Petr Blatný; Radek Bidlo; Alexander Meduna

# Automata with two-sided pushdowns defined over free groups generated by reduced alphabets

*Kybernetika*, Vol. 43 (2007), No. 3, 265--278

Persistent URL: http://dml.cz/dmlcz/135773

KYBERNETIKA — VOLUME *43* (2007), NUMBER 3, PAGES 265 – 278

# AUTOMATA WITH TWO–SIDED PUSHDOWNS DEFINED OVER FREE GROUPS GENERATED BY REDUCED ALPHABETS

Petr Blatný, Radek Bidlo and Alexander Meduna

This paper introduces and discusses a modification of pushdown automata. This modification is based on two-sided pushdowns into which symbols are pushed from both ends. These pushdowns are defined over free groups, not free monoids, and they can be shortened only by the standard group reduction. We demonstrate that these automata characterize the family of recursively enumerable languages even if the free groups are generated by no more than four symbols.

*Keywords:* pushdown automata, modifications, recursively enumerable languages
*AMS Subject Classification:* 68Q05, 68Q45

## 1. INTRODUCTION

Undoubtedly, the pushdown automata fulfill a crucial role in the automata theory. Viewed as a language acceptor, pushdown automaton consists of an input tape, a read head, a pushdown and a finite state control. The input tape is divided into squares, each of which contains one symbol of an input string. The finite control is represented by a finite set of states together with a finite set of computational rules. According to these rules, pushdown automaton changes states, moves the read head on the tape to the right and replaces the top symbol on the pushdown by an arbitrary sequence of another symbols contained in the pushdown alphabet. Every next computational step is performed with respect to the current symbol under the read head on the input tape, the current state and the current symbol on the top of the pushdown. The input string is accepted by the pushdown automaton, if its last symbol is read from the input tape and (1) a state marked as final is reached, or, (2) the pushdown is empty, or, (3) a state marked as final is reached and the pushdown is empty. Note that the acceptance method is defined for every pushdown automaton and these three methods are equivalent. In other words, every pushdown automaton with one of the three acceptance method can be transformed to another two pushdown automatons which accept every string with the other two acceptance methods and define the same language as the original automaton.

The automata theory has modified pushdown automata in many different ways including various modifications concerning their pushdown stores. To give some ex-

amples, we mention some of their modifications. To understand the first one, we introduce the pushdown *turn*. Consider a pushdown automaton and two consecutive moves performed by this automaton. If during the first move the automaton does not shorten its pushdown, while during the second move it shortens it, then the pushdown automaton makes a *turn* during the second move. If the pushdown automaton performs no more than one turn in every computation, it is called as one-turn pushdown automaton. One-turn pushdown automata represent an important restricted version of automata, and the formal language theory has studied their properties in detail (see Section 5.7 in [9] and Section 6.1 in [4]).

Let us introduce another modification of pushdown automata. By attaching an additional pushdown, the pushdown automaton is extended to the two-pushdown automaton. A two-pushdown automaton consists of a finite state control, an input tape with its read head, and two pushdowns. During a move, two-pushdown automaton rewrites the top symbols of both pushdowns; otherwise, it works by analogy with a pushdown automaton. It is proved that two-pushdown automata are more powerful than pushdown automata (see [13]).

By putting together the previously mentioned variations of pushdown automata (i. e. one-turn pushdown automata and two-pushdown automata), the simultaneously one-turn two pushdown automata can be introduced (see [14]). If the two-pushdown automaton makes a turn in both its pushdowns in one computational step, this turn is *simultaneous*. A two-pushdown automaton is simultaneously one-turn if it makes either no turn or one simultaneous turn in its pushdowns during any computation. As expected, this modification changes the power of pushdown automata.

There are another modifications of pushdown automata in the automata theory. Some of them can be found in [5, 6, 7, 8, 14, 15], and [17].

The present paper continues with this vivid topic and introduces automata with two-sided pushdowns. As their name indicates, we can insert symbol into these pushdowns from both ends. The two-sided pushdown automaton thus consists of an input tape with read head, a finite state control, and a two-sided pushdown. Every computational step is performed according to the current symbol under the read head on the input tape, the current state and the current symbols on both tops of the two-sided pushdown.

Pushdowns are usually defined over free monoids generated by the pushdown alphabets of the pushdown automata under the operation of concatenation. However, in this paper, we leave this concept and define these two-sided pushdowns over free groups rather than free monoids. To put it more precisely, we require that during every move, the string representing the current pushdown is over the free group generated by the pushdown alphabet under the operation of concatenation, and the standard group reduction is the only way by which the pushdown string can be shorten. We demonstrate that the pushdown automata modified in this way are as powerful as the Turing machines. In fact, they characterize the family of recursively enumerable languages. Moreover, the free groups are generated by no more than four symbols.

## 2. DEFINITIONS

This paper assumes that the reader is familiar with the language theory and algebra (see [1, 3, 4, 9, 11, 12, 13], and [18]).

Next, this section recalls only the notions used in this paper.

For a set, $W$, $card(W)$ denotes its cardinality. For an alphabet, $V$, $V^*$ represents the free monoid generated by $V$ under the operation of concatenation. Furthermore, $V^\circ$ represents the free group generated by $V$ under the operation of concatenation. The unit of $V^\circ$ is denoted by $\varepsilon$. For every string, $w \in V^\circ$, there is the *inverse string of* $w$, denoted by $\overline{w}$, with the property that $w\overline{w} = \overline{w}w = \varepsilon$. For $w \in V^*$ or $w \in V^\circ$, $|w|$ denotes the length of $w$.

The inverse string of $w = a_1 a_2 \dots a_n$, where $a_i \in V$ for $i = 1, 2, \dots, n$, $n \geq 0$, is defined as $\overline{w} = \overline{a_n a_{n-1}} \dots \overline{a_1}$. The string is said to be *reduced*, if it contains no pairs of the form $x\overline{x}$ or $\overline{x}x$, where $x, \overline{x} \in V^\circ$.

For example, if $V = \{a, b, c, \overline{a}, \overline{b}, \overline{c}\}$, then the inverse string of $bcaa \in V^\circ$ is $\overline{aacb} \in V^\circ$. Because $\overline{a}a = a\overline{a} = \varepsilon$, $\overline{b}b = b\overline{b} = \varepsilon$ and $\overline{c}c = c\overline{c} = \varepsilon$, it is obvious that $bcaaaacb = \overline{aacb}bcaa = \varepsilon$.

