Theoretical and Applied Informatics
Vol. 27 (2015), no. 1, pp. 25–47
DOI: 10.20904/271025

Ordered Pure Multi-Pushdown Automata

Alexander Meduna[1*] Ondřej Soukup[1†] Petr Zemek[1‡]

[1]Brno University of Technology, Faculty of Information Technology,
IT4I Centre of Excellence, Božetěchova 1/2, 612 66 Brno, Czech Republic

Abstract In the presented paper we discuss pure versions of pushdown automata that have no extra non-input symbols. More specifically, we study pure multi-pushdown automata, which have several pushdown lists. We restrict these automata by the total orders defined over their pushdowns or alphabets and determine the accepting power of the automata restricted in this way. Moreover, we explain the significance of the achieved results and relate them to some other results in the automata theory.

Keywords pure multi-pushdown automata; total orders; accepting power;

Received 11 JAN 2016 **Revised** 02 MAR 2016 **Accepted** 03 APR 2016

1 Introduction

Formal language theory introduced pure grammars — grammars that have only terminal symbols without any extra non-terminal symbols [1, 2, 3] — several decades ago, and it has investigated them since. However, while most language-generating grammatical models have their accepting counterparts based upon automata, pure grammars lacked them. Therefore, to fill this gap, the language theory has recently introduced and discussed pure versions of pushdown automata — pushdown automata that do not possess any extra pushdown non-input symbols [4, 5] — because their classical versions, in which non-input symbols may occur, fullfil a crucially important role in automata theory and its applications, such as parsing.

More specifically, multi-pushdown automata, which may have several pushdown lists, were investigated intensively [6, 7, 8, 9, 10, 11, 12], incuding their pure versions [4, 5]. The presented paper continues the study of pure multi-pushdown automata by introducing and investigating three new versions.

*E-mail: meduna@fit.vutbr.cz
†E-mail: isoukup@fit.vutbr.cz
‡E-mail: izemek@fit.vutbr.cz

First, we discuss ordinary pure multi-pushdown automata. It is proved that their one-pushdown versions characterize the family of context-free languages, while their two-or-more-pushdown versions are computational complete — they are as powerful as Turing machines.

Second, we restrict the way these automata work by introducing total order $\preceq$ over their pushdowns. In essence, during any computation, after using pushdown i, only pushdown j satisfying $i \preceq j$ can be used throughout the rest of the computation. The acceptance of an input is successfully completed by emptying all its pushdowns and entering a final state. The paper demonstrates that these pushdown automata with any number of pushdown lists characterize the family of context-free languages.

Finally, we introduce and study the total order $\trianglelefteq$ over the alphabets of these automata. We require that during any computation, the symbols of any pushdown string are ordered according to $\trianglelefteq$. The study proves that the automata restricted in this way define only a proper subfamily of the family of context-free languages.

The above-mentioned concepts and results are of some interest in view of other studies on the same subject, including the publications summarized next. Multi-pushdown automata have been defined and studied in [13]; additionally, this paper have studied their stateless versions. In [5], pure multi-pushdown automata that perform complete pushdown pops are introduced and studied. Papers [14, 15, 16, 17] from the area of formal verification study multi-pushdown automata with ordered pushdowns, where the pop operation can be performed only on the first nonempty pushdown.

2 Preliminaries

We assume that the reader is familiar with formal language theory (see [18, 19, 20]). For every positive integer n, let I_n denote the set $\{1, 2, \ldots, n\}$. For a set P, $\mathrm{card}(P)$ denotes the cardinality of P. A binary relation $\preceq$ over P is a *total order* if and only if $\preceq$ is antisymmetric, transitive, and total. For an alphabet (finite nonempty set) V, V^* represents the free monoid generated by V under the operation of concatenation. The unit of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V under the operation of concatenation. If $\mathrm{card}(V) = 1$, then V is a *unary alphabet*. For $x \in V^*$, $|x|$ denotes the length of x, $\mathrm{reversal}(x)$ denotes the reversal (mirror image) of x, and $\mathrm{alph}(x)$ denotes the set of symbols occurring in x. For $K \subseteq V^*$, we define $\mathrm{alph}(K) = \bigcup_{x \in K} \mathrm{alph}(x)$. If $\mathrm{card}\big(\mathrm{alph}(K)\big) = 1$, then K is a *unary language*.

A *pushdown automaton* (*PDA*) is a septuple

$$M = \big(Q, \Sigma, \Gamma, R, s, Z, F\big),$$

where Q is a finite set, Σ is an alphabet such that $Q \cap \Sigma = \emptyset$, Γ is an alphabet such that $\Sigma \subset \Gamma$, $R \subseteq \Gamma^* \times Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^* \times Q$ is a finite relation, $Z \in \Gamma - \Sigma$, $s \in Q$, and $F \subseteq Q$. The components Q, Σ, Γ, R, s, Z, and F are called the set of *states*, the *input alphabet*, the *pushdown alphabet*, the set of *rules*, the *start state*, the *initial pushdown symbol*, and the set of *final states*, respectively. Instead of $(z, p, a, w, q) \in R$, we write $zpa \to wq \in R$ throughout the paper. Let $\# \notin Q \cup \Sigma \cup \Gamma$ be a *bottom marker*. The *direct move relation* over $\{\#\}\Gamma^* Q \Sigma^*$, symbolically denoted by $\vdash_M$, is defined as follows: $\#yzpax \vdash_M \#ywqx$ in M if and only if $\#yzpax, \#ywqx \in \{\#\}\Gamma^* Q \Sigma^*$

and $zpa \to wq \in R$. Let $\vdash_M^m$ and $\vdash_M^*$ denote the mth power of $\vdash_M$, for some $m \geq 1$, and the reflexive-transitive closure of $\vdash_M$, respectively. The *language accepted by* M is denoted by $L(M)$ and defined as

$$L(M) = \{w \in \Sigma^* \mid \#Zsw \vdash_M^* \#f, f \in F\}$$

A *two-pushdown automaton* (2-*PDA*) [11] is an octuple

$$M = (Q, \Sigma, \Gamma, R, s, Z_1, Z_2, F).$$

where Q, Σ, Γ, s, and F are defined as in a pushdown automaton, $R \subseteq \Gamma \times \Gamma \times Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^* \times \Gamma^* \times Q$ is a finite relation, and $Z_1, Z_2 \in \Gamma - \Sigma$. The components Q, Σ, Γ, R, s, Z_1, Z_2, and F are called the set of *states*, the *input alphabet*, the *pushdown alphabet*, the set of *rules*, the *start state*, the *initial symbol of pushdown 1*, the *initial symbol of pushdown 2*, and the set of *final states*, respectively. Instead of $(A_1, A_2, p, a, w_1, w_2, q) \in R$, we write $A_1|A_2pa \to w_1|w_2q \in R$ throughout the paper. Let $\# \notin Q \cup \Sigma \cup \Gamma$ be a *bottom marker*. The *direct move relation* over $\{\#\}\Gamma^*\{\#\}\Gamma^*Q\Sigma^*$, symbolically denoted by $\vdash_M$, is defined as follows: $\#y_1A_1\#y_2A_2pax \vdash_M \#y_1w_1\#y_2w_2qx$ in M if and only if $\#y_1A_1\#y_2A_2pax$, $\#y_1w_1\#y_1w_1qx \in \{\#\}\Gamma^*\{\#\}\Gamma^*Q\Sigma^*$ and $A_1|A_2pa \to w_1|w_2q \in R$. Let $\vdash_M^m$ and $\vdash_M^*$ denote the mth power of $\vdash_M$, for some $m \geq 1$, and the reflexive-transitive closure of $\vdash_M$, respectively. The *language accepted by* M is denoted by $L(M)$ and defined as

$$L(M) = \{w \in \Sigma^* \mid \#Z_1\#Z_2sw \vdash_M^* \#\#f, f \in F\}.$$

The families of regular languages, context-free languages, and recursively enumerable languages are denoted by **RG**, **CF**, and **RE**, respectively. Recall that pushdown automata characterize **CF** and that two-pushdown automata characterize **RE**.

