Excel@FIT 2015
http://excel.fit.vutbr.cz

# On #-rewriting systems with pure pushdowns

Jiří Kučera*

**Abstract**
The aim of this paper is to present a modification of #-rewriting systems, called *#-rewriting systems with pure pushdowns*, and their relation to state grammars. Briefly, the #-rewriting systems with pure pushdowns are #-rewriting systems equiped with pushdowns containing only terminal symbols (herefrom comes their denomination as *pure*), and a pushdown containing terminal and nonterminal symbols, which is used as an auxiliary workspace. It will be shown that this extension leads to relation between programmed grammars and state grammars.

**Keywords:** rewriting systems — state grammars — pure pushdowns — infinite hierarchy

**Supplementary Material:** *N/A*

*ikucera@fit.vutbr.cz,*Faculty of Information Technology, Brno University of Technology*

## 1. Introduction

#-rewriting systems were introduced in [1] by Křivka, Meduna, and Schönecker, and their main purpose was to extend the formal language theory with formal model based on a combination of automata and grammars. Like grammars, #-rewriting systems are generative devices, but they do not use any nonterminal symbols. Like automata, they use finetely many states. The # symbol in the #-rewriting system's name is called a *bounder*. The bounder together with state are used to select the suitable rule to be applied and a position in configuration to be rewritten. If the number of occurences of # in configuration is limited by some positive integer $k$, then the #-rewriting systems have an *index* $k$. In [1] was proved that the family of languages generated by #-rewriting systems of index $k$ and the family of languages generated by programmed grammars of index $k$ are identical.

This paper extends the #-rewriting systems with *pure pushdowns*. It is shown that such an extension leads to two important results: (i) the family of languages generated by #-rewriting systems with pure pushdowns of index $k$ and the family of languages generated by state grammars of index $k$ are identical, and (ii) since the #-rewriting systems with pure pushdowns keep properties of #-rewriting systems, the family of languages generated by programmed grammars of index $k$ is included in the family of languages generated by state grammars of index $k$.

## 2. Preliminaries

It is assumed that the reader is familiar with the basic notions of the formal language theory (see [2, 3, 4]). Let $\Sigma$ be an alphabet. Then, $\Sigma^*$ represents the free monoid generated by $\Sigma$ under the operation of concatenation, with $\varepsilon$ as the unit of $\Sigma^*$. For $w \in \Sigma^*$, $alph(w)$ denotes the set of all symbols from $\Sigma$ contained in $w$. For $w \in \Sigma^*$ and $W \subseteq \Sigma$, $occur(w,W)$ denotes the number of occurences of symbols from $W$ in $w$. The special case $occur(w,\{a\})$, where $a \in \Sigma$, is written for brevity as $occur(w,a)$. By $\mathscr{L}(\mathrm{PG},k)$ is denoted the family of languages generated by programmed grammars of index $k$.

A *state grammar*[5] is a sixtuple

$$G = (V,T,K,P,S,s),$$

where $V$ is a total alphabet, $T \subset V$ is an alphabet of terminals, $K$ is a finite set of states, $V \cap K = \emptyset$, $P \subseteq (V-T) \times K \times V^* \times K$ is a finite set of rules, $S \in (V-T)$ is a start symbol, and $s \in K$ is a start state. Instead of $(A,p,x,q) \in R$, where $A \in (V-T)$, $p,q \in K$, and $x \in V^*$, is preferred the notation $(A,p) \to (x,q) \in R$. Let $(A,p) \to (x,q) \in R$. Then, $(uAv,p) \Rightarrow (uxv,q)\,[(A,p) \to (x,q)]$, or shortly $(uAv,p) \Rightarrow (uxv,q)$, is a *direct derivation step* in $G$ if and only if for every $(B,p) \to (y,t) \in R$, $B \notin alph(u)$, where $p,q,t \in K$,