A queue grammar (see [2]) is a sextuple, $Q = (V, T, W, F, s, P)$, where $V$ and $W$ are alphabets satisfying $V \cap W = \emptyset$, $T \subseteq V$, $F \subseteq W$, $s \in (V - T)(W - F)$, and $P \subseteq V \times (W - F) \times V^* \times W$ is a finite relation such that for every $a \in V$, there exists an element $(a, b, x, c) \in P$. If $u, v \in V^*W$, $u = arb$, $v = rxc$, $a \in V$, $r, x \in V^*$, $b, c \in W$, and $(a, b, x, c) \in P$, then $u \Rightarrow v[(a, b, x, c)]$ in $Q$, or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of $Q$, $L(Q)$, is defined as $L(Q) = \{w : s \Rightarrow^* wf, w \in T^*, f \in F\}$.

As an example, consider a queue grammar $G = (V, T, W, F, s, P)$, where $V = \{S, A, a, b\}$, $T = \{a, b\}$, $W = \{Q, f\}$, $F = \{f\}$, $s = SQ$ and $P = \{p_1, p_2\}$, $p_1 = (S, Q, Aaa, Q)$ and $p_2 = (A, Q, bb, f)$. Then, there exists a derivation

$$s = SQ \Rightarrow AaaQ[p_1] \Rightarrow aabbf[p_2]$$

in this queue grammar, which generates $aabb$.

A left-extended queue grammar (see [14]) is similar to an ordinary queue grammar except that it records the members of $V$ used when it works. Formally, a left-extended queue grammar is a six-tuple, $Q = (V, T, W, F, s, P)$, where $V, T, W, F$, and $s$ have the same meaning as in a queue grammar. $P \subseteq V \times (W - F) \times V^* \times W$ is a finite relation (as opposed to an ordinary queue grammar, this definition does not require that for every $a \in V$, there exists an element $(a, b, x, c) \in P$). Furthermore, assume that $\# \notin V \cup W$. If $u, v \in V^*\{\#\}V^*W$ so that $u = w\#arb$, $v = wa\#rxc$, $a \in V$, $r, x, w \in V^*$, $b, c \in W$, and $(a, b, x, c) \in P$, then $u \Rightarrow v[(a, b, x, c)]$ in $Q$, or, simply, $u \Rightarrow v$. In the standard manner, extend $\Rightarrow$ to $\Rightarrow^n$, where $n \geq 0$; then, based on $\Rightarrow^n$, define $\Rightarrow^+$ and $\Rightarrow^*$. The language of $Q$, $L(Q)$, is defined as $L(Q) = \{v : \#s \Rightarrow^* w\#vf$ for some $w \in V^*$, $v \in T^*$ and $f \in F\}$.

For example, consider a left-extended queue grammar, which has the same components as the previously mentioned queue grammar $G$. Then, there exists a derivation

$$\#s = \#SQ \Rightarrow S\#AaaQ[p_1] \Rightarrow AS\#aabbf[p_2]$$

in this left-extended queue grammar, which generates *aabb*. Moreover, this type of queue grammar saves symbols from the first components of productions which were used in the derivation.

A string-reading two-sided pushdown automaton over a free group is an eight-tuple, $M = (Q, \Sigma, \Gamma, R, z, Z_1, Z_2, F)$, where $Q$ is a finite set of states, $\Sigma$ is an input alphabet, $\Gamma$ is a pushdown alphabet, $Q \cap (\Sigma \cup \Gamma) = \emptyset$, $R$ is a finite set of rules of the form $u_1|u_2qw \rightarrow v_1|v_2p$ with $u_1, u_2 \in \Gamma$, $v_1, v_2 \in \Gamma^\circ$, $p, q \in Q$, and $w \in \Sigma^*$, $z \in Q$ is the start state, $Z_1 \in \Gamma$ is the start symbol of the left-hand side of the pushdown, $Z_2 \in \Gamma$ is the start symbol of the right-hand side of the pushdown, and $F \subseteq Q$ is a set of final states. A configuration of $M$ is any string of the form $vqy$, where $v \in \Gamma^\circ$, $y \in \Sigma^*$, and $q \in Q$. If $u_1|u_2qw \rightarrow v_1|v_2p \in R$, $y = u_1hu_2qwz$, and $x = v_1hv_2pz$, where $u_1, u_2 \in \Gamma$, $h, v_1, v_2 \in \Gamma^\circ$, $q, p \in Q$, and $w, z \in \Sigma^*$, then $M$ makes a move from $y$ to $x$ in $M$, symbolically written as $y \vdash x[u_1|u_2qw \rightarrow v_1|v_2q]$, or, simply, $y \vdash x$. In the standard manner, extend $\vdash$ to $\vdash^n$, where $n \geq 0$; based on $\vdash^n$, define $\vdash^+$ and $\vdash^*$. We call $Z_1Z_2zw \vdash^* vqx$ a computation, where $v \in \Gamma^\circ$, $q \in Q$, $w, x \in \Sigma^*$; a computation of the form $Z_1Z_2zw \vdash^* \varepsilon f$ with $f \in F$ is a successful computation. The language of $M$, $L(M)$, is defined as $L(M) = \{w : Z_1Z_2zw \vdash^* \varepsilon f$, where $f \in F, w \in \Sigma^*\}$.

A two-sided pushdown automaton over a free group is a string-reading two-sided pushdown automaton over a free group, $M = (Q, \Sigma, \Gamma, R, z, Z_1, Z_2, F)$, in which every $u_1|u_2qw \rightarrow v_1|v_2p \in R$ satisfies $0 \leq |w| \leq 1$, where $u_1, u_2 \in \Gamma$, $v_1v_2 \in \Gamma^\circ$, $q, p \in Q$, and $w \in \Sigma^*$.

## 3. RESULTS

In this section, we study the power of the modified versions of pushdown automata defined in Section 2. We demonstrate that they are as powerful as the Turing machines. In fact, they characterize the family of recursively enumerable languages even if the free groups over which their pushdowns are defined are generated by no more than four symbols.