For every unary language L, $L \in$ **RG** [21], there exists a finite automaton accepting L — there is no need for any pushdown storage. Thus, in what follows, we suppose every lagnuage is non-unary.

3 Pure Multi-Pushdown Automata

In this section, we define pure multi-pushdown automata and prove that with a single pushdown, they characterize the family of context-free languages. Then, we prove that pure pushdown automata with two or more pushdowns characterize the family of recursively enumerable languages — that is, they are Turing complete.

3.1 Definitions and Examples

First, we define pure multi-pushdown automata. Next, we illustrate them by an example.

Definition 1. A *pure n-pushdown automaton* (*n-PPDA*), where $n \geq 1$, is a $(5+n)$-tuple

$$M = (Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F),$$

where

- Q is a finite set of *states*;
- Σ is the *input alphabet* ($Q \cap \Sigma = \emptyset$);
- R is a finite set of *rules* of the form

$$izpa \to wq,$$

  where $i \in I_n$, $p, q \in Q$, and $z, w \in \Sigma^*$;
- $s \in Q$ is the *start state*;
- $Z_i \in \Sigma$ is the *initial symbol of pushdown* i, for $i = 1, 2, \ldots, n$;
- $F \subseteq Q$ is a set of *final states*. □

Definition 2. Let $M = (Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F)$ be an n-PPDA, for some $n \geq 1$. Let $\# \notin Q \cup \Sigma$ be a *bottom marker*. The *direct move relation* over $\big(\{\#\}\Gamma^*\big)^n Q\Sigma^*$ is denoted by $\vdash_M$ and defined as follows:

$$\#y_1z_1\#y_2z_2 \cdots \#y_nz_npax \vdash_M \#y_1w_1\#y_2w_2 \cdots \#y_nw_nqx$$

if and only if

$$\#y_1z_1\#y_2z_2 \cdots \#y_nz_npax,\ \#y_1w_1\#y_2w_2 \cdots \#y_nw_nqx \in \big(\{\#\}\Gamma^*\big)^n Q\Sigma^*,$$

$iz_ipa \to w_iq \in R$ for some $i \in I_n$, and $w_j = z_j$ for $j \in I_n - \{i\}$. Let $\vdash_M^m$ and $\vdash_M^*$ denote the mth power of $\vdash_M$, for some $m \geq 1$, and the reflexive-transitive closure of $\vdash_M$, respectively. □

Definition 3. Let $M = (Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F)$ be an n-PPDA, for some $n \geq 1$. The *language accepted by* M, denoted by $L(M)$, is defined as

$$L\big(M\big) = \big\{w \in \Sigma^* \mid \#Z_1\#Z_2 \cdots \#Z_nsw \vdash_M^* \underbrace{\#\#\ldots\#}_{n \text{ times}} f, f \in F\big\}.$$ □

We illustrate the previous definitions by the following example.

Example 1. Consider the 2-PPDA

$$M = \big(\{s_0, s_1, q_0, q_1, q_2, q_3, f_0, f_1\}, \{a, b, c\}, R, s_0, c, c, \{f_1\}\big),$$

where R contains the following rules:

$1cs_0 \to ccs_1$	$1acq_1c \to cq_2$
$2cs_1 \to ccq_0$	$2bcq_2 \to cq_3$
$1cq_0a \to acq_0$	$1acq_3c \to cq_2$
$2cq_0b \to bcq_1$	$1ccq_3 \to f_0$
$2cq_1b \to bcq_1$	$2ccf_0 \to f_1$

To give an insight into the way M works, observe that M uses the first pushdown to store as. Then, it makes use of the second pushdown to store bs. Finally, it compares the number of input cs with the contents of both pushdowns. Clearly, $L(M) = \{a^n b^n c^n \mid n \geq 1\}$. For example, the string $aabbcc$ is accepted by M in the following way:

$$\begin{array}{llll} \#c\#cs_0aabbcc & \vdash_M \#cc\#cs_1aabbcc & \vdash_M \#cc\#ccq_0aabbcc & \vdash_M \\ \#cac\#ccq_0abbcc & \vdash_M \#caac\#ccq_0bbcc & \vdash_M \#caac\#cbcq_1bcc & \vdash_M \\ \#caac\#cbbcq_1cc & \vdash_M \#cac\#cbbcq_2c & \vdash_M \#cac\#cbcq_3c & \vdash_M \\ \#cc\#cbcq_2 & \vdash_M \#cc\#ccq_3 & \vdash_M \#\#ccf_0 & \vdash_M \\ \#\#f_1 & & & \end{array}$$

□

For $n \geq 1$, let ${}_n\mathbf{PPDA}$ denote the family of languages accepted by n-PPDAs. Set

$$\mathbf{PMPDA} = \bigcup_{i=1}^{\infty} {}_i\mathbf{PPDA}.$$

3.2 Accepting Power

In this section, we prove that 1-PPDAs characterize the family of context-free languages, and that 2-PPDAs characterize the family of recursively enumerable languages.

Lemma 1. *Let K be a context-free language. Then, there is a 1-PPDA, $M = (Q,$ alph(K), R, s, Z, $F)$, such that $L(M) = K$.*

Proof. For any context-free language K, there exists a PDA M such that $L\big(M\big) = K$. Next, we show how to simulate M by a 1-PPDA. Let

$$M = \big(Q, \Sigma, \Gamma, R, s, Z, F\big)$$

be the PDA with $\Sigma =$ alph(K). Without any loss of generality, suppose that a,$b \in \Sigma$, $a \neq b$. Let $\Sigma \cup \Gamma = \{c_0, c_1, c_2, \ldots, c_n\}$, where $n =$ card$((\Sigma \cup \Gamma) - \{Z\})$ and $c_0 = Z$. Define the homomorphism τ from $(\Sigma \cup \Gamma)^*$ to $\{a,b\}^*$ as $\tau(c_i) = ab^i$, for $i = 0, 1, 2, \ldots, n$. Construct the 1-PPDA

$$M' = \big(Q, \Sigma, R', s, \tau(Z), F\big),$$

where

$$R' = \big\{1\tau(z)pd \to \tau(w)q \mid zpd \to wq \in R\big\}.$$

Notice that $\tau(Z) = a$. To prove that $L(M) = L(M')$, we establish two claims. Claim 1 demonstrates how M' simulates M. Claim 2 demonstrates the converse simulation.

Claim 1. If $\#Zsw \vdash_M^k \#uqv$, where $q \in Q$, $v, w \in \Sigma^*$, $u \in \Gamma^*$, for some $k \geq 0$, then $\#\tau(Z)sw \vdash_{M'}^* \#\tau(u)qv$.

Proof. This claim is established by induction on $k \geq 0$.

Basis. Let $k = 0$. Then, for $\#Zsw \vdash_M^0 \#Zsw$ with $w \in \Sigma^*$, there is $\#\tau(Z)sw \vdash_{M'}^0 \#\tau(Z)sw$, so the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 0$ such that the claim holds for all sequences of moves of length m, where $0 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#Zsw \vdash_M^{k+1} \#u'q'v',$$

where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#Zsw \vdash_M^k \#uqv \vdash_M \#u'q'v',$$

where $u \in \Gamma^*$, $q \in Q$, $v \in \Sigma^*$. Then, there exists a rule $xqa \to x'q' \in R$, where $u = yx$, $u' = yx'$ and $v = av'$, which was used to perform the $(k + 1)$th move. By the construction of M', there exists a corresponding rule $1\tau(x)qa \to \tau(x')q' \in R'$. By the induction hypothesis,

$$\#\tau(Z)sw \vdash_{M'}^* \#\tau(u)qv.$$

Thus, by using $1\tau(x)qa \to \tau(x')q'$, M' can make the move

$$\#\tau(u)qv \vdash_{M'} \#\tau(u')q'v'.$$

Notice that $\tau(u) = \tau(y)\tau(x)$. The resulting configuration of M' corresponds to the new configuration of M and the claim holds. □

Claim 2. If $\#\tau(Z)sw \vdash_{M'}^k \#\tau(u)qv$, where $q \in Q$, $v, w \in \Sigma^*$, $u \in \Gamma^*$, for some $k \geq 0$, then $\#Zsw \vdash_M^* \#uqv$.

