Introduction

Today, distributed systems which use asynchronous communication are ubiquitous—the Internet is a prime example. However, there has been very little work on studying the finite-state behaviour of such systems. In particular, this area lacks a satisfactory automata-theoretic framework. In contrast, automata theory for systems with synchronous communication is well developed via Zielonka's

⋆ Partly supported by IFCPAR Project 1502-1.

⋆⋆ Currently on leave at Department of Computer Science, State University of New York at Stony Brook, NY 11794-4400, USA. E-mail: kumar@cs.sunysb.edu.

asynchronous automata [Z87] and the connections to Mazurkiewicz trace theory [M78].

In [MNRS98], we introduce *networks of message-passing automata* as a model for distributed systems which communicate via message-passing. Each node in the network is a finite-state process. The number of different types of messages used by the system is assumed to be finite. This is not unreasonable if we distinguish "control" messages from "data" messages.

In our model, channels may reorder or delay messages, though messages are never lost. Since messages may be reordered, the state of each channel can be represented by a finite set of counters which record the number of messages of each type that have been sent along the channel but are as yet undelivered. The nodes cannot test if a counter's value is zero—this restriction captures the intuition that it is not practical for a node to decide that another process has *not* sent a message, since messages may be delayed arbitrarily.

Though each node in the network is finite-state, the overall system is potentially infinite-state since counter values are unbounded. Our goal is to characterise when such a network is "effectively finite-state". This is important because finite-state networks are amenable to verification using automated tools [H91].

To make precise the notion of a network being "effectively finite-state", we use formal language theory. The behaviour of the network is described in terms of its interaction with the environment. This can be represented as a formal language over a finite alphabet of possible interactions. Our goal then is to characterise when the language accepted by a network is regular.

In [MNRS98], we assume that each node interacts independently with its environment. Thus, the behaviour of the overall network is described as a language consisting of tuples of strings. The main result is that the language accepted by a *robust* message-passing network, whose behaviour is insensitive to message delays and differences in speed between nodes, can be "represented" by a sequential regular language.

Here, we adopt an interleaved approach and record the interactions of a network with its environment from the point of view of a sequential observer. In this framework, it is sufficient to concentrate on the global states of the system and regard the entire network as a single automaton equipped with a set of counters. Our main result is that a language L accepted by a message-passing automaton is regular if and only if the complement of L is also accepted by a message-passing automaton. *This is more general than requiring that L be robust in the sense of [MNRS98]—see Section 5.1.* We also demonstrate an alternative characterisation in terms of deterministic message-passing automata. Along the way, we establish a variety of results about message-passing automata, including pumping lemmas which are useful for showing when languages are *not* recognisable by these automata.

The paper is organised as follows. In the next section we define message-passing automata and establish some basic results about them. In Section 3 we prove a Contraction Lemma which leads to the decidability of the emptiness problem and the fact that the languages accepted by message-passing automata are not closed under complementation. Section 4 describes a family of pumping lemmas which are exploited in Section 5 to prove our main results concerning the regularity of languages accepted by message-passing automata. In the final section, we discuss in detail the connection between our results and those in Petri net theory and point out directions for future work. We have had to omit many proofs in this extended abstract. Full proofs and related results can be found in [MNRS97,MNRS98].

2 Message-Passing Automata

Natural numbers and tuples As usual, $\mathbb{N}$ denotes the set $\{0, 1, 2, \ldots\}$ of natural numbers. For $i, j \in \mathbb{N}$, $[i..j]$ denotes the set $\{i, i+1, \ldots, j\}$, where $[i..j] = \emptyset$ if $i > j$. We compare k-tuples of natural numbers component-wise. Let $\overline{m} = \langle m_1, m_2, \ldots, m_k \rangle$ and $\overline{n} = \langle n_1, n_2, \ldots, n_k \rangle$. Then $\overline{m} \leq \overline{n}$ iff $m_i \leq n_i$ for each $i \in [1..k]$.

Message-passing automata A *message-passing automaton* $\mathcal{A}$ is a tuple $(Q, \Sigma, \Gamma, T, q_{\text{in}}, F)$, where:

- Q is a finite set of *states*, with *initial state* q_{in} and *accepting states* $F \subseteq Q$.
- Σ is a finite *input alphabet*.
- Γ is a finite set of *counters*. We use $C, C', \ldots$ to denote counters. With each counter C, we associate two symbols, C^+ and C^-. We write $\Gamma^{\pm}$ to denote the set $\{C^+ | C \in \Gamma\} \cup \{C^- | C \in \Gamma\}$.
- $T \subseteq Q \times (\Sigma \cup \Gamma^{\pm}) \times Q$ is the *transition relation*.

Configurations A *configuration* of $\mathcal{A}$ is a pair (q, f) where $q \in Q$ and $f : \Gamma \to \mathbb{N}$ is a function which records the values stored in the counters. If the counters are $C_1, C_2, \ldots, C_k$ then we represent f by an element $\langle f(C_1), f(C_2), \ldots, f(C_k) \rangle$ of $\mathbb{N}^k$. By abuse of notation, the k-tuple $\langle 0, 0, \ldots, 0 \rangle$ is uniformly denoted $\overline{0}$, for all values of k.

We use χ to denote configurations. If $\chi = (q, f)$, $Q(\chi)$ denotes q and $F(\chi)$ denotes f. Further, for each counter C, $C(\chi)$ denotes the value $f(C)$.

Moves Each move of a message-passing automaton consists of either reading a letter from its input or manipulating a counter. Reading from the input represents interaction with the environment. Incrementing and decrementing counters correspond to sending and reading messages, respectively.

Formally, a message-passing automaton *moves* from configuration χ to configuration χ' on $d \in \Sigma \cup \Gamma^{\pm}$ if $(Q(\chi), d, Q(\chi')) \in T$ and one of the following holds:

- $d \in \Sigma$ and $F(\chi) = F(\chi')$.
- $d = C^+$, $C(\chi') = C(\chi) + 1$ and $C'(\chi) = C'(\chi')$ for every $C' \neq C$.
- $d = C^-$, $C(\chi') = C(\chi) - 1 \geq 0$ and $C'(\chi) = C'(\chi')$ for every $C' \neq C$.

Such a move is denoted $\chi \xrightarrow{(q,d,q')} \chi'$—that is, transitions are labelled by elements of T. Given a sequence of transitions $t_1 t_2 \ldots t_n = (q_1, d_1, q_2)(q_2, d_2, q_3)\ldots(q_n, d_n, q_{n+1})$, the corresponding sequence $d_1 d_2 \ldots d_n$ over $\Sigma \cup \Gamma^{\pm}$ is denoted $\alpha(t_1 t_2 \ldots t_n)$.

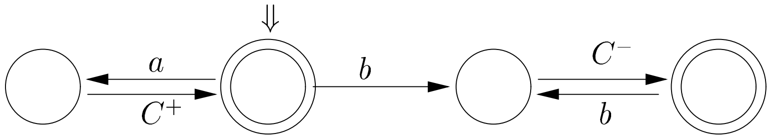
Computations, runs and languages A *computation* of $\mathcal{A}$ is a sequence $\chi_0 \xrightarrow{t_1} \chi_1 \xrightarrow{t_2} \ldots \xrightarrow{t_n} \chi_n$. We also write $\chi_0 \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi_n$ to indicate that there is a computation labelled $t_1 t_2 \ldots t_n$ from χ_0 to χ_n. Notice that χ_0 and $t_1 t_2 \ldots t_n$ uniquely determine all the intermediate configurations $\chi_1, \chi_2, \ldots, \chi_n$. If the transition sequence is not relevant, we just write $\chi_0 \Longrightarrow \chi_n$. As usual, $\chi \overset{t_1 t_2 \ldots t_n}{\Longrightarrow}$ denotes that there exists χ' such that $\chi \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi'$ and $\chi \Longrightarrow$ denotes that there exists χ' such that $\chi \Longrightarrow \chi'$.

For $K \in \mathbb{N}$, a *K-run* of $\mathcal{A}$ is a computation $\chi_0 \Longrightarrow \chi_n$ where $C(\chi_0) \leq K$ for each $C \in \Gamma$.

If δ is a string over $\Sigma \cup \Gamma^{\pm}$, $\delta\restriction_{\Sigma}$ denotes the subsequence of letters from Σ in δ. Let $w = a_1 a_2 \ldots a_k$ be a string over Σ. A *run of $\mathcal{A}$ over w* is a 0-run $\chi_0 \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi_n$ where $Q(\chi_0) = q_{\text{in}}$ and $\alpha(t_1 t_2 \ldots t_n)\restriction_{\Sigma} = w$. The run is said to be *accepting* if $Q(\chi_n) \in F$. The string w is *accepted* by $\mathcal{A}$ if $\mathcal{A}$ has an accepting run over w. The *language accepted* by $\mathcal{A}$, denoted $L(\mathcal{A})$, is the set of all strings over Σ accepted by $\mathcal{A}$.

