BASIC PROPERTIES OF *N*–LANGUAGES

Martin Čermák
Doctoral Degree Programme (3), FIT BUT
E-mail: icermak@fit.vutbr.cz

Supervised by: Alexander Meduna
E-mail: meduna@fit.vutbr.cz

Abstract: This paper investigates theory of *n*–languages, where *n*–languages are given by sets of *n*–tuples of strings. In the present paper, two *n*–accepting move–restricted automata systems are defined. The automata systems are given by pushdown or finite automata with move–restricting set. By this set, the systems control which moves can be used at the same time. The paper discuses some basic properties of the class of *n*–languages defined by the automata systems.

Keywords: finite automata, pushdown automata, automata system, *n*–language, closure properties, control computation

1 INTRODUCTION

The theory of formal languages investigates various formal model systems using several cooperating components (see [1, 4, 7, 8]). Usually, the systems define ordinary formal languages. Among them, it is a canonical multigenerative context–free grammar system (see [5, 3, 2]), where each component generates its own string—that is, the grammar system generates *n*–touple of string (so–called *n*–string), and only if the generation succeed, then final strings are given from the *n*–string by a defined operation on the *n*–strings. A similar approaches was applied on pushdown automata, when two type of *n*–accepting restricted pushdown automata systems, so that the systems accept *n*–strings instead of ordinary strings, was defined in [9]. The *n*–accepting automata systems and the canonical multigenerative grammar systems open the new area of the formal language theory. This paper continues with researching this type of systems by introducing *n*–accepting finite automata system, and discuses some basic properties of classes of so–called *n*–languages, which finite automata systems can recognize. Specifically, this paper investigates some closure properties and relationship between the class of *n*–languages defined by *n*–accepting restricted finite automata systems and the class of *n*–languages defined by pushdown automata systems.

2 PRELIMINARIES

In this paper, we assume that the reader is familiar with formal language theory (see [6]).

For a set, Q, $|Q|$ denotes the cardinality of Q. For an alphabet, V, V^* represents the free monoid generated by V. The identity of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V. For $w \in V^*$, $|w|$ denotes the length of w, w^R denotes the mirror image of w.

A *finite automaton* is a five–tuple $M = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is an alphabet, $q_0 \in Q$ is the initial state, δ is a finite set of rules of the form $qa \to p$, where $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $F \subseteq Q$ is a set of final states. A configuration of M is any word from $Q\Sigma^*$. For any configuration qay, where $y \in \Sigma^*$, $q \in Q$ and any $qa \to p \in \delta$, M makes a move from configuration qay to configuration py according to $qa \to p$, written as $qay \Rightarrow py\,[qa \to p]$, or, simply, $qay \Rightarrow py$. If $x, y \in Q\Sigma^*$ and $m > 0$, then $x \Rightarrow^m y$ if there exists a sequence $x_0 \Rightarrow x_1 \Rightarrow \ldots \Rightarrow x_m$, where $x_0 = x$ and $x_m = y$. Then we say $x \Rightarrow^+ y$ if there exists $m > 0$ such that $x \Rightarrow^m y$ and $x \Rightarrow^* y$ if $x = y$ or $x \Rightarrow^+ y$. If $w \in \Sigma^*$ and $q_0 w \Rightarrow^* f$, where $f \in F$, then w is accepted by M and $q_0 w \Rightarrow^* f$ is an acceptance of w in M. The language of M is defined as $\mathcal{L}(M) = \{w \in \Sigma^* : q_0 w \Rightarrow^* f \text{ is an acceptance of } w\}$.