Proof. This claim is established by induction on $k \geq 0$.

Basis. Let $k = 0$. Then, for $\#\tau(Z)sw \vdash_{M'}^0 \#\tau(Z)sw$, where $w \in \Sigma^*$, there is $\#Zsw \vdash_M^0 \#Zsw$, so the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 0$ such that the claim holds for all sequences of moves of length m, where $0 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#\tau(Z)sw \vdash_{M'}^{k+1} \#\tau(u')q'v',$$

where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#\tau(Z)sw \vdash_{M'}^k \#\tau(u)qv \vdash_{M'} \#\tau(u')q'v',$$

where $u \in \Gamma^*$, $q \in Q$, $v \in \Sigma^*$. Then, there exists a rule $1\tau(x)qa \to \tau(x')q' \in R'$, where $\tau(u) = \tau(y)\tau(x)$, $\tau(u') = \tau(y)\tau(x')$, and $v = av'$, which was used to perform the $(k + 1)$th move.

By the construction of M', this rule was introduced from a rule $xqa \to x'q' \in R$. By the induction hypothesis,

$$\#Zsw \vdash_M^* \#uqv.$$

Thus, by using $xqa \to x'q'$, M can make the move

$$\#uqv \vdash_M \#u'q'v'.$$

The resulting configuration of M corresponds to the new configuration of M' and the claim holds. □

Consider a special case of Claim 1 when $u = v = \varepsilon$ and $q \in F$. Then, M accepts w. Since $\tau(u) = \varepsilon$, M' also accepts w. So, $L(M) \subseteq L(M')$.

Similarly, in the special case of Claim 2 when $\tau(u) = v = \varepsilon$ and $q \in F$, M' accepts w. Since $u = \varepsilon$, M also accepts w. So, $L(M') \subseteq L(M)$.

Hence, $L(M) = L(M')$, and the lemma holds. □

Theorem 1. $_1$**PPDA** = **CF**

Proof. The inclusion **CF** $\subseteq$ $_1$**PPDA** follows from Lemma 1. The opposite inclusion, $_1$**PPDA** $\subseteq$ **CF**, follows directly from the definitions of 1-PPDAs and PDAs. Hence, $_1$**PPDA** = **CF**, so the theorem holds. □

Next, we turn our attention to 2-PPDAs.

Lemma 2. *Let K be a recursively enumerable language. Then, there is a 2-PPDA, $M = (Q,$ alph(K), R, s, Z_1, Z_2, $F)$, such that $L(M) = K$.*

Proof. For any recursively enumerable language K, there exists a 2-PDA M such that $L(M) = K$. We introduce a 2-PPDA M' that simulates M. Let

$$M = \big(Q, \Sigma, \Gamma, R, s, Z_1, Z_2, F\big)$$

be the 2-PDA with $\Sigma =$ alph(K). Without any loss of generality, suppose that a,$b \in \Sigma$, $a \neq b$ and ith rule in R is labeled with r_i, for $i \in I_{\mathrm{card}(R)}$. Let $\Sigma \cup \Gamma \cup \{Z\} = \{c_0, c_1, c_2, \ldots, c_n\}$, where $n = \mathrm{card}(\Sigma \cup \Gamma)$, $Z \notin \Sigma \cup \Gamma$, and $c_0 = Z$. Define the homomorphism τ from $(\Sigma \cup \Gamma \cup \{Z\})^*$ to $\{a,b\}^*$ as $\tau(c_i) = ab^i$, for $i = 0, 1, \ldots, n$. Construct the 2-PPDA

$$M' = \big(Q', \Sigma, R', s', \tau(Z), \tau(Z), F\big)$$

as follows. Initially, set $Q' = Q \cup \{s', s''\}$ $(s', s'' \notin Q)$ and $R' = \emptyset$. Perform (1) and (2), given next:

(1) add $1\tau(Z)s' \to \tau(Z_1)s''$, $2\tau(Z)s'' \to \tau(Z_2)s$ to R';

(2) for each $r_i : a|bpd \to u|wq \in R$,

(a) add r_i to Q',

(b) add $1\tau(a)pd \to \tau(u)r_i$, $2\tau(b)r_i \to \tau(w)q$ to R'.

Before proving that $L(M) = L(M')$, let us give an insight into the construction. Every computation of M' begins by performing two moves by using a special starting rules from (1). Since the homomorphism τ encodes the start symbols Z_1 and Z_2 with the strings containing more than one symbol, M' uses this starting rules to push $\tau(Z_1)$ and $\tau(Z_2)$ onto its pushdowns. Therefore, the resulting configuration corresponds to the starting configuration of M. Then, for every rule from R, in (2), we add two new rules to R', which act in a similar way but use the homomorphism τ for encoding the pushdown symbols. M use both pushdowns simultaneously in every move, while M' can use only one. Therefore, every rule $r_i \in R$ is simulated with two rules.

To prove the identity $L(M) = L(M')$, we establish two claims. Claim 3 demonstrates how M' simulates M. Claim 4 shows the converse simulation.

Claim 3. If $\#Z_1\#Z_2sw \vdash_M^k \#u_1\#u_2qv$, where $q \in Q$, $w, v \in \Sigma^*$, $u_1, u_2 \in \Gamma^*$, for some $k \geq 0$, then $\#\tau(Z)\#\tau(Z)s'w \vdash_{M'}^* \#\tau(u_1)\#\tau(u_2)qv$.

Proof. This claim is established by the induction on $k \geq 0$.

Basis. Let $k = 0$. Then, for $\#Z_1\#Z_2sw \vdash_M^0 \#Z_1\#Z_2sw$, where $w \in \Sigma^*$, there is

$$\#\tau(Z)\#\tau(Z)s'w \vdash_{M'} \#\tau(Z_1)\#\tau(Z)s'' \vdash_{M'} \#\tau(Z_1)\#\tau(Z_2)sw$$

so the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 0$ such that the claim holds for all sequences of moves of length m, where $0 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#Z_1\#Z_2sw \vdash_M^{k+1} \#u_1'\#u_2'q'v',$$

where $w, v' \in \Sigma^*$, $u_1', u_2' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#Z_1\#Z_2sw \vdash_M^k \#u_1\#u_2qv \vdash_M \#u_1'\#u_2'q'v',$$

where $v \in \Sigma^*$, $u_1, u_2 \in \Gamma^*$, $q \in Q$. Then, there exists a rule

$$r_i : x_1|x_2qa \to x_1'|x_2'q' \in R,$$

where $u_1 = y_1x_1$, $u_1' = y_1x_1'$, $u_2 = y_2x_2$, $u_2' = y_2x_2'$, $v = av'$, which was used to perform the $(k+1)$th move. By the construction of M', there exist two corresponding rules

$$1\tau(x_1)qa \to \tau(x_1')r_i, 2\tau(x_2)r_i \to \tau(x_2')q' \in R'.$$

By the induction hypothesis,

$$\#\tau(Z)\#\tau(Z)s'w \vdash_{M'}^* \#\tau(u_1)\#\tau(u_2)qv.$$

Consequently, by using previous two rules M' can make the moves

$$\#\tau(u_1)\#\tau(u_2)qv \vdash_{M'} \#\tau(u_1')\#\tau(u_2)r_iv' \vdash_{M'} \#\tau(u_1')\#\tau(u_2')q'v',$$

and the resulting configuration of M' corresponds to the new configuration of M. Therefore, the claim holds. □

In Claim 4, without any loss of generality, we do not consider any sequence of moves of M' of odd length. From the construction of M', every odd move leads to an intermediate state, while simulating one move of M, which is never final. Additionally, there is always exactly one applicable rule, so the computation never ends after odd number of moves, neither fails nor accepts.

