Fundamenta Informaticae 116 (2012) 129–140
DOI 10.3233/FI-2012-674
IOS Press

A Survey of Results on Stateless Multicounter Automata

Oscar H. Ibarra[*][†]
Department of Computer Science, University of California
Santa Barbara, CA 93106, USA
ibarra@cs.ucsb.edu

Ömer Eğecioğlu
Department of Computer Science, University of California
Santa Barbara, CA 93106, USA
omer@cs.ucsb.edu

Abstract. A stateless multicounter machine has m-counters operating on a one-way input delimited by left and right end markers. A move of the machine depends only on the symbol under the input head and the sign pattern of the counters. An input string is accepted if, when the input head is started on the left end marker with all counters zero, the machine eventually reaches the configuration where the input head is on the right end marker with all the counters again zero.

We bring together a number of results on stateless multicounter automata of various different types: deterministic, nondeterministic, realtime (the input head moves right at every step), or non-realtime. We investigate realtime and non-realtime machines in both deterministic and nondeterministic cases with respect to the number of counters and reversals. In addition to hierarchy results, we also consider closure properties and the connections to stateless multihead automata.

Keywords: Stateless multicounter machines, reversal-bounded, realtime, non-realtime, hierarchy, stateless multihead automata, closure properties.

1. Introduction

There has been recent interest in studying stateless machines (which is to say machines with only one state), see e.g. [15], because of their connection to certain aspects of membrane computing and P

[*]Address for correspondence: Department of Computer Science, University of California, Santa Barbara, CA 93106, USA
[†]Research supported in part by NSF Grants CCF-1143892 and CCF-1117708.

systems, a subarea of molecular computing that was introduced in a seminal paper by Gheorge Păun [13] (see also [14]). A membrane in a P system consists of a multiset of objects drawn from a given finite type set $\{a_1, \ldots, a_k\}$. The system has no global state (i.e., it is stateless) and works on the evolution of objects in a massively parallel way. Thus, the membrane can be modeled as having counters $x_1, \ldots, x_k$ to represent the multiplicities of objects of types $a_1, \ldots, a_k$, respectively, and the P system can be thought of as a counter machine in a nontraditional form: without states, and with parallel counter increments/decrements. However in this paper, we only consider stateless machines with sequential counter increments/decrements.

Since stateless machines have no states, the move of such a machine depends only on the symbol(s) scanned by the input head(s) and the local portion of the memory unit(s). Acceptance of an input string has to be defined in a different way. For example, in the case of a pushdown automaton (PDA), acceptance is by "null" stack. It is well known that nondeterministic PDAs with states are equivalent to stateless nondeterministic PDAs [6]. However, this is not true for the deterministic case [10]. For Turing Machines, where acceptance is when the machine enters a halting configuration, it can be shown that the stateless version is less powerful than those with states. In [4, 9, 15] the computing power of stateless multihead automata were investigated with respect to decision problems and head hierarchies. For these devices, the input is provided with left and right end markers. The move depends only on the symbols scanned by the input heads. The machine can be deterministic, nondeterministic, one-way, two-way. An input is accepted if, when all heads are started on the left end marker, the machine eventually reaches the configuration where all heads are on the right end marker. In [11], various types of stateless restarting automata and two-pushdown automata were compared to the corresponding machines with states. Properties of a variant in which the machine is a stateless multicounter Watson-Crick automaton [5] is considered in [1, 2, 12].

In this paper, we consider the computing power of stateless multicounter machines with reversal-bounded counters, stating known properties of these machines, including a subset of the proofs of the results. Such a machine has m-counters. It operates on a one-way input delimited by left and right end markers. The move of the machine depends only on the symbol under the input head and the signs of the counters, which indicate if the counter is zero or not. An input string is accepted if, when the input head is started on the left end marker with all counters zero, the machine eventually reaches the configuration where the input head is on the right end marker with all the counters again zero. Moreover, in most interesting cases, the machine is *k-reversal bounded* [7]: in other words for a specified k, no counter makes more than k pairs of consecutive increase/decrease alternations between increasing mode and decreasing mode (i.e. k successive pairs of increase content/decrease content stages) in any computation, accepting or not.

