Internal Report TCS-97-4
September, 1997

Message-Passing Automata and Asynchronous Communication
(Preliminary Version)

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

Abstract

This paper is a step towards developing a new automata-theoretic framework for describing distributed finite-state systems with asynchronous communication. If we assume that messages can be delayed arbitrarily in transit, it is reasonable to model the global behaviour of such systems in terms of finite-state automata equipped with blind counters—that is, counters which cannot be tested for zero.

We analyse the languages accepted by such automata and show that it is decidable whether the language of such an automaton is empty. We also develop a variety of pumping lemmas which can be used to show that certain languages are not accepted by these automata.

Our main result is that the subclass of languages accepted by these automata which is closed under complementation is precisely the class of regular languages. In the context of asynchronous protocols, our result implies that robust finite-state protocols use bounded buffers. In other words, messages are used *only* for handshaking—that is, for coordinating the interaction between different processes and the environment.

It is well known that automata with blind counters are closely related to Petri nets. However, our definition of languages is more appropriate for reasoning about asynchronous communication and is *different* from the definition used in the theory of Petri nets.

[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]Dept of Computer Sc. and Engg., Indian Institute of Technology, Bombay 400 076, India. E-mail: sohoni@cse.iitb.ernet.in

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

This paper is a step towards developing an automata-theoretic framework for describing distributed finite-state systems with asynchronous communication. Earlier attempts at defining such models of asynchronous systems deal primarily with infinite-state systems—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 focussed on issues far removed from those which are traditionally considered in the study of finite-state systems.

The setting for our work is as follows. Consider a system in which a collection of finite-state machines communicate by sending messages via buffered channels. Suppose that there are only finitely many different kinds of messages. Messages may experience arbitrary delays in transit, though they always eventually reach their recipient. At an abstract level, Internet protocols such as the SMTP mail protocol fit into this paradigm. Another example is the protocol used in a banking network to exchange information between ATMs and the bank's distributed databases.

Since messages may get reordered in transit, the state of such a system is completely described by the state of the components and the number of messages of each kind which have been sent but are as yet undelivered. Thus, at a global level, such systems can be treated as finite-state automata equipped with a finite number of counters, one for each type of message. The only operations permitted on the counters are increment and decrement (corresponding to sending and receiving the appropriate type of message respectively). The automaton cannot test if a counter's value is zero—this restriction captures the intuition that it is not practical for a component to make a decision based on the assumption that another process has *not* sent a message, since messages may be delayed arbitrarily.

With this motivation, we define *counter automata* and study the languages they accept. Each move of a counter automaton consists of either reading a letter from the input or manipulating a counter. Reading from the input represents the interaction of the underlying distributed system with its environment. We study the languages (over the input alphabet) accepted by these automata.

Our main concern is when such a language is regular. An automaton which accepts a regular language represents a communication protocol whose interaction with the environment is regular. Such a protocol essentially uses only bounded buffers. Our main result is that a language L accepted by a counter automaton is regular if and only if the complement of L is also accepted by a counter automaton. In the context of asynchronous protocols, our result implies that robust finite-state protocols use only bounded buffers. In other words, messages are used *only* for hand-shaking, to coordinate the interaction between different processes and the environment. Along the way, we develop a variety of tools and techniques for reasoning about counter automata, including a number of pumping lemmas which are useful for showing when languages are *not* recognisable by counter

automata.

The automata we consider are closely related to automata with blind counters, studied by Greibach [G78]. In turn, these automata are closely related to Petri nets [J86a, J86b]. Some of the techniques we develop are analogous to well-known results in Petri net theory, such as the covering tree construction of Karp and Miller [KM69]. However, our definition of languages is more appropriate for reasoning about asynchronous communication and is *different* from the definition used in Petri net theory. Towards the end of the paper, we discuss the connection between our framework and Petri net languages.

Recently, Abdulla and Jonsson have also studied decision problems for distributed systems with asynchronous communication [AJ93, AJ94]. However, they work in a setting where messages are delivered in the order in which they are sent. This means that the channels are unbounded, fifo buffers. With such a strong model, most interesting questions become undecidable. The results of [AJ93] show that the fifo model can be made tractable by assuming that messages may be lost in transit. With lossy channels, questions such as reachability of configurations and equivalence with respect to finite-state automata can be decided, though certain other questions remain undecidable [AJ94]. While their results are incomparable with ours, since the two models are orthogonal, we remark that all their positive results hold for our model as well.

The paper is organised as follows. In the next section we define counter automata and prove some basic results about them. In Section 3 we prove a Contraction Lemma which leads to decidability of the emptiness problem and the fact that the languages accepted by counter automata are not closed under complementation. Section 4 develops a family of pumping lemmas which are exploited in Section 5 to prove our main result concerning the regularity of languages accepted by counter 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.

2 Counter Automata

Natural numbers and tuples As usual, $\mathbb{N}$ denotes the set $\{0, 1, 2, \ldots\}$ of natural numbers. If $i, j \in \mathbb{N}$, $[i..j]$ denotes the set $\{i, i+1, \ldots, j\}$ with the convention that $[i..j] = \emptyset$ 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 $\overline{n} = \langle n_1, n_2, \ldots, n_k \rangle$ be k-tuples of natural numbers. Then $\overline{m} \leq \overline{n}$ iff $m_i \leq n_i$ for each $i \in [1..k]$.

Counter automata A *counter 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 Γ^+ for the set $\{C^+ | C \in \Gamma\}$, Γ^- for $\{C^- | C \in \Gamma\}$ and $\Gamma^\pm$ for $\Gamma^+ \cup \Gamma^-$.
- $T \subseteq Q \times (\Sigma \cup \Gamma^\pm) \times Q$ 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 $\langle 0, 0, \ldots, 0\rangle$, representing the function which assigns 0 to all counters, is uniformly denoted $\overline{0}$, for all values of k.

The function f *dominates* the function f', written $f \geq f'$, if $f(C) \geq f'(C)$ for every counter C. The function f *strictly dominates* the function f', written $f > f'$, if $f \geq f'$ and there is a counter C such that $f(C) > f'(C)$.

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 The 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 $\chi \xrightarrow{(q,d,q')} \chi'$—in other words, transitions are labelled by elements of T rather than elements of $\Sigma \cup \Gamma^{\pm}$. 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 of letters $d_1 d_2 \ldots d_n$ from $\Sigma \cup \Gamma^{\pm}$ is denoted $\alpha(t_1 t_2 \ldots t_n)$.

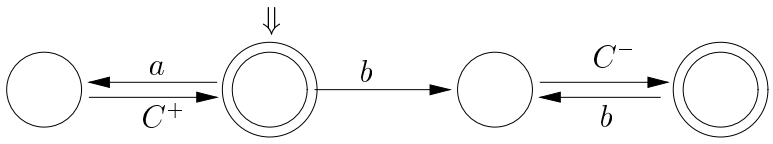
Computations, runs and languages A *computation* of $\mathcal{A}$ is a sequence $\chi_0 \xrightarrow{t_1} \chi_1 \xrightarrow{t_2} \ldots \xrightarrow{t_n} \chi_n$. We also write $\chi_0 \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} \chi_n$ to indicate that there is a computation labelled $t_1 t_2 \ldots t_n$ 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 *counter recognisable* if there is a counter 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 counter 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.*



2.1 Non-determinism versus determinism

Deterministic Counter Automata A counter 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, it is easy to see that any relaxation of the definition will allow deterministic automata to simulate non-deterministic automata in a trivial manner.

For instance, suppose we naïvely define a deterministic automaton to be one in which no state has two outgoing transitions with the same label. This definition would permit a deterministic automaton to choose between a counter move and another transition (which may or may not be a counter move). We can then simulate a choice between two transitions $t_1 = (q, d, q_1)$ and $t_2 = (q, d, q_2)$ with the same label by adding a dummy counter C. Instead of choosing directly between t_1 and t_2, the new automaton will first choose between t_1 and a move (q, C^+, q') leading to a new state q'. We can then simulate t_2 by adding a transition (q', d, q_2). Thus, the original choice between t_1 and t_2 is replaced by a cascaded choice involving the dummy counter C.

It is interesting to observe that a similar strong definition of determinism is used in the study of Petri net languages [J86a].

We have the following characterisation of languages accepted by deterministic counter automata.

Proposition 2.2 *Let $\mathcal{A}$ be a deterministic counter automaton. Then, either $L(\mathcal{A})$ is regular or there exists a word $w \notin L(\mathcal{A})$ such that every extension of w also does not belong to $L(\mathcal{A})$.*

Proof: Let $\mathcal{A}$ be a deterministic counter automaton. For each input word w, either $\mathcal{A}$ admits no run over w or it admits a unique sequence of runs $\rho_1, \rho_2, \ldots$, (which may be infinite) over w such that for each $i \geq 1$, ρ_{i+1} extends ρ_i by one transition involving a counter operation.

A word w is said to be *blocked* in the automaton $\mathcal{A}$ if $\mathcal{A}$ does not permit an infinite sequence of runs $\rho_1, \rho_2, \ldots$, over w. If w is blocked, there exists a unique state where $\mathcal{A}$ "gets stuck" when processing w. We denote this state q_w.

Since $\mathcal{A}$ is deterministic, we know that if q_w has any outgoing transitions, either the set of outgoing transitions at q_w is labelled by distinct letters from Σ or there is only a single outgoing transition labelled by an element of $\Gamma^{\pm}$. In the latter case, it must be that the transition is labelled C^-, for some $C \in \Gamma$, because a move labelled C^+ is always enabled. We say that w is Γ-*blocked* in $\mathcal{A}$ if w is blocked in $\mathcal{A}$ and q_w has an outgoing transition labelled C^-, for some $C \in \Gamma$.

Returning to the statement to be proved, if $L(\mathcal{A}) = \Sigma^*$, then $L(\mathcal{A})$ is regular. Thus, the interesting case is when $\overline{L(\mathcal{A})} = \Sigma^* \setminus L(\mathcal{A})$ is non-empty.

Case 1: If there exists w in $\Sigma^* \setminus L(\mathcal{A})$ which is Γ-blocked, then any extension of w must also be Γ-blocked. Thus all extensions of w also lie outside $L(\mathcal{A})$.