$A, B \in (V - T)$, and $u, v, x, y \in V^*$. In addition, if $n$ is a positive integer satisfying $occur(uA, V - T) \leq n$, then $(uAv, p) \Rightarrow (uxv, q)\,[(A, p) \rightarrow (x, q)]$ is said to be *n-limited*, symbolically written as

$$(uAv, p)\ {}_n\Rightarrow\ (uxv, q)\,[(A, p) \rightarrow (x, q)],$$

or $(uAv, p)\ {}_n\Rightarrow\ (uxv, q)$ in brief. Let $\Rightarrow^*$ and $\Rightarrow^+$ denote the reflexive and transitive closure of $\Rightarrow$, and the transitive closure of $\Rightarrow$, respectively. Similarly for ${}_n\Rightarrow$. The *language generated* by $G$, $L(G)$, is defined as

$$L(G) = \{w \in T^* \mid (S, s) \Rightarrow^* (w, q), q \in K\}.$$

Furthermore, for every $n \geq 1$, is defined

$$L(G, n) = \{w \in T^* \mid (S, s)\ {}_n\Rightarrow^*\ (w, q), q \in K\}.$$

The family of languages generated by state grammars and by state grammars in $k$-limited way are denoted as $\mathscr{L}(\mathrm{ST})$, and $\mathscr{L}(\mathrm{ST}, k)$, respectively.

Let $I$ denote a set of all positive integers. A *#-rewriting system*[1] is a quadruple $M = (Q, \Sigma, s, R)$, where $Q$ is a finite set of states, $\Sigma$ is an alphabet containing special symbol # called a *bounder*, $Q \cap \Sigma = \emptyset$, $s \in Q$ is a start state and $R \subseteq Q \times I \times \{\#\} \times Q \times \Sigma^*$ is a finite set of *rules*. A rule $(p, n, \#, q, x) \in R$, where $p, q \in Q$, $n \in I$ and $x \in \Sigma^*$, is written as $r\colon\ p_n\# \rightarrow qx$, where $r$ is its unique label. A *configuration* of $M$ is a word from $Q\Sigma^*$. Let $pu\#v$ and $quxv$ be two configurations of $M$, and let $r\colon\ p_n\# \rightarrow qx$ be a rule from $R$, where $p, q \in Q$, $u, v, x \in \Sigma^*$, $n \in I$ and $occur(u, \#) = n - 1$. Then, $M$ makes a *derivation step* from $pu\#v$ to $quxv$ by using $r\colon\ p_n\# \rightarrow qx$, symbolically written as $pu\#v \Rightarrow quxv\,[r]$ or simply $pu\#v \Rightarrow quxv$. The reflexive and transitive closure of $\Rightarrow$, and the transitive closure of $\Rightarrow$ are denoted by $\Rightarrow^*$, and $\Rightarrow^+$, respectively. The *language generated* by $M$, $L(M)$, is defined as

$$L(M) = \{w \mid s\# \Rightarrow^* qw, q \in Q, w \in (\Sigma - \{\#\})^*\}.$$

Let $k$ be a positive integer. A #-rewriting system $M$ is of *index* $k$ if and only if $s\# \Rightarrow^* qy$ implies $occur(y, \#) \leq k$. The family of languages generated by #-rewriting systems and by #-rewriting systems of index $k$ are denoted as $\mathscr{L}(\#\mathrm{RS})$, and $\mathscr{L}(\#\mathrm{RS}, k)$, respectively.

## 3. Results

In this section are defined a #-rewriting systems with pure pushdowns and investigated their generative capacity. The rigorous proofs of theorems introduced in this section are ommited, since they are the main matter of the extended version of this paper which is in preparation to be submitted to scientific journal. Instead, only the brief explanations of basic ideas of proofs are presented.