A language over Σ is said to be *message-passing recognisable* if there is a message-passing automaton with input alphabet Σ that accepts this language.

Example 2.1. Let $L_{\text{ge}} \subseteq \{a, b\}^*$ be given by $\{a^m b^n \mid m \geq n\}$. This language is message-passing recognisable. Here is an automaton for L_{ge}. The initial state is indicated by $\Downarrow$ and the final states have an extra circle around them.



The following result is basic to analysing the behaviour of message-passing automata. It follows from the fact that any infinite sequence of N-tuples of natural numbers contains an infinite increasing subsequence. We omit the proof.

Lemma 2.2. *Let X be a set with M elements and $\langle x_1, f_1\rangle, \langle x_2, f_2\rangle, \ldots, \langle x_m, f_m\rangle$ be a sequence over $X \times \mathbb{N}^N$ such that each coordinate of f_1 is bounded by K and for $i \in [1..m-1]$, f_i and f_{i+1} differ on at most one coordinate and this difference is at most 1. There is a constant ℓ which depends only on M, N and K such that if $m \geq \ell$, then there exist $i, j \in [1..m]$ with $i < j$, $x_i = x_j$ and $f_i \leq f_j$.*

Weak pumping constant We call the bound ℓ for M, N and K from the preceding lemma the *weak pumping constant* for (M, N, K), denoted $\pi_{M,N,K}$.

It is easy to see that if $\langle M', N', K\rangle \leq \langle M, N, K\rangle$, then $\pi_{M',N',K'} \leq \pi_{M,N,K}$.

3 A Contraction Lemma

Lemma 3.1 (Contraction). *For every message-passing automaton $\mathcal{A}$, there is a constant k such that if $\chi_0 \stackrel{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ is a computation of $\mathcal{A}$, with $m > k$, then there exist i and j, $m-k \le i < j \le m$, such that $\chi'_0 \stackrel{t_1 \ldots t_i t_{j+1} \ldots t_m}{\Longrightarrow} \chi'_{m-(j-i)}$ is also a computation of $\mathcal{A}$, with with $\chi'_\ell = \chi_\ell$ for $\ell \in [0..i]$ and $Q(\chi_\ell) = Q(\chi'_{\ell-(j-i)})$ for all $\ell \in [j..m]$.*

Proof Sketch. Let $\mathcal{A}$ have M states and N counters. We show that k can be chosen to be $\pi_{M,N,0}$. Let $\chi_0 \stackrel{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ be a computation of $\mathcal{A}$, with $m > \pi_{M,N,0}$. We define a sequence $f_m, f_{m-1}, \ldots, f_0$ of functions from Γ to $\mathbb{N}$ as follows:

$$f_m(C) = 0, \text{for all } C \in \Gamma$$

$$\text{For } i \in [0..m-1],\ f_i(C) = \begin{cases} f_{i+1}(C) & \text{if } \alpha(t_{i+1}) \notin \{C^+, C^-\} \\ f_{i+1}(C)+1 & \text{if } \alpha(t_{i+1}) = C^- \\ \max(0, f_{i+1}(C)-1) & \text{if } \alpha(t_{i+1}) = C^+ \end{cases}$$

We claim, without proof, that for each i, the function f_i represents the minimum counter values required to execute the transition sequence $t_{i+1}t_{i+2}\ldots t_m$.

Claim: $\forall i \in [1..m]$, $(Q(\chi_i), f) \stackrel{t_{i+1} t_{i+2} \ldots t_m}{\Longrightarrow}$ iff $f \ge f_i$.

Corollary to Claim: For each counter C and for each position $i \in [1..m]$, $C(\chi_i) \ge f_i(C)$.

Consider the sequence of N-tuples $f_m, f_{m-1}, \ldots f_0$. Since its length exceeds $\pi_{M,N,0}$, by Lemma 2.2 there exist positions i and j, $m \ge j > i \ge m-\pi_{M,N,0}$ such that $f_j \le f_i$ and $Q(\chi_j) = Q(\chi_i)$. By the Corollary to Claim, for each counter C, $C(\chi_i) \ge f_i(C) \ge f_j(C)$. Thus, $\chi_i \stackrel{t_{j+1} t_{j+2} \ldots t_m}{\Longrightarrow}$ whereby $\chi_0 \stackrel{t_1 t_2 \ldots t_i}{\Longrightarrow} \chi_i \stackrel{t_{j+1} t_{j+2} \ldots t_m}{\Longrightarrow} \chi'_{m-(j-i)}$ is a valid computation of $\mathcal{A}$ for some configuration $\chi'_{m-(j-i)}$. Since $Q(\chi_j) = Q(\chi_i)$ and the computations $\chi_j \stackrel{t_{j+1} t_{j+2} \ldots t_m}{\Longrightarrow} \chi_m$ and $\chi_i \stackrel{t_{j+1} t_{j+2} \ldots t_m}{\Longrightarrow} \chi'_{m-(j-i)}$ are labelled by the same sequence of transitions, it follows that $Q(\chi_\ell) = Q(\chi'_{\ell-(j-i)})$ for each $\ell \in [j..m]$, as required. □

Corollary 3.2. *A message-passing automaton $\mathcal{A}$ with M states and N counters has an accepting computation iff it has an accepting computation whose length is bounded by $\pi_{M,N,0}$.*

It is possible to provide an explicit upper bound for $\pi_{M,N,K}$ for all values of M, N, and K. This fact, coupled with the preceding observation, yields the following result (which can also be derived from the decidability of the reachability problem for Petri nets).

Corollary 3.3. *The emptiness problem for message-passing automata is decidable.*

Corollary 3.4. *Message-passing recognisable languages are not closed under complementation.*

Proof Sketch. We saw earlier that $L_{\mathrm{ge}} = \{a^m b^n \mid m \geq n\}$ is message-passing recognisable. We show that the language $L_{\mathrm{lt}} = \{a^m b^n \mid m < n\}$ is not message-passing recognisable. Suppose that L_{lt} is accepted by an automaton $\mathcal{A}_{\mathrm{lt}}$ with M states and N counters. Consider the string $w = a^J b^{J+1}$ where $J = \pi_{M,N,0}$ and let $\rho : \chi_0 \stackrel{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi_n$ be an accepting run of $\mathcal{A}_{\mathrm{lt}}$ on w. By applying the Contraction Lemma (repeatedly, if necessary) to ρ, we can obtain an accepting run ρ' of $\mathcal{A}_{\mathrm{lt}}$ over a word of the form $a^J b^K$, where $K \leq J$, thus contradicting the assumption that $L(\mathcal{A}_{\mathrm{lt}}) = L_{\mathrm{lt}}$. □

4 A Collection of Pumping Lemmas

Our main result is based on a series of pumping lemmas, which we present in this section. For reasons of space, we do not provide any proofs. More details may be found in [MNRS97,MNRS98].

Change vectors For a string w and a symbol x, let $\#_x(w)$ denote the number of times x occurs in w. Let v be a sequence of transitions. Recall that $\alpha(v)$ denotes the corresponding sequence of letters. For each counter C, define $\Delta_C(v)$ to be $\#_{C^+}(\alpha(v)) - \#_{C^-}(\alpha(v))$. The *change vector* associated with v, denoted Δv, is given by $\langle \Delta_C(v) \rangle_{C \in \Gamma}$.

Proposition 4.1. *Let $\mathcal{A} = (Q, \Sigma, \Gamma, T, q_{\mathrm{in}}, F)$ be a message-passing automaton.*

(i) For any computation $\chi \stackrel{v}{\Longrightarrow} \chi'$ of $\mathcal{A}$ and any counter $C \in \Gamma$, $|\Delta_C(v)| \leq |v|$.

(ii) For any configuration χ and sequence of transitions v, $\chi \stackrel{v}{\Longrightarrow}$ iff for each prefix u of v and each counter $C \in \Gamma$, $C(\chi) + \Delta_C(u) \geq 0$.

(iii) Let $\chi \stackrel{u}{\Longrightarrow} \chi' \stackrel{v}{\Longrightarrow}$ with $Q(\chi) = Q(\chi')$ and $n \in \mathbb{N}$ such that, for every counter $C \in \Gamma$, either $\Delta_C(u) \geq 0$ or $C(\chi) \geq n|u| + |v|$. Then, $\chi \stackrel{u^n v}{\Longrightarrow}$.

Proof.

(i) This follows from the fact that each move can change a counter value by at most 1.

