On One Language with Connection to Determinism and Bounded Deleting

Martin Procházka

Abstract: The separation of two properties of formal languages which are studied by means of deleting automata with restart operation (DR-automata) is presented.

A DR-automaton is a device with a finite state control unit and a head with a lookahead window that moves from the left to the right along a finite list of items. The DR-automaton can change the state in its control unit after each move to the right and it can also delete the item scanned by its head moving the head to the right neighbour of the deleted item. The DR-automaton can also execute a restart operation that sets its control unit to the initial state and relocates its head with the lookahead to the beginning of the list.

DR-automata were used to study hierarchies of various classes of languages and for their separations. Separation of determinism and nondeterminism is in great interest. In the separation theorems particular languages play important role. This article is focused on the language $L = \{a^m b^n \mid 0 \leq m \leq n \leq 2 \cdot m\}$. Originaly, this language was considered as a candidate for the separation of classes recognized by deterministic and nondeterministic versions of DR-automata. In this article we show that determinism by itself does not disable recognition of L by DR-automaton. The second feature that characterizes subclasses of DR-automata that cannot recognize L is bounded deleting introduced in this article.

Key Words: DR-automata, separation of classes of languages, determinism, bounded deleting.

Mathematics Subject Classification: 68Q45

1. Introduction

This article is devoted to the separation of two properties of formal languages which are studied by means of deleting automata with restart operation (DR-automata). Motivation for DR-automata comes from natural language analysis and they were introduced in [5].

A DR-automaton is a device with a finite state control unit and a head with a lookahead window that moves from the left to the right along a finite list of items. The DR-automaton can change the state in its control unit after each move to the right and it can also delete the item scanned by its head moving the head to the

Supported by the Grant Agency of the Czech Republic, Grant-No. 201/99/0236

right neighbour of the deleted item. The DR-automaton can also execute a restart operation that sets its control unit to the initial state and relocates its head with the lookahead to the beginning of the list.

DR-automata were used to study hierarchies of various classes of languages and for their separations. In the separation theorems particular languages play important role. The choice of proper separation language can expose the essence of these separations.

In roots of formal language theory concrete languages and their features were in a focus of study. It is evident when reading separation theorems contained in [1] and [2]. Such a language often formalizes some properties of natural languages (like English) and the result that this language does not belong to some class can easily be translated back into natural languages. Natural language analysis was, in fact, the main motivation for formal language theory and, as we have already mentioned, it remains an important motivation for DR-automata, too.

The style of this article resembles the style of separation theorems in articles from the beginnings of formal language theory. Results of this article are presented in the form $L \in \mathcal{L}$ or $L \notin \mathcal{L}'$ (where $\mathcal{L}$, $\mathcal{L}'$ are certain subclasses of languages recognized by DR-automata) instead of the contemporary form $\mathcal{L} \setminus \mathcal{L}' \neq \emptyset$.

This style emphasizes the language we are focused on:

$$L = \{a^m b^n \mid 0 \leq m \leq n \leq 2 \cdot m\}.$$

Originaly, this language was considered as a candidate for the separation of classes recognized by deterministic and nondeterministic versions of DR-automata. In this article we show that determinism by itself does not disable recognition of L by DR-automaton. The second feature that characterizes subclasses of DR-automata that cannot recognize L is bounded deleting introduced in this article.

Let us give a short preview of this article:

DR-automata, their deterministic version, and DR-automata with bounded deleting are defined in section 2.

Section 3 illustrates a relation of L to $CFL$and one subclass of DR-languages. Observations gathered in this section serves as a motivation for finding better separation line between automata that can and that cannot recognize the language L.

Main results of this article are stated in section 4. There it is proved that any DR-automaton both deterministic and with bounded deleting cannot recognize the language L. But if we give up one of these properties L becomes recognizable.

2. Definitions and basic facts

Deleting automata with restart operation (DR-automata) serve as a tool for study of subclasses of context-sensitive languages. They were introduced in [5].

We present definitions informally in the same way as in [5]; the formal technical details could be added in a standard way of automata theory.

A *deleting automaton with a restart operation*, a *DR-automaton* for short, is a device defined as a tuple $M = (Q, \Sigma, k, I, q_0, Q_A)$. It has a finite state control unit (with the state set Q) and one head moving on a finite linear (doubly linked) list of items. The first item always contains a special symbol ¢, the last one another special symbol \$, and each other item contains a symbol from a finite alphabet Σ (not containing ¢, \$).