A *pushdown automaton* is a septuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$, where Q is a finite set of states, Σ is an alphabet, $q_0 \in Q$ is the initial state, Γ is a pushdown alphabet, δ is a finite set of rules of the form $Zqa \rightarrow \gamma p$, where $p, q \in Q$, $Z \in \Gamma$, $a \in \Sigma \cup \{\varepsilon\}$, $\gamma \in \Gamma^*$ and $Z_0 \in \Gamma$ is the initial pushdown symbol. A configuration of M is any word from $\Gamma^* Q \Sigma^*$. For any configuration $xAqay$, where $x \in \Gamma^*$, $y \in \Sigma^*$, $q \in Q$ and any $Aqa \rightarrow \gamma p \in \delta$, M makes a move from configuration $xAqay$ to configuration $x\gamma py$ according to $Aqa \rightarrow \gamma p$, written as $xAqay \Rightarrow x\gamma py\,[Aqa \rightarrow \gamma p]$, or, simply, $xAqay \Rightarrow x\gamma py$. If $x, y \in \Gamma^* Q \Sigma^*$ and $m > 0$, then $x \Rightarrow^m y$ if there exists a sequence $x_0 \Rightarrow x_1 \Rightarrow \ldots \Rightarrow x_m$, where $x_0 = x$ and $x_m = y$. Then we say $x \Rightarrow^+ y$ if there exists $m > 0$ such that $x \Rightarrow^m y$ and $x \Rightarrow^* y$ if $x = y$ or $x \Rightarrow^+ y$. If $w \in \Sigma^*$ and $Z_0 q_0 w \Rightarrow^* f$, where $f \in Q$, then w is accepted by M and $Z_0 q_0 w \Rightarrow^* f$ is an acceptance of w in M. The language of M is defined as $\mathcal{L}(M) = \{w \in \Sigma^* : Z_0 q_0 w \Rightarrow^* f \text{ is an acceptance of } w\}$.

3 DEFINITIONS

3.1 n–ACCEPTING AUTOMATA SYSTEMS

An n–accepting move–restricted finite automata system (n–MAM$^{\text{FA}}$) and n–accepting move–restricted pushdown automata system (n–MAM$^{\text{PDA}}$) are $n+1$–tuples $\vartheta = (M_1 \ldots, M_n, \Psi)$ with M_i as a finite automaton and pushdown automaton for all $i = 1, \ldots, n$, respectively, and with Ψ as a finite set of n–tuples of the form $(r_1, \ldots, r_n)$, where for each $j = 1, \ldots, n$, $r_j \in \delta_j$ in M_j.

3.2 n–CONFIGURATION

Let n be a positive integer, $\vartheta = (M_1, \ldots, M_n, \Psi)$ be an n–MAM$^{\text{FA}}$ or n–MAM$^{\text{PDA}}$. An n–configuration is defined as an n-tuple $\chi = (x_1, \ldots, x_n)$, where for all $i = 1, \ldots, n$, x_i is a configuration of M_i.

3.3 MOVE

Let n be a positive integer, $\vartheta = (M_1, \ldots, M_n, \Psi)$ be an n–MAM$^{\text{FA}}$ or n–MAM$^{\text{PDA}}$. Let $\chi = (x_1, \ldots, x_n)$ and $\chi' = (x'_1, \ldots, x'_n)$ be two n-configurations, for all $i = 1, \ldots, n$, $x_i \Rightarrow x'_i[r_i]$ in M_i, and $(r_1, \ldots, r_n) \in \Psi$. Then, ϑ moves from n–configuration χ to χ', denoted $\chi \vdash \chi'$, and in the standard way, $\vdash^*$ and $\vdash^+$ denote the transitive-reflexive and the transitive closure of $\vdash$, respectively.

3.4 n–LANGUAGE OF n–MAM$^{\text{FA}}$

Let n be a positive integer, $\vartheta = (M_1, \ldots, M_n, \Psi)$ be an n–MAM$^{\text{FA}}$ and all $i = 1, \ldots, n$, $M_i = (Q_i, \Gamma_i, \delta_i, s_i, F_i)$ be a finite automaton. Let $\chi_0 = (s_1\omega_1, \ldots, s_n\omega_n)$ be the start n–configuration and $\chi_f = (q_1, \ldots, q_n)$ be a finish n–configuration of n–MAM$^{\text{FA}}$, where for all $i = 1, \ldots, n$, $q_i \in F_i$, $\omega_i \in \Sigma^*$. The n–language of n–accepting finite automata system is defined as $n\text{-}L(\vartheta) = \{(\omega_1, \ldots, \omega_n) : \chi_0 \vdash^* \chi_f\}$.

3.5 n–LANGUAGE OF n–MAM$^{\text{PDA}}$

Let n be a positive integer, $\vartheta = (M_1, \ldots, M_n, \Psi)$ be an n–MAM$^{\text{PDA}}$ and for all $i = 1, \ldots, n$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, \emptyset)$ be a pushdown automaton accepting input strings by empty pushdown. Let $\chi_0 = (z_{1,0}s_1\omega_1, \ldots, z_{n,0}s_n\omega_n)$ be the start n–configuration and $\chi_f = (q_1, \ldots, q_n)$ be a finish n–configuration of n–MAM$^{\text{PDA}}$, where for all $i = 1, \ldots, n$, $q_i \in Q_i$, $\omega_i \in \Sigma^*$. The n–language of n–accepting pushdown automata system is defined as $n\text{-}L(\vartheta) = \{(\omega_1, \ldots, \omega_n) : \chi_0 \vdash^* \chi_f\}$.