2. Stateless Multicounter Machines

A deterministic stateless (one-way) m-counter machine operates on an input of the form $\cent w\$$, where $\cent$ and \$ are the left and right end markers for the input w. At the start of the computation, the input head is on the left end marker $\cent$ and all m counters are zero. The moves of the machine are described by a set of rules of the form: $(x, s_1, .., s_m) \rightarrow (d, e_1, \ldots, e_m)$, where $x \in \Sigma \cup \{\cent, \$\}$, Σ is the input alphabet, s_i = sign of counter C_i (0 or 1 for positive), d = 0 or 1 (direction of the move of the input head: d = 0 means don't move, d =1 means move the head one cell to the right), and $e_i = +, -$, or 0 (increment counter i

by 1, decrement counter i by 1, or do not change counter i), with the restriction that $e_i = -$ is applicable only if s_i = 1. For a deterministic machine, no two rules have the same left hand sides.

The input w is accepted if the machine reaches the configuration where the input head is on the right end marker $ and all counters are zero. The machine is *k-reversal* if it has the property that each counter makes at most k "full" alternations between increasing mode and decreasing mode and vice-versa on any computation (accepting or not). Thus, e.g., $k = 2$ means the counter can only go from increasing to decreasing to increasing to decreasing. A machine is *reversal-bounded* if it is k-reversal for some k.

In the general case, when the input head reaches $\$$, the machine can continue computing until all counters eventually become zero for an input that is accepted. A special case is when the machine is *realtime*: in this case $d = 1$ for each rule, i.e., the input head moves right at each step. This means that when the input head reaches $\$$, all the counters must be zero for the input to be accepted. Deterministic realtime machines were investigated in [3], where hierarchies with respect to the number of counters and number of reversals were studied. In *non-realtime* machines d can be 0 or 1. In particular, when the input head reaches $\$$, the machine can continue computing until all counters become zero, and then accept. A stateless multicounter machine is nondeterministic if different rules are allowed to have identical left hand sides. Hierarchies and properties of deterministic realtime machines were studied in [8].

It can be shown that stateless realtime multicounter machines are quite powerful, even in the unary input alphabet case of $\Sigma = \{a\}$. We report the following basic characterization of the languages accepted by realtime multicounter machines over a unary alphabet. A proof of this characterization result can be found in [3].

Theorem 2.1. Every language over $\Sigma = \{a\}$ accepted by a stateless realtime multicounter machine M is of the form $a^r(a^s)^*$ for some $r, s \geq 0$.

3. 1-Reversal Machines

The simplest reversal-bounded machines are 1-reversal. In any accepting computation, the contents of the counters may increase, followed by a decrease to zero. The machine can be realtime or non-realtime.

3.1. 1-Reversal Realtime Machines

For 1-reversal realtime machines accepting only a singleton language of the form $L = \{a^n\}$. The precise value of the maximum such n can be determined. The program of the machine achieving this n is unique (up to relabeling of the counter indices). Note that we can interpret a^n as the "maximum" number that a 1-reversal m-counter machine can count. In [3], an upper bound on how high a 1-reversal, m-counter machine can count is provided. The construction proves

Theorem 3.1. We can construct a 1-reversal realtime machine M_m^* with m counters which accepts the singleton $\{a^n\}$ with

$$n = (m-1)2^m + m \ . \tag{1}$$

The value of n given in (1) is the maximal value that a single reversal m counter machine can count. Furthermore, the program of any machine that achieves this bound is unique up to relabeling of the counters.

For 1-reversal realtime machines $m+1$ counters is better than m counters. Here we no longer assume that the language accepted is a singleton (or finite), nor the alphabet is unary.

Theorem 3.2. Suppose L is accepted by a realtime 1-reversal machine with m counters. Then L is accepted by a realtime 1-reversal machine with $m + 1$ counters. Furthermore the containment is strict.

Proof:
Given a realtime 1-reversal machine M with m counters that accepts M, we can view M as an $m + 1$ counter machine which behaves exactly like M on the first m counters, and never touches the $(m+1)$-st counter. Since the acceptance of an input string is defined by entering \$ when all counters are zero, this machine is also 1-reversal and accepts L. By theorem 3.1, the singleton $\{a^n \mid n = m2^{m+1} + m + 1\}$ is accepted by the 1-reversal machine M^*_{m+1}. Since $(m - 1)2^m + m < m2^{m+1} + m + 1$ for $m > 0$, this language is not accepted by any 1-reversal realtime machine with m-counters. □

3.2. 1-Reversal Non-realtime Machines

Clearly, any language accepted by a realtime machine can be accepted by a non-realtime machine. However the latter is strictly more powerful. For $w \in \Sigma^*$ and $a \in \Sigma$, we define $|w|_a$ as the number of occurrences of a in w.