Claim 4. If $\#\tau(Z)\#\tau(Z)s'w \vdash_{M'}^{2k} \#\tau(u_1)\#\tau(u_2)qv$, where $q \in Q$, $w, v \in \Sigma^*$, $u_1, u_2 \in \Gamma^*$, for some $k \geq 1$, then $\#Z_1\#Z_2sw \vdash_M^* \#u_1\#u_2qv$.

Proof. This claim is established by the induction on $k \geq 1$.

Basis. Let $k = 1$. By the construction of M', at the beginning of every computation, there is the single applicable rule

$$1\tau(Z)s' \to \tau(Z_1)s'' \in R',$$

and by its application M' performs the move

$$\#\tau(Z)\#\tau(Z)s'w \vdash_{M'} \#\tau(Z_1)\#\tau(Z)s''w$$

where $w \in \Sigma^*$. In state s'', there is also the single applicable rule

$$2\tau(Z)s'' \to \tau(Z_2)s \in R'.$$

Thus, M' performs the move

$$\#\tau(Z_1)\#\tau(Z)s''w \vdash_{M'} \#\tau(Z_1)\#\tau(Z_2)sw.$$

Then, there is

$$\#Z_1\#Z_2sw \vdash_M^0 \#Z_1\#Z_2sw.$$

Therefore, the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 1$ such that the claim holds for all sequences of moves of length m, where $1 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#\tau(Z)\#\tau(Z)s'w \vdash_{M'}^{2(k+1)} \#\tau(u_1')\#\tau(u_2')q'v',$$

where $w, v' \in \Sigma^*$, $u_1', u_2' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#\tau(Z)\#\tau(Z)s'w \vdash_{M'}^{2k} \#\tau(u_1)\#\tau(u_2)qv \vdash_{M'},$$

$$\#\tau(u_1')\#\tau(u_2)r_iv' \vdash_{M'} \#\tau(u_1')\#\tau(u_2')q'v',$$

where $v \in \Sigma^*$, $u_1, u_2 \in \Gamma^*$, $q \in Q$, $i \in I_{\mathrm{card}(R)}$. Then, there exist a rule

$$1\tau(x_1)qa \to \tau(x_1')r_i, 2\tau(x_2)r_i \to \tau(x_2')q' \in R',$$

with the identities $\tau(u_1) = \tau(y_1)\tau(x_1)$, $\tau(u_1') = \tau(y_1)\tau(x_1')$, $\tau(u_2) = \tau(y_2)\tau(x_2)$, $\tau(u_2') = \tau(y_2)\tau(x_2')$, and $v = av'$, which was used to perform the $(k+1)$th and $(k+2)$th move. By the construction of M', these rules were introduced by a rule

$$x_1|x_2qa \to x_1'|x_2'q' \in R.$$

By the induction hypothesis,

$$\#Z_1\#Z_2sw \vdash_M^k \#u_1\#u_2qv.$$

Thus, by using $x_1|x_2qa \to x_1'|x_2'q'$, M can make the move

$$\#u_1\#u_2qv \vdash_M \#u_1'\#u_2'q'v'.$$

The resulting configuration of M corresponds to the new configuration of M' and the claim holds. □

Consider a special case of Claim 3, where $u_1 = u_2 = v = \varepsilon$ and $q \in F$. Then, M accepts w. Since $\tau(u_1) = \tau(u_2) = \varepsilon$, M' also accepts w. So, $L(M) \subseteq L(M')$.

Similarly, in a special case of Claim 4, where $\tau(u_1), \tau(u_2), v = \varepsilon$ and $q \in F$, M' accepts w. Since $u_1, u_2 = \varepsilon$, M accepts w as well. So, $L(M') \subseteq L(M)$.

Hence, $L(M) = L(M')$, and the lemma holds. □

Theorem 2. $_2\mathbf{PPDA} = \mathbf{RE}$

Proof. The inclusion $\mathbf{RE} \subseteq {}_2\mathbf{PPDA}$ follows from Lemma 2. The opposite inclusion, ${}_2\mathbf{PPDA} \subseteq \mathbf{RE}$, can be obtained by standard simulations. Hence, ${}_2\mathbf{PPDA} = \mathbf{RE}$, so the theorem holds. □

From Theorem 2, we obtain the following corollary.

Corollary 1. $\mathbf{PMPDA} = \mathbf{RE}$ □

4 Pure Multi-Pushdown Automata with Ordered Pushdowns

In the previous section, we studied pure multi-pushdown automata. In the present section, we restrict the way they use their pushdowns. We introduce a total order $\preceq$ over their pushdowns. After using the ith pushdown, the automaton can work only with the jth pushdown, where $i \preceq j$. We prove that these restricted versions of pure multi-pushdown automata characterize the family of context-free languages independently of the number of their pushdowns.

4.1 Definitions and Examples

Next, we define pure multi-pushdown automata with ordered pushdowns and illustrate them by an example.

Definition 4. For $n \geq 1$, a *pure n-pushdown-ordered multi-pushdown automaton* (n-*PPOPDA*) is a pair

$$\mathcal{H} = (M, \preceq),$$

where M is an n-PPDA, and $\preceq$ is a total order over I_n. □

Definition 5. Let $\mathcal{H} = (M, \preceq)$ be an n-PPOPDA, for some $n \geq 1$, where $M = (Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F)$. The *direct move relation* over $I_n(\{\#\}\Gamma^*)^n Q\Sigma^*$ is denoted by $\vdash_{\mathcal{H}}$ and defined as follows:

$$k\#y_1z_1\#y_2z_2\cdots\#y_nz_npax \vdash_{\mathcal{H}} i\#y_1w_1\#y_2w_2\cdots\#y_nw_nqx,$$

if and only if

$$k\#y_1z_1\#y_2z_2\cdots\#y_nz_npax,\ i\#y_1w_1\#y_2w_2\cdots\#y_nw_nqx \in I_n\big(\{\#\}\Gamma^*\big)^n Q\Sigma^*,$$

$iz_ipa \to w_iq \in R$, $k \preceq i$, and $w_j = z_j$ for $j \in I_n - \{i\}$. Let $\vdash_{\mathcal{H}}^m$ and $\vdash_{\mathcal{H}}^*$ denote the mth power of $\vdash_{\mathcal{H}}$, for some $m \geq 1$, and the reflexive-transitive closure of $\vdash_{\mathcal{H}}$, respectively. □

Definition 6. Let $\mathcal{H} = (M, \preceq)$ be an n-PPOPDA, for some $n \geq 1$, where $M = (Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F)$. The *language accepted by* $\mathcal{H}$, denoted by $L(\mathcal{H})$, is defined as

$$L\big(\mathcal{H}\big) = \big\{w \in \Sigma^* \mid 1\#Z_1\#Z_2\cdots\#Z_nsw \vdash_{\mathcal{H}}^* n\underbrace{\#\#\ldots\#}_{n \text{ times}}f, f \in F\big\}.$$ □

We illustrate the previous definitions by the next example.