3.6 Classes of n–Languages

- $\mathscr{L}(n\text{–MAM}^{\text{FA}}) = \{n\text{–}L : n\text{–}L$ is an n–language of $n\text{–MAM}^{\text{FA}}\}$
- $\mathscr{L}(n\text{–MAM}^{\text{PDA}}) = \{n\text{–}L : n\text{–}L$ is an n–language of $n\text{–MAM}^{\text{PDA}}\}$

4 RESULTS

4.1 Theorem

If L_1 and $L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, then $L_1 \cup L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$.

Proof: Consider two n–languages L_1 and L_2. If L_1 and $L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, then there are n–MAM$^{\text{FA}}$s, $\vartheta_1 = (M_{1,1},\ldots,M_{1,n},\Psi_1)$ and $\vartheta_2 = (M_{2,1},\ldots,M_{2,n},\Psi_2)$, such that $L_1 = L(\vartheta_1)$, $L_2 = L(\vartheta_2)$ and for all $i = 1,2$ and $j = 1,\ldots,n$, $M_{i,j} = (Q_{i,j},\Sigma_{i,j},\delta_{i,j},s_{i,j},F_{i,j})$ is a component of ϑ_i. For these two automata systems, we can construct $\vartheta_{12} = (M_{12,1},\ldots,M_{12,n},\Psi_{12})$, with $M_{12,j} = (Q_{12,j},\Sigma_{12,j},\delta_{12,j}, s_{12,j},F_{12,j}))$, in the following way: for all $i = 1,2$ and $j = 1,\ldots,n$, $Q_{12,j} = Q_{1,j} \cup Q_{2,j} \cup \{s_{12,j}\}$, where $s_{12,j}$ is the new start state of jth automaton, $\delta_{12,j} = \delta_{1,j} \cup \delta_{2,j} \cup \{p_{1,j} : s_{12,j} \to s_{1,j}, p_{2,j} : s_{12,j} \to s_{2,j}\}$, $\Sigma_{12,j} = \Sigma_{1,j} \cup \Sigma_{2,j}$, $F_{12,j} = F_{1,j} \cup F_{2,j}$, and $\Psi_{12} = \Psi_1 \cup \Psi_2 \cup \{(p_{1,1},\ldots,p_{1,n}),(p_{2,1},\ldots,(p_{2,n})\}$. From set Ψ_{12} follows that the first move has to be $(s_{12,1}\omega_1,\ldots,s_{12,n}\omega_n) \vdash (s_{i,1}\omega_1,\ldots,\ s_{i,n}\omega_n)$ for $i = 1,2$, and for $\omega_j \in \Sigma_{12,j}$ with $j = 1,\ldots,n$. Therefore, $(\omega_1,\ldots,\omega_n)$ is in $L(\vartheta_{12})$ iff $(\omega_1,\ldots,\omega_n) \in L(\vartheta_1)$ or $(\omega_1,\ldots,\omega_n) \in L(\vartheta_2)$—that is, $(\omega_1,\ldots,\omega_n) \in L(\vartheta_{12})$ iff $(\omega_1,\ldots,\omega_n) \in L_1 \cup L_2$. □

4.2 Lemma

For $n \geq 2$, n–language $n\text{–}L = \{(a^ib^j, a^jb^i(,\varepsilon)^{(n-2)}) : i,j = 0,1,\ldots,m\}$ is not in $\mathscr{L}(n\text{–MAM}^{\text{FA}})$.

Proof Idea: From definition of n–accepting move–restricted finite automata system, it can be seen that only chance how to compare symbols through components is read them step by step at the same time (or in a quasi parallel way). Hence, for comparing as in the first component and bs in the second one, the second component has to skip all as, and then the system can compare as and bs. After this, there is no possibility how to compare as in the second component with bs in the first component because finite automata can not be returned on the start position. Similar problem becomes when the system starts with comparing bs in the first component and as in the second one. The other components can not help, because they read no input symbols. Hence, $n\text{–}L$ is not belong to $\mathscr{L}(n\text{–MAM}^{\text{FA}})$. □

4.3 Corollary

$\mathscr{L}(n\text{–MAM}^{\text{FA}})$ for all $n \geq 2$, is not close under intersection.