Theorem 3.3. The language $L = \{w \mid w \in \{a, b\}^*, |w|_a = |w|_b\}$ can be accepted by a stateless non-realtime 1-reversal 2-counter machine M but not by a stateless realtime k-reversal m-counter machine for any $k, m \geq 1$.

Proof:
M has counters C_1 and C_2. On input ¢w\$, M reads the input and stores the number of a's (resp., b's) it sees in C_1 (resp., C_2). When the input head reaches \$, the counters are decremented simultaneously while the head remains on \$. M accepts if and only if the counters become zero at the same time.

Suppose L is accepted by some realtime k-reversal m-counter machine M'. Let x be a string with $|x|_a = |x|_b > 0$. Then x is accepted by M', i.e., M' on input ¢x\$, starts with the input head on ¢ with all counters zero, computes, and accepts after reading the last symbol of x with all counters again at zero. Consider now giving input xab to M'. After processing x, all counters are zero. Clearly, after processing symbol a, at least one counter of M' must increment; otherwise (i.e., if all counters remain at zero), M' will accept all strings of the form xa^i for all i, a contradiction. Then after processing b, all counters must again be zero, since xab is in L. It follows that on input xab, at least one counter made an additional reversal than on input x. Repeating the argument, we see that for some i, $x(ab)^i$ will require at least one counter to make $k + 1$ reversals. Therefore M' cannot be k-reversal for any k. □

Theorem 3.3 can be made stronger. Call a non-realtime reversal-bounded multicounter machine *restricted* if it can only accept an input when the input head first reaches the right end marker \$ and all counters are zero. Hence, there is no applicable rule when the input head is on \$. However, the machine can be non-realtime (i.e., need not move at each step) when the head is not on \$. The machine can also be nondeterministic. An argument similar to the proof of Theorem 3.3 can be used to prove the following result:

Corollary 3.4. $L = \{w \mid w \in \{a,b\}^*, |w|_a = |w|_b\}$ cannot be accepted by any stateless restricted nondeterministic non-realtime reversal-bounded multicounter machine.

Finally there is also an example of a unary singleton language that is accepted by a non-realtime 1-reversal machine that does not seem to be acceptable by a realtime 1-reversal machine. Our construction uses a technique in [9] where it was shown that the language can be accepted by a stateless $(2m+1)$-head machine.

Theorem 3.5. For every $m \geq 1$, the singleton language $L = \{a^{2^m-1}\}$ can be accepted by a stateless non-realtime 1-reversal $2(m+1)$-counter machine M.

Proof:
Let the counters be $0, 1, \ldots, 2m+1$. Counters $2, \ldots, 2m+1$ form m pairs $(i, i+m)$. Initially all counters are zero and the input head is on the left end marker. We describe the computation of M on input ¢a^n\$ in two phases.

Loading phase:

In this phase, the input head is moved to the right while simultaneously counters $1, \ldots, 2m+1$ are incremented by 1 for every right move of the head. When the input head reaches \$, counter 0 has value zero and counters $1, \ldots, 2m+1$ have value $n+1$. Then M enters the next phase.

Computing phase:

When this phase is entered, the input head is on the right end marker, counter 0 has value zero and counters $1, \ldots, 2m+1$ each have value $n+1$. We will refer to counter 1 as the head counter. The input head remains on the right end marker during this phase.

First, counter 0 is incremented by 1, counters $1, 2, \ldots, m+1$ (i.e., the main counter and the first counter from each pair) are decremented by 1, while counter $m+2, \ldots, 2m+1$ (second components of all pairs) remain unchanged with value $n+1$.

Then counters $(m+1, 2m+1)$ (that is, the last pair), whose difference (in value) is one, are decremented until $m+1$ becomes zero. From here, counters $1, 2, \ldots, m$ (the main counter and the first components of all unused pairs) are decremented simultaneously with counter $2m+1$, until counter $2m+1$ becomes zero. This will take only one step, and after that counters $1, 2, \ldots, m$ will have value $n-1$, counters $m+2, \ldots, 2m$ will have value $n+1$, while counters $m+1$ and $2m+1$ will have value zero.