Example 2. Let $\mathcal{H} = (M, \preceq)$, where $M = (\{q\}, \{a, b, c, d\}, R, q, c, a, \{q\})$, be a 2-PPDA with $1 \preceq 2$ and R containing the following rules:

$1cqa \to caq$	$2aqc \to acq$
$1cqb \to cbq$	$2aqd \to adq$
$1aqa \to aaq$	$2cqc \to ccq$
$1bqb \to bbq$	$2dqd \to ddq$
$1aqb \to q$	$2cqd \to q$
$1bqa \to q$	$2dqc \to q$
$1cq \to q$	$2aq \to q$

First, $\mathcal{H}$ checks the equality of the number of the occurrences of as and bs on the first pushdown and then cs and ds on the second pushdown. The language of $\mathcal{H}$ is

$$L(\mathcal{H}) = \big\{w \in \{a,b,c,d\}^* \mid w = w_1w_2, w_1 \in \{a,b\}^*, \#_a(w_1) = \#_b(w_1), w_2 \in \{c,d\}^*, \#_c(w_2) = \#_d(w_2)\big\},$$

where $\#_t(v)$ denotes the number of occurrences of t in v. Notice that there is no need to change a state to ensure a separation of w_1 and w_2. Indeed, the pushdown order given by $\preceq$ prevents a mixture of as and bs with cs and ds. For example, the string $abbacdcd$ is accepted by M in the following way:

$$\begin{array}{llll} 1\#c\#aqabbacdcd & \vdash_{\mathcal{H}} 1\#ca\#aqbbacdcd & \vdash_{\mathcal{H}} 1\#c\#aqbacdcd & \vdash_{\mathcal{H}} \\ 1\#cb\#aqacdcd & \vdash_{\mathcal{H}} 1\#c\#aqcdcd & \vdash_{\mathcal{H}} 1\#\#aqcdcd & \vdash_{\mathcal{H}} \\ 2\#\#acqdcd & \vdash_{\mathcal{H}} 2\#\#aqcd & \vdash_{\mathcal{H}} 2\#\#acqd & \vdash_{\mathcal{H}} \\ 2\#\#aq & \vdash_{\mathcal{H}} 2\#\#q & & \end{array}$$

□

For $n \geq 1$, let ${}_n\mathbf{PPOPDA}$ denote the family of languages accepted by n-PPOPDAs. Set

$$\mathbf{PPOMPDA} = \bigcup_{i=1}^{\infty} {}_i\mathbf{PPOPDA}.$$

4.2 Accepting Power

In this section, we prove that n-PPOPDAs characterize the family of context-free languages.

Lemma 3. ${}_n\mathbf{PPOPDA} \subseteq \mathbf{CF}$, *for any* $n \geq 1$.

Proof. We show that we can simulate any n-PPOPDA by a PDA, for any $n \geq 1$. Let $\mathcal{H} = (M, \preceq)$ be an n-PPOPDA, for some $n \geq 1$, where

$$M = \big(Q, \Sigma, R, s, Z_1, Z_2, \ldots, Z_n, F\big).$$

Without any loss of generality, we assume that $1 \preceq 2 \preceq \cdots \preceq n$. We introduce the PDA

$$M' = \big(Q', \Sigma, \Gamma, R', s', Z, \{f\}\big),$$

defined in the following way. Initially, set

$$\begin{array}{lcl} Q' & = & \{q_i \,|\, q \in Q, i \in I_n\} \cup \{s', f\} \ \big(\{s', f\} \cap Q = \emptyset\big) \\ \Gamma & = & \Sigma \cup \{Z\} \ (Z \notin \Sigma) \\ R' & = & \emptyset \end{array}$$

To finish the construction, perform (1) through (4), given next:

(1) add $Zs' \to ZZ_1s_1$ to R';

(2) for each $q \in Q$ and each $i \in I_{n-1}$, add $Zq_i \to ZZ_{i+1}q_{i+1}$ to R';

(3) for each $q \in F$, add $Zq_n \to f$ to R';

(4) for each $izpa \to wq \in R$, add $zp_ia \to wq_i$ to R'.

Before proving that $L(\mathcal{H}) = L(M')$, let us give an insight into the construction. $\mathcal{H}$ uses its pushdowns in the order given by $\preceq$. Every configuration of $\mathcal{H}$ keeps the information about the current working pushdown implicitly. M' keeps this information in its states. If the current state is q_i, M' simulates the work with the ith pushdown in the given order. There are two states with a special purpose: the initial s' and final f.

We use the new symbol Z as the bottom of the pushdown of M'. It is never removed except in the last terminating rule. It is also used to determine possible moves further in the simulation to the next pushdown. When Z is on the top of the pushdown, the currently simulated pushdown is empty. $\mathcal{H}$ can make a move to the next pushdown even if the current one is not empty yet; however, no such computation could be accepting. Indeed, the future emptying of the pushdown is

not possible. M' blocks instead of simulating the unsuccessful computation. The only permitted move in the pushdown order is when the currently simulated pushdown is empty. Then, the simulation is possibly accepting.

For every rule from R, we introduce the corresponding rules to R'. Additionally, we define rules simulating the pushdown order. Every accepting computation of $\mathcal{H}$ is divided into n phases, and in the ith phase, for $i \in I_n$, $\mathcal{H}$ uses only the ith pushdown, with respect to $\preceq$. Therefore, the simulation is also divided into n phases. The ith phase, for $i \in I_n$, is simulated as follows:

(I) Phase 1 starts with the rule from (1), phase i for $i > 1$ with the rule from (2). The start symbol Z_i of the ith pushdown of $\mathcal{H}$ is inserted to the pushdown of M'. Notice that except the first phase, the previously simulated $(i-1)$th pushdown must be empty before. We determine this by checking that Z is on the top of the pushdown.

(II) The simulation continues with rules from (4), where for a rule of the form $zp_ja \to wq_j$, it holds that $i = j$.

(III) If the symbol Z is on the top of the pushdown, the simulated pushdown is empty and the simulation continues back to (I) except the phase n, where the simulation can successfully finish by using a rule from (3).

To prove the identity $L(\mathcal{H}) = L(M')$, we establish two claims. Claim 5 demonstrates how to simulate $\mathcal{H}$ by M'. Claim 6 demonstrates the converse simulation.

In Claim 5, without any loss of generality, we do not consider any computation of $\mathcal{H}$, which manipulates the ith pushdown before emptying the jth one, where $i > j$. Since future emptying is not possible, such computation is obviously not accepting. This restriction is without any loss of generality. In Claim 5, we only show that M' covers all accepting computations of $\mathcal{H}$.

Claim 5. If $1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash_{\mathcal{H}}^{k} i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv$, where $w, v \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$, $i \in I_n$, and all jth pushdowns, with $j > i$, contain only the symbol Z_j, for some $k \geq 0$, then $\#Zs'w \vdash_{M'}^{*} \#Zuq_iv$.

Proof. This claim is established by induction on $k \geq 0$.

Basis. Let $k = 0$. Then, for

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash_{\mathcal{H}}^{0} 1\#Z_1\#Z_2\#\ldots\#Z_nsw,$$

where $w \in \Sigma^*$, there is

$$\#Zs'w \vdash_{M'} \#ZZ_1s_1w.$$

Therefore, the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 0$ such that the claim holds for all sequences of moves of length m, where $0 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash_{\mathcal{H}}^{k+1} j\#^ju'\#Z_{j+1}\#\ldots\#Z_nq'v',$$

where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q' \in Q$. Since $k+1 \geq 1$, this sequence of moves can be written in the form

$$\begin{array}{lll} 1\#Z_1\#Z_2\#\ldots\#Z_nsw & \vdash_{\mathcal{H}}^{k} & i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv \\ & \vdash_{\mathcal{H}} & j\#^ju'\#Z_{j+1}\#\ldots\#Z_nq'v' \end{array}$$

where $v \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$.

First, consider $i \neq j$. Observe that i and j denote the numbers of the current working pushdown. The order $\preceq$ ensures that $i \leq j$, so, $i < j \leq n$. If $i+1 < j$, the $(i+1)$th pushdown remains nonempty and the computation is not accepting. We do not consider such a computation. Summarily, $i+1 = j$. The $(k+1)$th move was then performed by using a rule $jZ_jqa \to u'q' \in R$, where $v = av'$. If the computation may be accepting, $u = \varepsilon$. Then, by the induction hypothesis,

$$\#Zs'w \vdash_{M'}^{*} \#Zq_iv.$$

By the construction of M', there exist two rules

$$Zq_i \to ZZ_{i+1}q_{i+1}, Z_{i+1}q_{i+1}a \to u'q'_{i+1} \in R'.$$

Thus, by using these rules, M' can make the moves

$$\#Zq_iv \vdash_{M'} \#ZZ_{i+1}q_{i+1}v \vdash_{M'} \#Zu'q'_{i+1}v',$$

and the resulting configuration corresponds to the new $\mathcal{H}$'s configuration.

