Automata with Modulo Counters and Nondeterministic Counter Bounds

Daniel Reidenbach and Markus L. Schmid [⋆]

Department of Computer Science, Loughborough University,
Loughborough, Leicestershire, LE11 3TU, United Kingdom
`{D.Reidenbach,M.Schmid}@lboro.ac.uk`

Abstract. We introduce and investigate Nondeterministically Bounded Modulo Counter Automata (NBMCA), which are two-way one-head automata that comprise a constant number of modulo counters, where the counter bounds are nondeterministically guessed, and this is the only element of nondeterminism. NBMCA are tailored to recognising those languages that are characterised by the existence of a specific factorisation of their words, e. g., pattern languages. In this work, we subject NBMCA to a theoretically sound analysis.

Keywords: Multi-head automata, Counter automata, Modulo counters, Stateless automata, Restricted nondeterminism

1 Introduction

In the present paper we introduce and study a novel automata model, the Nondeterministically Bounded Modulo Counter Automata (NBMCA for short), which comprise several two-way input heads and a number of counters. These NBMCA are suitable algorithmic tools for recognising those languages that are characterised by the existence of a specific factorisation of their words, e. g., pattern languages, and are a generalisation of the Janus automata that have been introduced and applied in [11] in order to investigate the membership problem for pattern languages. In [11], NBMCA with exactly two input heads are used. In the present work we focus on NBMCA with only one head, since we can easily simulate several input heads by just a single one. For every counter of an NBMCA an individual counter bound is provided, and every counter can only be incremented and counts modulo its counter bound. The current counter values and counter bounds are hidden from the transition function, which can only check whether a counter has reached its bound. By performing a reset on a counter, the automaton nondeterministically guesses a new counter bound between 0 and $|w|$, where w is the input word. This guessing of counter bounds is the only possible nondeterministic step of NBMCA, and the transition function is defined completely deterministically. We can interpret the counter bounds as

[⋆] Corresponding author.

positions of the input and, by means of the counter values, the input head can be moved to these positions.

Two aspects of this approach seem to be particularly worth studying. Firstly, all additional resources the automaton is equipped with, namely the counters, are tailored to storing positions in the input word. We can observe that this aspect is not really new; in fact, the idea of separating the mechanisms of storing positions from the functionality of actually processing the input is formalised in the models of partially blind multi-head automata (see, e. g., Ibarra and Ravikumar [7]), Pebble Automata (see, e. g., Chang et al. [1]) and automata with sensing heads (see, e. g., Petersen [10]). Given this similarity between NBMCA and established automata models regarding their emphasis on storing positions in the input word, there is still one difference: the counters of NBMCA are quite limited in their ability to change the positions they represent, since their values can merely be incremented, and their bounds are guessed. The question arises whether or not, for automata using counters as additional resources, their ability to count in both directions is essential with respect to the expressive power.

The second aspect is that the nondeterminism of NBMCA, which merely allows positions in the input word to be guessed, differs quite substantially from the common nondeterminism of automata, which provides explicit computational alternatives. Nevertheless, automata often use their nondeterminism to actually guess a certain position of the input. For example, a pushdown automaton that recognises $\{ww^R \mid w \in \Sigma^*\}$ needs to perform an unbounded number of guesses even though only one specific position, namely the middle one, of the input needs to be found. Despite this observation, the nondeterminism of NBMCA might be weaker, as it seems to *solely* refer to positions in the input. Hence, we also investigate the question of whether or not it is essential that the nondeterminism is explicitly provided by a nondeterministic transition function in order to exploit it to the full extent, in terms of expressive power.

In order to understand the character of these novel, and seemingly limited, resources NBMCA can use, the present paper compares the expressive power of these automata to that of the well-established, and seemingly less restricted, models of multi-head and counter automata. Furthermore, we study some basic decision problems for NBMCA as well as stateless versions of NBMCA, with and without restricted nondeterminism.

Note that, due to space constraints, all proofs have been omitted.