The head has a lookahead 'window' of fixed length k ($k \geq 0$); it means that besides the current item, M also scans the next k right neighbor items (or simply the end of the word when the distance to \$is less than k).

A *configuration* of M is written as a tuple (u, q, v) where $u \in \{\lambda\} \cup ¢\Sigma^*$ (λ denoting the empty word) is the contents of the working list from the left sentinel till (but not including) the position of the head, $q \in Q$ is the current state and $v \in (¢\Sigma^*\$ \cup \Sigma^*\$)$ is the contents of the working list from the scanned item until the right sentinel. The *initial configuration with the word* $w \in \Sigma^*$ is of the form $(\lambda, q_0, ¢w\$)$ where q_0 is a fixed initial state (and the head scans the left sentinel ¢ with its k right neighbors).

A computation of M is controlled by a finite collection I of *instructions*. An instruction is of the type

$$(q, au) \to (q', MVR) \quad \text{or} \quad (q, au) \to (q', DEL) \quad \text{or} \quad (q, au) \to RST.$$

Such an instruction is applicable when the control unit is in the state $q \in Q$, the head is attached to an item with $a \in \Sigma \cup \{¢,\$\}$ and scans also the lookahead $u \in$ $\in (\Sigma \cup \{\$\})^*$, $|u| \leq k$ (where $|u|$ denotes the length of u); its meaning is in the case of:

MVR – to change the current state to the prescribed state q' and to move the head to the right neighbor item,

DEL – to change the current state to the prescribed state q' and to delete the scanned item while placing the head to the right neighbor of the deleted item (here $a \notin \{¢, \$\}$);

RST – to restart, i.e. to switch to the initial state q_0 and to place the head on the most left item (containing ¢).

We suppose that each state $q \in Q$ is either *nonhalting* – for any (possible) a, u there is at least one instruction with the left-hand side (q, au) – or *halting* – there is no instruction with q in the left-hand side. The set of halting states is composed from the set Q_A of *accepting states* and the set of *rejecting states*. According to the state, we also classify *configurations* as *nonhalting* or *halting (accepting, rejecting)*.

Definition 2.1. (Deterministic DR-automaton) *A DR-automaton is deterministic (det-DR-automaton for short) if there are no different instructions with the same left-hand side.*

In the usual way, we define the (derivation) relations $\vdash_M$, $\vdash_M^*$ on the set of configurations of M. A *word $w \in \Sigma^*$ is accepted by M* if there is a computation which starts in the initial configuration with w and finishes in an accepting configuration, i.e. if $(\lambda, q_0, ¢w\$) \vdash_M^* (w_1, q, w_2)$ where $q \in Q_A$. $L(M) \subseteq \Sigma^*$ denotes the language consisting of all words accepted by M; we say that *M recognizes the language $L(M)$*.

Note that when starting with an initial configuration $(\lambda, q_0, ¢w\$)$, the configuration following a restart (RST) is again an initial configuration $(\lambda, q_0, ¢w'\$)$. We will suppose that the (new) word w' is always strictly shorter than w (something was deleted out of w) – this can be easily ensured at any M by 'remembering' the performing of a DEL-operation.

Under the mentioned condition, any computation of a DR-automaton is finite and proceeds in certain *cycles*: by a cycle we mean the (part of a) computation starting in an initial configuration and finishing in another initial configuration or in a halting configuration while not entering an initial configuration in between. The cycle finishing in a halting configuration is a *halting (accepting* or *rejecting) cycle*.

We write $u \Rightarrow_M v$ to denote the fact that there exists a cycle of M starting in the initial configuration with the word u and finishing in the initial configuration with the word v; the relation $\Rightarrow^*_M$ is then the reflexive and transitive closure of $\Rightarrow_M$ ($u \Rightarrow^*_M v$ means $(\lambda, q_0, ¢u\$) \vdash^*_M (\lambda, q_0, ¢v\$)$).

The next two obvious claims express the basic properties of the relation $\Rightarrow^*_M$.

Claim 2.2. (The error preserving property (for all DR-automata)) *Let M be a DR-automaton, and $u \Rightarrow^*_M v$ for some words u, v. If $u \notin L(M)$, then $v \notin L(M)$.*

Claim 2.3. (The correctness preserving property (for *det*-DR-automata)) *Let M be a deterministic DR-automaton and $u \Rightarrow^*_M v$ for some words u, v. If $u \in L(M)$, then $v \in L(M)$.*

Definition 2.4. (DR-automaton with bounded deleting) *A DR-automaton M is a DR-automaton with bounded deleting (D_cR -automaton) if there is a constant c such that at most c items of the working list are deleted in any cycle of M.*

In the proofs it is often useful to assume a DR-automaton M in the *weak cyclic form* – i.e. any word of $L(M)$ longer than k is not accepted by a one-cycle computation (there is one restart at least). The assumption is justified by the following claim (for proof see [4]).