Then the next pair $(m, 2m)$ is taken, and the same sequence of steps is repeated. Note that the difference in values between these counters is now 2. The result is that counters m and $2m$ are decremented to zero, while counters $1, 2, \ldots, m-1$ will have value $n-3$. This is continued with the rest of the pairs, until the following configuration is reached: counter 0 has value 1, counters 1 and 2 have values 2^{m-1}, counter $m+2$ has value $n+1$, and all other counters are zero.

From here, counters 2 and $m+2$ are decremented until counter 2 becomes zero. At this point, counters 1 and $m+2$ have same value if and only if the length of the string is $2^m - 1$. After that counters 1 and $m+2$ are decremented and the input is accepted if and only if these counters become zero at the same time. This happens if and only if the input has length $2^m - 1$. □

4. *k*-Reversal Machines

Now we consider the properties of k-reversal machines, $k \geq 1$. The next result gives an upper bound on the maximal n that is countable by a k-reversal m-counter machine.

Theorem 4.1. If the upper bound on n for a realtime 1-reversal m-counter machine is $f(m)$, then $f((2k-1)m)$ is an the upper bound on n for a k-reversal m-counter machine.

Proof:
We sketch the proof. Let $L = \{a^n\}$ be a singleton language accepted by a k-reversal m-counter machine M. We will show how we can construct from M a 1-reversal $(2k-1)m$-counter machine M' that makes at least as many steps as M and accepts a language $L' = \{a^{n'}\}$ for some $n' \geq n$. The result then follows. The construction of M' from M is based on the following ideas:

1. Consider first the case $k = 2$. Assume for now that the counters reverse from decreasing to increasing at different times.

2. Let C be a counter in M that makes 2 reversals. We associate with C three counters C, T, C' in M'. Initially, $T = C' = 0$.

3. C in M' simulates C in M as long as C does not decrement. When C decrements, T is set to 1 (i.e., it is incremented). Then as long as C does not increment the simulation continues.

4. When C in M increments, C in M' is decremented while simultaneously incrementing C' until C becomes zero. During the decrementing process all other counters remain unchanged. But to make M' operate in realtime, its input head always reads an a during this process.

5. When the counter C of M' becomes zero, T is set to zero (i,e., it is decremented), and C' is incremented by 1.

6. Then the simulation continues with C' taking the place of C. Counters C and T remain at zero and no longer used.

So if C in M makes 2 reversals, we will need three 1-reversal counters C, T, C' in M'. If C makes 3 reversals, we will need five 1-reversal counters C, T, C', T', C'' in M'. In general, if C makes k reversals, we will need $(2k-1)$ 1-reversal counters in M'. It follows that if there are m counters where each counter makes k reversals, we will need $(2k-1)m$ 1-reversal counters. If some of the counters "reverse" (to increasing) at the same time, we handle them systematically one at a time, by indexing the counters. □

From Theorem 4.1 and Theorem 3.1 we obtain:

Corollary 4.2. If $L = \{a^n\}$ is accepted by a realtime k-reversal m-counter machine, then

$$n \leq ((2k-1)m - 1)2^{(2k-1)m} + (2k-1)m \; .$$

It is also true that the number of counters matters for realtime k-reversal machines. The following result is proved in [3].

Theorem 4.3. For any fixed k, there is a language accepted by a realtime k-reversal $(m+1)$-counter machine which is not accepted by any k-reversal m-counter machine.

The next result, also shown in [3], gives a hierarchy with respect to the number of reversals for any fixed m, but some assumption on the number of reversals k is necessary for the hierarchy to be guaranteed.

Theorem 4.4. For any fixed m and $k < 2^{m-1}/m$, there is a language accepted by a realtime $(k+1)$-reversal m-counter machine which is not accepted by any k-reversal m-counter machine.

5. Counter and Reversal Hierarchies

For *non-realtime* machines, the following result proved in [3] relates the number of reversals to counters.

Theorem 5.1. If a language L is accepted by a stateless non-realtime k-reversal m-counter machine then it can be accepted by a stateless non-realtime 1-reversal $(2k-1)m$-counter machine.

First we prove that there is a hierarchy with respect to the number of counters for stateless non-realtime machines.

Lemma 5.2. For $k, m \geq 1$, there is a unique maximal number $f(k,m)$ such that the singleton language $L = \{a^{f(k,m)}\}$ is accepted by a stateless non-realtime k-reversal m-counter machine. (We refer to L as "maximal".)

Proof:
Follows from the fact that the singleton language $\{a\}$ is accepted by a non-realtime 1-reversal 1-counter machine and the fact that the number of non-realtime k-reversal m-counter machines is finite, depending only on k and m. □

Theorem 5.3. For $m \geq 1$, $m+1$ counters can do more than m counters for stateless non-realtime k-reversal machines.