A *rewriting system with pure pushdowns* of index $k$ is a construct

$$M = (Q, \Sigma, \Gamma, s, S, \#, \$, R),$$

where $Q$ is a finite set of states, $\Sigma$ is an input alphabet, $\Gamma$ is a pushdown alphabet, $\Sigma \subset \Gamma$, $Q \cap \Gamma = \emptyset$, $s \in Q$ is a start state, $S \in \Gamma$ is an initial pushdown symbol, $\# \notin (Q \cup \Sigma \cup \Gamma)$ is a *pure pushdown bounder*, $\$ \notin (Q \cup \Sigma \cup \Gamma)$ is a *pushdown bounder*, and $R$ is a finite set of rules of the following forms:

1. $p_n\# \rightarrow qx$, where $p, q \in Q$, $n$ is a positive integer, and $x \in (\Sigma \cup \{\#\})^*$;
2. $p\# \rightarrow qA$, where $p, q \in Q$, and $A \in \Gamma - \Sigma$;
3. $p\# \leftarrow qA$, where $p, q \in Q$, and $A \in \Gamma - \Sigma$;
4. $pa \rightarrow qa$, where $p, q \in Q$, and $a \in \Sigma$;
5. $pa \leftarrow qa$, where $p, q \in Q$, and $a \in \Sigma$.

A *configuration* of $M$ is a word

$$\chi \in Q(\Sigma \cup \{\#\})^*\$(\Sigma \cup \Gamma)^*.$$

There are five types of derivation steps possible in $M$:

1. Let $pu\#v\$\alpha$ and $quxv\$\alpha$ be two configurations of $M$, $p, q \in Q$, $u, v, x \in (\Sigma \cup \{\#\})^*$, $\alpha \in (\Sigma \cup \Gamma)^*$, and $occur(pu\#v\$\alpha, \#) \leq k$. Let $r\colon\ p_n\# \rightarrow qx$ be a rule from $R$. Then, $M$ makes a *derivation step* from $pu\#v\$\alpha$ to $quxv\$\alpha$ using $r$, symbolically written
$$pu\#v\$\alpha \Rightarrow quxv\$\alpha\,[r]$$
or simply $pu\#v\$\alpha \Rightarrow quxv\$\alpha$, if and only if there are exactly $n - 1$ occurences of # in $u$.
2. Let $pu\#\$\alpha$ and $qu\$A\alpha$ be two configurations of $M$, $p, q \in Q$, $u \in (\Sigma \cup \{\#\})^*$, $A \in \Gamma - \Sigma$, and $\alpha \in (\Sigma \cup \Gamma)^*$. Let $r\colon\ p\# \rightarrow qA$ be a rule from $R$. Then, $M$ makes a *derivation step* from $pu\#\$\alpha$ to $qu\$A\alpha$ using $r$, symbolically written
$$pu\#\$\alpha \Rightarrow qu\$A\alpha\,[r]$$
or simply $pu\#\$\alpha \Rightarrow qu\$A\alpha$.
3. Let $pu\$A\alpha$ and $qu\#\$\alpha$ be two configurations of $M$, $p, q \in Q$, $u \in (\Sigma \cup \{\#\})^*$, $A \in \Gamma - \Sigma$, and $\alpha \in (\Sigma \cup \Gamma)^*$. Let $r\colon\ p\# \leftarrow qA$ be a rule from $R$. Then, $M$ makes a *derivation step* from $pu\$A\alpha$ to $qu\#\$\alpha$ using $r$, symbolically written
$$pu\$A\alpha \Rightarrow qu\#\$\alpha\,[r]$$
or simply $pu\$A\alpha \Rightarrow qu\#\$\alpha$.

4. Let $pua\$\alpha$ and $qu\$a\alpha$ be two configurations of $M$, $p,q \in Q$, $u \in (\Sigma \cup \{\#\})^*$, $a \in \Sigma$, and $\alpha \in (\Sigma \cup \Gamma)^*$. Let $r\colon\ pa \to qa$ be a rule from $R$. Then, $M$ makes a *derivation step* from $pua\$\alpha$ to $qu\$a\alpha$ using $r$, symbolically written

$$pua\$\alpha \Rightarrow qu\$a\alpha\,[r]$$

or simply $pua\$\alpha \Rightarrow qu\$a\alpha$.