Case 2: Suppose that no word w in $\overline{L(\mathcal{A})}$ is Γ-blocked. Then, from $\mathcal{A}$ we can construct a finite-state automaton $\mathcal{A}'$ over the alphabet Σ which has ε-transitions. The automaton $\mathcal{A}'$ has the same set of states, initial state and final states as $\mathcal{A}$. For each transition t of the form (q, d, q') in $\mathcal{A}$, we have a corresponding transition $t' = (q, d', q')$ in $\mathcal{A}'$, where $d' = d$ if $d \in \Sigma$ and $d' = \varepsilon$ if $d \in \Gamma^{\pm}$.

Since $\mathcal{A}$ is deterministic, at each state of $\mathcal{A}'$ which has outgoing transitions, either the set of outgoing transitions is labelled by distinct letters from Σ or there is a single outgoing transition labelled ε. In other words, for every word w, either $\mathcal{A}'$ does not admit a run over w or $\mathcal{A}'$ admits a unique sequence of runs $\rho_1, \rho_2, \ldots$, over w such that for each $i \geq 1$, ρ_{i+1} extends ρ_i by a transition labelled ε.

We claim that $L(\mathcal{A}') = L(\mathcal{A})$ and hence $L(\mathcal{A})$ is regular. It is easy to see that each computation $\chi_0 \xrightarrow{t_1} \chi_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} \chi_n$ of $\mathcal{A}$ can be simulated by a run $Q(\chi_0) \xrightarrow{t'_1} Q(\chi_1) \xrightarrow{t'_2} \cdots \xrightarrow{t'_n} Q(\chi_n)$ of $\mathcal{A}'$, where for each $i \in [1..n]$, t'_i is the transition corresponding to t_i as described above. Since the initial and final states of $\mathcal{A}'$ are the same as those of $\mathcal{A}$, it follows that $L(\mathcal{A}) \subseteq L(\mathcal{A}')$.

To see that $L(\mathcal{A}') \subseteq L(\mathcal{A})$, assume that there is a word $w \in L(\mathcal{A}') \setminus L(\mathcal{A})$. Then, $\mathcal{A}'$ admits an accepting run $\rho' : q_{\text{in}} \xrightarrow{t'_1} q_1 \xrightarrow{t'_2} \cdots \xrightarrow{t'_n} q_n$ over w, with q_n a final state. From our construction of $\mathcal{A}'$, it follows that there is a maximal prefix $\sigma' : q_{\text{in}} \xrightarrow{t'_1} q_1 \xrightarrow{t'_2} \cdots \xrightarrow{t'_k} q_k$ of ρ', with $k < n$, such that $\mathcal{A}$ admits a 0-run $\sigma : \chi_0 \xrightarrow{t_1} \chi_1 \xrightarrow{t_2} \cdots \xrightarrow{t_k} \chi_k$ over w with the following properties:

- $Q(\chi_0) = q_{\text{in}}$.
- For each $i \in [1..k]$, $Q(\chi_i) = q_i$ and t'_i is the transition corresponding to t_i as specified in the construction of $\mathcal{A}'$.
- w is blocked in $\mathcal{A}$ and $q_w = q_k$.

Since σ cannot be extended in $\mathcal{A}$ while σ' can be extended in $\mathcal{A}'$, it must be the case that t_{k+1} corresponds to a move of the form C^- for a counter C whose value at χ_k is 0. This implies that w is Γ-blocked in $\mathcal{A}$, which is a contradiction.

Hence, $L(\mathcal{A}') = L(\mathcal{A})$ and $L(\mathcal{A})$ is regular.

□

Corollary 2.3 *Non-deterministic counter automata are strictly more powerful than deterministic counter automata.*

Proof: Consider the language $L \subseteq \{a, b\}^*$ given by

$$L = \{w \mid w = w_1 a^m b^n a w_2, \text{ where } w_1, w_2 \in \{a, b\}^* \text{ and } m \geq n \geq 1\}.$$

It is not difficult to transform the automaton which accepts $L_{\text{ge}} = \{a^m b^n \mid m \geq n\}$ into a non-deterministic counter automaton which accepts L. We argue that L cannot be accepted by *any* deterministic counter automaton. L is clearly not regular. Thus, by the previous proposition, for L to be accepted by a deterministic automaton, it must be the

case that there is a word $w \notin L$ such that every suffix of w is also not in L. However, 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 counter 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.

2.2 Some useful results

The following observations are basic to analysing the behaviour of counter automata. We first need the following terminology: a sequence $\overline{n}_1, \overline{n}_2, \ldots$ of k-tuples of natural numbers is said to be *non-decreasing* if $\overline{n}_1 \leq \overline{n}_2 \leq \cdots$.

Proposition 2.4 *Every infinite sequence of k-tuples of natural numbers has an infinite non-decreasing subsequence.*

Proof: The proof is by induction on k.

Basis: When $k = 1$, we have a sequence of natural numbers. If the sequence is bounded then some value appears infinitely often (by the pigeon-hole principle). On the other hand, if the sequence is unbounded, it is obvious that it contains a strictly increasing infinite subsequence.

Induction step: If we project the sequence of k-tuples onto its first $k-1$ components, we can apply the induction hypothesis to extract an infinite subsequence which is nondecreasing in these $k-1$ coordinates. We look at the corresponding subsequence in our original sequence of k-tuples and examine the kth coordinate of each element in the sequence. By an argument similar to the basis case, there must be an infinite subsequence which is non-decreasing on the kth coordinate as well.

□

Corollary 2.5 *There is no infinite set of k-tuples of natural numbers that is pairwise incomparable.*

Lemma 2.6 *Let T be a finitely branching infinite tree whose nodes are labelled by k-tuples from $\mathbb{N}$. For each $i \in \mathbb{N}$ there is a number μ_i such that along any path of length μ_i starting at the root of T, the corresponding sequence of labels $n_1, n_2, \ldots, n_{\mu_i}$ has a nondecreasing subsequence of length i.*

Proof: Suppose there exists $i \in \mathbb{N}$ for which there is no such μ_i. In other words, for each $j \in \mathbb{N}$ there is a path of length j starting at the root whose labels $n_1, n_2, \ldots, n_j$ do not contain a non-decreasing subsequence of length i.

Call a node t in T *bad* if the labels along the unique path from the root to t do not have a non-decreasing subsequence of length i. Clearly the parent of a bad node is also bad. Thus the set of bad nodes forms a subtree of T. By our assumption that there is no

μ_i corresponding to i, there must be bad nodes at each level in the tree. Hence the set of bad nodes forms an infinite subtree of T.

By König's Lemma there is an infinite path in T all of whose nodes are bad. The labels along this path do not have any non-decreasing subsequence of length greater than or equal to i. This contradicts Proposition 2.4. □

Lemma 2.7 *Let $\mathcal{A}$ be a counter automaton with M states and N counters and let $K \in \mathbb{N}$. Then, there exists $\ell \in \mathbb{N}$, such that for any K-run $\chi_0 \xrightarrow{t_1} \chi_1 \ldots \xrightarrow{t_\ell} \chi_\ell$ of $\mathcal{A}$, there are two configurations χ_i and χ_j, $0 \leq i < j \leq \ell$, such that $Q(\chi_i) = Q(\chi_j)$ and $F(\chi_i) \leq F(\chi_j)$.*

Proof: Construct a tree T whose nodes are labelled by $\mathbb{N}^N$ as follows.

- The root x_0 is labelled $\langle \overline{0} \rangle$.
- For each vector $\overline{v} = \langle m, \overline{n} \rangle$ where $m \in [1..M]$ and $\overline{n}(i) \leq K$ for all $i \in [1..N]$, construct a child x_0^v of the root labelled by $\overline{n}$.
- Let x be a node labelled $\langle m, \overline{n} \rangle$. For each vector $v = \langle m', \overline{n}' \rangle$ where $m' \in [1..M]$ and $\overline{n}'$ differs from $\overline{n}$ in at most one coordinate by at most 1, construct a child x^v of x labelled v.

Clearly, T is a finitely branching tree. Hence, by Lemma 2.6, for each natural number p there is a number μ_p such that, if $x_0 x_1 \ldots x_p \xrightarrow{t_1} \chi_1 \xrightarrow{t_2} \cdots \xrightarrow{t_{\mu_p}} \chi_{\mu_p}$ is a K-run of $\mathcal{A}$ then the sequence $F(\chi_0), F(\chi_1), \ldots, F(\chi_{\mu_p})$ has a non-decreasing subsequence of length p.

Thus, if $\chi_0 \Longrightarrow \chi_{\mu_{M+1}}$ is any run of $\mathcal{A}$, then there are positions $0 \leq k_1 < k_2 < \ldots < k_{M+1} \leq \mu_{M+1}$ such that $F(\chi_{k_1}) \leq F(\chi_{k_2}) \ldots \leq F(\chi_{k_{M+1}})$. By the pigeon-hole principle, there are positions k_r and k_s, $1 \leq r < s \leq M+1$, such that $Q(\chi_{k_r}) = Q(\chi_{k_s})$. To prove the lemma, set $\ell = \mu_{M+1}$, $\chi_i = \chi_{k_r}$ and $\chi_j = \chi_{k_s}$.

□

Weak pumping constant Notice that the bound ℓ established in the preceding lemma depends only on the values M, N and K and is independent of the actual structure of the automaton. Let $\pi_{M,N,K}$ denote the bound ℓ. We refer to $\pi_{M,N,K}$ as the *weak pumping constant* for (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 counter automaton $\mathcal{A}$, there is a constant k such that if $\chi_0 \overset{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ is a computation of $\mathcal{A}$, with $m > k$, then there exist i and j, $m-k \leq i < j \leq m$, such that $\chi'_0 \overset{t_1 \ldots t_i t_{j+1} \ldots t_m}{\Longrightarrow} \chi'_{m-(j-i)}$ 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: 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 N-tuples of natural numbers as follows:

$$\begin{array}{rcl} f_m(n) & = & 0, \text{for all } n \in [1..N] \\ \text{For } i \in [0..m{-}1],\ f_i(n) & = & \left\{ \begin{array}{ll} f_{i+1}(n) & \text{if } \alpha(t_{i+1}) \notin \{C_n^+, C_n^-\} \\ f_{i+1}(n){+}1 & \text{if } \alpha(t_{i+1}) = C_n^- \\ \max(0, f_{i+1}(n){-}1) & \text{if } \alpha(t_{i+1}) = C_n^+ \end{array} \right. \end{array}$$

We next show that 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 $f \geq f_i$.

Proof of Claim: By induction on $m{-}i$.

Basis: If $i = m$ there is nothing to prove.