Second, consider $i = j$. Then, we can rewrite the $\mathcal{H}$'s sequence of moves to the form

$$\begin{array}{lll} 1\#Z_1\#Z_2\#\ldots\#Z_nsw & \vdash_{\mathcal{H}}^{k} & i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv \\ & \vdash_{\mathcal{H}} & i\#^iu'\#Z_{i+1}\#\ldots\#Z_nq'v' \end{array}$$

The $(k+1)$th move was performed by a rule $ixqa \to x'q' \in R$, where $u = yx$, $u' = yx'$. By the construction of M', there exists a rule from (4), $xq_ia \to x'q_i \in R'$. By the induction hypothesis,

$$\#Zs'w \vdash_{M'}^{*} \#Zuq_iv.$$

By using $xq_ia \to x'q_i$, M' can perform the move

$$\#Zuq_iv \vdash_{M'} \#Zu'q'_iv'.$$

The resulting configuration corresponds to the new $\mathcal{H}$'s configuration.

By combining the previous two cases, Claim 5 holds. □

Claim 6. If $\#Zs'w \vdash_{M'}^{k} \#Zuq_iv$, where $w, v \in \Sigma^*$, $u \in \Gamma^*$, $q_i \in Q'$, $i \in I_n$, for some $k \geq 1$. Then, $1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash_{\mathcal{H}}^{*} j\#^iu\#Z_{i+1}\#\ldots\#Z_nqv$, $j \leq i$.

Proof. This claim is established by induction on $k \geq 1$.

Basis. Let $k = 1$. By the construction of M', at the beginning of every computation, there is the single applicable rule

$$Zs' \to ZZ_1s_1 \in R',$$

and by its application M' performs the move

$$\#Zs'w \vdash_{M'} \#ZZ_1sw,$$

where $w \in \Sigma^*$. Then, there is

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash^0_{\mathcal{H}} 1\#Z_1\#Z_2\#\ldots\#Z_nsw,$$

and the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 1$ such that the claim holds for all sequences of moves of length m, where $1 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#Zs'w \vdash^{k+1}_{M'} \#Zu'q'_jv',$$

where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q'_j \in Q'$. Since $k+1 \geq 1$, this sequence can be written in the form

$$\#Zs'w \vdash^k_{M'} \#Zuq_iv \vdash_{M'} \#Zu'q'_jv',$$

where $v \in \Sigma^*$, $u \in \Gamma^*$, $q_i \in Q'$. Next, we cover these two cases: $i = j$ and $i + 1 = j$. By the construction of M', there is no other possibility.

First, suppose that $i + 1 = j$. Then, the $(k+1)$th move of M' was performed by a rule from (2), which is of the form

$$Zq_i \to ZZ_{i+1}q_{i+1},$$

so, $u = \varepsilon$, $u' = Z_{i+1}$, $q' = q$, $v' = v$. Thus, the sequence of moves of M' can be written as

$$\#Zs'w \vdash^k_{M'} \#Zq_iv \vdash_{M'} \#ZZ_{i+1}q_{i+1}v.$$

By the induction hypothesis we have

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash^*_{\mathcal{H}} i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv.$$

Since $u = \varepsilon$, we can rewrite this sequence of moves to the form

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash^*_{\mathcal{H}} i\#^{i+1}Z_{i+1}\#\ldots\#Z_nqv$$

which corresponds to the new configuration of M'.

Second, suppose that $i = j$. Then, the sequence of moves of M' is

$$\#Zs'w \vdash^k_{M'} \#Zuq_iv \vdash_{M'} \#Zu'q'_iv'.$$

The $(k+1)$th move of M' was performed by a rule from (4), $xq_ia \to x'q'_i \in R'$, where $u = yx$, $u' = yx'$, $v = av'$. By the construction of M', this rule was added to R' from a rule $ixqa \to x'q' \in R$. By the induction hypothesis,

$$1\#Z_1\#Z_2\#\ldots\#Z_nsw \vdash^*_{\mathcal{H}} i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv.$$

Therefore, by using $ixqa \to x'q'$, $\mathcal{H}$ can perform an additional move

$$i\#^iu\#Z_{i+1}\#\ldots\#Z_nqv \vdash_{\mathcal{H}} i\#^iu'\#Z_{i+1}\#\ldots\#Z_nq'v',$$

and enter the configuration corresponding to the new configuration of M'.

Hence, we covered both possible cases and the claim holds. □

Consider a special case of Claim 5 when

$$1\#Z_1\#Z_2\#\ldots\#Z_n sw \vdash^k_{\mathcal{H}} n\#^n uqv,$$

where $u = v = \varepsilon$, $q \in F$, for some $k \geq 0$. Then, $\mathcal{H}$ accepts w. By this claim

$$\#Zs'w \vdash^*_{M'} \#Zq_n.$$

Since $q \in F$ and by the construction of M', there exists a rule from (3), $Zq_n \to f \in R'$, and by using this rule, M' can also accept w. So, $L(\mathcal{H}) \subseteq L(M')$.

As a special case of Claim 6, M' can make the sequence of moves

$$\#Zs'w \vdash^k_{M'} \#Zq_n,$$

for some $k \geq 1$. Then, if $q \in F$, by the construction of M', there exists a rule from (3), $Zq_n \to f \in R'$, and M' can accept w by its application. However, by this claim

$$\#Z_1\#Z_2\#\ldots\#Z_n sw \vdash^*_{\mathcal{H}} \#^n q.$$

So, $\mathcal{H}$ accepts w as well. Hence, $L(M') \subseteq L(\mathcal{H})$.

Since $L(\mathcal{H}) \subseteq L(M')$ and $L(M') \subseteq L(\mathcal{H})$, we have that $L(\mathcal{H}) = L(M')$, and the lemma holds. □

Lemma 4. *Let K be a context-free language. Then, there is a 1-PPOPDA, $\mathcal{H} = (M, \preceq)$, where $M = (Q,$ alph(K), R, s, Z $F)$, such that $L(M) = K$.*

Proof. This Lemma follows directly from Lemma 1. Indeed, observe that every 1-PPDA is a 1-PPOPDA. □

Theorem 3. *${}_n$***PPOPDA** = **CF**, *for any $n \geq 1$.*

Proof. Let $n \geq 1$. By Lemma 3, ${}_n$**PPOPDA** $\subseteq$ **CF**. By Lemma 4, **CF** $\subseteq {}_n$**PPOPDA**. Hence, ${}_n$**PPOPDA** = **CF**, so the theorem holds. □

From Theorem 3, we obtain the following corollary.

Corollary 2. **PPOMPDA** = **CF** □

5 Ordered Pure Pushdown Automata

In the previous section, we studied pure multi-pushdown automata with a total order over their pushdowns. In the present section, we turn our attention to an order over the input symbols. We introduce an order $\trianglelefteq$ and restrict the way the automaton pushes symbols onto its pushdown. If a is the topmost pushdown symbol, only b such that $a \trianglelefteq b$ can be pushed onto the pushdown. We prove that such restricted pure pushdown automata characterize only a proper subfamily of the family of context-free languages.

5.1 Definitions and Examples

Next, we define ordered pure pushdown automata and illustrate them by an example.

Definition 7. An *ordered pure pushdown automaton* (*OPPDA*) is a pair

$$\mathcal{H} = \big(M, \trianglelefteq\big),$$

where $M = (Q, \Sigma, R, s, Z, F)$ is a 1-PPDA, and $\trianglelefteq$ is a total order over Σ. Since there is only a single pushdown, instead of $izpa \to wq \in R$, we write just $zpa \to wq \in R$. □

Definition 8. Let $\mathcal{H} = (M, \trianglelefteq)$ be an OPPDA, where $M = (Q, \Sigma, R, s, Z, F)$. The *direct move relation* over $\{\#\}\Gamma^*Q\Sigma^*$ is denoted by $\vdash_{\mathcal{H}}$ and defined as

$$\#yzpax \vdash_{\mathcal{H}} \#ywqx,$$

if and only if $\#yzpax, \#ywqx \in \{\#\}\Gamma^*Q\Sigma^*$, $zpa \to wq \in R$, and

$$c_1 \trianglelefteq c_2 \trianglelefteq \cdots \trianglelefteq c_{|yw|},$$

where $yw = c_1c_2 \cdots c_{|yw|}$. Let $\vdash^m_{\mathcal{H}}$ and $\vdash^*_{\mathcal{H}}$ denote the mth power of $\vdash_{\mathcal{H}}$, for some $m \geq 1$, and the reflexive-transitive closure of $\vdash_{\mathcal{H}}$, respectively. □

Definition 9. Let $\mathcal{H} = (M, \trianglelefteq)$ be an OPPDA, where $M = (Q, \Sigma, R, s, Z, F)$. The *language accepted by* $\mathcal{H}$, denoted by $L(\mathcal{H})$, is defined as

$$L\big(\mathcal{H}\big) = \big\{w \in \Sigma^* \mid \#Zsw \vdash^*_{\mathcal{H}} \#f, f \in F\big\}.$$ □

We illustrate the previous definitions by an example.