**Lemma 1.** For every recursively enumerable language, $L$, there exists a left-extended queue grammar, $G = (V, T, W, F, s, P)$, such that $L(G) = L$ and every $(A, q, x, p) \in P$ satisfies $A \in (V - T)$, $q \in (W - F)$, and $x \in ((V - T)^* \cup T^*)$. Formal proof is described in [14].

**Corollary 1.** Let $G = (V, T, W, F, Sq_0, P)$ be a left-extended queue grammar satisfying the properties given in Lemma 1. Grammar $G$ generates every $w \in L(G)$ in this way

$$
\begin{array}{lll}
 & \#Sq_0 & \\
\Rightarrow & x_1\#y_1q_1 & [p_1] \\
\Rightarrow & x_2\#y_2q_2 & [p_2] \\
 & \vdots &
\end{array}
$$

$$
\begin{array}{lll}
 & \vdots & \\
\Rightarrow & x_k \# y_k q_k & [p_k] \\
\Rightarrow & x_{k+1} \# y_{k+1} z_1 q_{k+1} & [p_{k+1}] \\
 & \vdots & \\
\Rightarrow & x_{k+j-1} \# y_{k+j-1} z_{j-1} q_{k+j-1} & [p_{k+j-1}] \\
\Rightarrow & x_{k+j} \# y_{k+j} z_j q_{k+j} & [p_{k+j}] \\
\Rightarrow & x_{k+j} y_{k+j} \# z_{j+1} q_{k+j+1} & [p_{k+j+1}]
\end{array}
$$

where $x_1, \ldots, x_{k+j} \in (V-T)^*$, $y_1, \ldots, y_{k+j-1} \in (V-T)^*$, $y_{k+j} \in (V-T)$, $z_1, \ldots, z_{j+1} \in T^*$, $z_{j+1} = w$, $q_1, \ldots, q_{k+j} \in (W-F)$, $q_{k+j+1} \in F$. $p_1, \ldots, p_k$ are of the form $(A, q, x, p)$, where $A \in (V-T)$, $p, q \in (W-F)$ and $x \in (V-T)^*$. $p_{k+1}, \ldots, p_{k+j}$ are of the form $(A, q, y, p)$, where $A \in (V-T)$, $p, q \in (W-F)$ and $y \in T^*$. The last used production, $p_{k+j+1}$, is of the form $(A, p, y, t)$, where $A \in (V-T)$, $p \in (W-F)$, $y \in T^*$ and $t \in F$.

For the proof of the main result in this paper presented later, we use a left-extended queue grammar, since the derivation method of these grammars is closer to the behaviour of our two-sided pushdowns. Recall that left-extended queue grammars characterize the family of recursively enumerable languages (see [14]). Observe that according to the definition of left-extended queue grammars, every symbol $A \in (V-T)$ which appears in the first component of any rule used in the derivation will be moved from the place right from # to the place left from #. Every string generated right from # in the queue grammar will be inserted into the two-sided pushdown from the right-hand side. Moreover, every symbol moved to the place left from # in the queue grammar is then inserted as inverse from the left-hand side on the two-sided pushdown. At the end of every successful computation, the left half of the two-sided pushdown is equal to the inverted right half, so it can be discharged by the group reduction.

**Theorem 1.** For every left-extended queue grammar, $G = (V, T, W, F, Sq_0, P)$, satisfying the properties described in Lemma 1, there exists a string-reading two-sided pushdown automaton over a free group with the reduced pushdown alphabet, $M = (Q, T, Z, R, z, 1, 1, F_M)$, such that $L(G) = L(M)$.

P r o o f . We construct a string-reading two-sided pushdown automaton over a free group with the reduced pushdown alphabet as follows.

*Construction.* Define the injections, $h : (V-T) \to \{0,1\}^{n+2}$ and $\overline{h} : (V-T) \to \{\overline{0}, \overline{1}\}^{n+2}$, where $n = \lceil \log_2(\underline{card}(V-T)) \rceil$, such that for every $A \in (V-T)$, $h(A) = \{0\}\{0,1\}^n\{0\}$ and $\overline{h}(A) = \overline{h(A)}$. Extend the domain of $h$ to $(V-T)^*$. After this extension, $h$ is now an injective homomorphism from $(V-T)^*$ to $(\{0\}\{0,1\}^n\{0\})^*$. Note that the inverses to $0$ and $1 \in V-T$ are $\overline{0}$ and $\overline{1} \in V-T$, respectively.

Construct the set of states, $Q$, the pushdown alphabet, $Z$, and the set of final states, $F_M$, as $Q = \{f, z\} \cup \{\langle q, 1\rangle, \langle q, 2\rangle | q \in W\}$, $Z = \{0, \overline{0}, 1, \overline{1}\}$, and $F_M = \{f\}$, respectively.

The set of rules, $R$, is constructed in the following way.

1) for the start axiom of $G$, $Sq_0$, where $S \in (V-T)$, $q_0 \in (W-F)$,
   add $1|1z \to 1|h(S)1\langle q_0, 1\rangle$ to $R$
2) for every $(A, q, x, p) \in P$, where $A \in (V-T)$, $p, q \in (W-F)$, $x \in (V-T)^*$,
   add $1|1\langle q, 1\rangle \to 1\overline{h}(A)|h(x)1\langle p, 1\rangle$ to $R$
3) for every $q \in W$
   add $1|1\langle q, 1\rangle \to 1|1\langle q, 2\rangle$ to $R$
4) for every $(A, q, y, p) \in P$, where $A \in (V-T)$, $p, q \in (W-F)$, $y \in T^*$,
   add $1|1\langle q, 2\rangle y \to 1\overline{h}(A)|1\langle p, 2\rangle$ to $R$
5) for every $(A, q, y, t) \in P$, where $A \in (V-T)$, $q \in (W-F)$, $y \in T^*$, $t \in F$,
   add $1|1\langle q, 2\rangle y \to \overline{h}(A)|\varepsilon f$ to $R$

The construction of $M$ is completed. For the next parts of this proof, we introduce the following notation. If $\langle q, 1\rangle$ is the actual state of $M$, we say that $M$ is in *nonterminal-generating* mode. Similarly, if $\langle q, 2\rangle$ is the actual state of $M$, we say that $M$ is in *terminal-reading* mode, where $q \in W$.