(ii) This follows immediately from the definition of a computation.

(iii) The proof is by induction on n.

Basis: For $n = 0$, there is nothing to prove.

Induction step: Let $n > 0$ and assume the result holds for $n-1$. We will show that $\chi \stackrel{u}{\Longrightarrow} \chi' \stackrel{u^{n-1}v}{\Longrightarrow}$.

From the assumption, we know that $\chi \stackrel{u}{\Longrightarrow} \chi'$. To show that $\chi' \stackrel{u^{n-1}v}{\Longrightarrow}$, we examine the value of each counter C at χ'. If $\Delta_C(u) < 0$, then $C(\chi) \geq$

$n|u| + v$. Since $C(\chi') = C(\chi') + \Delta_C(u)$ and $|\Delta_C(u)| \le |u|$, it follows that $C(\chi') \ge (n-1)|u| + v$. From the induction hypothesis, we can then conclude that $\chi' \overset{u^{n-1}v}{\Longrightarrow}$.

□

Pumpable decomposition Let $\mathcal{A}$ be a message-passing automaton with N counters and let $\rho : \chi_0 \overset{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ be a computation of $\mathcal{A}$. A decomposition $\chi_0 \overset{u_1}{\Longrightarrow} \chi_{i_1} \overset{v_1}{\Longrightarrow} \chi_{j_1} \overset{u_2}{\Longrightarrow} \chi_{i_2} \overset{v_2}{\Longrightarrow} \chi_{j_2} \overset{u_3}{\Longrightarrow} \cdots \overset{u_n}{\Longrightarrow} \chi_{i_n} \overset{v_n}{\Longrightarrow} \chi_{j_n} \overset{u_{n+1}}{\Longrightarrow} \chi_m$ of ρ is said to be *pumpable* if it satisfies the following conditions:

(i) $n \le N$.
(ii) For each $k \in [1..n]$, $Q(\chi_{i_k}) = Q(\chi_{j_k})$.
(iii) For each v_k, $k \in [1..n]$, Δv_k is non-zero and has at least one positive entry.
(iv) Let C be a counter and $k \in [1..n]$ such that $\Delta_C(v_k)$ is negative. Then, there exists $\ell < k$ such that $\Delta_C(v_\ell)$ is positive.

We refer to $v_1, v_2, \ldots, v_n$ as the *pumpable blocks* of the decomposition. We say that a counter C is *pumpable* if $\Delta_C(v_i) > 0$ for some pumpable block v_i. The following lemma shows that all the pumpable counters of a pumpable decomposition are simultaneously unbounded. We omit the proof. (This is similar to a well-known result of Karp and Miller in the theory of vector addition systems [KM69].)

Lemma 4.2 (Counter Pumping). *Let $\mathcal{A}$ be a message-passing automaton and ρ a K-run of $\mathcal{A}$, $K \in \mathbb{N}$, with a pumpable decomposition of the form $\chi_0 \overset{u_1}{\Longrightarrow} \chi_{i_1} \overset{v_1}{\Longrightarrow} \chi_{j_1} \overset{u_2}{\Longrightarrow} \chi_{i_2} \overset{v_2}{\Longrightarrow} \chi_{j_2} \cdots \overset{u_n}{\Longrightarrow} \chi_{i_n} \overset{v_n}{\Longrightarrow} \chi_{j_n} \overset{u_{n+1}}{\Longrightarrow} \chi_m$. Then, for any $I, J \in \mathbb{N}$, with $I \ge 1$, there exist $\ell_1, \ell_2, \ldots, \ell_n \in \mathbb{N}$ and a K-run ρ' of $\mathcal{A}$ of the form $\chi'_0 \overset{u_1}{\Longrightarrow} \chi'_{i'_1} \overset{v_1^{\ell_1}}{\Longrightarrow} \chi'_{j'_1} \overset{u_2}{\Longrightarrow} \chi'_{i'_2} \overset{v_2^{\ell_2}}{\Longrightarrow} \chi'_{j'_2} \cdots \overset{u_n}{\Longrightarrow} \chi'_{i'_n} \overset{v_n^{\ell_n}}{\Longrightarrow} \chi'_{j'_n} \overset{u_{n+1}}{\Longrightarrow} \chi'_p$ such that ρ' satisfies the following properties:*

(i) $\chi_0 = \chi'_0$.
(ii) $Q(\chi'_p) = Q(\chi_m)$.
(iii) For $i \in [1..n]$, $\ell_i \ge I$.
(iv) For every counter C, $C(\chi'_p) \ge C(\chi_m)$.
(v) Let Γ_{pump} *be the set of pumpable counters in the pumpable decomposition of* ρ. *For each counter* $C \in \Gamma_{\text{pump}}$, $C(\chi'_p) \ge J$.

Proof. The proof is by induction on n, the number of pumpable blocks in the decomposition.

Basis: If $n = 0$, there is nothing to prove.

Induction step: Let $n > 0$ and assume the lemma holds for all decompositions with $n-1$ pumpable blocks. For each counter C, let $J_C = \max(J, C(\chi_m))$.

By the induction hypothesis, for all $I', J' \in \mathbb{N}$, $I' \geq 1$, we can transform the prefix $\sigma : \chi_0 \stackrel{u_1}{\Longrightarrow} \chi_{i_1} \stackrel{v_1}{\Longrightarrow} \chi_{j_1} \stackrel{u_2}{\Longrightarrow} \cdots \stackrel{v_{n-1}}{\Longrightarrow} \chi_{j_{n-1}} \stackrel{u_n}{\Longrightarrow} \chi_{i_n}$ of ρ into a K-run $\sigma' : \chi'_0 \stackrel{u_1}{\Longrightarrow} \chi'_{i'_1} \stackrel{v_1^{\ell_1}}{\Longrightarrow} \chi'_{j'_1} \stackrel{u_2}{\Longrightarrow} \cdots \stackrel{v_{n-1}^{\ell_{n-1}}}{\Longrightarrow} \chi'_{j'_{n-1}} \stackrel{u_n}{\Longrightarrow} \chi'_{i'_n}$ satisfying the conditions of the lemma. We shall choose I' and J' so that the transition sequence $v_n^{\ell_n} u_{n+1}$ can be appended to σ' to yield the run claimed by the lemma.

To fix values for I' and J', we first estimate the value of ℓ_n, the number of times we need to pump v_n to satisfy all the conditions of the lemma. Let $\Gamma^n_{\text{pos}} = \{C \mid \Delta_C(v_n) > 0\}$. It is sufficient if the number ℓ_n is large enough for each counter $C \in \Gamma^n_{\text{pos}}$ to exceed J_C at the end of the new computation. For a counter $C \in \Gamma^n_{\text{pos}}$ to be above J_C at the end of the computation, it is sufficient for C to have the value $J_C + |u_{n+1}|$ after $v_n^{\ell_n}$. By the induction hypothesis, the value of C before $v_n^{\ell_n}$ is at least $C(\chi_{i_n})$. Hence, it would take $\lceil \frac{J_C + |u_{n+1}| - C(\chi_{i_n})}{\Delta_C(v_n)} \rceil$ iterations of v_n for C to reach the required value after $v_n^{\ell_n}$. On the other hand, we should also ensure that $\ell_n \geq I$. Thus, it is safe to set ℓ_n to be the maximum of I and $\max_{C \in \Gamma^n_{\text{pos}}} \lceil \frac{J_C + |u_{n+1}| - C(\chi_{i_n})}{\Delta_C(v_n)} \rceil$.

We set $I' = I$ and estimate a value for J' such that $\chi'_{i'_n} \stackrel{v_n^{\ell_n} u_{n+1}}{\Longrightarrow} \chi'_p$ with each counter $C \in (\Gamma \setminus \Gamma^n_{\text{pos}})$ achieving a value of at least $C(\chi_m)$ at χ'_p and each counter $C \in (\Gamma_{\text{pos}} \setminus \Gamma^n_{\text{pos}})$ achieving a value of at least J_C at χ'_p.

By the induction hypothesis, $Q(\chi'_{i'_n}) = Q(\chi_{i_n})$ and $F(\chi'_{i'_n}) \geq F(\chi_{i_n})$. Since $\chi_{i_n} \stackrel{v_n u_{n+1}}{\Longrightarrow}$, it follows that $\chi'_{i'_n} \stackrel{v_n u_{n+1}}{\Longrightarrow}$. By Proposition 4.1 (iii), to ensure that $\chi'_{i'_n} \stackrel{v_n^{\ell_n} u_{n+1}}{\Longrightarrow} \chi'_p$, it is sufficient to raise each counter C with $\Delta_C(v_n) < 0$ to a value of at least $\ell_n|v_n| + |u_{n+1}|$ at $\chi'_{i'_n}$. If $\Delta_C(v_n) < 0$ then, by the definition of pumpable decompositions, $\Delta_C(v_i) > 0$ for some $i \in [1..n{-}1]$, so C gets pumped above J' in σ'.