Proof:
Clearly, any language accepted by a non-realtime k-reversal m-counter machine can be accepted by a k-reversal $(m+1)$-counter machine.

Now let M a non-realtime k-reversal m-counter machine accepting the maximal language $\{a^n\}$ (for some n). Such a languages exists by the above lemma. Let the counters of M be $C_1, \ldots, C_m$. We will construct a non-realtime k-reversal $(m+1)$-counter machine M' accepting $\{a^{n+1}\}$. It would then follow that $m+1$ counters are better than m counters.

M' will have counters $C_1, \ldots, C_m, C_{m+1}$ and its rules are defined as follows:

1. If $(¢, s_1, \ldots, s_m) \to (0, e_1, \ldots, e_m)$ is in M, then
   $(¢, s_1, \ldots, s_m, 0) \to (0, e_1, \ldots, e_m, 0)$ is in M'.

2. If $(¢, s_1, \ldots, s_m) \to (1, e_1, \ldots, e_m)$ is in M, then
   $(¢, s_1, \ldots, s_m, 0) \to (1, e_1, \ldots, e_m, 1)$ is in M'.

3. If $(a, s_1, \ldots, s_m) \to (d, e_1, \ldots, e_m)$ is in M, then $(a, s_1, \ldots, s_m, 1) \to (1, 0, \ldots, 0, -1)$ is in M'.

4. If $(x, s_1, \ldots, s_m) \to (d, e_1, \ldots, e_m)$ is in M, then $(x, s_1, \ldots, s_m, 0) \to (d, e_1, \ldots, e_m, 0)$ is in M' for $x \in \{a, \$\}$.

$\square$

Note that the above result and proof hold for the realtime case. In fact, in the construction, case 1 does not apply; in case 2, $s_1 = \cdots = s_m = 0$, and in case 4, $x = a$ and $d = 1$. As in the realtime case, the following result can be shown [8]:

Theorem 5.4. For any fixed m and $k < 2^{\frac{m}{2}-1}/m$, there is a language accepted by a stateless $(k+1)$-reversal m-counter machine which is not accepted by any stateless k-reversal m-counter machine.

6. Closure Properties

Theorem 6.1. The class of languages accepted by stateless deterministic non-realtime k-reversal multicounter machines is closed under intersection, union, and complementation.

Proof:
Let M_1 and M_2 be two such machines.

Intersection:
Let M_1 and M_2 have m and n counters, respectively. We construct a machine M which simulates these machines in parallel. M has $m+n$ counters to simulate the counters of M_1 and M_2, using the following rules:

1. If $(x, s_1, \ldots, s_m) \to (d, e_1, \ldots, e_m)$ in M_1 and $(x, s'_1, \ldots, s'_n) \to (d, e'_1, \ldots, e'_n)$ in M_2, then $(x, s_1, \ldots, s_m, s'_1, \ldots, s'_n) \to (x, e_1, \ldots, e_m, e'_1, \ldots, e'_n)$ in M.

2. If $(x, s_1, \ldots, s_m) \to (0, e_1, \ldots, e_m)$ in M_1 and $(x, s'_1, \ldots, s'_n) \to (1, e'_1, \ldots, e'_n)$ in M_2, then $(x, s_1, \ldots, s_m, s'_1, \ldots, s'_n) \to (x, e_1, \ldots, e_m, 0, \ldots, 0)$ in M.

3. If $(x, s_1, \ldots, s_m) \to (1, e_1, \ldots, e_m)$ in M_1 and $(x, s'_1, \ldots, s'_n) \to (0, e'_1, \ldots, e'_n)$ in M_2, then $(x, s_1, \ldots, s_m, s'_1, \ldots, s'_n) \to (x, 0, \ldots, 0, e'_1, \ldots, e'_n)$ in M.

Complementation and Union:
Given M_1, we construct a machine M which accepts the complement of the language accepted by M_1. In the addition to the m counters of M_1, M uses a new counter T. Before the simulation, M sets T to 1. Then M simulates M_1. If M_1 does not accept the input, either by getting stuck at some point on the input or reaching \$ and not able to zero all the counters, M decrements all the counters to zero and sets T to 0. Closure under union follows, since the class of languages is closed under intersection. $\square$

We believe a "pumping lemma" type result for stateless deterministic non-realtime reversal-bounded multicounter machines can be shown, but we have not quite verified all the details. Such a lemma would be of the form:

Suppose L is the language accepted by a stateless deterministic non-realtime reversal-bounded m-counter machine over a unary alphabet. If L is infinite, then there exists some $n_0 \geq 0$ such that for $n \geq n_0$, $a^n \in L$ implies $a^{n+1} \in L$.