Proof: Consider two 2–languages $L_1 = \{(a^ib^j, a^jb^k) : i,j,k \geq 0\}$ and $L_2 = \{(a^ib^j, a^kb^i) : i,j,k \geq 0\}$. Both of them belong to $\mathscr{L}(n\text{–MAM}^{\text{FA}})$, because we can construct 2–MAM$^{\text{FA}}$s $\vartheta_1 = (M_1,M_2,\Psi_1)$ and $\vartheta_2 = (M_1,M_2,\Psi_2)$ such that $L(\vartheta_1) = L_1$ and $L(\vartheta_2) = L_2$. All four automata are given by the definition $M = (\{q_1,q_2\},\{a,b\},\{r_1 : q_1a \to q_1, r_2 : q_1 \to q_1, r_3 : q_1b \to q_2, r_4 : q_2b \to q_2, r_5 : q_2 \to q_2\}, s_i, \{q_1,q_2\})$, and $\Psi_1 = \{(r_1,r_2),(r_3,r_1),(r_4,r_1),(r_5,r_3),(r_5,r_4)\}$ and $\Psi_2 = \{(p,q) : (q,p) \in \Psi_1\}$. The intersection of $L(\vartheta_1)$ and $L(\vartheta_2)$ is 2–language $L_3 = \{(a^ib^j, a^jb^i) : i,j = 0,1,\ldots,m\}$. Lemma 4.2 says that $L_3 \notin \mathscr{L}(n\text{–MAM}^{\text{FA}})$, and therefore, $\mathscr{L}(2\text{–MAM}^{\text{FA}})$ is not close under intersection.

In general, for $n \geq 2$, consider n–languages $K_1 = \{(a^ib^j, a^jb^k(,\varepsilon)^{n-2}) : i,j,k \geq 0\}$ and $K_2 = \{(a^ib^j, a^kb^i(,\varepsilon)^{n-2}) : i,j,k \geq 0\}$. From Lemma 4.2, $K_1 \cap K_2 \notin \mathscr{L}(n\text{–MAM}^{\text{FA}})$. □

4.4 COROLLARY

$\mathscr{L}(n\text{–MAM}^{\text{FA}})$ for all $n \geq 2$ is not close under complementation.

Proof: By contradiction. Suppose that $\mathscr{L}(n\text{–MAM}^{\text{FA}})$ for all $n \geq 2$ is close under complementation. Let $L_1, L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$. From Theorem 4.1 it follows that $L_1 \cup L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, and by supposition, $\overline{(L_1 \cup L_2)} \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$ as well. From De Morgan's law, $\overline{(L_1 \cup L_2)} = (\overline{L_1} \cap \overline{L_2})$, but it is contradiction, because $\mathscr{L}(n\text{–MAM}^{\text{FA}})$ for all $n \geq 2$, is not close under intersection. □

4.5 COROLLARY

$\mathscr{L}(n\text{–MAM}^{\text{FA}}) \subsetneq \mathscr{L}(n\text{–MAM}^{\text{PDA}})$.

Proof: The inclusion $\mathscr{L}(n\text{–MAM}^{\text{FA}}) \subseteq \mathscr{L}(n\text{–MAM}^{\text{PDA}})$ is clear from the definition of n–MAM$^{\text{FA}}$ and n–MAM$^{\text{PDA}}$. It remains to prove that $\mathscr{L}(n\text{–MAM}^{\text{FA}}) \neq \mathscr{L}(n\text{–MAM}^{\text{PDA}})$.