Any counter C such that $\Delta_C(v_n) \geq 0$ will surely exceed $C(\chi_m)$ at χ'_p. On the other hand, a counter C such that $\Delta_C(v_n) < 0$ can decrease by at most $\ell_n|v_n| + |u_{n+1}|$ after $\chi'_{i'_n}$.

Putting these two facts together, it suffices to set J' to $\ell_n|v_n| + |u_{n+1}| + \max_{\{C \mid \Delta_C(v_n) < 0\}} J_C$.

Let $\rho' : \chi'_0 \stackrel{u_1}{\Longrightarrow} \chi'_{i'_1} \stackrel{v_1^{\ell_1}}{\Longrightarrow} \chi'_{j'_1} \stackrel{u_2}{\Longrightarrow} \cdots \stackrel{u_n}{\Longrightarrow} \chi'_{i'_n} \stackrel{v_n^{\ell_n}}{\Longrightarrow} \chi'_{j'_n} \stackrel{u_{n+1}}{\Longrightarrow} \chi'_p$. By the induction hypothesis, we know that $\chi'_0 = \chi_0$ and for $i \in [1..n{-}1]$, $\ell_i \geq I$. By construction, $\ell_n \geq I$ as well. We have also ensured that for every counter C, $C(\chi'_p) \geq C(\chi_m)$ and for every counter $C \in \Gamma_{\text{pos}}$, $C(\chi'_p) \geq J$. The fact that $Q(\chi'_p) = Q(\chi_m)$ follows from the fact that each v_n loop brings the automaton back to $Q(\chi'_{i'_n}) = Q(\chi_{i_n})$, and the fact that both ρ and ρ' go through the same sequence of transitions u_{n+1} at the end of the computation. $\square$

Having shown that all pumpable counters of a pumpable decomposition can be simultaneously raised to arbitrarily high values, we describe a sufficient condition for a K-run to admit a non-trivial pumpable decomposition.

Strong pumping constant For each $M, N, K \in \mathbb{N}$, we define the *strong pumping constant* $\Pi_{M,N,K}$ by induction on N as follows (recall that $\pi_{M,N,K}$ denotes the weak pumping constant for (M, N, K)):

$$\forall M, K \in \mathbb{N}. \quad \Pi_{M,0,K} = 1$$
$$\forall M, N, K \in \mathbb{N}. \; \Pi_{M,N+1,K} = \Pi_{M,N,\pi_{M,N+1,K}+K} + \pi_{M,N+1,K} + K$$

Lemma 4.3 (Decomposition). *Let $\mathcal{A}$ be a message-passing automaton with M states and N counters and let $K \in \mathbb{N}$. Let $\rho : \chi_0 \stackrel{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ be any K-run of $\mathcal{A}$. Then, there is a pumpable decomposition $\chi_0 \stackrel{u_1}{\Longrightarrow} \chi_{i_1} \stackrel{v_1}{\Longrightarrow} \chi_{j_1} \stackrel{u_2}{\Longrightarrow} \chi_{i_2} \stackrel{v_2}{\Longrightarrow} \chi_{j_2} \cdots \stackrel{u_n}{\Longrightarrow} \chi_{i_n} \stackrel{v_n}{\Longrightarrow} \chi_{j_n} \stackrel{u_{n+1}}{\Longrightarrow} \chi_m$ of ρ such that for every counter C, if $C(\chi_j) > \Pi_{M,N,K}$ for some $j \in [0..m]$, then there exists $k \in [1..n]$, such that $\Delta_C(v_k)$ is positive.*

To prove this lemma, we need the following result.

Proposition 4.4. *Let $\mathcal{A}$ be a message-passing automaton with M states and N counters and let $\rho : \chi_0 \Longrightarrow \chi_n$ be a K-run of $\mathcal{A}$ in which some counter value exceeds $\pi_{M,N,K} + K$. Then, there is a prefix $\sigma : \chi_0 \Longrightarrow \chi_s$ of ρ such that:*

- *For each $m \in [0..s]$ and every counter C, $C(\chi_m) < \pi_{M,N,K} + K$.*
- *There exists $r \in [0..s-1]$, such that $\sigma : \chi_0 \Longrightarrow \chi_r \Longrightarrow \chi_s$, $Q(\chi_r) = Q(\chi_s)$ and $F(\chi_r) < F(\chi_s)$.*

Proof. Suppose that the lemma does not hold. Let $\rho : \chi_0 \stackrel{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi_n$ be a computation of minimum length which fails to satisfy the lemma. Since the initial counter values in ρ are bounded by K and some counter value exceeds $\pi_{M,N,K} + K$ in ρ, it must be the case that the length of ρ is at least $\pi_{M,N,K}$.

By the definition of $\pi_{M,N,K}$, there exist i and j, $i < j \leq \pi_{M,N,K}$ such that $Q(\chi_i) = Q(\chi_j)$ and $F(\chi_i) \leq F(\chi_j)$. Since ρ is a K-run and $j \leq \pi_{M,N,K}$, all counter values at the configurations $\chi_0, \chi_1, \ldots, \chi_j$ must be bounded by $\pi_{M,N,K} + K$. If $F(\chi_i) < F(\chi_j)$, ρ would satisfy the lemma with $r = i$ and $s = j$, so it must be the case $F(\chi_i) = F(\chi_j)$.

Since $\chi_i = \chi_j$, we can construct a shorter computation $\rho' = \chi_0 \stackrel{t_1 t_2 \ldots t_i}{\Longrightarrow} \chi_i \stackrel{t_{j+1}}{\longrightarrow} \chi_{j+1} \stackrel{t_{j+2}}{\longrightarrow} \cdots \stackrel{t_n}{\longrightarrow} \chi_n$. It is easy to see that the same counter whose value exceeded $\pi_{M,N,K} + K$ in ρ must also exceed $\pi_{M,N,K} + K$ in ρ'—the only configurations visited by ρ which are not visited by ρ' are those in the interval $\chi_{i+1}, \chi_{i+2}, \ldots \chi_j$. However, we have already seen that all counter values in $\chi_0, \chi_1, \ldots, \chi_j$ are bounded by $\pi_{M,N,K} + K$.

It is clear that if ρ' satisfies the lemma, then so does ρ. On the other hand, if ρ' does not satisfy the lemma, then ρ is not a minimum length counterexample to the lemma. In either case we obtain a contradiction. $\square$

We now return to the proof of the Decomposition Lemma.

Proof. (of Lemma 4.3) The proof is by induction on N, the number of counters.

Basis: If $N=0$, set $n=0$ and $u_1=\rho$.

Induction step: Let Γ_{gt} denote the set of counters whose values exceed $\Pi_{M,N,K}$ in the K-run ρ.

If $\Gamma_{\mathrm{gt}}=\emptyset$, we set $n=0$ and $u_1=\rho$.

Otherwise, by Proposition 4.4, we can find positions r and s in ρ such that $\chi_0 \overset{u'}{\Longrightarrow} \chi_r \overset{v'}{\Longrightarrow} \chi_s \Longrightarrow \chi_m$, with $Q(\chi_r)=Q(\chi_s)$, $F(\chi_r)<F(\chi_s)$ and all counter values at $\chi_0,\chi_1,\ldots,\chi_s$ bounded by $\pi_{M,N,K}+K$.

Let Σ be the input alphabet of $\mathcal{A}$ and Γ its set of counters. Fix a counter C' in which increases strictly between χ_r and χ_s—that is, $C'(\chi_s)>C'(\chi_r)$. By our choice of χ_r and χ_s, such a counter must exist. Construct an automaton $\mathcal{A}'$ with input alphabet $\Sigma\cup\{C'^{+},C'^{-}\}$ and counters $\Gamma\setminus\{C'\}$. The states and transitions of $\mathcal{A}'$ are the same as those of $\mathcal{A}$. In other words, $\mathcal{A}'$ behaves like $\mathcal{A}$ except that it treats moves involving the counter C' as input letters.

Consider the computation $\chi_s \overset{t_{s+1}t_{s+2}\ldots t_m}{\Longrightarrow} \chi_m$ of $\mathcal{A}$. It is easy to see that there is a corresponding computation $\rho' : \chi'_s \overset{t_{s+1}t_{s+2}\ldots t_m}{\Longrightarrow} \chi'_m$ of $\mathcal{A}'$ such that for each $k\in[s..m]$, $Q(\chi_k)=Q(\chi'_k)$ and for each counter $C\neq C'$, $C(\chi_k)=C(\chi'_k)$.