It would then follow that the language $L = \{a^{2n} \mid n \geq 1\}$ cannot be accepted by any stateless deterministic non-realtime reversal-bounded multicounter machine.

7. Equivalence to Stateless Multihead Automata

It turns out that stateless non-realtime reversal-bounded multicounter machines over a unary alphabet are equivalent to stateless multihead automata.

A stateless m-head machine (over unary input) M operates on an input $¢a^n\$$, where $¢$ and \$ are the left and right end markers. The initial configuration is when all m heads are on the left end marker $¢$, and the accepting configuration is when all m heads reach the right end marker \$, which we assume is a halting configuration. The moves are defined by a set of rules of the form: $(\ell_1, .., \ell_m) \rightarrow (d_1, \ldots, d_m)$ where ℓ_i is the symbol under head i (can be $¢, a, \$$), d_i = 0 or 1 (direction of move of head i : no move or move right one cell). Note that since the machine is deterministic, no two rules can have the same left hand sides. Also there is no rule with left hand side $(\$, \ldots, \$)$.

Lemma 7.1. Any stateless multihead automaton M can be converted to an equivalent stateless non-realtime 1-reversal multicounter machine M'.

Proof:
Let the heads of M be $H_1, \ldots, H_m$. M' will have an input head and $2m$ counters $C_1, \ldots, C_m, T_1, \ldots, T_m$, which are initially zero. When given $¢a^n\$$, M' reads the input while simultaneously incrementing the m counters $C_1, \ldots, C_m$. When the input head reaches the right end marker \$, each C_i will have value $n + 1$. Then M' simulates M. The input head of M' remains on \$ during the simulation. Counter C_i simulates the actions of head H_i. Moving head H_i one cell to the right is simulated by decrementing C_i. Note that at the start of the simulation, $T_1, \ldots, T_m$ are zero. This corresponds to the configuration when all the heads of M are on the left end marker $¢$. When C_i is first decremented (corresponding to H_i moving right of $¢$), T_i is set to 1. When the counters $C_1, \ldots, C_m$ become zero (corresponding to all heads $H_1, \ldots, H_m$ reaching \$), the T_i's are decremented and the input is accepted. □

Lemma 7.2. Any stateless non-realtime 1-reversal multicounter machine can be converted to an equivalent stateless multihead automaton M'.

Proof:
(Sketch.) First consider a stateless non-realtime 1-reversal 1-counter machine M with input head H and counter C. We construct a stateless multihead automaton M' equivalent to M. M' will have 6 heads $H_1, H_2, C_1, C_2, T_1, T_2$. Initially, all heads of M' are on $¢$. M' simulates M as follows:

1. Heads H_1 and C_1 simulate head H and counter C of M, respectively, where "incrementing" C corresponds to "moving" C_1 to the right on the input.

2. When C decrements, M' moves T_1 right to the next symbol (indicating a new situation).

3. M' restarts the simulation of M (from the beginning) but now using H_2 and C_2 (to simulate H and C) and at same time C_1 is moved along with C_2 when the latter is incrementing.

4. When C_2 decrements, M' moves T_2 right to the next symbol (indicating yet again another situation) and suspend the simulation, i.e., H_2 and C_2 are not moved. At this time, if C_2 is in position d, then C_1 is in position $2d$, i.e., the "distance" between these heads is d.

5. C_1 and C_2 are moved to the right in parallel until C_1 reaches \$. Note that the "distance" between C_1 (which is now on \$) and C_2 is still d.

6. Then M' uses H_2 and C_2 to resume the simulation of M, but C_2 now simulates the decreasing phase of counter C of M by moving right on the input. C_2 reaching \$ indicates that counter C of M has value 0.

M' accepts the language accepted by M. When M has several 1-reversal counters, the construction of M' above can be generalized. We omit the details. □

From Theorem 5.1 and the above lemmas, we have the following characterization:

Theorem 7.3. A language L over a unary alphabet is accepted by a stateless non-realtime reversal-bounded multicounter machine if and only if it can be accepted by a stateless multihead automaton.