Induction step:

By the induction hypothesis, $(Q(\chi_{i+1}), f') \stackrel{t_{i+2} t_{i+3} \ldots t_m}{\Longrightarrow}$ iff $f' \geq f_{i+1}$.

Suppose that $f \geq f_i$. We have to show that $(Q(\chi_i), f) \stackrel{t_{i+1} t_{i+2} \ldots t_m}{\Longrightarrow}$. We first argue that there is a move $(Q(\chi_i), f) \stackrel{t_{i+1}}{\longrightarrow} (Q(\chi_{i+1}), f')$. Since we know that $(Q(\chi_i), t_{i+1}, Q(\chi_{i+1}))$ is a transition of $\mathcal{A}$, the only reason for forbidding such a move is that $\alpha(t_{i+1}) = C_n^-$ for some counter C_n and $f(n) = 0$. However, if $\alpha(t_{i+1}) = C_n^-$, we know that $f_i(n) = f_{i+1}(n) + 1 \geq 1$. Since $f \geq f_i$, $f(n) \geq 1$ as well.

Consider the function f'. We shall show that $f' \geq f_{i+1}$. From the induction hypothesis, it then follows that $(Q(\chi_i), f) \stackrel{t_{i+1}}{\longrightarrow} (Q(\chi_{i+1}), f') \stackrel{t_{i+2} t_{i+3} \ldots t_m}{\Longrightarrow}$, whereby $(Q(\chi_i), f) \stackrel{t_{i+1} t_{i+2} \ldots t_m}{\Longrightarrow}$.

To check that $f' \geq f_{i+1}$, we consider all possible values for $\alpha(t_{i+1})$.

(i) $\alpha(t_{i+1}) \in \Sigma$: Then $f' = f \geq f_i = f_{i+1}$.

(ii) $\alpha(t_{i+1}) \in \{C_n^-, C_n^+\}$: Then

- For $\ell \neq n$, $f'(\ell) = f(\ell) \geq f_i(\ell) = f_{i+1}(\ell)$.
- If $\alpha(t_{i+1}) = C_n^-$, then $f'(n) = f(n) - 1 \geq f_i(n) - 1 = f_{i+1}(n)$.
- If $\alpha(t_{i+1}) = C_n^+$, then $f'(n) = f(n){+}1 \geq f_i(n){+}1 = \max(1, f_{i+1}(n)) \geq f_{i+1}(n)$.

Thus, for each $\ell \in [1..n]$, $f'(\ell) \geq f_{i+1}(\ell)$.

Conversely, suppose that $(Q(\chi_i), f) \stackrel{t_{i+1} t_{i+2} \ldots t_m}{\Longrightarrow}$. We have to establish that $f \geq f_i$. We know that $(Q(\chi_i), f) \stackrel{t_{i+1}}{\longrightarrow} (Q(\chi_{i+1}), f') \stackrel{t_{i+2} t_{i+3} \ldots t_m}{\Longrightarrow}$ and, by the induction hypothesis, $f' \geq f_{i+1}$. As before, we examine all possible values of $\alpha(t_{i+1})$.

(i) $\alpha(t_{i+1}) \in \Sigma$: Then $f = f' \geq f_{i+1} = f_i$.

(ii) $\alpha(t_{i+1}) \in \{C_n^-, C_n^+\}$: Then

- For $\ell \neq n$, $f(\ell) = f'(\ell) \geq f_{i+1}(\ell) = f_i(\ell)$.
- If $\alpha(t_{i+1}) = C_n^-$, then $f(n) = f'(n) + 1 \geq f_{i+1}(n) + 1 = f_i(n)$.
- If $\alpha(t_{i+1}) = C_n^+$, then $f(n) = f'(n) - 1 \geq f_{i+1}(n) - 1$. Since $f(n) \geq 0$ and $f_i(n) = \max(0, f_{i+1}(n) - 1)$, $f(n) \geq f_i(n)$.

Thus, for each $\ell \in [1..n]$, $f(\ell) \geq f_i(\ell)$.

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

Consider the sequence $f_m, f_{m-1}, \ldots f_0$. Since its length exceeds $\pi_{M,N,0}$, by Lemma 2.7 there exist positions i and j, $m \geq j > i \geq m - \pi_{M,N,0}$ such that $f_j \leq f_i$ and $Q(\chi_j) = Q(\chi_i)$. By the Corollary to Claim, for each counter C_n, $C_n(\chi_i) \geq f_i(n) \geq f_j(n)$. 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 counter automaton $\mathcal{A}$ with M states and N counters has an accepting computation iff it has an accepting computation whose length is bounded by $\pi_{M,N,0}$.*

In the Appendix, we give a constructive proof of Lemma 2.7 which provides 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.

Corollary 3.3 *The emptiness problem for counter automata is decidable.*

We remark, however, that a result of Lipton [L76] from the theory of Petri net languages implies that the emptiness problem for counter recognisable languages is EXPSPACE-hard.

Corollary 3.4 *Counter recognisable languages are not closed under complementation.*

Proof: We saw earlier that $L_{\text{ge}} = \{a^m b^n \mid m \geq n\}$ is counter recognisable. Let $\mathcal{A}$ be an automaton which accepts L_{ge}. We can easily extend $\mathcal{A}$ to accept $L'_{\text{ge}} = L_{\text{ge}} \cup \{w \mid w \text{ is not of the form } a^m b^n\}$. The complement of the language L'_{ge} is the language $L_{\text{lt}} = \{a^m b^n \mid m < n\}$.

Suppose that L_{lt} were counter recognisable. Let $\mathcal{A}_{\text{lt}}$ be an automaton which accepts L_{lt}. Let M be the number of states in $\mathcal{A}_{\text{lt}}$ and N the number of counters used by $\mathcal{A}_{\text{lt}}$. 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 $\mathcal{A}_{\text{lt}}$ on w. By applying the Contraction Lemma (repeatedly, if necessary) to ρ, we can obtain an accepting run ρ' of $\mathcal{A}_{\text{lt}}$ over a word of the form $a^J b^K$, where $K \leq J$, thus

contradicting the assumption that $L(\mathcal{A}_{\mathrm{lt}}) = L_{\mathrm{lt}}$. (The reason we may need to use the Contraction Lemma more than once to obtain a suitable ρ' is that when we apply the Lemma once, the sequence of moves deleted may fail to contain any transition labelled b. However, if this happens, the resulting run will continue to have a suffix containing $\pi_{M,N,0} + 1$ moves labelled b, so we can apply the Contraction Lemma repeatedly until at least one transition labelled b is deleted.) □

4 A Collection of Pumping Lemmas

Change vectors For a string w over a set X and a symbol $x \in X$, $\#_x(w)$ denotes 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 counter 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 $|\Delta_C(u)| \leq |u|$, it follows that $C(\chi') \geq (n-1)|u| + v$. From the induction hypothesis, we can then conclude that $\chi' \stackrel{u^{n-1}v}{\Longrightarrow}$.

□

Pumpable decomposition Let $\mathcal{A}$ be a counter 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) $n \leq N$.

(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. If C is a counter such that $\Delta_C(v_i) > 0$ for some pumpable block v_i, we say that C is a *pumpable counter*.

Proposition 4.2 *Let $\mathcal{A}$ be a counter automaton and $\rho : \chi_0 \overset{t_1 t_2 \ldots t_m}{\Longrightarrow} \chi_m$ be a computation of $\mathcal{A}$. Consider a pumpable decomposition $\chi_0 \overset{u_1}{\Longrightarrow} \chi_{i_1} \overset{v_1}{\Longrightarrow} \chi_{j_1} \overset{u_2}{\Longrightarrow} \cdots \overset{u_n}{\Longrightarrow} \chi_{i_n} \overset{v_n}{\Longrightarrow} \chi_{j_n} \overset{u_{n+1}}{\Longrightarrow} \chi_m$ of ρ. Then, for $r \in [1..n]$, $\chi_0 \overset{u_1}{\Longrightarrow} \chi_{i_1} \overset{v_1}{\Longrightarrow} \chi_{j_1} \overset{u_2}{\Longrightarrow} \cdots \overset{u_r}{\Longrightarrow} \chi_{i_r} \overset{v_r}{\Longrightarrow} \chi_{j_r} \overset{u_{r+1}}{\Longrightarrow} \chi_{i_{r+1}}$ is a pumpable decomposition of $\rho_r : \chi_0 \overset{u_1 v_1 \ldots u_r v_r u_{r+1}}{\Longrightarrow} \chi_{i_{r+1}}$.*

Proof: Immediate, from the definition of pumpable decompositions. □

Lemma 4.3 (Counter Pumping) *Let $\mathcal{A}$ be an 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 $I \geq 1$, 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 $i \in [1..n]$, $\ell_i \geq I$.*

(iv) *For every counter C, $C(\chi'_p) \geq C(\chi_m)$.*

(v) *Let Γ_{pos} be the set of pumpable counters in the pumpable decomposition of ρ. For each counter $C \in \Gamma_{\text{pos}}$, $C(\chi'_p) \geq J$.*

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. □

The preceding lemma shows that all the pumpable counters in a pumpable decomposition are simultaneously unbounded. This is analogous to a well-known result of Karp and Miller in the theory of vector addition systems [KM69]. They show how to associate a finite object called a covering tree with each vector addition system. The covering tree

can be used to decide whether a set of coordinates of the vector addition system is simultaneously unbounded. See Section 6 for a more detailed discussion of the connection of our work to vector addition systems.

Corollary 4.4 *Let $\mathcal{A}$ be an automaton and ρ a K-run of $\mathcal{A}$, $K \in \mathbb{N}$, with a pumpable decomposition of the form*

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

(i) For any $I \in \mathbb{N}$, with $I \geq 1$, there exist $\ell_1, \ell_2, \ldots, \ell_{n-1} \in \mathbb{N}$ and a K-run ρ' of $\mathcal{A}$ of the form

$$\chi'_0 \stackrel{u_1}{\Longrightarrow} \chi'_{i'_1} \stackrel{v_1^{\ell_1}}{\Longrightarrow} \chi'_{j'_1} \stackrel{u_2}{\Longrightarrow} \chi'_{i'_2} \stackrel{v_2^{\ell_2}}{\Longrightarrow} \chi'_{j'_2} \cdots \stackrel{u_{n-1}}{\Longrightarrow} \chi'_{i'_{n-1}} \stackrel{v_{n-1}^{\ell_{n-1}}}{\Longrightarrow} \chi'_{j'_{n-1}} \stackrel{u_n}{\Longrightarrow} \chi'_{i'_n} \stackrel{v_n^I}{\Longrightarrow} \chi'_{j'_n} \stackrel{u_{n+1}}{\Longrightarrow} \chi'_p$$

such that $\chi_0 = \chi'_0$, $Q(\chi'_p) = Q(\chi_m)$ and $F(\chi'_p) \geq F(\chi_m)$.