From Proposition 4.4, we know that ρ' is in fact a $(\pi_{M,N,K}{+}K)$-run of $\mathcal{A}'$. Further, for every counter C in $\Gamma_{\mathrm{gt}}\setminus\{C'\}$, there exists a $j\in[s..m]$, such that $C(\chi'_j)=C(\chi_j)>\Pi_{M,N,K}>\Pi_{M,N-1,\pi_{M,N,K}+K}$. (In the K-run ρ, no counter could have exceeded $\Pi_{M,N,K}$ before χ_s because Proposition 4.4 guarantees that all counter values at $\chi_0,\chi_1,\ldots,\chi_s$ are bounded by $\pi_{M,N,K}{+}K$.) By the induction hypothesis, we can find a pumpable decompostion

$$\chi'_s \overset{u'_1}{\Longrightarrow} \chi'_{i'_1} \overset{v'_1}{\Longrightarrow} \chi'_{j'_1} \overset{u'_2}{\Longrightarrow} \chi'_{i'_2} \overset{v'_2}{\Longrightarrow} \chi'_{j'_2} \overset{u'_3}{\Longrightarrow} \cdots \overset{u'_p}{\Longrightarrow} \chi'_{i'_p} \overset{v'_p}{\Longrightarrow} \chi'_{j'_p} \overset{u'_{p+1}}{\Longrightarrow} \chi_m$$

of ρ' such that if C is a counter with $C(\chi'_j)>\Pi_{M,N-1,\pi_{M,N,K}+K}$ for some $j\in[s..m]$, then there exists $k\in[1..p]$ such that $\Delta_C(v'_k)$ is positive.

Consider the corresponding computation

$$\chi_s \overset{u'_1}{\Longrightarrow} \chi_{i'_1} \overset{v'_1}{\Longrightarrow} \chi_{j'_1} \overset{u'_2}{\Longrightarrow} \chi_{i'_2} \overset{v'_2}{\Longrightarrow} \chi_{j'_2} \cdots \overset{u'_p}{\Longrightarrow} \chi_{i'_p} \overset{v'_p}{\Longrightarrow} \chi_{j'_p} \overset{u'_{p+1}}{\Longrightarrow} \chi_m$$

of $\mathcal{A}$. In this computation, for each $k\in[1..p]$, $Q(\chi_{i'_k})=Q(\chi'_{i'_k})=Q(\chi'_{j'_k})=Q(\chi_{j'_k})$. Further, for each $C\in\Gamma_{\mathrm{gt}}\setminus\{C'\}$, $C(\chi_{i'_k})=C(\chi'_{i'_k})$ and $C(\chi_{j'_k})=C(\chi'_{j'_k})$.

We prefix the computation $\chi_s \overset{u'_1v'_1\ldots u'_{p+1}}{\Longrightarrow} \chi_m$ with the K-run $\chi_0 \overset{u'}{\Longrightarrow} \chi_r \overset{v'}{\Longrightarrow} \chi_s$ which we used to identify χ_s and χ_r. We then assert that the composite K-run

$$\chi_0 \overset{u'}{\Longrightarrow} \chi_r \overset{v'}{\Longrightarrow} \chi_s \overset{u'_1}{\Longrightarrow} \chi_{i''_1} \overset{v'_1}{\Longrightarrow} \chi_{j''_1} \overset{u'_2}{\Longrightarrow} \chi_{i''_2} \overset{v'_2}{\Longrightarrow} \chi_{j''_2} \cdots \overset{u'_p}{\Longrightarrow} \chi_{i''_p} \overset{v'_p}{\Longrightarrow} \chi_{j''_p} \overset{u'_{p+1}}{\Longrightarrow} \chi_m.$$

provides the decomposition

$$\chi_0 \overset{u_1}{\Longrightarrow} \chi_{i_1} \overset{v_1}{\Longrightarrow} \chi_{j_1} \overset{u_2}{\Longrightarrow} \chi_{i_2} \overset{v_2}{\Longrightarrow} \chi_{j_2} \cdots \overset{u_n}{\Longrightarrow} \chi_{i_n} \overset{v_n}{\Longrightarrow} \chi_{j_n} \overset{u_{n+1}}{\Longrightarrow} \chi_m$$

of ρ claimed in the statement of the lemma. In other words, $u_1 = u'$, $v_1 = v'$, $\chi_{i_1} = \chi_r$ and $\chi_{j_1} = \chi_s$, while for $k \in [2..n]$, $u_k = u'_{k-1}$, $v_k = v'_{k-1}$, $\chi_{i_k} = \chi_{i'_{k-1}}$ and $\chi_{j_k} = \chi_{j'_{k-1}}$.

Let us verify that this decomposition satisfies all the conditions required by the lemma.

First we verify that this decomposition is pumpable.

- Since $p \leq N-1$, it is clear than $n = p+1 \leq N$.
- By construction $Q(\chi_{i_1}) = Q(\chi_r) = Q(\chi_s) = Q(\chi_{j_1})$. For $k \in [2..n]$, $Q(\chi_{i_k}) = Q(\chi_{i'_{k-1}}) = Q(\chi_{j'_{k-1}}) = Q(\chi_{j_k})$.
- We know that $\Delta v_1 = \Delta v'$ is non-zero and strictly positive by the choice of v'. For $k \in [2..n]$, we know that $\Delta_C(v_k) = \Delta_C(v'_{k-1})$ for $C \neq C'$. Since we have already established that $\Delta v'_{k-1}$ is non-zero and has at least one positive entry for $k \in [2..n]$, it follows that the corresponding change vectors Δv_k are also non-zero and have at least one positive entry.
- Let C be a counter and $k \in [1..n]$ such that $\Delta_C(v_k)$ is negative. Since $\Delta v_1 = \Delta v'$ is positive by the choice of v, it must be that $k \in [2..n]$. If $C \neq C'$, then $\Delta_C(v'_{k-1}) = \Delta_C(v_k)$ is negative. In this case, we already know that there exists $\ell \in [2..k-1]$, such that $\Delta_C(v'_{\ell-1}) = \Delta_C(v_\ell)$ is positive.
  On the other hand, if $C = C'$, it could be that $\Delta_{C'}(v'_z)$ is negative for all $z \in [1..p]$, since C' is treated as an input letter rather than as a counter in the automaton $\mathcal{A}'$. However, we know that $\Delta_{C'}(v_1) = \Delta_{C'}(v')$ is positive by the choice of v' and C', so C' also satisfies the condition of the lemma.

Finally, let C be a counter such that $C(\chi_j) > \Pi_{M,N,K}$ for some $j \in [1..m]$. If $C \neq C'$, then $C(\chi_j) > \Pi_{M,N-1,\pi_{M,N,K}+K}$ for some $j \in [s..m]$, so we already know that $\Delta_C(v'_{k-1}) = \Delta_C(v_k)$ is positive for some $k \in [2..n]$. On the other hand, if $C = C'$, we know that $\Delta_C(v_1) = \Delta_C(v')$ is positive by the choice of v' and C'.

□

The Counter Pumping Lemma allows us to pump blocks of transitions in a computation. However, it is possible for a pumpable block to consist solely of invisible transitions which increment and decrement counters. Using the Decomposition Lemma, we can prove a more traditional kind of pumping lemma, stated in terms of input strings. We omit the proof.

Lemma 4.5 (Visible Pumping). *Let L be a message-passing recognisable language. There exists $n \in \mathbb{N}$ such that for all input strings w, if $w \in L$ and $|w| \geq n$ then w can be written as $w_1w_2w_3$ such that $|w_1w_2| \leq n$, $|w_2| \geq 1$ and $w_1w_2^iw_3 \in L$ for all $i \geq 1$.*

Another consequence of Lemmas 4.2 and 4.3 is a strict hierarchy theorem for message-passing automata, whose proof we omit.