Claim 2.5. *For any DR-automaton M, with lookahead k, there exists a DR-automaton M', with some lookahead n, $n \geq k$, such that M' is in the weak cyclic form and $L(M) = L(M')$.*

We want to recall a definition of restarting automata as a special case of DR-automata. For this aim, it is technically convenient to consider the following innocent generalization of the basic definition: the form of instructions is generalized to

$$(q, au) \to (q', \omega) \quad \text{or} \quad (q, au) \to \omega, RST$$

where ω is a sequence of at most $|aw|$ (occurrences of) operation symbols MVR and DEL. Besides the state change it prescribes a sequence of moving right and deleting to be performed (and finished by restarting in the second case).

This can be easily simulated by the original DR-automaton. Nevertheless it enables the following definition of the *restarting automaton*, R *-automaton* for short, which was introduced in [3]: just put the restriction that ω *does not* contain DEL

in the instructions $(q, au) \to (q', \omega)$, and it *does* contain DEL in the instructions $(q, au) \to \omega, RST$.

The weak cyclic form claim (2.5) holds for R -automata as well (cf. [4]). Inspecting the proof of this claim we can see that its construction preserves both determinism and bounded deleting.

For brevity, we introduce the following notation. DR denotes the class of all (nondeterministic) deleting automata with a restart operation. D_cR denotes the class of all DR-automata with bounded deleting. Prefix *det-* denotes the deterministic version. For any class $\mathcal{A}$ of automata, $\mathcal{L}(\mathcal{A})$ denotes the class of languages recognizable by automata from $\mathcal{A}$, and an *$\mathcal{A}$-language* is a language from $\mathcal{L}(\mathcal{A})$. CFL denotes the class of context-free languages, $DCFL$ the class of deterministic context-free languages.

3. L, CFL and $\mathcal{L}$(R)

In this section we illustrate a relation of L to $CFL$and one subclass of DR-languages, namely $\mathcal{L}$(R). Context-free grammars have a property that could by considered as a complement to bounded deleting. This property could be called 'bounded inserting' or 'bounded pumping' and it has a close connection to pumping lemma. Therefore, the relation of L to context-free languages is discussed, too.

Most of up to date studied subclasses of DR-automata feature bounded deleting. There are nondeteriministic automata from these subclasses that recognize L in contrast to their deterministic versions. This is also a reason why determinism was viewed as a key feature that disables recognition of L. In section 4 we will see that this is not the case. We have to add some other restriction to determinism, namely bounded deleting.

In this section we prove the statements bellow.

$$L \in CFL$$
$$L \in \mathcal{L}(\ R\ )$$

Assertions that $L \notin \mathcal{L}(det - mon - R) = DCFL$, $L \in \mathcal{L}(mon - R)$ are direct consequences of statements above or could be proved in the same way. For definition of monotonicity (abbreviation *mon*) and its basic properties see [4].

3.1. $L \in CFL$

It is easy to see that $L \in CFL$. Let us consider a grammar with the following rules:

$$S \to aSb \mid aSbb \mid \lambda$$

Let the rule $S \to aSb$ is applied just m-times and the rule $S \to aSbb$ just n-times in the derivation of a word w. Then $w = a^{m+n}b^{m+2\cdot n}$ and $w \in L$.

On the other hand, any word $a^m b^{m+n}$, where $0 \le n \le m$ can be derived from S using the rule $S \to aSb$ $(m-n)$-times and the rule $S \to aSbb$ n-times.

3.2. $L \in \mathcal{L}(\mathrm{R})$

A nondeterministic R -automaton M that recognizes L is based on the same idea as a context-free grammar generating L. Such an automaton accepts words λ, *ab*, *abb*, and *aabb* in one cycle. All other words of length less then 5 are rejected immediately. At least one cycle is executed on all words with a prefix from aa^+bbb. The automaton moves along the prefix of symbols a until *aabbb* is scanned and reduces *aabbb* nondeterministically into either *abb* or *ab*. All other words of length at least 5 are rejected in one cycle.