2 Definitions

Let $\mathbb{N}$ denote the set of all positive integers and let $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. The symbols $\subseteq$ and $\subset$ refer to subset and proper subset relation, respectively. For an arbitrary alphabet Σ, a *word* (*over* Σ) is a finite sequence of symbols from Σ, and ε stands for the *empty word*. The symbol Σ^+ denotes the set of all nonempty words over Σ, and $\Sigma^* := \Sigma^+ \cup \{\varepsilon\}$. For the *concatenation* of two words u, v we write $u \cdot v$ or simply uv, and u^k denotes the k-fold concatenation of u. The notation $|K|$ stands for the size of a set K or the length of a word K.

For an arbitrary class of automata, such as the set DFA of deterministic finite automata, the expression “a DFA” refers to any automaton from DFA. For an arbitrary automaton M, $L(M)$ denotes the set of all words accepted by M and, for an arbitrary class A of automata, let $\mathcal{L}(A) := \{L(M) \mid M \in A\}$. For every $k \in \mathbb{N}$ let 1DFA(k), 2DFA(k), 1NFA(k) and 2NFA(k) denote the class of *deterministic one-way*, *deterministic two-way*, *nondeterministic one-way* and *nondeterministic two-way automata* with k input heads, respectively. For a comprehensive survey on multi-head automata the reader is referred to Holzer et al. [3] and to the references therein.

Next, we define the central automata model of this paper. A *Nondeterministically Bounded Modulo Counter Automaton*, NBMCA(k) for short, is a two-way one-head automaton with k counters. More precisely, it is a tuple $M := (k, Q, \Sigma, \delta, q_0, F)$, where $k \in \mathbb{N}$ is the number of *counters*, Q is a finite nonempty set of *states*, Σ is a finite nonempty alphabet of *input symbols*, $q_0 \in Q$ is the *initial state* and $F \subseteq Q$ is the set of *accepting states*. The mapping $\delta : Q \times \Sigma \times \{\mathtt{t_0}, \mathtt{t_1}\}^k \rightarrow Q \times \{-1, 0, 1\} \times \{0, 1, \mathtt{r}\}^k$ is called the *transition function*. Instead of writing transitions in the form $\delta(C) = S$, we use the notation $C \rightarrow_\delta S$. If δ is obvious from the context, we simply write $C \rightarrow S$. An input to M is any word of the form ¢w\$, where $w \in \Sigma^*$ and the symbols ¢, \$ (referred to as *left* and *right endmarker*, respectively) are not in Σ. Let $(p, b, s_1, \ldots, s_k) \rightarrow_\delta (q, r, d_1, \ldots, d_k)$. We call the element b the *scanned input symbol* and r the *input head movement*. For each $j \in \{1, 2, \ldots, k\}$, the element $s_j \in \{\mathtt{t_0}, \mathtt{t_1}\}$ is the *counter message of counter* j, and d_j is called the *counter instruction for counter* j. The transition function δ of an NBMCA(k) determines whether the input heads are moved to the left ($r_i = -1$), to the right ($r_i = 1$) or left unchanged ($r_i = 0$), and whether the counters are incremented ($d_j = 1$), left unchanged ($d_j = 0$) or reset ($d_j = \mathtt{r}$). In case of a reset, the counter value is set to 0 and a new counter bound is nondeterministically guessed between 0 and the current input length. Hence, every counter is bounded, but these bounds are chosen in a nondeterministic way. In order to define the language accepted by an NBMCA, we need to define the concept of an NBMCA computation.

Let M be an NBMCA and $w := a_1 \cdot a_2 \cdot \cdots \cdot a_n$, $a_i \in \Sigma$, $1 \leq i \leq n$. A *configuration of* M *(on input* w*)* is an element of $\widehat{C}_M := \{[q, h, (c_1, C_1), \ldots, (c_k, C_k)] \mid q \in Q, 0 \leq h \leq n+1, 0 \leq c_i \leq C_i \leq n, 1 \leq i \leq k\}$. The pair (c_i, C_i), $1 \leq i \leq k$, describes the current configuration of the i^{th} counter, where c_i is the *counter value* and C_i the *counter bound*. The element h is called the *input head position*.