(ii) For any $I \in \mathbb{N}$, with $I \geq 1$ and any $k \in [1..n]$, there exist $\ell_1, \ell_2, \ldots, \ell_{k-1} \in \mathbb{N}$ and a K-run ρ' of $\mathcal{A}$ of the form

$$\chi'_0 \stackrel{u_1}{\Longrightarrow} \chi'_{i'_1} \stackrel{v_1^{\ell_1}}{\Longrightarrow} \chi'_{j'_1} \stackrel{u_2}{\Longrightarrow} \cdots \stackrel{u_{k-1}}{\Longrightarrow} \chi'_{i'_{k-1}} \stackrel{v_{k-1}^{\ell_{k-1}}}{\Longrightarrow} \chi'_{j'_{k-1}} \stackrel{u_k}{\Longrightarrow} \chi'_{i'_k} \stackrel{v_k^I}{\Longrightarrow} \chi'_{j'_k} \stackrel{u_{k+1}v_{k+1}\ldots u_n v_n u_{n+1}}{\Longrightarrow} \chi'_p$$

such that $\chi_0 = \chi'_0$, $Q(\chi'_p) = Q(\chi_m)$ and $F(\chi'_p) \geq F(\chi_m)$.

Proof: The first statement follows by setting $J = 0$ when defining ℓ_n in the proof of the Counter Pumping Lemma. The second result is then immediate. We omit the details. □

We have shown that all counters which increase within the pumpable blocks of a pumpable decomposition can be simultaneously raised to arbitrarily high values. We next 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)):

$$\begin{array}{rrcl} \forall M, K \in \mathbb{N}. & \Pi_{M,0,K} & = & 1 \\ \forall M, N, K \in \mathbb{N}. & \Pi_{M,N+1,K} & = & \Pi_{M,N,\pi_{M,N+1,K}+K} + \pi_{M,N+1,K} + K \end{array}$$

Lemma 4.5 (Decomposition) *Let $\mathcal{A}$ be an 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 4.6 *Let $\mathcal{A}$ be a counter 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. □

We now return to the proof of the Decomposition Lemma.

Proof: (of Lemma 4.5) 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 $\Gamma_{\text{gt}} = \emptyset$, we set $n = 0$ and $u_1 = \rho$.

Otherwise, by Proposition 4.6, we can find positions r and s in ρ such that $\chi_0 \stackrel{u'}{\Longrightarrow} \chi_r \stackrel{v'}{\Longrightarrow} \chi_s \Longrightarrow \chi_m$, 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 4.6, we know that ρ' is in fact a $(\pi_{M,N,K}+K)$-run of $\mathcal{A}'$. Further, for every counter C in $\Gamma_{\text{gt}} \setminus \{C'\}$, 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 4.6 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 $C \in \Gamma_{\text{gt}} \setminus \{C'\}$, $C(\chi_{i'_k}) = C(\chi'_{i'_k})$ and $C(\chi_{j'_k}) = C(\chi'_{j'_k})$.

We prefix the computation $\chi_s \overset{u'_1 v'_1 \ldots u'_{p+1}}{\Longrightarrow} \chi_m$ 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 we stated earlier 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.

Lemma 4.7 (Visible Pumping) *Let L be a counter 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 $i \geq 1$.*

Proof: Let $\mathcal{A} = (Q, \Sigma, \Gamma, T, q_{\text{in}}, F)$ be a counter automaton which accepts L. From $\mathcal{A}$, we construct a new automaton $\mathcal{A}'$ by adding a new counter C_{vis} which is incremented each time an input letter is read.

Formally, $\mathcal{A}' = (Q', \Sigma, \Gamma', T', q_{\text{in}}, F)$ where:

- $Q' = Q \cup \{q_t \mid t = (q, d, q') \in T \text{ and } d \in \Sigma\}$.
- $\Gamma' = \Gamma \cup \{C_{\text{vis}}\}$.
- $T' = \{(q, d, q') \in T \mid d \notin \Sigma\} \cup \{(q, C^+_{\text{vis}}, q_t), (q_t, d, q') \mid t = (q, d, q') \in T, d \in \Sigma\}$.

It is clear that $L(\mathcal{A}') = L(\mathcal{A}) = L$. Let $M = |Q'|$ and $N = |\Gamma'|$. Set $n = \Pi_{M,N,0}$.

Let $\chi_0 \stackrel{x}{\Longrightarrow} \chi_f \stackrel{y}{\Longrightarrow} \chi_g$ be an accepting run of $\mathcal{A}'$ on w, where $|w| \geq \Pi_{M,N,0}$ and $|\alpha(x)\restriction_\Sigma| = \Pi_{M,N,0}$. Let $\chi_0 \stackrel{u_1v_1\ldots u_nv_nu_{m+1}}{\Longrightarrow} \chi_f$ be the pumpable decomposition of $\chi_0 \stackrel{x}{\Longrightarrow} \chi_f$ given by the Decomposition Lemma.

Since C_{vis} attains the value $\Pi_{M,N,0}$ along $\chi_0 \stackrel{x}{\Longrightarrow} \chi_f$, there is a pumpable block v_i, $i \in [1..m]$, such that $\Delta_{C_{\text{vis}}}(v_i) > 0$. Choose the first such block. Then $\Delta_{C_{\text{vis}}}(v_j) = 0$ for all $j < i$.

Each pumpable block defines a cycle in $\mathcal{A}'$. However, the structure of $\mathcal{A}'$ ensures that a cycle has a move labelled C^+_{vis} iff it also has a move labelled by an input letter. Thus, v_i contains at least one move with a label from Σ, while $\alpha(v_j)\restriction_\Sigma$ is empty for each $j < i$.

Let $w_1 = \alpha(u_1v_1u_2\ldots u_i)$, $w_2 = \alpha(v_i)$ and $w_3 = \alpha(u_{i+1}v_{i+1}\ldots v_nu_{n+1}y)$.

By Corollary 4.4 (ii), for each $I \in \mathbb{N}$, there is a run $\chi_0 \stackrel{u_1v_1^{\ell_1}\ldots u_iv_i^Iu_{i+1}v_{i+1}\ldots u_mv_mu_{m+1}}{\Longrightarrow} \chi'_f$ with $Q(\chi'_f) = Q(\chi_f)$ and $F(\chi'_f) \geq F(\chi_f)$.

This means that $\rho : \chi_0 \overset{u_1 v_1^{\ell_1} \ldots u_i v_i^I u_{i+1} v_{i+1} \ldots u_m v_m u_{m+1}}{\Longrightarrow} \chi'_f \overset{y}{\Longrightarrow} \chi'_g$ is an accepting run on $\alpha(u_1 v_1^{\ell_1} \ldots u_i v_i^I u_{i+1} \ldots u_m v_m u_{m+1} z) \restriction_\Sigma$. Since $\alpha(v_j) = \varepsilon$ for all $j < i$, it follows that $\alpha(u_1 v_1^{\ell_1} \ldots u_{i-1} v_{i-1}^{\ell_{i-1}} u_i) = \alpha(u_1 v_1 \ldots u_{i-1} v_{i-1} u_i) = w_1$. Thus, ρ is an accepting run over $w_1 w_2^I w_3$, as claimed by the lemma.

□

Example 4.8 *The language $L = \{a^p \mid p \text{ is prime}\}$ is not counter recognisable.*

Proof: Suppose L is counter recognisable. Let p be a prime larger that n, the pumping constant for L specified by Lemma 4.7. Then, we can write p as $x + y + z$ such that $a^{x+my+z} \in L$ for all $m \geq 1$. Choose $m = p + 1$. Then $a^{x+(p+1)y+z} \in L$, though $x + (p+1)y + z = x + y + z + py = (1 + y)p$ is not a prime! □

One difference between the preceding lemma and the traditional pumping lemma for regular languages is that in the context of counter recognisable languages, for a pumpable string uvw, we must have at least one iteration of the pumpable segment v to ensure that the resulting string uv^iw is in the language, whereas for regular languages, $uv^0w = uw$ is also guaranteed to be in the language.

Lemma 4.9 (Counter Hierarchy) *For $k \in \mathbb{N}$, let $\mathcal{L}_k$ be the set of languages recognisable by counter automata with k counters. Then, for all k, $\mathcal{L}_k \subsetneq \mathcal{L}_{k+1}$.*

Proof: Define $L_{k+1} = \{a_0^{n_0} a_1^{n_1} \cdots a_k^{n_k} a_{k+1}^{n_{k+1}} \mid n_0 \geq n_1 \geq \cdots \geq n_{k+1}\}$. It is not difficult to construct a counter automaton with k+1 counters which accepts L_{k+1}. However, there is no k-counter machine which accepts this language.

Suppose $\mathcal{A}$ is a k-counter machine which accepts L_{k+1}. As in the proof of Lemma 4.7, we extend $\mathcal{A}$ with a new counter C_{k+1} which is incremented precisely when a_{k+1} is read from the input. Let the new machine $\mathcal{A}'$ have M states.

Consider an accepting run ρ of $\mathcal{A}'$ on a string $w = a_0^{n_0} a_1^{n_1} \ldots a_{k+1}^{n_{k+1}}$ with $n_{k+1} \geq \Pi_{M,k+1,0}$. By Lemma 4.5, the run ρ has a pumpable decomposition $\chi_0 \overset{u_1 v_1 \ldots u_m v_m u_{m+1}}{\Longrightarrow} \chi_f$ such that $m \leq k$+1 and $\Delta_{C_{k+1}}(v_i) > 0$ for some $i \in [1..m]$.