*Basic Idea.* $M$ simulates derivations in the left-extended queue grammar, $G$, and codes the symbols from $V-T$ on its pushdown in a binary way. First, consider that in $G$, $w\#Avp$ is the actual sentential form, where $w, v \in (V-T)^*$, $A \in (V-T)$, and $p \in (W-F)$. Then, the corresponding configuration of $M$ is $1\overline{h}(w)h(w)h(A)h(v)1\langle p, 1\rangle\omega$, where $\omega \in T^*$. Let $(A, p, x, q) \in P$, where $x \in (V-T)^*$ Then, $w\#Avp \Rightarrow wA\#vxq$ in $G$. In this case, $M$ must be in the nonterminal-generating mode and the corresponding $M$'s rule is by construction $1|1\langle p, 1\rangle \to 1\overline{h}(A)|h(x)1\langle q, 1\rangle \in R$. By using this rule, $M$ moves to a new configuration of the form $1\overline{h}(A)\overline{h}(w)h(w)h(A)h(v)h(x)1\langle q, 1\rangle\omega$. Observe that $A$ is encoded by $\overline{h}$ and the resulting binary string is inserted to the left-hand side of pushdown. Next, $x$ is encoded by $h$ and the result is inserted into the pushdown from the right-hand side.

Second, let $w\#Avup$ is the actual sentential form, where $u \in T^*$, and $(A, p, y, q) \in P$, $y \in T^*$. Then, $w\#Avup \Rightarrow wA\#vuyq$ in $G$. By construction, the corresponding $M$'s rule is $1|1\langle p, 2\rangle y \to 1\overline{h}(A)|1\langle q, 2\rangle \in R$ and $M$ makes a transition $1\overline{h}(w)h(w)h(A)h(v)1\langle p, 2\rangle y\omega' \vdash 1\overline{h}(A)\overline{h}(w)h(w)h(A)h(v)1\langle q, 2\rangle\omega'$, where $\omega' \in T^*$. Note that in this case, $M$ must be in terminal-reading mode. In this mode, only the encoded $A$, $\overline{h}(A)$, is inserted into the left-hand side of the pushdown.

In other words, every $A \in (V-T)$, that is generated behind the # in $G$, is inserted as $h(A)$ into the right-hand side of the pushdown. Note that all these symbols in the left-extended queue grammar $G$ satisfying the Lemma 1 are moved in front of # during every successful derivation. That is the reason why $M$ inserts their encoded inverses into the left-hand side of the pushdown to correctly simulate the derivation in $G$. To make the pushdown empty, $M$ uses the inverses from the free group.

Let us now present an example of automaton construction for better understanding. Consider a left-extended queue grammar, $G = (V, T, W, F, s, P)$, where $V = \{S, A, B, a, b\}$, $T = \{a, b\}$, $W = \{Q, f\}$, $F = \{f\}$, $s = SQ$ and $P = \{p_1, p_2, p_3\}$, $p_1 = (S, Q, AB, Q)$, $p_2 = (A, Q, aa, Q)$ and $p_3 = (B, Q, bb, f)$. For example, $G$ generates sentence *aabb* by the following derivation.

$$\#s = \#SQ \Rightarrow S\#ABQ[p_1] \Rightarrow SA\#BaaQ[p_2] \Rightarrow SAB\#aabbf[p_3]$$

We construct a string-reading two-sided pushdown automaton over a free group with the reduced pushdown alphabet, $M = (Q, T, Z, R, z, 1, 1, F_M)$, as follows:

- $Q = \{z, f, \langle Q, 1\rangle\ \langle Q, 2\rangle\}$,
- $T = \{a, b\}$,
- $Z = \{0, \overline{0}, 1, \overline{1}\}$,
- $R = \{$

| | | |
|---|---|---|
| $p_1$: | $1\|1z \to 1\|h(S)1\langle Q, 1\rangle$ | for the start axiom $SQ$, |
| $p_2$: | $1\|1\langle Q, 1\rangle \to 1\overline{h}(S)\|h(AB)1\langle Q, 1\rangle$ | for $(S, Q, AB, Q) \in P$, |
| $p_3$: | $1\|1\langle Q, 1\rangle \to 1\|1\langle Q, 2\rangle$ | for $Q \in W$, |
| $p_4$: | $1\|1\langle Q, 2\rangle aa \to 1\overline{h}(A)\|1\langle Q, 2\rangle$ | for $(A, Q, aa, Q) \in P$, |
| $p_5$: | $1\|1\langle Q, 2\rangle bb \to \overline{h}(B)\|\varepsilon f$ | for $(B, Q, bb, f) \in P\}$, |

- $F_M = \{f\}$

Encoding $h$ of symbols from $(V - T)$:

$$h(S) = 00010 \quad \overline{h}(S) = \overline{01000}$$
$$h(A) = 00110 \quad \overline{h}(A) = \overline{01100}$$
$$h(B) = 01000 \quad \overline{h}(B) = \overline{00010}$$

Now, observe the acceptance progress of the string $aabb$ by $M$.

| two-sided pushdown | state | input | production |
|---|---|---|---|
| $11$ | $z$ | $aabb$ | |
| $1h(S)1$ | $\langle Q, 1\rangle$ | $aabb$ | $p_1$ |
| $1\overline{h}(S)h(S)h(AB)1$ | $\langle Q, 1\rangle$ | $aabb$ | $p_2$ |
| $1h(A)h(B)1$ | $\langle Q, 1\rangle$ | $aabb$ | free group reduction |
| $1h(A)h(B)1$ | $\langle Q, 2\rangle$ | $aabb$ | $p_3$ |
| $1\overline{h}(A)h(A)h(B)1$ | $\langle Q, 2\rangle$ | $bb$ | $p_4$ |
| $1h(B)1$ | $\langle Q, 2\rangle$ | $bb$ | free group reduction |
| $\overline{h}(B)h(B)$ | $f$ | $\varepsilon$ | $p_5$ |
| $\varepsilon$ | $f$ | $\varepsilon$ | free group reduction |

Since the two-sided pushdown is empty and the current state $f$ is the final state, $aabb$ is accepted by $M$.