5. Let $pu\$a\alpha$ and $qua\$\alpha$ be two configurations of $M$, $p,q \in Q$, $u \in (\Sigma \cup \{\#\})^*$, $a \in \Sigma$, and $\alpha \in (\Sigma \cup \Gamma)^*$. Let $r\colon\ pa \leftarrow qa$ be a rule from $R$. Then, $M$ makes a *derivation step* from $pu\$a\alpha$ to $qua\$\alpha$ using $r$, symbolically written

$$pu\$a\alpha \Rightarrow qua\$\alpha\,[r]$$

or simply $pu\$a\alpha \Rightarrow qua\$\alpha$.

As usual, the reflexive and transitive closure of $\Rightarrow$, and the transitive closure of $\Rightarrow$ are denoted by $\Rightarrow^*$, and $\Rightarrow^+$, respectively.

The *language generated* by $M$, $L(M)$, is defined as

$$L(M) = \{w \in \Sigma^* \mid s\#\$S \Rightarrow^* qw\$, q \in Q\}.$$

The family of languages generated by #-rewriting systems with pure pushdowns of index $k$ are denoted as $\mathscr{L}(\#\mathrm{RS}_{\mathrm{PD}}, k)$.

In the following theorems are summarized the obtained results.

**Theorem 3.1.** *For every $k \geq 1$,*

$$\mathscr{L}(\#\mathrm{RS}_{\mathrm{PD}}, k) = \mathscr{L}(\mathrm{ST}, k).$$

*Proof (Basic Idea).* Every state grammar of index $k$ can be simulated by some #-rewriting system with pure pushdowns (#RSPD) of index $k$. The state and first $k$ nonterminal symbols of the configuration of state grammar are encoded into the state of the configuration of #RSPD. More precisely, if

$$(u_1A_1u_2A_2\ldots A_ku_{k+1}\alpha, p)$$

is a configuration of the state grammar of index $k$, where $u_i$ is a string over the alphabet of terminal symbols, $1 \leq i \leq k+1$ , $A_j$ is nonterminal symbol, $1 \leq j \leq k$, $\alpha$ is a string over the total alphabet, and $p$ is a state, the corresponding configuration of #RSPD of index $k$ will be

$$\langle pA_1A_2\ldots A_k\rangle u_1\#u_2\#\ldots\#u_{k+1}\$\alpha.$$

During a simulation, the #RSPD of index $k$ must handle the cases when nonterminal symbols in the corresponding configuration of the state grammar of index $k$ are generated or erased.

Conversely, every #RSPD of index $k$ can be simulated by some state grammar of index $k$ as follows (the simulation of one derivation step): To simulate a $p_n\# \to qx$ rule, state grammar scans for the $n$th # symbol of its configuration, replaces it for $x$, adjusts the number of occurences of # symbols in its configuration (by the simulation of the #RSPD's rules of form 2 to 5), and changes the state $p$ to the state $q$. □

**Theorem 3.2.** *For every $k \geq 1$,*

$$\mathscr{L}(\#\mathrm{RS}, k) \subseteq \mathscr{L}(\#\mathrm{RS}_{\mathrm{PD}}, k).$$

*Proof.* Follows from the observation that #-rewriting systems with pure pushdowns of index $k$ are extensions of #-rewriting systems of index $k$. □

The conclusion of the previous two theorems and from the results in [1] is the following corollary.

**Corollary 3.3.** *For every $k \geq 1$,*

$$\mathscr{L}(\mathrm{PG}, k) \subseteq \mathscr{L}(\mathrm{ST}, k).$$

## 4. Acknowledgements

This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070), the TAČR grant TE01020415, and the BUT grant FIT-S-14-2299.

## References

[1] Z. Křivka, A. Meduna, and R. Schönecker. Generation of languages by rewriting systems that resemble automata. *International Journal of Foundations of Computer Science*, 17(5):1223–1229, 2006.

[2] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages: Word, Language, Grammar*, volume 1. Springer-Verlag, Berlin, 1997.

[3] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages: Linear Modeling: Background and Application*, volume 2. Springer-Verlag, Berlin, 1997.

[4] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages: Beyond Words*, volume 3. Springer-Verlag, Berlin, 1997.

[5] T. Kasai. An hierarchy between context-free and context-sensitive languages. *Journal of Computer and System Sciences*, 4:497–508, 1970.