By Lemma 4.3, for each $I \geq 1$, there exist $\ell_1, \ell_2, \ldots, \ell_m \in \mathbb{N}$ such that each $\ell_j \geq I$ and $\chi_0 \overset{u_1 v_1^{\ell_1} \ldots u_m v_m^{\ell_m} u_{m+1}}{\Longrightarrow} \chi'_f$ is an accepting run. From the structure of words in L_{k+1}, it follows that for each pumpable block v_i, $\alpha(v_i)$ contains at most one of the visible letters $\{a_0, a_1, \ldots, a_{k+1}\}$. Since $m \leq k$+1 at least one letter from $\{a_0, a_1, \ldots, a_{k+1}\}$ does not appear in $\bigcup_{i \in [1..m]} \alpha(v_i)$. Also, since $\Delta_{C_{k+1}}(v_i) > 0$ for some $i \in [1..m]$, from the structure of $\mathcal{A}'$ it follows that a_{k+1} *does* appear in $\bigcup_{i \in [1..m]} \alpha(v_i)$.

Let a_r be a letter which does not appear in $\bigcup_{i \in [1..m]} \alpha(v_i)$ and let v_j be the block such that a_{k+1} appears in $\alpha(v_j)$. By Corollary 4.4 (ii), for all $I \geq 1$, there exist $\ell_1, \ell_2, \ldots, \ell_{j-1}$ with each $\ell_j \geq I$ such that $\chi_0 \overset{u_1 v_1^{\ell_1} \ldots v_{j-1}^{\ell_{j-1}} u_j v_j^I u_{j+1} v_{j+1} \ldots u_m v_m^{\ell_m} u_{m+1}}{\Longrightarrow} \chi'_g$ is an accepting run of $\mathcal{A}'$. Choose $I = n_r + 1$.

Thus $w' = \alpha(u_1 v_1^{\ell_1} \ldots v_{j-1}^{\ell_{j-1}} u_j v_j^{n_r+1} u_{j+1} v_{j+1} \ldots u_m v_m^{\ell_m} u_{m+1})) \restriction_\Sigma \in L(\mathcal{A}')$. But, w' has only n_r a_r's and at least n_r+1 a_{k+1}'s, which violates the definition of L_{k+1}.

Thus, there is *no* k-counter machine which accepts L_{k+1}.

□

The proof above requires alphabets of size $k+2$ to separate $\mathcal{L}_k$ from $\mathcal{L}_{k+1}$. However, it is not difficult to tighten the proof to establish a strict hierarchy for alphabets of size 3.

5 Characterising Regularity of Counter Recognisable Languages

Automata with bounded counters

Let $\mathcal{A} = (Q, \Sigma, \Gamma, T, q_{\text{in}}, F)$ be a counter 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:

- $Q[K] = Q \times \{f \mid f : \Gamma \longrightarrow [0..K]\}$.
- $Q[K]_{\text{in}} = (q_{\text{in}}, \overline{0})$.
- $F[K] = Q_f \times \{f \mid f : \Gamma \longrightarrow [0..K]\}$.
- 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 $f'(C) = \begin{cases} f(C)-1 & \text{if } f(C) < K \\ K & \text{otherwise} \end{cases}$

Notice that 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 $t_1^{-1} t_2^{-1} \ldots t_n^{-1}$. Note that 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 counter 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 counter automata

Product automaton Let $\mathcal{A}_1 = (Q_1, \Sigma_1, \Gamma_1, T_1, q^1_{\text{in}}, F_1)$ and $\mathcal{A}_2 = (Q_2, \Sigma_2, \Gamma_2, T_2, q^2_{\text{in}}, F_2)$ be two counter automata. The *product automaton* $\mathcal{A}_1 \times \mathcal{A}_2$ is the structure $(Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \Gamma_1 \cup \Gamma_2, T_1 \times T_2, (q^1_{\text{in}}, q^2_{\text{in}}), F_1 \times F_2)$, 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) $((q^1_{\text{in}}, \overline{0}), (q^2_{\text{in}}, \overline{0})) \overset{t_1 t_2 \ldots t_n}{\Longrightarrow} ((q_1, f_1), (q_2, f_2))$ *is a computation of* $\mathcal{A}_1 \times \mathcal{A}_2$ *if and only if* $(q^1_{\text{in}}, \overline{0}) \overset{\pi_1(t_1 t_2 \ldots t_n)}{\Longrightarrow} (q_1, f_1)$ *and* $(q^2_{\text{in}}, \overline{0}) \overset{\pi_2(t_1 t_2 \ldots t_n)}{\Longrightarrow} (q_2, f_2)$ *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* $L(\mathcal{A}_1 \times \mathcal{A}_2) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

Regularity and closure under complementation

Let $L \subseteq \Sigma^*$ be a language such that both L and its complement $\overline{L}$ are accepted by counter automata. Let $L = L(\mathcal{A})$ and $\overline{L} = L(\overline{\mathcal{A}})$, where we can assume that $\mathcal{A}$ and $\overline{\mathcal{A}}$ use disjoint sets of counters. Then 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 5.3 $L(\mathcal{A}[K] \times \overline{\mathcal{A}}) = \emptyset$.

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 4.3 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 become full along $\hat{\rho}$. Thus, we can extend $\hat{\rho}$ to an accepting run σ by appending the sequence of transitions $u_{m+1}u_{m+2}\ldots u_n$ 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. □

Theorem 5.4 *Let L be a language over Σ. L and $\overline{L}$ are counter recognisable iff L is regular.*

Proof: 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 counter recognisable then the language is regular.

The converse is obvious: if L is a regular language, we can find finite-state automata recognising both L and $\overline{L}$. Since finite-state automata are trivial examples of counter automata, both L and $\overline{L}$ are counter recognisable. □

Observe that our construction is effective—given 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.

Regularity and closure under reversal

Suppose L is recognised by the *deterministic* counter automaton $\mathcal{A}$ and L-reverse is accepted by the counter automaton $\mathcal{B}$. We will show that there is a constant τ, depending on the number of states M and the number of counters N of $\mathcal{A}$, such that $\mathcal{A}[\tau]$ recognises L.

Overview

The proof has two parts.

Part 1. We assume that $L(\mathcal{A}[\tau]) \neq L$ and conclude from this that there exist strings $\beta, \gamma \in \Sigma^*$ such that

$$\forall i \geq 1.\ \exists \alpha \in \Sigma^*.\ \exists n.\ [\alpha\beta^i\gamma \in L\ \&\ \forall j \geq n.\ \alpha\beta^j\gamma \notin L].$$

Part 2. We consider the reverse of the language L. Part 1 shows that there exist strings $\hat{\beta}, \hat{\gamma} \in \Sigma^*$ such that

$$\forall i \geq 1.\ \exists \alpha \in \Sigma^*.\ \exists n.\ [\hat{\gamma}\hat{\beta}^i\hat{\alpha} \in L\text{-reverse}\ \&\ \forall j \geq n.\ \hat{\gamma}\hat{\beta}^j\hat{\alpha} \notin L\text{-reverse}]. \tag{1}$$

On the other hand, we will show that for all counter automata $\mathcal{B}$ and all $\beta, \gamma \in \Sigma^*$, there exists an n such that if $\gamma\beta^i\alpha \in L(\mathcal{B})$ for some $i \geq n$, then $\gamma\beta^j\alpha \in L(\mathcal{B})$ for infinitely many j. Thus, it follows from (1) that L-reverse is not counter recognisable. This contradiction shows that our assumption $L \neq L(\mathcal{A}[\tau])$ (of Part 1) is false. We thus have $L = L(\mathcal{A}[\tau])$, and in particular, that L is regular.

Notation.

- While analysing the computation of counter automata, we will permit counters to assume negative values. We refer to such computations as *free runs,* and use $\chi_0 \overset{t}{\rightsquigarrow} \chi_m$ to denote the free run corresponding to the sequence of transitions t, starting from configuration χ_0 and ending at configuration χ_m, passing through configurations χ_i.
- Let $\mathcal{A}$ be a counter automaton with M states and N counters. We say that the counter C is *saturated* in the free run $\chi_0 \overset{u}{\rightsquigarrow} \chi_m$, if for some $i \in [0..m]$, $C(\chi_i) \geq \Pi_{M,N,0}$, and for all $j < i$, $C(\chi_j) \geq 0$.

Constants. In the rest of this section we write Π for $\Pi_{M,N,0}$. Let

$$\begin{aligned} \kappa &= M \cdot \Pi^N + 1; \\ \tau &= N \cdot \kappa. \end{aligned}$$

Part 1

Lemma 5.5 *Suppose t is a sequence of transitions such that $(q_{\text{in}}, \overline{0}) \overset{t}{\Longrightarrow}$ is an accepting run of $\mathcal{A}[\tau]$ but not of $\mathcal{A}$. Then, we have $t = uvw$ with the free run*

$$\chi_0 \overset{u}{\rightsquigarrow} \chi_i \overset{v}{\rightsquigarrow} \chi_j \overset{w}{\rightsquigarrow} \chi_n,$$

where $0 < i < j < n$, such that

(i) For some counter $\hat{C}$, $\Delta_{\hat{C}}(v) < 0$.

(ii) $Q(\chi_i) = Q(\chi_j)$.

(iii) If for some counter C, $\Delta_C(v) \neq 0$ or C assumes a negative value on the free run of $\mathcal{A}$ corresponding to t, then C is saturated in the free run $\mathcal{A} \overset{u}{\rightsquigarrow}$.

Proof: Let $t = t_1 t_2 \ldots t_n$. Consider the free run corresponding to t,

$$\rho : (q_{\text{in}}, \overline{0}) = \chi_0 \overset{t_1}{\rightsquigarrow} \chi_1 \overset{t_2}{\rightsquigarrow} \cdots \overset{t_n}{\rightsquigarrow} \chi_n.$$

Since this is not an accepting run of $\mathcal{A}$, there is a $j < n$ and a counter C such that $C(\chi_j) < 0$. Since $(q_{\text{in}}, \overline{0}) \overset{t}{\Longrightarrow}$ is an accepting run of $\mathcal{A}[\tau]$, there must be an $i < j$, where $C(\chi_i) = \tau$. Choose ℓ to be the maximum i such that there is a counter $\hat{C}$ satisfying the following two conditions.

- $\hat{C}(\chi_i) = \tau$.
- For some $j > i$, $\hat{C}(\chi_j) < 0$.

Let m be the minimum $j > \ell$ such that $\hat{C}(\chi_j) = 0$. Note that $m < n$, because there is a $j > \ell$ where $\hat{C}(\chi_j) < 0$.