Lemma 4.6 (Counter Hierarchy). *For $k \in \mathbb{N}$, let $\mathcal{L}_k$ be the set of languages recognisable by message-passing automata with k counters. Then, for all k, $\mathcal{L}_k \subsetneq \mathcal{L}_{k+1}$.*

5 Regularity of Message-Passing Recognisable Languages

Automata with bounded counters

Let $\mathcal{A} = (Q, \Sigma, \Gamma, T, q_{\text{in}}, F)$ be a message-passing automaton. For $K \in \mathbb{N}$, define $\mathcal{A}[K] = (Q[K], T[K], Q[K]_{\text{in}}, F[K])$ to be the finite-state automaton over the alphabet $\Sigma \cup \Gamma^{\pm}$ given by:

- $Q[K] = Q \times \{f \mid f : \Gamma \longrightarrow [0..K]\}$, with $Q[K]_{\text{in}} = (q_{\text{in}}, \overline{0})$.
- $F[K] = F \times \{f \mid f : \Gamma \longrightarrow [0..K]\}$.
- If $(q, d, q') \in T$, then $((q, f), d, (q', f')) \in T[K]$ where:
  - If $d \in \Sigma$, $f' = f$.
  - If $d = C^+$, $f'(C') = f(C')$ for all $C' \neq C$ and
  $$f'(C) = \begin{cases} f(C)+1 & \text{if } f(C) < K \\ K & \text{otherwise.} \end{cases}$$
  - If $d = C^-$, $f'(C') = f(C')$ for all $C' \neq C$, $f(C) \geq 1$ and
  $$f'(C) = \begin{cases} f(C)-1 & \text{if } f(C) < K \\ K & \text{otherwise.} \end{cases}$$

Each transition $t = ((q, f), d, (q', f')) \in T[K]$ corresponds to a unique transition $(q, d, q') \in T$, which we denote t^{-1}. For a sequence of transitions $t_1 t_2 \ldots t_n$, we write $(t_1 t_2 \ldots t_n)^{-1}$ for $t_1^{-1} t_2^{-1} \ldots t_n^{-1}$. For any sequence $t_1 t_2 \ldots t_n$ of transitions in $T[K]$, $\alpha(t_1 t_2 \ldots t_n) = \alpha((t_1 t_2 \ldots t_n)^{-1})$. Moreover, if $(q_0, f_0') \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} (q_n, f_n')$ and $(q_0, f_0) \overset{(t_1 t_2 \ldots t_n)^{-1}}{\Longrightarrow} \chi_n$, then $Q(\chi_n) = q_n$.

Thus, the finite-state automaton $\mathcal{A}[K]$ behaves like a message-passing automaton except that it deems any counter whose value attains a value K to be "full" . Once a counter is declared to be full, it can be decremented as many times as desired. The following observations are immediate.

Proposition 5.1. *(i) If* $(q_0, f_0') \xrightarrow{t_1'} (q_1, f_1') \xrightarrow{t_2'} \cdots \xrightarrow{t_n'} (q_n, f_n')$ *is a computation of* $\mathcal{A}$ *then,* $(q_0, f_0) \xrightarrow{t_1} (q_1, f_1) \xrightarrow{t_2} \cdots \xrightarrow{t_n} (q_n, f_n)$ *is a computation of* $\mathcal{A}[K]$ *where*

- $t_1' t_2' \ldots t_n' = (t_1 t_2 \ldots t_n)^{-1}$.
- $\forall C \in \Gamma.\ \forall i \in [1..n].\ f_i(C) = \begin{cases} f_i'(C) & \text{if } f_j'(C) < K \text{ for all } j \leq i \\ K & \text{otherwise} \end{cases}$

(ii) Let $(q_0, f_0) \xrightarrow{t_1} (q_1, f_1) \xrightarrow{t_2} \cdots \xrightarrow{t_n} (q_n, f_n)$ *be a computation of* $\mathcal{A}[K]$*. Then there is a maximal prefix* $t_1 t_2 \ldots t_\ell$ *of* $t_1 t_2 \ldots t_n$ *such that there is a computation* $(q_0, f_0') \xrightarrow{t_1^{-1}} (q_1, f_1') \xrightarrow{t_2^{-1}} \ldots \xrightarrow{t_\ell^{-1}} (q_\ell, f_\ell')$ *of* $\mathcal{A}$ *with* $f_0 = f_0'$*. Moreover, if* $\ell < n$*, then for some counter* C*,* $\alpha(t_{\ell+1}') = C^-$*,* $f_\ell'(C) = 0$ *and there is a* $j < \ell$ *such that* $f_j'(C) = K$*.*

(iii) Let $L(\mathcal{A}[K])$ *be the language over* $\Sigma \cup \Gamma^{\pm}$ *accepted by* $\mathcal{A}[K]$*. Let* $L_\Sigma(\mathcal{A}[K]) = \{w{\restriction}_\Sigma \mid w \in L(\mathcal{A}[K])\}$*. Then,* $L(\mathcal{A}) \subseteq L_\Sigma(\mathcal{A}[K])$*.*

Synchronised products of message-passing automata

Product automata Let $\mathcal{A}_i = (Q_i, \Sigma_i, \Gamma_i, T_i, q_{\text{in}}^i, F_i)$, $i = 1,2$, be a pair of message-passing automata. The *product automaton* $\mathcal{A}_1 \times \mathcal{A}_2$ is the structure $(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \Gamma_1 \cup \Gamma_2, T_1 \times T_2, (q_{\text{in}}^1, q_{\text{in}}^2), F_1 \times F_2)$, where $((q_1, q_2), d, (q_1', q_2')) \in T_1 \times T_2$ iff one of the following holds:

- $d \in (\Sigma_1 \cup \Gamma_1) \cap (\Sigma_2 \cup \Gamma_2)$ and $(q_i, d, q_i') \in T_i$ for $i \in \{1,2\}$.
- $d \in (\Sigma_1 \cup \Gamma_1) \setminus (\Sigma_2 \cup \Gamma_2)$, $(q_1, d, q_1') \in T_1$ and $q_2 = q_2'$.
- $d \in (\Sigma_2 \cup \Gamma_2) \setminus (\Sigma_1 \cup \Gamma_1)$, $(q_2, d, q_2') \in T_2$ and $q_1 = q_1'$.

For $t = ((q_1, q_2), d, (q_1', q_2')) \in T$ and $i \in \{1,2\}$, let $\pi_i(t)$ denote (q_i, d, q_i') if $d \in (\Sigma_i \cup \Gamma_i)$ and the empty string ε otherwise. As usual, $\pi_i(t_1 t_2 \ldots t_n)$ is just $\pi_i(t_1)\pi_i(t_2)\ldots\pi_i(t_n)$. Thus, for a sequence of transitions $\rho = t_1 t_2 \ldots t_n$ over $T_1 \times T_2$, $\pi_1(\rho)$ and $\pi_2(\rho)$ denote the projections of ρ onto the transitions of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively. Clearly, $\alpha(t_1 t_2 \ldots t_n)\restriction_{(\Sigma_i \cup \Gamma_i)} = \alpha(\pi_i(t_1 t_2 \ldots t_n))$ for $i \in \{1,2\}$.

We shall often write a configuration $((q_1, q_2), f)$ of $\mathcal{A}_1 \times \mathcal{A}_2$ as a pair of configurations $((q_1, f_1), (q_2, f_2))$ of $\mathcal{A}_1$ and $\mathcal{A}_2$, where f_1 and f_2 are restrictions of f to Γ_1 and Γ_2 respectively.

The following observations are easy consequences of the definition of product automata.

Proposition 5.2. *(i)* $((q_{\text{in}}^1, \overline{0}), (q_{\text{in}}^2, \overline{0})) \stackrel{t_1 t_2 \ldots t_n}{\Longrightarrow} ((q_1, f_1), (q_2, f_2))$ *is a computation of* $\mathcal{A}_1 \times \mathcal{A}_2$ *if and only if* $(q_{\text{in}}^1, \overline{0}) \stackrel{\pi_1(t_1 t_2 \ldots t_n)}{\Longrightarrow} (q_1, f_1)$ *and* $(q_{\text{in}}^2, \overline{0}) \stackrel{\pi_2(t_1 t_2 \ldots t_n)}{\Longrightarrow} (q_2, f_2)$ *are computations of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *respectively.*
(ii) If $\Sigma_1 = \Sigma_2$ *and* $\Gamma_1 \cap \Gamma_2 = \emptyset$, *then* $L(\mathcal{A}_1 \times \mathcal{A}_2) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

5.1 Regularity and closure under complementation

Our first characterisation of regular message-passing recognisable languages is the following.

Theorem 5.3. *Let* L *be a language over* Σ. L *and* $\overline{L}$ *are message-passing recognisable iff* L *is regular.*

This result is related to the main result of [MNRS98] which states that if we record the behaviour of message-passing systems as tuples of sequences, every *robust* system is effectively finite-state. In the sequential setting, a language L would be robust in the sense of [MNRS98] if there were a single automaton $\mathcal{A}$ with accept and reject states such that for each word $w \in L$, *every* run of $\mathcal{A}$ on w leads to an accept state and for each word $w \notin L$, *every* run of $\mathcal{A}$ on w leads to a reject state. Here, the requirement on L is much weaker—all we demand is that both L and $\overline{L}$ be accepted independently by message-passing automata, possibly with very different state spaces.