An *atomic move* of M is denoted by the relation $\vdash_{M,w}$ over the set of configurations. Let $(p, b, s_1, \ldots, s_k) \rightarrow_\delta (q, r, d_1, \ldots, d_k)$. Then, for all c_i, C_i, $1 \leq i \leq k$, where $c_i < C_i$ if $s_i = \mathtt{t_0}$ and $c_i = C_i$ if $s_i = \mathtt{t_1}$, and for every h, $0 \leq h \leq n+1$, with $a_h = b$, we define $[p, h, (c_1, C_1), \ldots, (c_k, C_k)] \vdash_{M,w} [q, h', (c'_1, C'_1), \ldots, (c'_k, C'_k)]$. Here, the elements h' and c'_j, C'_j, $1 \leq j \leq k$, are defined in the following way. $h' := h + r$ if $0 \leq h + r \leq n+1$ and $h' := h$ otherwise. For each $j \in \{1, \ldots, k\}$, if $d_j = \mathtt{r}$, then $c'_j := 0$ and, for some $m \in \{0, 1, \ldots, n\}$, $C'_j := m$. If, on the other hand, $d_j \neq \mathtt{r}$, then $C'_j := C_j$ and $c'_j := c_j + d_j \bmod (C_j + 1)$.

In order to describe a *sequence of (atomic) moves of M (on input w)* we use the reflexive and transitive closure of the relation $\vdash_{M,w}$, denoted by $\vdash^*_{M,w}$. M accepts the word w if and only if $\widehat{c}_0 \vdash^*_{M,w} \widehat{c}_f$, where $\widehat{c}_0 := [q_0, 0, (0, C_1), \ldots, (0, C_k)]$ for some $C_i \in \{0, 1, \ldots, |w|\}$, $1 \leq i \leq k$, is an *initial configuration*, and $\widehat{c}_f := [q_f, h, (c_1, C_1), \ldots (c_k, C_k)]$ for some $q_f \in F$, $0 \leq h \leq n+1$ and $0 \leq c_i \leq C_i \leq n$, $1 \leq j \leq k$, is a *final configuration.* In every computation of an NBMCA, the counter bounds are nondeterministically initialised, and the only nondeterministic step an NBMCA is able to perform during the computation consists in guessing a new counter bound for some counter.

3 Expressive Power, Hierarchy and Decidability

An NBMCA can be regarded as a finite state control with additional resources. Thus, it is quite similar to classical nondeterministic multi-head automata. The essential differences between the models are those addressed in Section 1. Hence, in order to gain insights with respect to the question of whether these differences affect the expressive power, we study the problem of simulating classical nondeterministic multi-head automata by NBMCA and vice versa. It is almost obvious that NBMCA can be simulated by nondeterministic multi-head automata as NBMCA can be interpreted as just a further restricted version of them. So multi-head automata intuitively seem to be more powerful.

Theorem 1. *For every $k \in \mathbb{N}$, $\mathcal{L}(\text{NBMCA}(k)) \subseteq \mathcal{L}(\text{2NFA}(2k+1))$.*

The converse question, i. e., whether arbitrary multi-head automata, and particularly their unrestricted nondeterminism, can be simulated by NBMCA, is more interesting. It can be done by using a modulo counter of the NBMCA in order to simulate an input head of the 2NFA(k) in the following way. The modulo counter first guesses $|w|$ as counter bound, which is done by reseting it and checking, by means of the input head, whether or not the guessed bound equals $|w|$, and then the counter value can be used in order to store the position of the input head. Since the counter value cannot be decremented, a decrement has to be performed by $|w| - 1$ increments.

However, for reasons that shall be explained later, we aim for a simulation that is more economic with respect to the usage of modulo counters. More presicely, we want to use a single modulo counter in order to store the positions of two input heads of a 2NFA(k), i. e., the counter value and the counter bound each represent a distinct input head position. A step of the 2NFA(k) is then simulated by first moving the input head of the NBMCA successively to all these positions stored by the counters and record the scanned input symbols in the finite state control. After that, all these positions stored by the counters must be updated according to the transition function of the 2NFA(k). It turns out that this is possible, but, since counter values cannot be decremented and counter bounds cannot be changed directly, the constructions are rather involved and require some technical finesse. Furthermore, we need an additional counter which is also used in order to simulate the possible nondeterministic choices of the 2NFA(k).