Let $x = t_1 t_2 \ldots t_\ell$, $y = t_{\ell+1} t_{\ell+2} \ldots t_m$ and $z = t_{m+1} t_{m+2} \ldots t_n$. Since $\hat{C}(\chi_\ell) = \tau$ and $\hat{C}(\chi_m) = 0$, in the computation

$$\chi_\ell \overset{t_{\ell+1}}{\rightsquigarrow} \chi_{\ell+1} \overset{t_{\ell+2}}{\rightsquigarrow} \cdots \overset{t_m}{\rightsquigarrow} \chi_m,$$

$\hat{C}$ takes all values in the range $[0..\tau]$. For $j \in [0, N]$, let k_j be the minimum $k \in [\ell..m]$ such that $\hat{C}(\chi_k) = \tau - j\kappa$. The computation on xy can then be written as

$$\rho' : \chi_0 \overset{y_0 = x}{\rightsquigarrow} \chi_\ell = \chi_{k_0} \overset{y_1}{\rightsquigarrow} \chi_{k_1} \overset{y_2}{\rightsquigarrow} \cdots \overset{y_N}{\rightsquigarrow} \chi_{k_N} = \chi_m.$$

Let Γ' be the set of counters that assume a value Π or bigger somewhere in ρ'. For $C \in \Gamma'$, let j_C be the minimum j such that C assumes a value Π or bigger in the segment of the above computation corresponding to y_j. In particular, $j_{\hat{C}} = 0$ because $\hat{C}(\chi_\ell) = \tau \geq \Pi$ and $\chi_0 \overset{y_0}{\rightsquigarrow} \chi_\ell$. There are only $|\Gamma'| \leq N$ counters, whereas there are $N+1$ segments. Thus, there is a $j \in [0..N]$ such that $j \neq j_C$ for all $C \in \Gamma'$. Let $\hat{\jmath}$ be the minimum such j; since $j_{\hat{C}} = 0$, we have $\hat{\jmath} \neq 0$. Consider the computation corresponding to $y_{\hat{\jmath}}$,

$$\rho'' : \chi'_0 \overset{t'_1}{\rightsquigarrow} \chi'_1 \overset{t'_2}{\rightsquigarrow} \cdots \overset{t'_h}{\rightsquigarrow} \chi'_h,$$

where $t'_i = t_{k_{\hat{\jmath}-1}+i}$, $\chi'_i = \chi_{k_{\hat{\jmath}-1}+i}$ and $h = k_{\hat{\jmath}} - k_{\hat{\jmath}-1}$. In this computation the value of $\hat{C}$ falls from $\tau - (\hat{\jmath}-1)\kappa$ to $\tau - \hat{\jmath}\kappa$, that is, by κ. For $i \in [0..\kappa]$, let p_i be the minimum $p \in [0..h]$ such that $\hat{C}(\chi'_p) = \hat{C}(\chi'_0) - i = \tau - (\hat{\jmath}-1)\kappa - i$. Then, ρ'' can be written as

$$\rho'' = \chi''_0 \overset{y'_1}{\rightsquigarrow} \chi''_1 \overset{y'_2}{\rightsquigarrow} \chi''_2 \overset{y'_3}{\rightsquigarrow} \cdots \overset{y'_\kappa}{\rightsquigarrow} \chi''_\kappa,$$

where $\chi''_i = \chi'_{p_i}$ for $i \in [0..\kappa]$. Let Γ'' be the set of counters whose values in ρ'' after χ''_0 are less than Π. Since $\kappa = M\Pi^N + 1$, there exist $r, s \in [1..\kappa]$, $r < s$, such that

(C1) $C(\chi''_r) = C(\chi''_s)$, for all $C \in \Gamma''$.

(C2) $Q(\chi''_r) = Q(\chi''_s)$.

The definition of χ''_i implies

(C3) $\hat{C}(\chi''_r) = \hat{C}(\chi_{k_{j-1}}) - r > \hat{C}(\chi_{k_{j-1}}) - s = \hat{C}(\chi''_s)$.

Now let

$$\begin{aligned} u &= x y_1 y_2 \ldots y_{\hat{\jmath}-1}\, y'_1 y'_2 \ldots y'_r; \\ v &= y'_{r+1} y'_{r+2} \ldots y'_s; \\ \text{and} \quad w &= y'_{s+1} y'_{s+2} \ldots y'_\kappa\, y_{\hat{\jmath}+1} y_{\hat{\jmath}+2} \ldots y_N\, z. \end{aligned}$$

We claim that this choice of u, v and w satisfies the requirements of the lemma. Clearly, $t = uvw$. Part (i) of the lemma follows immediately from condition (C3) above; part (ii) follows from condition (C2). We now consider part (iii).

If C becomes negative in the free run ρ, then it must assume the value τ somewhere before that, because $(q_{\text{in}}, \overline{0}) \stackrel{t}{\Longrightarrow}$ is an accepting run in $\mathcal{A}[\tau]$. By the maximality of ℓ, this happens at or before χ_ℓ. Thus, C is saturated in the free run $\chi_0 \stackrel{u}{\rightsquigarrow}$. Next, we consider counters C such that $\Delta_C(v) \neq 0$. We may assume that C does not become negative in ρ, for we have just taken care of all such counters. By condition (C1) above, $C \notin \Gamma''$. That is, C takes a value Π or bigger after χ''_0 in the computation ρ''. Thus, $j_C \leq \hat{\jmath}$. Since $\hat{\jmath} \neq j_C$ for all C, we have $j_C < \hat{\jmath}$. Hence, C is saturated in the free run $\chi_0 \stackrel{u}{\rightsquigarrow}$. □

We need the following pumping lemma for free runs.

Lemma 5.6 *Suppose $\chi_0 \stackrel{t}{\rightsquigarrow} \chi_m$ is a free run of $\mathcal{A}$, where every counter that assumes a negative value is saturated. Let Γ' be the set of counters saturated in this run. Then, for all K, $\mathcal{A}$ has a run $\chi'_0 \stackrel{t'}{\Longrightarrow} \chi'_{m'}$ such that $\chi_0 = \chi'_0$, $Q(\chi'_{m'}) = Q(\chi_m)$, $F(\chi'_{m'}) \geq F(\chi_m)$ and $C(\chi'_{m'}) \geq K$ for all $C \in \Gamma'$.*

Proof: Similar to the proof of the Counter Pumping Lemma. Omitted. □

Proof of Part 1. If $L \neq L(\mathcal{A}[\tau])$, then there exists a string $a \in L(\mathcal{A}[\tau]) \setminus L$. Let $(q_{\text{in}}, \overline{0}) \stackrel{t}{\Longrightarrow}$ be the shortest accepting run of $\mathcal{A}[\tau]$ such that $\alpha(t)\restriction_\Sigma = a$. Clearly, $\chi_0 \stackrel{t}{\rightsquigarrow}$ is a free run of $\mathcal{A}$, but since $a \notin L$, this is not an accepting run of $\mathcal{A}$. Using Lemma 5.5, we obtain a decomposition $t = uvw$ such that the free run

$$\chi_0 \stackrel{u}{\rightsquigarrow} \chi_\ell \stackrel{v}{\rightsquigarrow} \chi_m \stackrel{w}{\rightsquigarrow} \chi_n,$$

$(0 < \ell < m < n)$ satisfies (i), (ii) and (iii). Let $\beta = \alpha(v)\restriction_\Sigma$ and $\gamma = \alpha(w)\restriction_\Sigma$. We first show that for all $i \geq 1$ there is an α such that $\alpha\beta^i\gamma \in L$.

Fix $i \geq 1$. Part (ii) implies that there is a free run of the form

$$\chi'_0 \stackrel{u}{\rightsquigarrow} \chi'_{\ell'} \stackrel{v^i}{\rightsquigarrow} \chi'_{m'} \stackrel{w}{\rightsquigarrow} \chi'_{n'},$$

where $\chi_0 = \chi'_0$ and $Q(\chi'_{n'}) = Q(\chi_n)$. Part (iii) implies that all counters that assume a negative value in this run are saturated in the initial segment $\chi'_0 \stackrel{u}{\rightsquigarrow} \chi'_{\ell'}$. We apply Lemma 5.6 to this initial segment with

$$K = \max_{C, l' \leq j \leq n'} |C(\chi'_j)|.$$

We obtain u' such that in the run $\chi_0 \stackrel{u'}{\Longrightarrow} \chi''$, $Q(\chi'') = Q(\chi'_{\ell'})$, $C(\chi'') \geq C(\chi'_\ell)$ for all C and $C(\chi'') \geq K$ for all counters saturated in $\chi'_0 \stackrel{u}{\rightsquigarrow} \chi'_{\ell'}$. Then, $\mathcal{A} \stackrel{u'v^iw}{\Longrightarrow}$ is an accepting run of $\mathcal{A}$. In particular, $\alpha(u'v^iw) = \alpha\beta^i\gamma \in L$, where $\alpha = \alpha(u')\restriction_\Sigma$.

It remains to show that $\alpha\beta^j\gamma \notin L$, for all large enough j. We first observe that $\alpha(v)\restriction_\Sigma$ is not empty. For otherwise, since $\mathcal{A}$ is deterministic, the set of states of $\mathcal{A}$ that appear in the run $\chi_m \stackrel{w}{\rightsquigarrow} \chi_n$ is a subset of the set of states that appear in the run $\chi_\ell \stackrel{v}{\rightsquigarrow} \chi_m$. This implies that $Q(\chi_n) = Q(\chi_j)$, for some $j \in [\ell..m]$, and, furthermore, that $\alpha(w)\restriction_\Sigma$ is empty. But then $\chi_0 \stackrel{t}{\rightsquigarrow}$ is not the shortest accepting run of $\mathcal{A}[\tau]$ with $\alpha(t)\restriction_\Sigma = a$. This contradiction shows that $\alpha(v)\restriction_\Sigma$ is not empty.

Clearly, for all $j \geq 1$ we have the free run $\chi_0 \stackrel{uv^jw}{\rightsquigarrow}$. Since $\Delta_{\hat{C}}(v) < 0$, for all large enough j this is not an accepting run of $\mathcal{A}$. Suppose $\alpha\beta^j\gamma \in L$ for some such j. Then, since $\mathcal{A}$ is deterministic, we have $uv^jw = t_1t_2$, where $\chi_0 \stackrel{t_1}{\Longrightarrow}$ is an accepting run of $\mathcal{A}$ and $\alpha(t_2)\restriction_\Sigma$ is empty, which implies that $|t_2| < |vw|$. Let $|t_2| = k$; we have $1 \leq k < |vw|$. Now $\chi_0 \stackrel{t_1}{\Longrightarrow}$ is an accepting run of $\mathcal{A}$. By comparing this run with the run $\chi_0 \stackrel{t}{\Longrightarrow} \chi_n$, we observe that $Q(\chi_{n-k})$ is a final state of $\mathcal{A}$. But then $\alpha(t_1t_2\ldots t_{n-k})\restriction_\Sigma = a$ and $\chi_0 \Rightarrow t_1t_2\ldots t_{n-k}$ is an accepting run of $\mathcal{A}[\tau]$, contradicting the minimality of t. Hence, $\alpha\beta^j\gamma \notin L$ for all large enough j. □

Part 2.