Example 3. Let $\mathcal{H} = (M, \trianglelefteq)$, where $M = (\{s, q, f\}, \{a, b, c\}, R, s, a, \{f\})$, be an OPPDA with $a \trianglelefteq b \trianglelefteq c$ and R containing the following rules:

$$\begin{array}{ll} sa \to as & aqa \to q \\ sb \to bs & bqb \to q \\ sc \to cs & cqc \to q \\ s \to q & aq \to f \end{array}$$

First, $\mathcal{H}$ pushes some symbols from the input onto the pushdown. Then, it nondeterministically changes its state and checks the equality of the reversal of the pushdown string with the rest of the input string. The language of $\mathcal{H}$ is

$$L(\mathcal{H}) = \big\{w_1w_2 \mid w_1 \in \{a\}^*\{b\}^*\{c\}^*, w_2 = \text{reversal}(w_1)\big\}.$$

The given order $\trianglelefteq$ ensures a separation of sequences of as, bs, and cs. For example, the string $aabccbaa$ is accepted by $\mathcal{H}$ in the following way:

$$\begin{array}{lll} \#asaabccbaa \vdash_{\mathcal{H}} \#aasabccbaa \vdash_{\mathcal{H}} \#aaasbccbaa \vdash_{\mathcal{H}} \\ \#aaabsccbaa \vdash_{\mathcal{H}} \#aaabcscbaa \vdash_{\mathcal{H}} \#aaabcqcbaa \vdash_{\mathcal{H}} \\ \#aaabqbaa \vdash_{\mathcal{H}} \#aaaqaa \vdash_{\mathcal{H}} \#aaqa \vdash_{\mathcal{H}} \\ \#aq \vdash_{\mathcal{H}} \#f \end{array}$$

□

Let **OPPDA** denote the family of languages accepted by OPPDAs.

5.2 Accepting Power

In this section, we prove that OPPDAs characterize only a proper subfamily of the family of context-free languages.

Lemma 5. OPPDA $\subseteq$ CF

Proof. Consider an OPPDA $\mathcal{H} = (M, \trianglelefteq)$, where

$$M = \big(Q, \Sigma, R, s, Z, F\big).$$

We show how to simulate $\mathcal{H}$ by a PDA. First, we give a construction of such a PDA. Then, we describe the idea underlying this construction. However, even before, without any loss of generality, we remove from R all rules of the form $vpa \to wq$, where w does not satisfy $\trianglelefteq$. Observe that such rules are never applicable. Let

$$M' = \big(Q', \Sigma, \Gamma, R', s', Z', F\big),$$

be the PDA constructed as follows. Initially, set $Q' = Q \cup \{s'\}$ $(s' \notin Q)$, $\Gamma = \Sigma \cup \{Z'\}$ $(Z' \notin \Sigma)$, and $R' = \emptyset$. Perform (1) through (4), given next:

(1) add $Z's' \to Z'Zs$ to R';

(2) for each $f \in F$, add $Z'f \to f$ to R';

(3) for each $vpa \to wq \in R$, add $Z'vpa \to Z'wq$ to R';

(4) for each $vpa \to wq \in R$ and for each $c \in \Sigma$, if $cw = cd_1d_2 \dots d_{|w|}$, where $d_i \in \Sigma$, and $c \trianglelefteq d_1 \trianglelefteq d_2 \trianglelefteq \cdots \trianglelefteq d_{|w|}$, add $cvpa \to cwq$ to R'.

Before we prove the identity $L(\mathcal{H}) = L(M')$, let us give an insight into the construction. M' has to simulate the order $\trianglelefteq$. We introduce a new pushdown symbol Z' to denote the bottom of the simulated pushdown. It is the start pushdown symbol and it is never removed until the computation succeeds.

The set of states Q is extended with a new start state s' to ensure the consistence of the starting configuration of $\mathcal{H}$ and M'. For this purpose, we introduce the rule $Z's' \to Z'Zs$ in (1).

Then, the simulation continues rule by rule in accordance with $\mathcal{H}$, but unlike $\mathcal{H}$, M' does not ensure the order of the pushdown symbols naturally. However, the rules are designed to satisfy

the simulated order $\trianglelefteq$. Roughly speaking, M' has to look one symbol deeper into the pushdown whether the following move does not violate $\trianglelefteq$. More precisely, every rule $vpa \to wq \in R$ is replaced with rules from (3) and (4). The resulting rules extend all the rules in R. The pushdown strings on both sides of each new rule contain an additional pushdown symbol. It is either Z' or $c \in \Sigma$ that preserves $\trianglelefteq$. Therefore, the rule to be applied is chosen according to the top pushdown string with one additional symbol, which remains in the pushdown. Then, the resulting pushdown configuration corresponds to the simulated one.

Finally, whenever Z' occurs on the top of the pushdown and the current state is $f \in F$, by the construction of M', there exists the rule $Z'f \to f \in R'$ from (2) and M' can accept.

To prove that $L(\mathcal{H}) = L(M')$, we establish two claims. Claim 7 shows how M' simulates $\mathcal{H}$. Claim 8 shows the converse simulation.

Claim 7. If $\#Zsw \vdash_{\mathcal{H}}^{k} \#uqv$, where $v, w \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$, for some $k \geq 0$, then $\#Z's'w \vdash_{M'}^{*} \#Z'uqv$.

Proof. This claim is established by induction on $k \geq 0$.

Basis. Let $k = 0$. Then, for $\#Zsw \vdash_{\mathcal{H}}^{0} \#Zsw$, where $w \in \Sigma^*$, there is

$$\#Z's'w \vdash_{M'} \#Z'Zsw,$$

by the rule from (1), so the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 0$ such that the claim holds for all sequences of moves of length m, where $0 \leq m \leq k$.

Induction Step. Consider any sequence of moves $\#Zsw \vdash_{\mathcal{H}}^{k+1} \#u'q'v'$, where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#Zsw \vdash_{\mathcal{H}}^{k} \#uqv \vdash_{\mathcal{H}} \#u'q'v'$$

where $v \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$. Then, there exits a rule $xqa \to x'q' \in R$, where $u = yx$, $u' = yx'$, $v = av'$, which was used to perform the $(k+1)$th move. Since both yx and yx' satisfy $\trianglelefteq$ and $y = y'c$, where $c \in \Sigma \cup \{\varepsilon\}$ and $y' = \Gamma^*$, by the construction of M', there exists a rule $c'xqa \to c'x'q' \in R'$, where, if $c = \varepsilon$, $c' = Z'$, otherwise $c' = c$. By the induction hypothesis,

$$\#Z's'w \vdash_{M'}^{*} \#Z'uqv.$$

Consequently, by using $c'xqa \to c'x'q'$, M' can make the move

$$\#Z'uqv \vdash_{M'} \#Z'u'q'v'.$$

The resulting configuration of M' corresponds to the new configuration of $\mathcal{H}$, and the claim holds. □

Claim 8. If $\#Z's'w \vdash_{M'}^{k} \#Z'uqv$, where $v, w \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$, $u = c_1c_2 \dots c_{|u|}$ and $c_1 \trianglelefteq c_2 \trianglelefteq \cdots \trianglelefteq c_{|u|}$, for some $k \geq 1$, then $\#Zsw \vdash_{\mathcal{H}}^{*} \#uqv$.