It remains to check that M recognizes L. It is easy to see that all the words accepted in one cycle (λ, *ab*, *abb*, and *aabb*) are just all the words from L of the length less than 5. Any word $a^m b^m$, where $m \geq 2$ can be reduced by M to the word $a^{m-1}b^{m-1}$ which is from L. Any word $a^m b^{m+n}$, where $m \geq 2$ and $1 \leq n \leq m$ can be reduced by M to $a^{m-1}b^{m-1+n-1}$ which is from L, too.

The word $w = a^m b^n$ can be reduced into either $a^{m-1}b^{n-1}$ or $a^{m-1}b^{n-2}$. If $w \notin L$ (i.e. $m > n$ or $2 \cdot m < n$) then none of the shorter words is in L. Moreover, any word from $aa^+bbb^+a\{a,b\}^*$ contains a subword *ba* after any reduction executed by M so that it remains out of L.

4. Determinism and bounded deleting

We prove that L cannot be recognized by any deterministic DR-automaton with bounded deleting but if one of these features is omitted then there is an automaton that recognizes L:

$$L \in \mathcal{L}(D_cR)$$
$$L \notin \mathcal{L}(det\text{-}D_cR)$$
$$L \in \mathcal{L}(det\text{-}DR)$$

4.1. $L \in \mathcal{L}(\ D_cR\ )$

We know that $L \in \mathcal{L}(\mathrm{R})$. Any R -automaton removes items from the working list within a single RSToperation in one cycle. Therefore a number of deleted items in each cycle is limited by $k+1$, where k is a size of its lookahead. It implies that $\mathcal{L}(\mathrm{R}) \subseteq \mathcal{L}(D_cR)$.

4.2. $L \notin \mathcal{L}(det\text{-}D_cR\ )$

Before we start to prove the statement from the title of this section we will introduce three notions those will help us in the rest of this section. These notions are *a loop*, *a deleting loop*, and *a full-scan form*. A *loop* is a part of computation that starts and ends in the same state and with the same scanned symbols. *A loop length* is a number of operations executed in the course of a loop. A *deleting loop* is a loop during which at least one DEL operation is executed. A DR-automaton is in the *full-scan form* if it restarts only when the right sentinel is scanned by its head. For any DR-automaton an equivalent DR-automaton in the full-scan form can be constructed. The construction is the following: Add a special new state q_{RST} and substitute the execution of RST operation with transition into this new state. In this state an automaton is allowed only to move to the right and restart when the right sentinel is scanned by its head.

Let us turn our attention to the proof of the statement $L \notin \mathcal{L}(det\text{-}D_cR)$. We will suppose that M is *det-*D_cR -automaton recognizing the language L with s states and the lookahead size k and that M is in the full-scan as well as the weak cyclic form. We will show that this assumption leads to a contradiction.

We study a behavior of M on words from $R \subseteq a^*b^*$ such that M executes at least one loop on symbols a and one loop on symbols b during a cycle on any word from R. This is sufficient due to the fact that we can construct a *det-DR*-automaton equivalent to M which behaves exactly like M on words from R, accepts in one cycle all the words from $L \cap \overline{R}$ (which is a finite set), and rejects in one cycle all other words (they constitute a regular set). So, there is a *det-DR*-automaton accepting all the words from $L \cap R$ and rejecting all the words from $R \setminus L$ iff there is a *det-DR*-automaton recognizing L.

On any word w with the number of both symbols a and b greater than $s + k + 1$ the computation of M on w has the following course:

- M starts on w and executes a loopless computation on a prefix of A_1 symbols a. During this loopless computation there remain x_1 symbols a in the working list.
- The loopless computation on symbols a ends by entering the loop. (Otherwise, M is not in the full-scan form.) This loop is not a deleting one, because of the bounded deleting. The length of this loop is ℓ_a.
- M leaves the loop when the first b is scanned. M cannot leave the loop earlier because of determinism. Depending on the state in the moment when the first b is scanned the first time (let's suppose that this state is q_a) M makes two decisions:

  1. How many out of k scanned symbols a it will delete?
  2. How it will continue the computation on symbols b?

  The state q_a can be derived from A_2 – the number of steps executed by M between following two configurations:

  - the last configuration in which M is in the first state of the loop and only symbols a are in the lookahead and
  - the first configuration in which first b appears in the lookahead.