To prove the theorem, we need an auxiliary result. Let $L \subseteq \Sigma^*$ be such that both L and its complement $\overline{L}$ are message-passing recognisable. Let $L = L(\mathcal{A})$ and $\overline{L} = L(\overline{\mathcal{A}})$, where we may assume that $\mathcal{A}$ and $\overline{\mathcal{A}}$ use disjoint sets of counters.

The language accepted by $\mathcal{A} \times \overline{\mathcal{A}}$ must be empty. Let M be the number of states of $\mathcal{A} \times \overline{\mathcal{A}}$ and N be the number of counters that it uses. Let K be a number greater than $\Pi_{M,N,0}$, the strong pumping constant for $(M, N, 0)$. Recall that $\mathcal{A}[K] = (Q[K], T[K], Q[K]_{\text{in}}, F[K])$ is a finite-state automaton *without* counters working on the input alphabet $\Sigma \cup \Gamma^{\pm}$.

Lemma 5.4. $L(\mathcal{A}[K] \times \overline{\mathcal{A}}) = \emptyset$.

Proof. Let $\mathcal{A}[K] = (Q[K], T[K], Q[K]_{\text{in}}, F[K])$ and $\overline{\mathcal{A}} = (\overline{Q}, \Sigma, \overline{\Gamma}, \overline{T}, \overline{q}_{\text{in}} \overline{F})$. Each computation ρ of $\mathcal{A}[K] \times \overline{\mathcal{A}}$ is of the form $((q_0, \overline{0}), (\overline{q}_0, \overline{0})) \xrightarrow{u_1} ((q_1, f_1), (\overline{q}_1, \overline{f}_1)) \xrightarrow{u_2} \cdots \xrightarrow{u_n} ((q_n, f_n), (\overline{q}_n, \overline{f}_n))$, where, for $i \in [0..n]$, $u_i \in T[K] \times \overline{T}$.

By Propositions 5.1 and 5.2, corresponding to the sequence $u_1 u_2 \ldots u_n$ there exists a maximal sequence of transitions $v_1 v_2 \ldots v_m$ of $\mathcal{A} \times \overline{\mathcal{A}}$ where:

- Each v_i belongs to $T \times \overline{T}$.
- For each $i \in [1..m]$, $\pi_2(v_i) = \pi_2(u_i)$.
- For each $i \in [1..m]$, $\pi_1(v_i) = \begin{cases} (\pi_1(u_i))^{-1} & \text{if } \pi_1(u_i) \neq \varepsilon \\ \varepsilon & \text{otherwise} \end{cases}$
- $\rho' : ((q_0, \overline{0}), (\overline{q}_0, \overline{0})) \xrightarrow{v_1} ((q_1, f'_1), (\overline{q}_1, \overline{f}_1)) \xrightarrow{v_2} \cdots \xrightarrow{v_m} ((q_m, f'_m), (\overline{q}_m, \overline{f}_m))$ is a computation of $\mathcal{A} \times \overline{\mathcal{A}}$.
- If $m < n$, then for some $\hat{C} \in \Gamma$, $\alpha(u_{m+1}) = \hat{C}^-$, $f'_m(\hat{C}) = 0$ and $f'_j(\hat{C}) = K$ for some $j \in [0..m]$.

Let us define the *residue length* of ρ to be $n - m$.

Suppose that $L(\mathcal{A}[K] \times \overline{\mathcal{A}})$ is non-empty. Since $L(\mathcal{A} \times \overline{\mathcal{A}})$ is empty, it is easy to see that any accepting run of $\mathcal{A}[K] \times \overline{\mathcal{A}}$ has a non-zero residue length. Without loss of generality, assume that the run ρ considered earlier is an accepting run of $\mathcal{A}[K] \times \overline{\mathcal{A}}$ whose residue length is minimal. Then, in the corresponding run ρ' of $\mathcal{A} \times \overline{\mathcal{A}}$, the counter $\hat{C} \in \Gamma$ attains the value K along ρ' and then goes to 0 at the end of the run so that the move labelled $\hat{C}^-$ is not enabled at $((q_m, f'_m), (\overline{q}_m, \overline{f}_m))$.

Since K exceeds the strong pumping constant for $\mathcal{A} \times \overline{\mathcal{A}}$, by Lemma 4.2 we can find an alternative run $\hat{\rho}' : ((q_0, \overline{0}), (\overline{q}_0, \overline{0})) \overset{v'_1 v'_2 \ldots v'_\ell}{\Longrightarrow} ((q'_\ell, f'_\ell), (\overline{q}'_\ell, \overline{f}'_\ell))$ with $(q'_\ell, \overline{q}'_\ell) = (q_m, \overline{q}_m)$, $f'_\ell(\hat{C}) \geq K$, and all other counter values at $(f'_\ell, \overline{f}'_\ell)$ at least as large as at $(f_m, \overline{f}'_m)$. In particular, *every* counter which exceeded the cutoff value K along ρ' is pumpable and thus exceeds K along $\hat{\rho}'$ as well.

By Propositions 5.1 and 5.2, we can construct a corresponding sequence of transitions $u'_1 u'_2 \ldots u'_\ell$ over $T[K] \times \overline{T}$ such that $\pi_1(v'_1 v'_2 \ldots v'_\ell) = (\pi_1(u'_1 u'_2 \ldots u'_\ell))^{-1}$ and $\pi_2(v'_1 v'_2 \ldots v'_\ell) = \pi_2(u'_1 u'_2 \ldots u'_\ell)$, where $\hat{\rho} : ((q_0, \overline{0}), (\overline{q}_0, \overline{0})) \overset{u'_1 u'_2 \ldots u'_\ell}{\Longrightarrow} ((q''_\ell, f''_\ell), (\overline{q}'_\ell, \overline{f}'_\ell))$ is a run of $\mathcal{A}[K] \times \overline{\mathcal{A}}$ with $(q''_\ell, \overline{q}'_\ell) = (q_m, \overline{q}_m)$ and $f''_\ell(C) \geq f_m(C)$ for each $C \in \Gamma$.

We already know that $\overline{f}'_\ell(C) \geq \overline{f}_m(C)$ for each $C \in \overline{\Gamma}$. Further, since every counter which exceeded the cutoff value K along ρ' also exceeds K along $\hat{\rho}'$, we know that any counter which has become full along ρ would also have saturated along $\hat{\rho}$. Thus, we can extend $\hat{\rho}$ to an accepting run σ by appending the sequence of transitions $u_{m+1}u_{m+2}\dots u_n$ which occur at the end of the accepting run ρ.

Recall that $\alpha(u_{m+1}) = \hat{C}^-$ and $f'_\ell(\hat{C}) \geq 1$ by our choice of $\hat{\rho}'$. From this, it follows that the residue length of the newly constructed accepting run σ is at least one less than the residue length of ρ, which is a contradiction, since ρ was assumed to be an accepting run of minimal residue length. □

We can now prove Theorem 5.3.

Proof. (of Theorem 5.3)
Let $L = L(\mathcal{A})$ and $\overline{L} = L(\overline{\mathcal{A}})$. Define $\mathcal{A}[K]$ as above. We claim that $L_\Sigma(\mathcal{A}[K]) = L(\mathcal{A})$. By Proposition 5.1, we know that $L(\mathcal{A}) \subseteq L_\Sigma(\mathcal{A}[K])$. On the other hand, from the previous lemma it follows that $L_\Sigma(\mathcal{A}[K]) \cap L(\overline{\mathcal{A}}) = \emptyset$. This implies that $L_\Sigma(\mathcal{A}[K]) \subseteq \overline{L(\overline{\mathcal{A}})}$, which means that $L_\Sigma(\mathcal{A}[K]) \subseteq L(\mathcal{A})$. So $L(\mathcal{A}) = L_\Sigma(\mathcal{A}[K])$. Since $\mathcal{A}[K]$ is a finite-state automaton, it follows that $L(\mathcal{A})$ is regular. Therefore, if a language and its complement are message-passing recognisable then the language is regular.

The converse is obvious. □

Observe that our construction is effective—given message-passing automata $\mathcal{A}$ and $\overline{\mathcal{A}}$ for L and $\overline{L}$ respectively, we can construct a finite-state automaton $\mathcal{A}[K]$ for L.

5.2 Regularity and determinacy

Our next characterisation of regularity is in terms of deterministic message-passing automata.

Deterministic Message-Passing Automata A message-passing automaton $\mathcal{A} = (Q, \Sigma, \Gamma, T, q_{\text{in}}, F)$ is said to be *deterministic* if the following two conditions hold:

- If $(q, d_1, q_1), (q, d_2, q_2) \in T$, with $d_1, d_2 \in \Sigma$, then $d_1 = d_2$ implies $q_1 = q_2$.
- If $(q, d_1, q_1), (q, d_2, q_2) \in T$, with $d_1 \in \Gamma^{\pm}$, then $d_1 = d_2$ and $q_1 = q_2$.