Rigorous proof. Next, we prove $L(G) = L(M)$, thus $L(G) \subseteq L(M)$ and $L(M) \subseteq L(G)$. First, we demonstrate Claims A, B and C to prove $L(G) \subseteq L(M)$.

***Claim A.*** If $A_1 \dots A_n \# B_1 \dots B_m u \Rightarrow^i A_1 \dots A_n B_1 \dots B_i \# B_{i+1} \dots B_m x_1 \dots x_i p$ in $G$, then $1\overline{h}(A_n) \dots \overline{h}(A_1) h(A_1) \dots h(A_n) h(B_1) \dots h(B_m) 1\langle u, 1\rangle \omega \vdash^i 1\overline{h}(B_i) \dots \overline{h}(B_1) \overline{h}(A_n) \dots \overline{h}(A_1) h(A_1) \dots h(A_n) h(B_1) \dots h(B_m) h(x_1) \dots h(x_i) 1\langle p, 1\rangle \omega$ in $M$, where $A_1, \dots, A_n, B_1, \dots, B_m \in (V - T)$, $x_1, \dots, x_i \in (V - T)^*$, $u, p \in (W - F)$, $n \geq 0$, $\omega \in T^*$, $0 \leq i \leq m$.

*Basis.* Let $i = 0$. Then $A_1 \dots A_n \# B_1 \dots B_m u \Rightarrow^0 A_1 \dots A_n \# B_1 \dots B_m u$ in $G$.

Clearly, $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega \vdash^0 1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega$ in $M$.

*Induction Hypothesis.* Assume that Claim A holds for every $i \leq l$, where $l$ is a positive integer.

*Induction Step.* Consider any derivation of the form $A_1\dots A_n\#B_1\dots B_mu \Rightarrow^{l+1} A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_mx_1\dots x_lx_{l+1}q$. Express this derivation as $A_1\dots A_n\#B_1\dots B_mu \Rightarrow^{l} A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_mx_1\dots x_lp \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_mx_1\dots x_lx_{l+1}q$ in $G$, where $0 \leq l \leq m$, $q \in (W-F)$.

By the induction hypothesis, $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle \omega \vdash^l 1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)1\langle p,1\rangle\omega \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)h(x_{l+1})1\langle q,1\rangle\omega$ in $M$. There is only one type of productions in $P$ able to perform the derivation $A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_mx_1\dots x_lp \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_mx_1\dots x_lx_{l+1}q$ in $G$, namely productions of the form $(B_{l+1},p,x_{l+1},q) \in P$, where $B_{l+1} \in (V-T)$, $p,q \in (W-F)$ and $x_{l+1} \in (V-T)^*$. Observe that by point 2 in construction, there is a rule $1|1\langle p,1\rangle \rightarrow 1\overline{h}(B_{l+1})|h(x_{l+1})1\langle q,1\rangle$ in $R$, so $1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)1\langle p,1\rangle\omega \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)h(x_{l+1})1\langle q,1\rangle\omega$ in $M$ and Claim A holds. □

***Claim B.*** If $A_1\dots A_n\#B_1\dots B_ma_1\dots a_ku \Rightarrow^i A_1\dots A_nB_1\dots B_i\#B_{i+1}\dots B_m a_1\dots a_kb_1\dots b_ip$ in $G$, then $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^i 1\overline{h}(B_i)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_i)h(B_{i+1})\dots h(B_m)1\langle p,2\rangle b_{i+1}\dots b_j$ in $M$, where $A_1,\dots,A_n,B_1,\dots,B_m \in (V-T)$, $a_1,\dots,a_k$, $b_1,\dots,b_j \in T^*$, $u,p \in (W-F)$, $0 \leq k$, $0 \leq i \leq j \leq m$.

*Basis.* Let $i=0$. Then $A_1\dots A_n\#B_1\dots B_ma_1\dots a_ku \Rightarrow^0 A_1\dots A_n\#B_1\dots B_m a_1\dots a_ku$ in $G$. Clearly, $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^0 1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j$ in $M$.

*Induction Hypothesis.* Assume that Claim B holds for every $i \leq l$, where $l$ is a positive integer.

*Induction Step.* Consider any derivation of the form $A_1\dots A_n\#B_1\dots B_ma_1\dots a_ku \Rightarrow^{l+1} A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_ma_1\dots a_kb_1\dots b_lb_{l+1}q$ and express this derivation as $A_1\dots A_n\#B_1\dots B_ma_1\dots a_ku \Rightarrow^l A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_ma_1\dots a_k b_1\dots b_lp \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_ma_1\dots a_kb_1\dots b_lb_{l+1}q$ in $G$, where $0 \leq k$, $0 \leq l \leq m$, $q \in (W-F)$.

By the induction hypothesis, $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^l 1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle p,2\rangle b_{l+1}\dots b_j \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle q,2\rangle b_{l+2}\dots b_j$ in $M$.

In this case, there is only one possibility how $G$ can make the derivation

$A_1 \dots A_n B_1 \dots B_l \# B_{l+1} \dots B_m a_1 \dots a_k b_1 \dots b_l p \Rightarrow A_1 \dots A_n B_1 \dots B_l B_{l+1}$ $\# B_{l+2} \dots B_m a_1 \dots a_k b_1 \dots b_l b_{l+1} q$. Observe that it is done by a production of the form $(B_{l+1}, p, b_{l+1}, q) \in P$, where $B_{l+1} \in (V - T)$, $p, q \in (W - F)$, $b_{l+1} \in T^*$. By point 4 in construction, there is a rule $1|1\langle p, 2\rangle b_{l+1} \to 1\overline{h}(B_{l+1})|1\langle q, 2\rangle$ in $R$, so $1\overline{h}(B_l) \dots \overline{h}(B_1)\overline{h}(A_n) \dots \overline{h}(A_1)h(A_1) \dots h(A_n)h(B_1) \dots h(B_m)1\langle p, 2\rangle b_{l+1} \dots b_j \vdash$ $1\overline{h}(B_{l+1})\overline{h}(B_l) \dots \overline{h}(B_1)\overline{h}(A_n) \dots \overline{h}(A_1)h(A_1) \dots h(A_n)h(B_1) \dots h(B_m)1\langle q, 2\rangle b_{l+2} \dots$ $b_j$ in $M$ and Claim B holds. □

***Claim C.*** If $A_1 \dots A_{n-1} \# A_n yq \Rightarrow A_1 \dots A_{n-1} A_n \# yzt$ in $G$, where $A_1, \dots, A_n \in (V - T)$, $y, z \in T^*$, $q \in (W - F)$, $t \in F$, then $1\overline{h}(A_{n-1}) \dots \overline{h}(A_1)h(A_1) \dots h(A_n)1\langle q, 2\rangle$ $z \vdash \overline{h}(A_n) \dots \overline{h}(A_1)h(A_1) \dots h(A_n)f = \varepsilon f$ in $M$, where $f \in F_M$.