Theorem 2. *For every $k \in \mathbb{N}$, $\mathcal{L}(\text{2NFA}(k)) \subseteq \mathcal{L}(\text{NBMCA}(\lceil \frac{k}{2} \rceil + 1))$.*

The above results show that neither the restrictions on the counters of NBMCA nor the special nondeterminism constitute a restriction on the expressive power. Thus, NBMCA can be used whenever classical multi-head automata can be applied, but due to their specific counters and nondeterminism they are particularly suitable algorithmic tools for recognising those languages that are characterised by the existence of a certain factorisation for their words, such as pattern languages (see [11]).

The tight use of the modulo counters in the previous simulation turns out to be worth the effort, as it allows us to prove a hierarchy result on the class NBMCA by applying a classical hierarchy result concerning multi-head automata (Monien [9]).

Corollary 1. *For every $k \in \mathbb{N}$, $\mathcal{L}(\text{NBMCA}(k)) \subset \mathcal{L}(\text{NBMCA}(k+2))$.*

Next, we investigate the decidability of the emptiness, infiniteness, universe, equivalence, inclusion and disjointness problem with respect to languages given by NBMCA. From the fact that all these problems are undecidable even for 1DFA(2) (cf., Holzer et al. [3]) and Theorem 2, it follows that all these problems are also undecidable for NBMCA. However, it is a common approach to further restrict automata models with undecidable problems in order to obtain subclasses with decidable problems. One respective option is to require the automata to be reversal bounded (see, e.g., Ibarra [4]). Hence, for all $m_1, m_2, l, k \in \mathbb{N}$, let (m_1, m_2, l)-REV-NBMCA(k) denote the class of NBMCA(k) that perform at most m_1 input head reversals, at most m_2 counter reversals and resets every counter at most l times in every accepting computation (here, input head reversals are defined in the same way as by Ibarra [4], whereas a counter reversal is an increment of the counter in case that it has already reached its counter bound). We can directly apply a result by Ibarra [4] about reversal-bounded counter machines in order to obtain the following:

Theorem 3. *The emptiness, infiniteness and disjointness problem for the class (m_1, m_2, l)-REV-NBMCA are decidable.*

Next, we investigate the decidability properties of (m, ∞, l)-REV-NBMCA, i. e., the number of counter reversals is not bounded anymore. This question is motivated as follows. Ibarra [4] shows for counter machines that if only the reversals of the input head are bounded and counter reversals are unrestricted, then the typical decision problems remain undecidable. However, while a counter reversal of a counter machine can happen anytime in the computation and for any possible counter value, a counter reversal of an NBMCA strongly depends on the current counter bound, i. e., as long as a counter is not reset, all the counter reversals of that counter happen at exactly the same counter value. Hence, the modulo counters of (m, ∞, l)-REV-NBMCA are still restricted, since the number of resets is bounded, and the question arises whether or not this restriction is strong enough to maintain positive decidability results. The following answers this question in the negative, even for small m and k, and no counter resets:

Theorem 4. *The emptiness, infiniteness, universe, equivalence, inclusion and disjointness problems are undecidable for* $(3, \infty, 0)$-REV-NBMCA(3).

4 NBMCA without States

In this section, we consider NBMCA without states. Stateless versions of automata have recently been introduced by Yang et al. [12], where they are compared to P-Systems. Ibarra et al. [6] and Frisco and Ibarra [2] investigate stateless multi-head automata, whereas Ibarra and Eğecioğlu [5] consider stateless counter machines. Kutrib et al. [8] study stateless restarting automata.

A *stateless* NBMCA (SL-NBMCA for short) can be regarded as an NBMCA with only one internal state that is never changed. Hence, the component referring to the state is removed from the transition function and transitions do not depend anymore on the state. As a result, the acceptance of inputs by accepting state is not possible anymore. So for stateless NBMCA we define the input to be accepted by a special accepting transition, i. e., the transition that does not change the configuration of the automaton anymore. On the other hand, if the automaton enters a configuration for which no transition is defined, then the input is rejected and the same happens if an infinite loop is entered. For example, $(b, s_1, \ldots, s_k) \rightarrow (r, d_1, \ldots, d_k)$ is a possible transition for an SL-NBMCA(k) and $(b, s_1, \ldots, s_k) \rightarrow (0, 0, 0, \ldots, 0)$ is an accepting transition. An SL-NBMCA(k) can be given as a tuple (k, Σ, δ) comprising the number of counters, the input alphabet and the transition function. We now consider an example for the languages $S_k := \{a^k, \varepsilon\}$, $k \in \mathbb{N}$. The following SL-NBMCA(5) recognises exactly S_3.