Lemma 5.7 *There exists a function $A(M, N, K, L)$ such that if $\rho : \chi_0 \stackrel{t}{\Longrightarrow}$ is a K-run of a counter automata $\mathcal{A}$ with M states and N counters such that*

- $|\alpha(t)\restriction_\Sigma| \leq L$ *and*
- $|t| \geq A(M, N, K, L)$,

then $t = uvw$ such that

- $\chi_0 \stackrel{u}{\Longrightarrow} \chi' \stackrel{v}{\Longrightarrow} \chi'' \stackrel{w}{\Longrightarrow}$;
- $Q(\chi') = Q(\chi'')$ *and* $F(\chi') \leq F(\chi'')$;
- $\alpha(v')$ *is non-empty but has no symbols from* Σ.

Proof: Let

$$A(M, N, K, L) = \begin{cases} \pi_{M,N,K} & \text{if } L = 0 \\ \pi_{M,N,K} + A(M, N, K + \pi_{M,N,K}, L - 1) & \text{if } L \geq 1 \end{cases}.$$

We will prove by induction on L that $A(M, N, K, L)$ defined above satisfies the requirements of the lemma.

Basis: If $L = 0$, the claim follows from the definition of the weak pumping constant.

Induction step: Let $\rho : \chi_0 \stackrel{t'}{\Longrightarrow} \chi_1 \stackrel{a}{\longrightarrow} \chi_2 \stackrel{t''}{\Longrightarrow}$, where t' is the maximal prefix of t with no symbols from Σ, and $a \in \Sigma$. If $|t'| \geq \pi_{M,N,K}$, then the claim follows from the definition of the weak pumping constant.

Otherwise, $\chi_2 \stackrel{t''}{\Longrightarrow}$ is a $(K + \pi_{M,N,K})$-run. Also, $|\alpha(t'')\restriction_\Sigma| = |\alpha(t)\restriction_\Sigma| - 1$ and

$$|t''| \geq t - \pi_{M,N,K} \geq A(M, N, K + \pi_{M,N,K}, L - 1).$$

The claim then follows from the induction hypothesis. □

Note. In the above lemma, we can assume that $|uv| \leq A(M,N,K,L)$, for we can always restrict ourselves to the prefix of t of length exactly $A(M,N,K,L)$.

Lemma 5.8 *Suppose $\mathcal{A}$ is a counter automaton with M states and N counters. Let $(l_i)_{i=0}^{\infty}$ be a sequence of non-negative integers. Then there exists a constant $B = B(M,N,K,(l_i))$ such that if*

$$\rho : \chi_0 \stackrel{v_0}{\Longrightarrow} \chi_1 \stackrel{v_1}{\Longrightarrow} \cdots \stackrel{v_{B-1}}{\Longrightarrow} \chi_B,$$

where $|v_i| \leq l_i$, then there exist $i, j \in [1,B]$, $i < j$, such that $Q(\chi_i) = Q(\chi_j)$ and $F(\chi_i) \leq F(\chi_j)$.

Proof: We will modify the proof of Lemma 2.7. Consider the following infinite tree T whose nodes are labelled by elements of $\mathbb{N}^N$. The root of T is labelled by $F(\chi_0)$. If a node at level i (the root is at level 0) is labelled by f, then v has one child for each N-tuple obtained by performing at most l_i increment and decrement operations on the components of f.

Clearly, T is finitely branching and infinite. By Lemma 2.6 there exists μ_k such that along any path in T of length μ_k starting at the root, the corresponding sequence of labels has a non-decreasing subsequence of length k. Now μ_k depends only on the tree T, which is unique if the label of the root and the sequence (l_i) are fixed. The label of the root has at most $(K+1)^N$ possibilities; hence there exists a function $\mu(k,M,N,K,(l_i))$, such that in every such tree in every path of length $\mu(k,M,N,K,(l_i))$ starting from the root, the corresponding labels have a non-decreasing sequence of length at least k.

We set $B(M,N,K,(l_i)) = \mu(M+2,M,N,K,(l_i))$ and complete the proof of the lemma by arguing as in Lemma 2.7. (We have $M+2$ and not $M+1$ because in the lemma we want $i \neq 0$.) □

Lemma 5.9 (Part 2) *Let $\mathcal{A}$ be a counter automaton with M states and N counters. There exists a constant $E = E(M,N,K,\ell)$ such that if $\chi_0 \stackrel{t}{\Longrightarrow}$ is an accepting K-run of $\mathcal{A}$, where $\alpha(t)\restriction_\Sigma = \gamma\beta^m\alpha$ $(|\beta|, |\gamma| \leq \ell)$, and $m \geq E$, then for infinitely many j there is an accepting run $\chi_0 \stackrel{t_j}{\Longrightarrow}$ with $\alpha(t_j)\restriction_\Sigma = \gamma\beta^j\alpha$.*

Proof: Define the sequence $(\ell_i(M,N,K))$ by

$$\begin{aligned} l_0 &= A(M,N,K); \\ l_i &= A(M,N,K+\ell_1+\ell_2+\cdots+\ell_{i-1}). \end{aligned}$$

Let $D(M,N,K) = B(M,N,K,(\ell_i(M,N,K)))$ and

$$\overline{\ell}(M,N,K) = \sum_{i=1}^{D(M,N,K)-1} l_i(M,N,K).$$

Then, $E(M,N,K)$ is defined by

$$E(M,N,K,\ell) = \begin{cases} B(M,N,K,(\ell)) & \text{if N=0.} \\ D(M,N,K) + E(M,N-1,K+\overline{\ell}(M,N,K),\ell) & \text{if } N>1 \end{cases}.$$

(Here (ℓ) denotes the infinite sequence all of whose terms are ℓ.)

We will use induction on the number of counters to show that E as defined above meets the requirements of the lemma.

Basis: If $N=0$, there are no counter moves. The claim then follows from Lemma 5.8.

Induction step: Assume t is minimal such that $\chi_0 \stackrel{t}{\Longrightarrow}$ is an accepting K-run of $\mathcal{A}$ with $\alpha(t)\restriction_\Sigma = \gamma\beta^m\alpha$ $(m \geq E)$. We may write this computation as

$$\chi_0 \stackrel{v_0}{\Longrightarrow} \chi_1 \stackrel{v_1}{\Longrightarrow} \chi_2 \stackrel{v_2}{\Longrightarrow} \cdots \stackrel{v_m}{\Longrightarrow} \chi_{m+1} \stackrel{w}{\Longrightarrow} \chi_f,$$

where $\alpha(v_0)\restriction_\Sigma = \gamma$, $\alpha(v_i)\restriction_\Sigma = \beta$ for $i \in [1..m]$ and $\alpha(w)\restriction_\Sigma = \alpha$. If $|v_i| \leq \ell_i$ for $i \in [0..D(M,N,K)-1]$, then the claim follows from Lemma 5.8.

Otherwise, there exists an $i \in [0..D(M,N,K)-1]$ such that $|v_i| \geq \ell_i$. Let i be the smallest with this property. By Lemma 5.7, for this i, we may write the computation on v_i as

$$\chi_i \stackrel{v}{\Longrightarrow} \chi' \stackrel{v'}{\Longrightarrow} \chi'' \stackrel{v''}{\Longrightarrow} \chi_{i+1},$$

where $Q(\chi') = Q(\chi'')$, $F(\chi') \leq F(\chi'')$, where $\alpha(v')$ is nonempty but has no input symbols. Since t is minimal, we have $F(\chi') < F(\chi'')$. Fix a counter C such that $\Delta_C(v') > 0$. Next, consider the computation after v', that is

$$\chi'' \stackrel{v''}{\Longrightarrow} \chi_{i+1} \stackrel{v_{i+1}}{\Longrightarrow} \chi_{i+2} \stackrel{v_{i+2}}{\Longrightarrow} \cdots \stackrel{v_m}{\Longrightarrow} \chi_{m+1} \stackrel{w}{\Longrightarrow} \chi_f.$$

Here we treat C^+ and C^- as elements of the input alphabet. As noted above, we may assume $|vv'| \leq \ell_i$. Thus, $|v_0 \ldots v_{i-1}vv'| \leq \overline{\ell}(M,N,K)$. Also, $\alpha(v''v_{i+1}\ldots v_m w)\restriction_\Sigma = \gamma'\beta^j\alpha$, where

$$j \geq E(M,N,K,\ell) - D(M,N,K) \geq E(M,N-1,K+\overline{\ell}(M,N,K),\ell).$$

We apply the induction hypothesis to this computation. For infinitely many j we obtain t'_j such that $X'' \stackrel{t'_j}{\Longrightarrow}$ is an accepting computation of $\mathcal{A}$ and $\alpha(t_j)\restriction_\Sigma = \gamma'\beta^j\alpha$.

Since $\Delta_C(v') > 0$, it is easily verified that for sufficiently large k and

$$t = v_0 v_1 \ldots v_{i-1} v v'^k v'' t'_j,$$

$\chi_0 \stackrel{t}{\Longrightarrow}$ is an accepting computation of $\mathcal{A}$. Since $\alpha(v')$ has no input symbols, this implies that for infinitely many j, there is a t_j such that $\chi_0 \stackrel{t_j}{\Longrightarrow}$ is an accepting computation of $\mathcal{A}$ and $\alpha(t_j)\restriction_\Sigma = \gamma\beta^j\alpha$. □

6 Discussion

In this concluding discussion, we point out similarities and differences between our work and earlier results from the theory of vector addition systems and Petri nets. We also identify some directions for further work in developing our model of finite-state distributed systems with asynchronous communication.

Vector addition systems

An n-coordinate *vector addition system (VAS)* consists of a finite set of *initial vectors* and a finite set of *transition vectors*. Each initial vector is an n-tuple of natural numbers and each transition vector is an n-tuple of integers. An n-tuple of natural numbers is *reachable* if it can be generated from an initial vector by performing a sequence of additions with vectors from the set of transitions while ensuring that each intermediate vector generated is non-negative.