Though this notion of determinism seems rather strong, any relaxation of the definition will allow deterministic automata to simulate non-deterministic automata in a trivial manner. If we permit the automaton to choose between a counter move and another transition (which may or may not be a counter move), we can add a spurious counter C and replace any non-deterministic choice between transitions $t_1 = (q, d, q_1)$ and $t_2 = (q, d, q_2)$ by a choice between t_1 and a move (q, C^+, q_C) involving C which leads to a new state where $t'_2 = (q_C, d, q_2)$ is enabled.

The following result characterises languages accepted by deterministic message-passing automata. Similar results have been demonstrated for deterministic Petri net languages [Pel87,V82].

Proposition 5.5. *Let $\mathcal{A}$ be a deterministic message-passing automaton. Then, either $L(\mathcal{A})$ is regular or there is a word $w \notin L(\mathcal{A})$ such that every extension of w also does not belong to $L(\mathcal{A})$.*

Proof Sketch. Let $\mathcal{A}$ be a deterministic message-passing automaton. We say that $\mathcal{A}$ is *Γ-blocked on a word* w if, while reading w, $\mathcal{A}$ gets stuck at a state with a single outgoing edge labelled C^-, for some counter C. If $\mathcal{A}$ is Γ-blocked on w, it is easy to see $\mathcal{A}$ is also Γ-blocked on w' for any extension w' of w, so every extension of w is outside $L(\mathcal{A})$.

If $\mathcal{A}$ is not Γ-blocked for any word w, we construct a finite-state automaton $\mathcal{A}'$ with ε-moves over Σ with the same state space as $\mathcal{A}$. In $\mathcal{A}'$, each counter move (q, d, q'), $d \in \Gamma^{\pm}$, is replaced by a ε-move (q, ε, q'). There is a natural correspondence between computations of $\mathcal{A}$ and runs of $\mathcal{A}'$. It is easy to see that $L(\mathcal{A}) \subseteq L(\mathcal{A}')$. Using the fact that $\mathcal{A}$ is not Γ-blocked for any word w, we can show that any accepting run of $\mathcal{A}'$ can be mapped back to an accepting run of $\mathcal{A}$. Thus $L(\mathcal{A}') \subseteq L(\mathcal{A})$. In other words, $L(\mathcal{A}') = L(\mathcal{A})$, so $L(\mathcal{A})$ is regular. □

The preceding characterisation implies that non-deterministic message-passing automata are strictly more powerful than deterministic message-passing automata. Consider the language $L = \{w \mid w = w_1 a^m b^n a w_2\}$ where $w_1, w_2 \in \{a, b\}^*$ and $m \geq n \geq 1$. This language is message-passing recognisable but violates the condition of Proposition 5.5: L is not regular and for any word $w \notin L$, we can always find an extension of w in L—for instance, $waba \in L$ for all $w \in \{a, b\}^*$.

Observe, however, that even deterministic message-passing automata are strictly more powerful than normal finite-state automata. For instance, the language L_{ge} of Example 2.1 is not regular but the automaton accepting the language is deterministic.

With these observations about deterministic automata behind us, we can now state our alternative characterisation of regularity. First, we need some notation. For a string w, let w^R denote the string obtained by reading w from right to left. For a language L, let L^R be the set of strings $\{w^R \mid w \in L\}$.

Theorem 5.6. *Let L be a message-passing recognisable language such that L^R is recognised by a deterministic message-passing automaton. Then, L is regular.*

The idea is to compute a constant τ which depends on the number of states M and the number of counters N of $\mathcal{A}$ and to then show that L is recognised by the finite-state automaton $L[\tau]$. The proof is quite involved and we omit it due to lack of space.

6 Discussion

Other models for asynchronous commmunication Many earlier attempts to model asynchronous systems focus on the infinite-state case—for instance, the port automaton model of Panangaden and Stark [PS88] and the I/O automaton model of Lynch and Tuttle [LT87]. Also, earlier work has looked at issues far removed from those which are traditionally considered in the study of finite-state systems.

Recently, Abdulla and Jonsson have studied decision problems for distributed systems with asynchronous communication [AJ93]. However, they model channels as unbounded, fifo buffers, a framework in which most interesting questions become undecidable. The results of [AJ93] show that the fifo model becomes tractable if messages may be lost in transit: questions such as reachability of configurations become decidable. While their results are, in general, incomparable to ours, we remark that their positive results hold for our model as well.

Petri net languages Our model is closely related to Petri nets [G78,J86]. We can go back and forth between labelled Petri nets and message-passing networks while maintaining a bijection between the firing sequences of a net N and the computations of the corresponding automaton $\mathcal{A}$.

There are several ways to associate a language with a Petri net [H75,J86,Pet81]. The first is to examine all firing sequences of the net. The second is to look at firing sequences which lead to a set of *final markings*. The third is to identify firing sequences which reach markings which *dominate* some final marking. The third class corresponds to message-passing recognisable languages.

A number of positive results have been established for the first class of languages—for instance, regularity is decidable [GY80,VV80]. On the other hand, a number of negative results have been established for the second class of languages—for instance, it is undecidable whether such a language contains *all* strings [VV80]. However, none of these results, positive or negative, carry over to the third class—ours is one of the few tangible results for this class of Petri net languages.

Directions for future work We believe that Theorem 5.6 holds even without the determinacy requirement on L^R. An interesting question is to develop a method for transforming sequential specifications in terms of message-passing automata into equivalent distributed specifications in terms of the message-passing network model of [MNRS98]. Another challenging question is the decidability of regularity for message-passing recognisable languages.

References

[AJ93] P.A. Abdulla and B. Jonsson: Verifying programs with unreliable channels, in *Proc. 8th IEEE Symp. Logic in Computer Science*, Montreal, Canada (1993).

[GY80] A. Ginzburg and M. Yoeli: Vector Addition Systems and Regular Languages, *J. Comput. System. Sci.* **20** (1980) 277–284

[G78] S.A. Greibach: Remarks on Blind and Partially Blind One-Way Multicounter Machines, *Theoret. Comput. Sci* **7** (1978) 311–324.

[H75] M. Hack: Petri Net Languages, *C.S.G. Memo 124*, Project MAC, MIT (1975).

[H91] G.J. Holzmann: *Design and validation of computer protocols*, Prentice Hall (1991).

[J86] M. Jantzen: Language Theory of Petri Nets, in W. Brauer, W. Reisig, G. Rozenberg (eds.), *Petri Nets: Central Models and Their Properties, Advances in Petri Nets, 1986, Vol 1*, Springer LNCS **254** (1986) 397–412.

[KM69] R.M. Karp and R.E. Miller: Parallel Program Schemata, *J. Comput. System Sci.*, **3** (4) (1969) 167–195.

[LT87] N.A. Lynch and M. Tuttle: Hierarchical Correctness Proofs for Distributed Algorithms, *Technical Report MIT/LCS/TR-387*, Laboratory for Computer Science, MIT (1987).

[M78] A. Mazurkiewicz: Concurrent Program Schemes and their Interpretations, *Report DAIMI-PB-78*, Computer Science Department, Aarhus University, Denmark (1978).

[MNRS97] M. Mukund, K. Narayan Kumar, J. Radhakrishnan and M. Sohoni: Message-Passing Automata and Asynchronous Communication, *Report TCS-97-4*, SPIC Mathematical Institute, Madras, India (1997).

[MNRS98] M. Mukund, K. Narayan Kumar, J. Radhakrishnan and M. Sohoni: Robust Asynchronous Protocols are Finite-State, *Proc. ICALP 98*, Springer LNCS (1998) *(to appear)*.

[PS88] P. Panangaden and E.W. Stark: Computations, Residuals, and the Power of Indeterminacy, in T. Lepisto and A. Salomaa (eds.), *Proc. ICALP '88*, Springer LNCS **317** (1988) 439–454.

[Pel87] E. Pelz: Closure Properties of Deterministic Petri Nets, *Proc. STACS 87*, Springer LNCS **247**, (1987) 371-382.

[Pet81] J.L. Peterson: *Petri net theory and the modelling of systems*, Prentice Hall (1981).

[VV80] R. Valk and G. Vidal-Naquet: Petri Nets and Regular Languages, *J. Comput. System. Sci.* **20** (1980) 299–325.

[V82] G. Vidal-Naquet: Deterministic languages for Petri nets, *Application and Theory of Petri Nets*, Informatik-Fachberichte **52**, Springer-Verlag (1982).

[Z87] W. Zielonka: Notes on Finite Asynchronous Automata, *R.A.I.R.O.—Inf. Théor. et Appl.*, **21** (1987) 99–135.