Consider 2–MAM$^{\text{PDA}}$, $\vartheta = (M_1, M_2, \Psi_1)$ with $\Psi = \{(1,1),(2,2),(3.2),(4,3),(5,3),(6,3), (7,3), (8,4),(9,1)\}$) and the pushdown automata defined in the following way: $M_1 = (\{s,q\}, \{a,b\}, \{\#,a\}, \{1.\#s \rightarrow s, 2.\#sa \rightarrow \#as, 3.asa \rightarrow aas, 4.\#sb \rightarrow \#q, 5.asb \rightarrow \#q, 6.\#qb \rightarrow \#q, 7.aqb \rightarrow \#q, 8.aq \rightarrow q, 9.\#q \rightarrow q\}, s, \#, \emptyset)$ and $M_2 = (\{s\}, \{a,b\}, \{\#\}, \{1.\#s \rightarrow s, 2.\#s \rightarrow \#s, 3.\#sa \rightarrow \#s, 4.\#sb \rightarrow \#s\}, s, \#, \emptyset)$. It is not hard to see that ϑ define 2–language $L = \{(a^i b^j, a^j b^i) : i, j = 0, 1, \ldots, m\}$ and works in this way: first, automaton M_2 loops in state s and reads no symbol, while M_1 shifts all as onto the pushdown. After pushing all as from the M_1's input onto the pushdown, M_1 and M_2 read bs and as, respectively, and by reading them at the same time, automata compare their number. If there is more as in M_2's input than bs in M_1's input, then the automata system is stoped and input is not accepted. Otherwise, M_1 skips to the other state and ϑ continues with comparing a's on the M_1's pushdown and b's in the M_2's input by removing as from the pushdown in M_1 and reading b's from M_2's input. Only if the input was of the form $(a^i b^j, a^j b^i)$ with $i, j = 0, 1, \ldots, m$, the automata system removes symbols # from M_1's and M_2's pushdowns, and ϑ accepts. Because Lemma 4.2 says that $L = \{(a^i b^j, a^j b^i) : i, j = 0, 1, \ldots, m\}$ is not in $\mathscr{L}(2\text{–MAM}^{\text{FA}})$, $\mathscr{L}(2\text{–MAM}^{\text{FA}}) \neq \mathscr{L}(2\text{–MAM}^{\text{PDA}})$. In general, for $n \geq 2$, there are n–languages $L = \{(a^i b^j, a^j b^i (,\varepsilon)^{(n-2)}) : i, j = 0, 1, \ldots, m\}$. These n–languages can be given by n–MAM$^{\text{PDA}}$, where the first two components are defined in the same way as M_1 and M_2 was. The other components loops without reading any symbols. Only in the last step, automata remove symbols # from their pushdowns. Hence, $L = \{(a^i b^j, a^j b^i (,\varepsilon)^{(n-2)}) : i, j = 0, 1, \ldots, m\} \in \mathscr{L}(n\text{–MAM}^{\text{PDA}})$—that is, $\mathscr{L}(n\text{–MAM}^{\text{FA}}) \neq \mathscr{L}(n\text{–MAM}^{\text{PDA}})$. □

4.6 THEOREM

If L_1 and $L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, then $L_1 . L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, where $L_1 . L_2 = \{(w_1 w'_1, \ldots, w_n w'_n) : (w_1, \ldots, w_n) \in L_1 \text{ and } (w'_1, \ldots, w'_n) \in L_2\}$.

Proof: Consider two n–languages L_1 and L_2. If L_1 and $L_2 \in \mathscr{L}(n\text{–MAM}^{\text{FA}})$, then there are n–MAM$^{\text{FA}}$s, $\vartheta_1 = (M_{1,1}, \ldots, M_{1,n}, \Psi_1)$ and $\vartheta_2 = (M_{2,1}, \ldots, M_{2,n}, \Psi_2)$, such that $L_1 = L(\vartheta_1)$, $L_2 = L(\vartheta_2)$ and for all $i = 1,2$ and $j = 1, \ldots, n$, $M_{i,j} = (Q_{i,j}, \Sigma_{i,j}, \delta_{i,j}, s_{i,j}, F_{i,j})$ is a component of ϑ_i. For these two automata systems, we can construct $\vartheta_{12} = (M_{12,1}, \ldots, M_{12,n}, \Psi_{12})$, with $M_{12,j} = (Q_{12,j}, \Sigma_{12,j}, \delta_{12,j}, s_{1,j}, F_{2,j})$), in the following way: for every $i = 1,2$ and $j = 1, \ldots, n$, $Q_{12,j} = Q_{1,j} \cup Q_{2,j}$, $\delta_{12,j} = \delta_{1,j} \cup \delta_{2,j} \cup \{p_j.f_j \rightarrow s_{2,j} : f_j \in F_{1,j}\}$, $\Psi_{12} = \Psi_1 \cup \Psi_2 \cup \{(p_1, \ldots, p_n)\}$, and $\Sigma_{12,j} = \Sigma_{1,j} \cup \Sigma_{2,j}$. From Definition 3.4, $(w_1, \ldots, w_n) \in L_1$ iff $(s_{1,1} w_1, \ldots, s_{1,n} w_n) \vdash^* (f_1, \ldots, f_n)$, where for all $i = 1, \ldots, n$, $f_i \in F_{1,i}$, in ϑ_1. Clearly, $(s_{1,1} w_1, \ldots, s_{1,n} w_n) \vdash^* (f_1, \ldots, f_n)$ in ϑ_{12} as well, and obviously, $(s_{1,1} w_1 w'_1, \ldots, s_{1,n} w_n w'_n) \vdash^* (f_1 w'_1, \ldots, f_n w'_n)$. As $(w'_1, \ldots, w'_n) \in L_2$ and because $(f_1 \rightarrow s_{2,1}, \ldots, s_{2,n}) \in \Psi_{12}$, $(f_1 w'_1, \ldots, f_n w'_n) \vdash (s_{2,1} w'_1, \ldots, s_{2,n} w'_n)$. Naturaly, $(s_{2,1} w'_1, \ldots, s_{2,n} w'_n) \vdash^* (f'_1, \ldots, f'_n)$ with $f'_i \in F_{2,i}$ in ϑ_2. Hence, $(s_{1,1} w_1 w'_1,$