Definition 1. *Let* $M_{S_3} := (5, \{a\}, \delta) \in$ SL-NBMCA(5)*, where* δ *is defined by* $(\cent, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}) \rightarrow_\delta (\mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{r})$, $(\mathtt{a}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_0}) \rightarrow_\delta (-\mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{1}, \mathtt{1})$, $(\cent, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_1}) \rightarrow_\delta (\mathtt{1}, \mathtt{1}, \mathtt{0}, \mathtt{0}, \mathtt{0}, \mathtt{0})$, $(\mathtt{a}, \mathtt{t_1}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_1}) \rightarrow_\delta (\mathtt{1}, \mathtt{0}, \mathtt{1}, \mathtt{0}, \mathtt{0}, \mathtt{0})$, $(\mathtt{a}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_0}, \mathtt{t_0}, \mathtt{t_1}) \rightarrow_\delta (\mathtt{1}, \mathtt{0}, \mathtt{0}, \mathtt{1}, \mathtt{0}, \mathtt{0})$, $(\mathtt{a}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_0}, \mathtt{t_1}) \rightarrow_\delta (\mathtt{1}, \mathtt{0}, \mathtt{0}, \mathtt{0}, \mathtt{1}, \mathtt{1})$, $(\$, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_1}, \mathtt{t_0}) \rightarrow_\delta (\mathtt{0}, \mathtt{0}, \mathtt{0}, \mathtt{0}, \mathtt{0}, \mathtt{0})$.

The question of whether or not states are really necessary for a model, i. e., whether it is possible to simulate automata by their stateless counterparts, is probably the most fundamental question about stateless automata. Regarding SL-NBMCA, we can observe that every NBMCA with states can be turned into an equivalent one without states. Hence, the loss of the finite state control does not lead to a reduced expressive power of the model.

Theorem 5. *For every* $M \in$ NBMCA(k)*,* $k \in \mathbb{N}$*, with a set of states* Q*, there exists an* $M' \in$ SL-NBMCA$(k + \lceil \log(|Q| + 1) \rceil + 2)$ *with* $L(M) = L(M')$.

For the simulation of NBMCA by SL-NBMCA as well as for the automaton M_{S_3} (see Definition 1), it is vital that certain counters have a counter bound of 1. Due to the lack of states, this need for counters to be initialised with a counter bound of 1 involves considerable technical challenges.

Next, we use the model of stateless NBMCA in order to investigate a more general question in automata theory regarding limited nondeterminism. Usually, the nondeterminism is mainly controlled by the finite state control, i. e., certain states allow nondeterministic steps whereas others enforce a deterministic transition. Hence, nondeterminism can be switched on and off and, thus, it is a resource the automaton may use, but it is not forced to. These considerations suggest that the finite state control plays an important role regarding restricted nondeterminism and it is not obvious what consequences, in this regard, an abolishment of the finite state control may have. In the following we try to answer this question by employing SL-NBMCA. As shown in the previous section, if we allow an unbounded number of modulo counters, a finite state control can be simulated and, thus, nondeterminism can be controlled in the usual way. Therefore we consider SL-NBMCA(1) and, furthermore, we assume the input head to operate in a one-way manner. In order to restrict the nondeterminism of the model, we simply limit the number of possible resets for the modulo counter. More precisely, in any computation the first k applications of a reset operation reset the counter in accordance with the definition, whereas every further application of a reset is simply ignored. We shall refer to this model by 1SL-NBMCA$_k$(1), where k stands for the number of possible resets.

This way of restricting automata is unusual compared to the common restrictions that are found in the literature. This can be illustrated by considering input head reversal bounded automata as an example (see, e. g., Ibarra [4]). An input head reversal bounded automaton is an automaton that can recognise each word of a language in such a way that the number of input head reversals is bounded. There is no need to require the input head reversals to be bounded in the non-accepting computations as well, as this does not constitute a further restriction. This is due to the fact that we can always use the finite state control to count the number of input head reversals in order to interrupt a computation in a non-accepting state as soon as the bound of input head reversals is exceeded. However, regarding stateless automata this is not necessarily possible anymore, and it seems that it is a difference whether a restriction is defined for all possible computations or only for the accepting ones. Our definition of bounded resets introduced above avoids these problems by slightly changing the model itself, i. e., in every computation it loses the ability to reset the counter after a number of resets. For every $k \in \mathbb{N}$, there are languages that require at least k resets:

Theorem 6. *For every $k \in \mathbb{N}$, there exists a language $L \in \mathcal{L}(\text{1SL-NBMCA}_k(1))$ with $L \notin \mathcal{L}(\text{1SL-NBMCA}_{k'}(1))$ for every $k' \in \mathbb{N}$, $k' < k$.*

Moreover, by applying Theorem 6 and a simple set-theoretic reasoning, we can show that there are languages that can be recognised by a 1SL-NBMCA$_k$(1), but cannot be recognised by any 1SL-NBMCA$_{k+1}$(1).

Theorem 7. *There exist infinitely many $k \in \mathbb{N}$ such that $\mathcal{L}(\text{1SL-NBMCA}_k(1))$ and $\mathcal{L}(\text{1SL-NBMCA}_{k+1}(1))$ are incomparable.*

The above results yield the following conclusions: For every $k \in \mathbb{N}$, there is a language that can be recognised by a 1SL-NBMCA(1) with k, but not with $k-1$

resets. This meets our expectation of nondeterminism being a useful resource enhancing the expressive power of automata. Theorem 7, on the other hand, does not fit with the usual results on restricted nondeterminism, as it shows that expressive power is lost by increasing the nondeterminism, i. e., for infinitely many $k \in \mathbb{N}$, there is a language that can be recognised by a 1SL-NBMCA(1) with k, but not with $k+1$ resets. Considering the strong restrictions of 1SL-NBMCA$_k$(1), it is maybe not surprising that without any states the nondeterminism cannot be controlled anymore and, thus, a result of the sort mentioned above can be obtained. However, proving this behaviour is quite involved and, to the knowledge of the authors, it is the first result in the literature that formally establishes such a connection between finite state control and nondeterminism.

References

1. J. H. Chang, O. H. Ibarra, M. A. Palis, and B. Ravikumar. On pebble automata. *Theoretical Computer Science*, 44:111–121, 1986.
2. P. Frisco and O. H. Ibarra. On stateless multihead finite automata and multihead pushdown automata. In *Proc. 13th International Conference on Developments in Language Theory, DLT 2009*, volume 5583 of *Lecture Notes in Computer Science*, pages 240–251, 2009.
3. M. Holzer, M. Kutrib, and A. Malcher. Complexity of multi-head finite automata: Origins and directions. *Theoretical Computer Science*, 412:83–96, 2011.
4. O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *Journal of the ACM*, 25:116–133, 1978.
5. O. H. Ibarra and Ö. Eğecioğlu. Hierarchies and characterizations of stateless multicounter machines. In *Proc. 15th Annual International Conference on Computing and Combinatorics, COCOON 2009*, volume 5609 of *Lecture Notes in Computer Science*, pages 408–417, 2009.
6. O. H. Ibarra, J. Karhumäki, and A. Okhotin. On stateless multihead automata: Hierarchies and the emptiness problem. *Theoretical Computer Science*, 411:581–593, 2010.
7. O. H. Ibarra and B. Ravikumar. On partially blind multihead finite automata. *Theoretical Computer Science*, 356:190–199, 2006.
8. M. Kutrib, H. Messerschmidt, and F. Otto. On stateless two-pushdown automata and restarting automata. *International Journal of Foundations of Computer Science*, 21:781–798, 2010.
9. B. Monien. Two-way multihead automata over a one-letter alphabet. *RAIRO Informatique Théorique*, 14:67–82, 1980.
10. H. Petersen. Automata with sensing heads. In *Proc. 3rd Israel Symposium on Theory of Computing and Systems*, pages 150 – 157, 1995.
11. D. Reidenbach and M. L. Schmid. A polynomial time match test for large classes of extended regular expressions. In *Proc. 15th International Conference on Implementation and Application of Automata, CIAA 2010*, volume 6482 of *Lecture Notes in Computer Science*, pages 241–250, 2011.
12. L. Yang, Z. Dang, and O. H. Ibarra. On stateless automata and p systems. *International Journal of Foundations of Computer Science*, 19:1259–1276, 2008.