In [KM69], Karp and Miller study various decision problems for vector addition systems. They show how to associate with each VAS a finite object called its covering tree. This can be used to solve a number of other questions, including whether the set of reachable vectors of the VAS is finite. These results have immediate applicability in the theory of Petri nets because a Petri net can be represented as a VAS. Our Counter Pumping Lemma (Lemma 4.3) is similar in spirit to Karp and Miller's covering tree result.

Petri net languages

It is well known that there is a strong connection between automata with blind counters and Petri nets [G78, J86a]. It is not difficult to show that we can go back and forth between labelled Petri nets and counter automata in such a way that given a net N and its corresponding counter automaton $\mathcal{A}$, there is bijection a between the firing sequences of N and the computations of $\mathcal{A}$. Thus, questions about Petri net languages can rephrased as questions about languages accepted by counter automata.

There are several ways to associate a language with a Petri net [H75, J86a]. The first is to just examine all firing sequences of the net. The second is to fix a set of final markings and look at the labels along firing sequences leading to these designated markings. The third possibility is to fix final markings but only require a firing sequence to lead to a marking which dominates a final marking rather than be exactly equal to a final marking. Following the terminology of [J86a], we designate the class of languages generated by these three definitions $\mathcal{L}$, $\mathcal{L}_0$ and $\mathcal{L}_1$ respectively. When transitions are allowed to have invisible labels, the corresponding classes of languages are designated $\mathcal{L}^\lambda$, $\mathcal{L}_0^\lambda$ and $\mathcal{L}_1^\lambda$ respectively.

In the setting of counter automata, the first definition corresponds to examining the set of computations of the automaton. The second definition yields a definition of accepting runs in terms of both final states and final counter values. The third definition corresponds more directly to the one we use in this paper—the final states are fixed but the final counter values are irrelevant.

In the theory of Petri net languages, a number of positive results have been established for the class $\mathcal{L}$—for instance, regularity is decidable [GY80, VV80]. On the other hand, it is quite easy to exhibit languages from the class $\mathcal{L}_1$ (and hence, counter recognisable languages) where the language itself is regular but the underlying language of transitions is not. Hence the results of [GY80, VV80] do not carry over to counter recognisable languages—in fact, the problem of deciding whether a counter recognisable language is regular is open.

At the other end of the spectrum, a number of negative results have been established for the class $\mathcal{L}_0$ (and hence also $\mathcal{L}_0^\lambda$). For instance, it is undecidable whether such a language is universal—that is, whether it consists of *all* strings [VV80]. However, this result crucially uses the fact that final markings must be reached exactly. The problem

of deciding whether a counter recognisable language is universal is open.

Few, if any, results have been proved for the classes $\mathcal{L}_1$ and $\mathcal{L}_1^\lambda$, which correspond most closely to counter recognisable languages. Our characterisation of the subclass closed under complementation is probably the *only* non-trivial result known for this class. Notice that this characterisation fails for the class $\mathcal{L}_0$—it is possible to construct nets for both the language L_{ge} of Example 2.1 and its complement, though L_{ge} is not regular.

Counter recognisable languages and message-passing

How can we interpret our results on counter recognisable languages in terms of distributed systems for asynchronous communication? We say that an asynchronous protocol is *robust* if it responds "sensibly" to any sequence of interactions with the environment—in other words, for any such sequence, it either accepts the sequence as valid or terminates with an error. When we model asynchronous protocols by counter automata, the strings accepted by the automaton correspond to sequences of interactions with the environment. For a robust protocol, both the set of interactions accepted by the protocol and the set of interactions rejected by the protocol are counter recognisable. Our characterisation of the complementation-closed subset of counter recognisable languages then tells us that all robust protocols use only bounded buffers. Any messages exchanged by processes following a robust protocol can be viewed as just hand-shakes which coordinate the interaction between the different processes and the environment.

One shortcoming of our model is that we implicitly sequentialise all the interactions of a distributed system into a sequence of global interactions. It would be more satisfying to build a theory where we separate the interaction of each process and allow our automata to read n-tuples of strings, where n is the number of processes. It is not obvious how to extend the notion of a robust protocol to this setting. One possibility is to use the fact we can keep track of the latest information each process has about every other process in a message-passing system using the algorithm proposed in [MNS95]. In the theory of synchronous communication, an analogous result is the key to generating a distributed finite-state system recognising an n-tuple of strings from a global description of such a system [MS94, Z87]. We have some preliminary results in this direction.

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).

[AJ94] P.A. Abdulla and B. Jonsson: Undecidability of verifying programs with unreliable channels, in S. Abiteboul, E. Shamir (eds.), *Proc. ICALP '94*, Springer LNCS **820** (1994) 316–327.

[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).

[J86a] 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.

[J86b] M. Jantzen: Complexity of Place/Transition 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) 413–434.

[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).

[L76] R.J. Lipton: The Reachability Problem Requires Exponential Space, *Research Report No 62*, Dept of Computer Science, Yale University (1976).

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

[MNS95] M. Mukund, K. Narayan Kumar and M. Sohoni: Keeping Track of the Latest Gossip in Message-Passing Systems, *Proc. Structures in Concurrency Theory (STRICT)*, Berlin 1995, Workshops in Computing Series, Springer-Verlag (1995) 249–263.

[MS94] M. Mukund and M. Sohoni: Gossiping, Asynchronous Automata and Zielonka's Theorem, *Report TCS-94-2*, School of Mathematics, SPIC Science Foundation, Madras, India (1994).

[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.

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

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

SPIC Mathematical Institute

Internal Reports (Theoretical Computer Science)

TCS-90-1 M. Mukund: *Expressiveness and Completeness of a Logic for Well Branching Prime Event Structures.*

TCS-90-2 M. Mukund and P.S. Thiagarajan: *An Axiomatization of Well Branching Prime Event Structures.*

TCS-90-3 K. Lodaya, M. Mukund, R. Ramanujam, P.S. Thiagarajan: *Models and Logics for True Concurrency.*

TCS-91-1 P.S. Thiagarajan (ed.): *Proceedings of National Seminar on Theoretical Computer Science, Madras, India, July 4–6, 1991.*

TCS-91-2 M. Mukund: *A Transition System Characterization of Petri Nets.*

TCS-91-3 P.W. Hoogers, H.C.M. Kleijn, P.S. Thiagarajan: *A Trace Semantics for Petri Nets.*

TCS-91-4 M. Nielsen, G. Rozenberg, P.S. Thiagarajan: *Elementary Transition Systems.*

TCS-91-5 M. Nielsen, G. Rozenberg, P.S. Thiagarajan: *Transition Systems, Event Structures and Unfoldings.*

TCS-92-1 M. Nielsen, G. Rozenberg, P.S. Thiagarajan: *Elementary Transition Systems and Refinement.*

TCS-92-2 M. Mukund and M. Nielsen: *CCS, Locations and Asynchronous Transition Systems.*

TCS-92-3 M. Mukund: *Transition System Models for Concurrency.*

TCS-93-2 M. Agrawal: *On the Isomorphism Problem for Weak Reducibilities.*

TCS-93-3 M. Mukund and M. Sohoni: *Keeping Track of the Latest Gossip: Bounded Time-Stamps Suffice.*

TCS-93-4 P.S. Thiagarajan: *A Trace Based Extension of PTL.*

TCS-93-5 N. Klarlund, M. Mukund, M. Sohoni: *Determinizing Asynchronous Automata.*

TCS-93-6 P.S. Thiagarajan: *TrPTL: A Trace Based Extension of Linear Time Temporal Logic.*

TCS-93-7 M. Agrawal and V. Arvind: *On Quasi-Linear Truth-Table Reductions to P-Selective Sets.*

TCS-93-8 K. Lodaya, R. Parikh, R. Ramanujam, P.S. Thiagarajan: *A Logical Study of Distributed Transition Systems.*

TCS-94-1 P.W. Hoogers, H.C.M. Kleijn, P.S. Thiagarajan: *A Event Structure Semantics for General Petri Nets.*

TCS-94-2 M. Mukund and M. Sohoni: *Gossiping, Asynchronous Automata and Zielonka's Theorem.*

TCS-94-3 R. Krishnan and S. Venkatesh: *Optimizing the Gossip Automaton.*

TCS-94-4 M. Agrawal, R. Krishnan and S. Venkatesh: *The Isomorphism Problem for 2-DFA Reductions.*

TCS-94-5 M. Agrawal and V. Arvind: *Geometric Sets of Low Information Content.*

TCS-95-1 M. Agrawal: *Self-reducibility Versus Prunability.*

TCS-95-2 M. Agrawal, P. Ramadevi and V Vinay: *A New Link Invariant and its Complexity.*

TCS-95-3 M. Mukund, K. Narayan Kumar and M. Sohoni: *Keeping Track of the Latest Gossip in Message-Passing Systems.*

TCS-95-4 P.S. Thiagarajan: *PTL over Product State Spaces.*

TCS-95-5 M. Agrawal: *DSPACE(n) $\stackrel{?}{=}$ NSPACE(n): A Degree Theoretic Characterization.*

TCS-95-6 N. Klarlund, M. Mukund, M. Sohoni: *Determinizing Asynchronous Automata on Infinite Inputs.*

TCS-95-7 S. Krishnamurthy, M. Mukund: *Implementing Causal Ordering with Bounded Time-stamps.*

TCS-96-1 M. Mukund, P.S. Thiagarajan: *Linear Time Temporal Logics over Mazurkiewicz Traces.*

TCS-96-2 M. Mukund: *Finite-state Automata on Infinite Inputs.*

TCS-97-1 Jesper G. Henriksen and P.S. Thiagarajan: *Dynamic Linear Time Temporal Logic.*

TCS-97-2 P.S. Thiagarajan and I. Walukiewicz: *An Expressively Complete Linear Time Temporal Logic for Mazurkiewicz Traces.*

TCS-97-3 Jesper G. Henriksen and P.S. Thiagarajan: *A Product Version of Dynamic Linear Time Temporal Logic.*

TCS-97-4 M. Mukund, K. Narayan Kumar, J. Radhakrishnan and M. Sohoni: *Message-Passing Automata and Asynchronous Communication.*

Copies of reports can be ordered from the following address:

SPIC Mathematical Institute
92, G.N. Chetty Road
T. Nagar
Madras 600 017

Email: `office@smi.ernet.in`