  Therefore, a number of symbols out of last k symbols a that remain in the working list is a function of A_2. This number will be refered as $x_2(A_2)$.
- M continues by a loopless computation on symbols b of the length $B_1(A_2)$ after which $y_1(A_2)$ symbols b remains in the working list.
- The loopless computation on symbols b ends by entering the loop. (Otherwise, M is not in the full-scan form.) This loop of length $\ell_b(A_2)$ deletes no b from the list.
- M leaves the loop when the right sentinel is scanned. M cannot leave the loop earlier because of determinism. The number of steps executed in the course of the last unfinished loop is B_2 and together with A_2 it determines $y_2(A_2, B_2)$ – the number of symbols that remain in the working list out of k last symbols b.

The cycle of M has the following form:

$$a^{A_1+(\ell_a\cdot m+A_2)+k}b^{B_1(A_2)+(\ell_b(A_2)\cdot n+B_2)+k} \Rightarrow$$
$$\Rightarrow a^{x_1+(\ell_a\cdot m+A_2)+x_2(A_2)}b^{y_1(A_2)+(\ell_b(A_2)\cdot n+B_2)+y_2(A_2,B_2)},$$

where $A_2 \in \{1,\dots,\ell_a\}$, $B_2 \in \{1,\dots,\ell_b(A_2)\}$.

The description of the cycle for words with enough symbols a and b can be further simplified using the least common multiple of loop-lengths $\ell_a, \ell_b(1), \cdots, \ell_b(\ell_a)$. When we put

$$\begin{aligned} \ell &= LCM(\ell_a, \ell_b(1), \dots, \ell_b(\ell_a)) \\ R &= k + max(A_1, B_1(1), \dots, B_1(\ell_a)) \end{aligned}$$

the cycle can be describe in the following way:

$$a^{\ell\cdot m+R+A}b^{\ell\cdot n+R+B} \Rightarrow a^{\ell\cdot m+x(A)}b^{\ell\cdot n+y(A,B)},$$

where $x(A) \le R+A$, $y(A,B) \le R+B$ and $A, B \in \{1,\dots,\ell\}$. Values of A_2 and B_2 can be easily reconstructed from values of A and B.

Equivalence (1) below expresses both error and correctness preserving properties.

$$\begin{gathered} \ell\cdot m+R+A \le \ell\cdot n+R+B \le 2\cdot(\ell\cdot m+R+A) \\ \Updownarrow \\ \ell\cdot m+x(A) \le \ell\cdot n+y(A,B) \le 2\cdot(\ell\cdot m+x(A)) \end{gathered} \tag{1}$$

Let us consider a special case of a cycle of automaton M when

$$\begin{aligned} R+A &= \ell\cdot z \\ R+B &= \ell\cdot z \\ x(A) &= \ell\cdot z - \Delta_a \\ y(A,B) &= \ell\cdot z - \Delta_b \end{aligned}$$

for some z and $\Delta_a, \Delta_b \ge 0$. Because both A and B ranges between 1 and ℓ it is obvious that there are values of these variables such that $R+A$ and $R+B$ are both the same multiples of ℓ. After substitution into (1) and several steps we get a condition (2) for our special case.

$$\begin{gathered} m \le n \le 2\cdot m+z \\ \Updownarrow \\ \frac{\Delta_b-\Delta_a}{\ell}+m \le n \le 2\cdot m+z+\frac{\Delta_b-2\cdot\Delta_a}{\ell} \end{gathered} \tag{2}$$

This condition holds for all $m, n \ge 1$. The last step eliminates m and n from inequalities in (2). We get the error and correctness preserving properties for our special case in the form of equations (3).

$$\left\lceil \frac{\Delta_b - \Delta_a}{\ell} \right\rceil = 0 = \left\lfloor \frac{\Delta_b - \Delta_a}{\ell} - \frac{\Delta_a}{\ell} \right\rfloor \tag{3}$$

The only possible choice for Δ_a and Δ_b is

$$\Delta_a = \Delta_b = 0$$

Therefore, M executes no DELoperation on all words from the infinite set

$$\{a^{\ell \cdot m+R+A} b^{\ell \cdot n+R+A} \mid m,n \geq 1 \text{ and } 1 \leq A \leq \ell \text{ and } (R+A) \equiv 0 \bmod \ell\}.$$

This is a contradiction with the weak cyclic form of M. So, there is no *det-*D_cR -automaton such that it recognizes the language L.