$\dots, s_{1,n} w_n w'_n) \vdash^* (f'_1, \dots, f'_n)$ in ϑ_{12}. The theorem holds. □

5 CONCLUSION

In this paper, we defined the new type of n–accepting move–restricted automata system with finite automata. On the class of n–languages defined by the system, we showed some fundamental closure properties. Specifically, the class of n–languages defined by n–accepting move–restricted pushdown automata system is closed over concatenation and union, and on the other hand, it is not closed over intersection and complement. Futhermore, we showed that the n–accepting move–restricted finite automata system is weaker than the n–accepting move–restricted pushdown automata system. Beside of examined closure properties, there are many other closer properties, which make an open research area. Especially, very useful it can be closure properties over n–union, n–intersection, and n–shuffle, where n before operators means that operators are used on each component. For example, n–union of $L_1 = \{(w_1, \dots, w_n)\}$ and $L_2 = \{(w'_1, \dots, w'_n)\}$ is n–language $L_3 = \{(x_1, \dots, x_n) : x_i \in \{w_i, w'_i\}$ for all $i = 1, \dots, n\}$.

6 ACKNOWLEDGEMENTS

This work was supported by the research plan MSM0021630528 and by MŠMT grant MEB041003.

REFERENCES

[1] R. Alur. Formal analysis of hierarchical state machines. *Lecture Notes in Computer Science*, 2772/2004:434–435, 2004.

[2] R. Lukáš. Power of multigenerative grammar systems. In *Second Doctoral Workshop on Mathematical and Engineering Methods in Computer Science (MEMICS 2006)*, pages 99–104, 2006.

[3] R. Lukáš and A. Meduna. Multigenerative grammar systems. In *Pre-proceedings 1st Doctoral Workshop on Mathematical and Engineering Methods in Comupter Science (MEMICS 2005)*, pages 85–87. Faculty of Informatics MU, 2005.

[4] C. Martín and V. Mitrana. Parallel communicating automata systems. *Journal of Applied Mathematics and Computing*, pages 237–257, 2008.

[5] A. Meduna and R. Lukáš. Multigenerative grammar systems. *Schedae Informaticae*, 2006(15):175–188, 2006.

[6] M. Meduna. *Automata and Languages: Theory and Applications*. Springer-Verlag, London, 2000.

[7] M. H. ter Beek, E. Csuhaj-Varjú, and V. Mitrana. Teams of pushdown automata. *Int. J. Comput. Math.*, 81(2):141–156, 2004.

[8] M. Čermák. Power decreasing derivation restriction in grammar systems. In *Proceedings of the 15th Conference and Competition STUDENT EEICT 2009 Volume 4*, pages 385–389. Faculty of Information Technology BUT, 2009.

[9] M. Čermák. Multilanguages and multiaccepting automata system. In *Proceedings of the 16th Conference and Competition STUDENT EEICT 2010 Volume 5*, pages 146–150. Faculty of Information Technology BUT, 2010.