8. Nondeterministic Machines and Semilinear Sets

Recall that in a nondeterministic machine some rules can have the same left hand sides. In this section, we characterize bounded languages accepted by stateless nondeterministic reversal-bounded non-realtime multicounter machines in terms of semilinear sets.

A language L is *bounded* if there are distinct symbols $a_1, \ldots, a_r$ such that $L \subseteq a_1^* \cdots a_r^*$. The Parikh map of L, $\psi(L)$, is defined to be the set of r-tuples of nonnegative integers $\{(i_1, \ldots, i_r) \mid a^{i_1} \cdots a^{i_r} \in L\}$.

Let $\mathbb{N}$ be the set of nonnegative integers and r be a positive integer. A subset Q of $\mathbb{N}^r$ is a *linear set* if there exist vectors $v_0, v_1, \ldots, v_t$ in $\mathbb{N}^r$ such that $Q = \{v \mid v = v_0 + a_1 v_1 + \cdots + a_t v_t,\ a_i \in \mathbb{N}\}$. A set $Q \subseteq \mathbb{N}^r$ is *semilinear* if it is a finite union of linear sets.

It is known that $L \subseteq a_1^* \cdots a_r^*$ is accepted by a nondeterministic non-realtime reversal-bounded multicounter machine with states if and only if $\psi(L)$ is semilinear. This result also holds for stateless machines. To avoid introducing additional notation, we illustrate the ideas with an example below.

Consider the linear set $Q = \{(2,1) + x(2,3) + y(1,0) \mid x, y \geq 0\}$. The bounded language corresponding to this set is $L = \{a^{2x+y+2} b^{3x+1} \mid x, y \geq 0\}$. We will construct a stateless nondeterministic non-realtime 1-reversal multicounter machine M accepting L. In the construction, we use some special types of counters, which we call *switches*. A switch starts at zero at the beginning of the computation (when the input head is on ¢), then it is incremented to 1 at some point during the computation, and finally set back to zero before acceptance.

M has counters $A_1, A_2, A_3, B_1, B_2, B_3$ and other counters used as switches. Given input ¢w\$, we may assume that $w = a^m b^n$ for some m, n; otherwise, we can use a switch counter to confirm that a b cannot be followed by an a.

On input $¢a^m b^n\$$, M operates as follows: while the input is on ¢, M increments A_1, A_2, B_1, B_2, B_3 simultaneously, x times, where x is chosen nondeterministically, after which (using some switches) increments counter A_3, y times, where y is chosen nondeterministically. Then it checks that $m = A_1 + A_2 + A_3 + 2$. This is done by first reading 2 a's (again using some switches). Then it reads the rest of the a-segment while decrementing A_1 until it becomes zero, then decrementing A_2 until it too becomes zero, and decrementing A_3 until it becomes zero. M's head will be on the first b if and only if $m = A_1 + A_2 + A_3 + 2$. Similarly, by reading the b-segment, M can check that $m = B_1 + B_2 + B_3 + 1$, and this holds if and only if the head reaches \$ when counter B_3 becomes zero.

If Q is a semilinear set we can construct a machine for each linear set and then combine these machines into one machine that nondeterministically selects one of the machines to simulate. (We will need to use additional switches for this.)

One can formalize the discussion above to prove the "if" part of the next result. The "only if" part follows from the fact that it holds for machines with states [7].

Theorem 8.1. $L \subseteq a_1^* \cdots a_r^*$ can be accepted by a stateless nondeterministic non-realtime reversal-bounded multicounter machine if and only if $\psi(L)$ is semilinear.

Corollary 8.2. $L \subseteq a_1^* \cdots a_r^*$ is accepted by a stateless nondeterministic non-realtime reversal-bounded multicounter machine with states if and only if it can be accepted by a stateless nondeterministic non-realtime reversal-bounded multicounter machine.

9. Examples for the Unbounded Reversal Case

The examples in this section are from [8], although the details of their construction are not provided. We begin with a stateless non-realtime counter machine that accepts the language $L = \{a^{2^i} \mid i \geq 0\}$. What is interesting is that this can be accepted by a machine with only 4 counters.

Here the input is $¢a_1 a_2 \cdots a_n\$$ with the read head initially on the left end marker ¢ and all m counters zero. The head moves to the right at each step. Depending on the symbol under the head and the signs of the counters, a counter is decremented (if positive), incremented, or left the same. Once the head reaches the \$ sign, further moves are possible, depending on the signs of the counters only. The machine accepts if the counters all become zero. Since further moves are allowed after the head reaches the \$, the machine is non-realtime.