Grammar $G$ performs the described derivation by a production of the form $(A_n, q, z, t) \in P$, where $A_n \in (V - T)$, $z \in T^*$, $q \in (W - F)$, $t \in F$. By point 5 of construction, there is a rule $1|1\langle q, 2\rangle z \to \overline{h}(A_n)|\varepsilon f$ in $R$, so the corresponding computational step described in Claim C indeed occurs in $M$, so Claim C holds. □

Claims A, B and C prove that $L(G) \subseteq L(M)$. Next, we demonstrate Claims D, E and F to prove $L(M) \subseteq L(G)$.

***Claim D.*** Automaton $M$ accepts every $w \in L(M)$ in this way

$11zw_1w_2 \dots w_r \vdash$
$1h(S)1\langle q_0, 1\rangle w_1w_2 \dots w_r \vdash$
$1\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)1\langle q_1, 1\rangle w_1w_2 \dots w_r \vdash$
$1\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)1\langle q_2, 1\rangle w_1w_2 \dots w_r \vdash$
$1\overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)h(X_1^3)h(X_2^3) \dots$
$h(X_{n_3}^3)1\langle q_3, 1\rangle w_1w_2 \dots w_r \vdash \dots$
$1\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)$
$h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)1\langle q_m, 1\rangle w_1w_2 \dots w_r \vdash$
$1\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)$
$h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)1\langle q_m, 2\rangle w_1w_2 \dots w_r \vdash$
$1\overline{h}(X_{j+1}^k)\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots$
$h(X_{n_2}^2)h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)1\langle q_{m+1}, 2\rangle w_2 \dots w_r \vdash$
$1\overline{h}(X_{j+2}^k)\overline{h}(X_{j+1}^k)\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots h(X_{n_1}^1)h(X_1^2)h(X_2^2)$
$\dots h(X_{n_2}^2)h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)1\langle q_{m+2}, 2\rangle w_3 \dots w_r \vdash$
$\dots$
$1\overline{h}(X_{n_m-1}^m) \dots \overline{h}(X_{j+2}^k)\overline{h}(X_{j+1}^k)\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots$
$h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots$
$\dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)1\langle q_{m+r-1}, 2\rangle w_r \vdash$
$\overline{h}(X_{n_m}^m)\overline{h}(X_{n_m-1}^m) \dots \overline{h}(X_{j+2}^k)\overline{h}(X_{j+1}^k)\overline{h}(X_j^k) \dots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \dots$
$h(X_{n_1}^1)h(X_1^2)h(X_2^2) \dots h(X_{n_2}^2)h(X_1^3)h(X_2^3) \dots h(X_{n_3}^3) \dots$
$\dots h(X_1^m)h(X_2^m) \dots h(X_{n_m}^m)f = \varepsilon f$

where $w = w_1w_2 \dots w_r$, $r \geq 1$, $w_1, \dots, w_r \in T^*$, $q_0, q_1, \dots, q_{m+r-1} \in (W - F)$, $X_1^1, \dots, X_{n_1}^1, X_1^2, \dots, X_{n_2}^2, \dots, X_1^m, \dots, X_{n_m}^m \in (V - T)$, $n_1, n_2, \dots, n_m \geq 0$, $0 \leq k \leq m$.

Proof of Claim D. We examine steps 1 through 5 of the construction of $R$. Note that in every successful computation, $M$ uses rules created in step $b$ before it uses rules created in step $b+1$, for $b = 1, \ldots, 4$.

In the first computational step, $M$ applies the production $1|1z \to 1|h(S)1\langle q_0, 1\rangle$ introduced in 1, where $Sq_0$ is the axiom of $G$. This is the only way by which $M$ can make the transition $11zw_1w_2 \ldots w_r \vdash 1h(S)1\langle q_0, 1\rangle w_1w_2 \ldots w_r$. Observe that this production is used exactly once during one successful computation. By this step, automaton is switched to the nonterminal-generating mode.

In the next part of computation, namely

$$1h(S)1\langle q_0, 1\rangle w_1w_2 \ldots w_r \vdash^*$$
$$1\overline{h}(X_j^k) \ldots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \ldots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \ldots h(X_{n_2}^2)$$
$$h(X_1^3)h(X_2^3) \ldots h(X_{n_3}^3) \ldots h(X_1^m)h(X_2^m) \ldots h(X_{n_m}^m)1\langle q_m, 1\rangle w_1w_2 \ldots w_r$$

$M$ uses rules of the form $1|1\langle q, 1\rangle \to 1\overline{h}(A)|h(x)1\langle p, 1\rangle$ constructed in 2, where $A \in (V-T)$, $x \in (V-T)^*$, $p, q \in (W-F)$. This part of computation is characterized by $M$'s states of the form $\langle q, 1\rangle$, $q \in (W-F)$. For the more detailed proof of this part, see Claim E.

By the next computational step,

$$1\overline{h}(X_j^k) \ldots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \ldots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \ldots h(X_{n_2}^2)$$
$$h(X_1^3)h(X_2^3) \ldots h(X_{n_3}^3) \ldots h(X_1^m)h(X_2^m) \ldots h(X_{n_m}^m)1\langle q_m, 1\rangle w_1w_2 \ldots w_r \vdash$$

$$1\overline{h}(X_j^k) \ldots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \ldots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \ldots h(X_{n_2}^2)$$
$$h(X_1^3)h(X_2^3) \ldots h(X_{n_3}^3) \ldots h(X_1^m)h(X_2^m) \ldots h(X_{n_m}^m)1\langle q_m, 2\rangle w_1w_2 \ldots w_r$$

$M$ switches to the terminal-reading mode by a rule of the form $1|1\langle q, 1\rangle \to 1|1\langle q, 2\rangle$ constructed in 3. Observe that this production is used exactly once during one successful computation. Since this production changes an actual state of automaton of the form $\langle q, 1\rangle$ to the state of the form $\langle q, 2\rangle$, $q \in (W-F)$, there is no further possibility of using any productions constructed in parts 1 through 3.