4.3. $L \in \mathcal{L}(det\text{-}DR)$

We construct a *det-*DR-automaton M that recognizes the language L. M works in the following way:

- M executes a cycle

$$a^{2\cdot m+x_1} b^{2\cdot n+y_1} \Rightarrow_M a^{m+x_2} b^{n+y_2} \tag{4}$$

  for all $m,n \geq 1$, $x_1, y_1 \in \{0,1\}$. Values of x_2 and y_2 are determined from values of x_1 and y_1 as stated in the following table:

x_1	0	0	1	1
y_1	0	1	0	1
x_2	0	−1	0	0
y_2	0	−1	−1	0

- M accepts in one cycle words λ, ab and rejects in one cycle all the words containing a subword ba, and words from a^+, aa^+b, b^+, and $abbb^+$.

Now, we will verify that cycle (4) preserves both errorness and correctness of the word the automaton M reduces. In the same way as in the previous subsection 4.2 the condition (5) expressing error and correctness preserving properties for the cycle (4) will be simplified obtaining an equivalent condition (6).

$$\begin{gathered} 2\cdot m + x_1 \leq \ 2\cdot n + y_1 \leq 2\cdot(2\cdot m + x_1) \\ \Updownarrow \\ m + x_2 \leq \ n + y_2 \ \leq 2\cdot(m + x_2) \end{gathered} \tag{5}$$

$$\begin{gathered} \frac{x_1 - y_1}{2} + m \leq n \leq 2\cdot m + \frac{2\cdot x_1 - y_1}{2} \\ \Updownarrow \\ x_2 - y_2 + m \leq n \leq 2\cdot m + 2\cdot x_2 - y_2 \end{gathered} \tag{6}$$

Resulting condition (6) is equivalent to the following conjunction of two equations (7).

$$\left\lfloor \frac{x_1 - y_1}{2} \right\rfloor = x_2 - y_2 \quad \text{and} \quad \left\lceil \frac{2 \cdot x_1 - y_1}{2} \right\rceil = 2 \cdot x_2 - y_2 \tag{7}$$

Table 1 shows that M satisfies condition (7). Therefore, M fulfills correctness and error preserving properties on words from $\{a^m b^n \mid m, n \geq 2\}$.

x_1	0	0	1	1
y_1	0	1	0	1
x_2	0	-1	0	0
y_2	0	-1	-1	0
$\lceil \frac{x_1 - y_1}{2} \rceil$	0	0	1	0
$x_2 - y_2$	0	0	1	0
$\lfloor \frac{2 \cdot x_1 - y_1}{2} \rfloor$	0	-1	1	0
$2 \cdot x_2 - y_2$	0	-1	1	0

Table 1: Verification of correctness and error preserving properties.

5. Conclusion

We presented the separation of two properties of formal languages which are studied by means DR-automata. In this separation the language $L = \{a^m b^n \mid 0 \leq m \leq n \leq$ $\leq 2 \cdot m\}$ was used. Originaly, this language was considered as a candidate for the separation of classes recognized by deterministic and nondeterministic versions of DR-automata. In this article we showed that L is recognized by *det-DR*-automaton but cannot be recognized by *det-DR*-automaton with bounded deleting that is introduced in this article.

References

[1] Chomsky N.: *Linguistics, Logic, Psychology, and Computers*; Computer Programming and Artificial Inteligence (an intensive course for practical scientists and engineers), Summer session, University of Michigan, 1958, pp. 424–454

[2] Chomsky N.: *Formal Properties of Grammars*; Handbook of Mathematical Psychology, 2, New York, Wiley, 1963, pp.323–418

[3] Jančar P., Mráz F., Plátek M., Vogel J.: *Restarting Automata*; Proc. FCT'95, Dresden, Germany, LNCS 965, Springer 1995, pp. 283–292

[4] Jančar P., Mráz F., Plátek M., Procházka M., Vogel J.: Restarting Automata, Marcus Grammars and Context-Free Languages; in: Dassow J., Rozenberg G., Salomaa A. (eds.): *Developments in Language Theory II*; World Scientific Publ., 1996, pp 102–111

[5] Jančar P., Mráz F., Plátek M., Procházka M., Vogel J.: *Deleting Automata with Restart Operation*; Proc. of the 3rd International Conference Development in Language Theory, Thessaloniki, July-26 1997, edited by Symeon Bozapalidis, Aristotle University of Thessaloniki, pp. 191–202

Author's address: Charles University Prague, Dept. of Theoretical Computer Science, Malostranské náměstí 25, 118 00 Prague 1, Czech Republic

E-mail: `martin_prochazka@hotmail.com`

Received: August 19, 1998