Let us consider the unary alphabet. The inputs are of the form $¢a^n\$$. We can show the following:

Proposition 9.1. The language $L = \{a^{2^i} \mid i \geq 0\}$ is accepted by a stateless non-realtime 4-counter machine.

It can also be proved that by adding a fifth counter, the construction for the proof of the above proposition can be modified to accept the language

$$L = \{a^n \mid n \text{ is a tower of 2's}\} .$$

Furthermore the singleton language $L = \{a^n \mid n = m$ levels of 2's $\}$ can be accepted by a machine with $\log m + 5$ counters. In these examples, the counter machines are non-realtime. Interestingly, one can show that similar results can be obtained for realtime machines.

The descriptions of these machines make use of the *sign-vectors* introduced in [3] to construct their explicit moves. They also make use of types of *modules,* that are constructed by using the patterns of the sign-vectors. These modules are *called* repeatedly with different counter values and *return* other counter values. Calls and returns are controlled by the patterns of the sign-vectors of the counters.

The constructions for the examples given here are not provided in this survey due to space limitations. They can be obtained from the authors.

References

[1] Ö. Eğecioğlu, L. Hegedüs, and B. Nagy. Stateless multicounter $5' \to 3'$ Watson-Crick Automata. In: Fifth International Conference on Bio-Inspired Computing: Theories and Applications, IEEE Press, pp. 1599–1606 (2010).

[2] Ö. Eğecioğlu, L. Hegedüs, and B. Nagy. Hierarchies of Stateless Multicounter $5' \to 3'$ Watson-Crick Automata Languages, *Fundamenta Informaticae*, Vol. 110, No. 1-4, pp. 111–123 (2011).

[3] Ö. Eğecioğlu and O. H. Ibarra. On stateless multicounter machines. Ambos-Spies, K., Löwe, B., Merkle, W. (eds.) CiE 2009, LNCS, Vol. 5635, pp. 178–187 (2009).

[4] P. Frisco and O. H. Ibarra. On stateless multihead finite automata and multihead pushdown automata. DLT 2009, LNCS, Vol. 5583, pp. 240–251 (2009).

[5] R. Freund, Gh. Păun, G. Rozenberg, and A. Salomaa. Watson-Crick Finite Automata. In: Third Annual DIMACS Symposium on DNA Based Computers, Philadelphia, pp. 535–546 (1994).

[6] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Series in Computer Science. Addison-Wesley, Reading, MA, 1979.

[7] O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *J. Assoc. for Computing Machinery*, 25, pp. 116–133 (1978).

[8] O. H. Ibarra and Ö. Eğecioğlu. Hierarchies and characterizations of stateless multicounter machines. In: Ngo, H.G. (ed.), COCOON 2009, LNCS, Vol. 5609, pp. 408–417 (2009).

[9] O. H. Ibarra, J. Karhumaki, and A. Okhotin. On stateless multihead automata: hierarchies and the emptiness problem. *Proceedings of 8th Latin American Symposium*, LNCS Vol. 4957, pp. 94–105 (2008).

[10] A. J. Korenjak and J. E. Hopcroft. Simple deterministic languages. *Proceedings of IEEE 7th Annu. Symp. on Switching and Automata Theory*, pp. 36–46 (1966).

[11] M. Kutrib, H. Messerschmidt, and Friedrich Otto. On stateless two-pushdown automata and restarting automata. *Pre-Proceedings of the 8th Automata and Formal Languages*, May (2008).

[12] B. Nagy, L. Hegedüs, and Ö. Eğecioğlu. Hierarchy Results On Stateless Multicounter $5' \to 3'$ Watson-Crick Automata, IWANN 2011, LNCS, Vol. 6691, pp. 465–472 (2011).

[13] Gh. Păun. Computing with Membranes. *Journal of Computer and System Sciences*, 61, 1, pp. 108–143 (2000) (and Turku Center for Computer Science-TUCS Report 208, November 1998, `www.tucs.fi`).

[14] Gh. Păun. *Computing with Membranes: An Introduction*, Springer, Berlin, 2002.

[15] L. Yang, Z. Dang, and O. H. Ibarra. On stateless automata and P systems. *Pre-Proceedings of Workshop on Automata for Cellular and Molecular Computing*, August (2007).