In the next part of computation, namely

$$1\overline{h}(X_j^k) \ldots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \ldots h(X_{n_1}^1)h(X_1^2)h(X_2^2) \ldots h(X_{n_2}^2)$$
$$h(X_1^3)h(X_2^3) \ldots h(X_{n_3}^3) \ldots h(X_1^m)h(X_2^m) \ldots h(X_{n_m}^m)1\langle q_m, 2\rangle w_1w_2 \ldots w_r \vdash^*$$

$$1\overline{h}(X_{n_m-1}^m) \ldots \overline{h}(X_{j+2}^k)\overline{h}(X_{j+1}^k)\overline{h}(X_j^k) \ldots \overline{h}(X_2^1)\overline{h}(X_1^1)\overline{h}(S)h(S)h(X_1^1)h(X_2^1) \ldots$$
$$h(X_{n_1}^1)h(X_1^2)h(X_2^2) \ldots h(X_{n_2}^2)h(X_1^3)h(X_2^3) \ldots h(X_{n_3}^3) \ldots$$
$$\ldots h(X_1^m)h(X_2^m) \ldots h(X_{n_m}^m)1\langle q_{m+r-1}, 2\rangle w_r$$

$M$ uses rules constructed in 4 and reads input strings of terminals. The detailed proof of this part of computation is described in Claim F.

The last computational step switches $M$ to the final state. It is done by a rule of the form $1|1\langle q, 2\rangle y \to \overline{h}(A)|\varepsilon f$ constructed in 5, where $q \in (W-T)$, $y \in T^*$, $A \in (V-T)$ and $f \in F_M$. After that, if the two-sided pushdown is empty by a group reduction and the input string is read, then $M$ accepts the input string. Otherwise, the input string is not accepted, since there is no rule with the left-hand side of the form $1|1fy$, where $f \in F_M$, $y \in T^*$, so Claim D holds. □

***Claim E.*** If $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega \vdash^i 1\overline{h}(B_i)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_i)1\langle p,1\rangle\omega$ in $M$, then $A_1\dots A_n\#B_1\dots B_m u \Rightarrow^i A_1\dots A_nB_1\dots B_i\#B_{i+1}\dots B_m x_1\dots x_i p$ in $G$, where $A_1,\dots,A_n,B_1,\dots,B_m \in (V-T)$, $x_1,\dots,x_i \in (V-T)^*$, $u,p \in (W-F)$, $0 \le i \le m$.

*Basis.* Let $i=0$. Then $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega \vdash^0 1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega$ in $M$. Clearly, $A_1\dots A_n\#B_1\dots B_m u \Rightarrow^0 A_1\dots A_n\#B_1\dots B_m u$ in $G$.

*Induction Hypothesis.* Assume that Claim E holds for every $i \le l$, where $l$ is a positive integer.

*Induction Step.* Consider any computation of the form $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega \vdash^{l+1} 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)h(x_{l+1})1\langle q,1\rangle\omega$ and express this derivation as $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,1\rangle\omega \vdash^l 1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)1\langle p,1\rangle\omega \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)h(x_{l+1})1\langle q,1\rangle\omega$ in $M$, where $q \in (W-F)$, $0 \le l \le m$.

By the induction hypothesis, $A_1\dots A_n\#B_1\dots B_m u \Rightarrow^l A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_m x_1\dots x_l p \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_m x_1\dots x_l x_{l+1} q$ in $G$. There is only one type of productions in $R$ able to perform the computation $1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)1\langle p,1\rangle\omega \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)h(x_1)\dots h(x_l)h(x_{l+1})1\langle q,1\rangle\omega$ in $M$, namely productions of the form $1|1\langle p,1\rangle \to 1\overline{h}(B_{l+1})|h(x_{l+1})1\langle q,1\rangle \in R$. Observe that by construction, there is a rule $(B_{l+1},p,x_{l+1},q)$ in $P$, so $A_1\dots A_n\#B_1\dots B_m u \Rightarrow^l A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_m x_1\dots x_l p \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_m x_1\dots x_l x_{l+1} q$ in $G$ and Claim E holds. □

***Claim F.*** If $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^i 1\overline{h}(B_i)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_i)h(B_{i+1})\dots h(B_m)1\langle p,2\rangle b_{i+1}\dots b_j$ in $M$, then $A_1\dots A_n\#B_1\dots B_m a_1\dots a_k u \Rightarrow^i A_1\dots A_nB_1\dots B_i\#B_{i+1}\dots B_m a_1\dots a_k b_1\dots b_i p$ in $G$, where $A_1,\dots,A_n,B_1,\dots,B_m \in V-T$, $a_1,\dots,a_k$, $b_1,\dots,b_j \in T^*$ and $p,u \in W-F$, $0 \le i \le m$.

*Basis.* Let $i=0$. Then $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^0 1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j$ in $M$. Clearly, $A_1\dots A_n\#B_1\dots B_m a_1\dots a_k u \Rightarrow^0 A_1\dots A_n\#B_1\dots B_m a_1\dots a_k u$ in $G$.

*Induction Hypothesis.* Assume that Claim F holds for every $i \le l$, where $l$ is a positive integer.

*Induction Step.* Consider any computation of the form
$1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^{l+1}$
$1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle q,2\rangle$
$b_{l+2}\dots b_j$ and express this derivation as $1\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots$
$h(B_m)1\langle u,2\rangle b_1\dots b_j \vdash^l 1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots$
$h(B_m)1\langle p,2\rangle b_{l+1}\dots b_j \vdash 1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)$
$h(B_1)\dots h(B_m)1\langle q,2\rangle b_{l+2}\dots b_j$ in $M$, where $0\leq l\leq j\leq m$, $q\in(W-F)$.

By the induction hypothesis, $A_1\dots A_n\#B_1\dots B_m a_1\dots a_k u \Rightarrow^l$
$A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_m a_1\dots a_k b_1\dots b_l p \Rightarrow$
$A_1\dots A_nB_1\dots B_lB_{l+1}\#B_{l+2}\dots B_m a_1\dots a_k b_1\dots b_l b_{l+1} q$ in $G$, where $0\leq l\leq m$.

In this case, there is the only way by which $M$ can make the computational step
$1\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle p,2\rangle b_{l+1}\dots b_j \vdash$
$1\overline{h}(B_{l+1})\overline{h}(B_l)\dots\overline{h}(B_1)\overline{h}(A_n)\dots\overline{h}(A_1)h(A_1)\dots h(A_n)h(B_1)\dots h(B_m)1\langle q,2\rangle$
$b_{l+2}\dots b_j$. Observe that it is done by a production of the form $1|1\langle p,2\rangle b_{l+1}$
$\to 1\overline{h}(B_{l+1})|1\langle q,2\rangle \in R$. By point 4 in construction, there is a production
$(B_{l+1},p,b_{l+1},q)\in P$ where $B_{l+1}\in(V-T)$, $p,q\in(W-F)$, $b_{l+1}\in T^*$, so
$A_1\dots A_nB_1\dots B_l\#B_{l+1}\dots B_m a_1\dots a_k b_1\dots b_l p \Rightarrow A_1\dots A_nB_1\dots B_lB_{l+1}\#$
$B_{l+2}\dots B_m a_1\dots a_k b_1\dots b_l b_{l+1} q$ in $G$ and Claim F holds. □

By Claims D, E and F, we proved that $L(M)\subseteq L(G)$. As a result, $L(G)=L(M)$, so Theorem 1 is proved.

**Theorem 2.** For every string-reading two-sided pushdown automaton over a free group with the reduced pushdown alphabet, $Q'$, there exists a two-sided pushdown automaton over a free group with the reduced pushdown alphabet, $Q$, such that $L(Q')=L(Q)$.

P r o o f. The formal proof of this theorem is simple and left to the reader. □

## 4. CONCLUSIONS

In this paper, we proved that the power of two-sided pushdown automata with pushdowns defined over free groups is equal to the power of Turing machines, so these automata generate the whole family of recursively enumerable languages. Moreover, the pushdown alphabet contains no more than four symbols. Note that the same result can be also reached with two-sided pushdowns defined over free monoids. This modification affects only the set of rules with their construction, and it is left for the reader.

Another modifications of pushdown automata have been studied in theory of automata and formal languages. We can mention simultaneously one-turn two-pushdown automata introduced in [14], regulated pushdown automata described in

[15], or finite-turn pushdown automata (see [8]). Very simple and natural modification of pushdown automata is also presented in [5] and [4], where there is the ability for pushdown reversal added. The main goal of all these modifications is to increase the generative power of ordinary pushdown automata. In our paper, we significantly increased the power and moreover, the number of transition rules was reduced by defining of the two-sided pushdowns over free groups.

ACKNOWLEDGEMENT

This work was supported by the Czech Science Foundation under Grant 201/04/0441.

(Received February 16, 2006.)

REFERENCES

[1] N. Jacobson: Basic Algebra. Second edition. W. H. Freeman, New York 1989.
[2] H. C. M. Kleijn and G. Rozenberg: On the generative power of regular pattern grammars. Acta Informatica *20* (1983), 391–411.
[3] A. V. Aho and J. D. Ullman: The Theory of Parsing, Translation and Compiling. Volume I: Parsing. Prentice Hall, Englewood Cliffs, NJ 1972.
[4] J. Autebert, J. Berstel, and L. Boasson: Context-Free Languages and Pushdown Automata. In: Handbook of Formal Languages (G. Rozenberg, and A. Salomaa, eds.), Springer, Berlin 1997.
[5] B. Courcelle: On jump deterministic pushdown automata. Math. Systems Theory *11* (1977), 87–109.
[6] S. A. Greibach: Checking automata and one-way stack languages. J. Comput. Systems Sci. *3* (1969), 196–217.
[7] S. Ginsburg, S. A. Greibach, and M. A. Harrison: One-way stack automata. J. Assoc. Comput. Mach. *14* (1967), 389–418.
[8] S. Ginsburg and E. Spanier: Finite-turn pushdown automata. SIAM J. Control *4* (1968), 429–453.
[9] M. A. Harrison: Introduction to Formal Language Theory. Addison-Wesley, Reading, Mass. 1978
[10] M. Holzer and M. Kutrib: Flip-pushdown automata: $k + 1$ pushdown reversals are better than $k$. In: Languages and Programming – ICALP 2003 (Lecture Notes in Computer Science 2719), Springer, Berlin 2003, pp. 490–501.
[11] H. R. Lewis and C. H. Papadimitriou: Elements of the Theory of Computation. Prentice-Hall, Englewood Cliffs, NJ 1981.
[12] J. C. Martin: Introduction to Languages and the Theory of Computation. McGraw-Hill, New York 1991.
[13] A. Meduna: Automata and Languages: Theory and Applications. Springer, London 2000.
[14] A. Meduna: Simultaneously one-turn two-pushdown automata. Internat. Computer Math. *82* (2003), 679–687.
[15] A. Meduna and D. Kolář: Regulated pushdown automata. Acta Cybernet. *14* (2000), 653–664.
[16] J. Sakarovitch: Pushdown automata with terminating languages. In: Languages and Automata Symposium, *RIMS 421* (1981), 15–29.
[17] P. Sarkar: Pushdown automaton with the ability to flip its stack. TR01-081, Electronic Colloquium on Computational Complexity (ECCC), November 2001.

[18] T. A. Sudkamp: Languages and Machines. Addison Wesley, Reading, Mass. 1988.
[19] L. Valiant: The equivalence problem for ceterministic finite turn pushdown automata. Inform. and Control *81* (1989), 265–279.

*Petr Blatný, Radek Biblo, and Alexander Meduna, Brno University of Technology, Faculty of Information Technology, Department of Information Systems, Božetěchova 2, 612 66 Brno. Czech Republic.*
*e-mails: blatny@fit.vutbr.cz, bidlor@fit.vutbr.cz, meduna@fit.vutbr.cz*