Proof. This claim is established by induction on $k \geq 1$.

Basis. Let $k = 1$. From the construction of M', there exists the only applicable rule from (1), $Z's' \to Z'Zs$. Since Z satisfies $\trianglelefteq$, then for

$$\#Z's'w \vdash_{M'} \#Z'Zsw,$$

where $w \in \Sigma^*$, there is $\#Zsw \vdash_{\mathcal{H}}^0 \#Zsw$ and the basis holds.

Induction Hypothesis. Suppose that there exists $k \geq 1$ such that the claim holds for all sequences of moves of length m, where $1 \leq m \leq k$.

Induction Step. Consider any sequence of moves

$$\#Z's'w \vdash_{M'}^{k+1} \#Z'u'q'v',$$

where $w, v' \in \Sigma^*$, $u' \in \Gamma^*$, $q' \in Q$. Since $k + 1 \geq 1$, this sequence can be written in the form

$$\#Z's'w \vdash_{M'}^{k} \#Z'uqv \vdash_{M'} \#Z'u'q'v',$$

where $v \in \Sigma^*$, $u \in \Gamma^*$, $q \in Q$. Then, there exits a rule $cxqa \to cx'q' \in R'$, where $u = ycx$, $u' = ycx'$, $c \in \Gamma \cup \{Z'\}$, $v = av'$, which was used to perform the $(k+1)$th move. Since yc and cx' satisfy $\trianglelefteq$, the order $\trianglelefteq$ is preserved. Then, by the construction of M', used rule was introduced to R' according to some rule $xqa \to x'q' \in R$. By the induction hypothesis,

$$\#Zsw \vdash_{\mathcal{H}}^{k} \#uqv.$$

Consequently, by using $xqa \to x'q'$, $\mathcal{H}$ can make the move

$$\#uqv \vdash_{\mathcal{H}} \#u'q'v'.$$

The resulting configuration of $\mathcal{H}$ corresponds to the new configuration of M' and the claim holds. □

Consider a special case of Claim 7 $\#Zsw \vdash_{\mathcal{H}}^{k} \#uqv$ when $u, v = \varepsilon$, $q \in F$. $\mathcal{H}$ accepts w. Then,

$$\#Z's'w \vdash_{M'}^{*} \#Z'q,$$

and, by the construction of M', there exists a rule $Z'q \to q$ from (2). M' can also accept w. Hence, $L(\mathcal{H}) \subseteq L(M')$.

In a special case of Claim 8, M' performs the sequence of moves

$$\#Z's'w \vdash_{M'}^{k} \#Z'uqv,$$

where $u, v = \varepsilon$, $q \in F$. Then, by the construction of M', there exists a rule $Z'q \to q$ from (2). By using this rule, M' can accept w. However, on the basis of $\#Zsw \vdash_{\mathcal{H}}^{*} \#uqv$ and, since $u = v = \varepsilon$ and $q \in F$, $\mathcal{H}$ accepts w as well. Hence, $L(M') \subseteq L(\mathcal{H})$.

Consequently, $L(\mathcal{H}) = L(M')$, and the lemma holds. □

Lemma 6. CF – OPPDA $\neq \emptyset$

Proof. Consider the language $L_R = \{w \,\text{reversal}(w) \mid w \in \{a,b\}^*\}$. Obviously, $L_R \in \mathbf{CF}$. Suppose that there exists an OPPDA $\mathcal{H} = (M, \trianglelefteq)$, where

$$M = \big(Q, \Sigma, R, s, Z, F\big),$$

and $L(\mathcal{H}) = L_R$. Additionally, let $uv \in L(\mathcal{H})$, where $|u| = |v|$. To check the equality of u with its reversal v, $\mathcal{H}$ has to remember u first. So, with every symbol of u read from the input, $\mathcal{H}$ has to enter a unique configuration different from all the previous configurations.

Now, suppose that $u = (a^i b^j)^k$, where $i, j > \text{card}(Q)$ and $k > \text{card}(\Sigma)$. To save the information about the lengths of the sequences of as and bs, $\mathcal{H}$ has to insert nonempty sequences of the symbols into the pushdown. Indeed, states are not enough to encode this. Each sequence on the pushdown must differ from the previous one—contain different symbols. With the mth sequence, where $m > \text{card}(\Sigma)$, $\mathcal{H}$ has to use the previously used symbol to differentiate the sequences. However, it is in the conflict with $\trianglelefteq$, which is a contradiction. Therefore, the lemma holds. □

Lemma 7. RG $\subseteq$ OPPDA

Proof. For any regular language K, there is a finite automaton $M = \big(Q, \Sigma, R, s, F\big)$ satisfying $L(M) = K$. Then, we can define the OPPDA $\mathcal{H} = (M', \trianglelefteq)$, where $M' = \big(Q, \Sigma, R', s, a, F\big)$ with a being an arbitrary symbol from Σ and $R' = R \cup \{as \to s\}$. Obviously, $L(\mathcal{H}) = K$, so the lemma holds. □

Lemma 8. OPPDA – RG $\neq \emptyset$

Proof. This lemma follows from Example 3 (recall that $L(\mathcal{H}) \notin \mathbf{RG}$). □

Theorem 4. RG $\subset$ OPPDA $\subset$ CF

Proof. By Lemma 5, **OPPDA** $\subseteq$ **CF**. By Lemma 6, **CF** – **OPPDA** $\neq \emptyset$. By Lemma 7, **RG** $\subseteq$ **OPPDA**. By Lemma 8, **OPPDA** – **RG** $\neq \emptyset$. Hence, **RG** $\subset$ **OPPDA** $\subset$ **CF**, so the theorem holds. □

6 Conclusions

In this final section, we briefly conclude all the achieved results. First, we studied pure versions of PDAs and proved that the absence of special pushdown symbols does not affect their accepting power; one-pushdown PPDAs still characterize **CF** and PPDAs with two or more pushdown lists characterize **RE** — that is they are Turing-complete. Next, we introduced an order above the pushdown lists of multi-pushdown PPDAs and proved that they characterize **CF** regardless of a number of their pushdown lists. Finally, we introduce PPDAs with ordered pushdown symbols. We proved that they characterize a superfamily of **RG**, however, only a subfamily of **CF**.

7 Acknowledgements

This work was supported by the following grants: MŠMT CZ1.1.00/02.0070, BUT FIT-S-11-2, TAČR TE01010415, and CEZ MŠMT MSM0021630528.

References

[1] A. Gabrielian. Pure grammars and pure languages. *Int. J. Comput. Math.*, 9:3–16, 1981. DOI: 10.1080/00207168108803224.

[2] E. Mäkinen. A note on pure grammars. *Inf. Process. Lett.*, 23:271–274, 1986. DOI: 10.1016/0020-0190(86)90085-2.

[3] H.A. Maurer, A. Salomaa, and D. Wood. Pure grammars. *Information and Control*, 44:47–72, 1980. DOI: 10.1016/S0019-9958(80)90131-X.

[4] T. Masopust and A. Meduna. On pure multi-pushdown automata that perform complete-pushdown pops. In *Proceedings of the 12th International Conference on Automata and Formal Languages*, pages 325–336, 2008.

[5] T. Masopust and A. Meduna. On pure multi-pushdown automata that perform complete-pushdown pops. *Acta Cybernetica*, 19(2):537–552, 2009.

[6] R. Bidlo, P. Blatný, and A. Meduna. Automata with two-sided pushdowns defined over free groups generated by reduced alphabets. *Kybernetika*, 43(3):21–35, 2007.

[7] M.A. Harrison and O.H. Ibarra. Multi-tape and multi-head pushdown automata. *Information and Control*, 13:433–470, 1968. DOI: 10.1016/S0019-9958(68)90901-7.

[8] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*, page 171. Addison-Wesley, 1979.

[9] O.H. Ibarra. *Generalizations of pushdown automata.* PhD thesis, University of California, Berkeley, 1967.

[10] D.C. Kozen. *Automata and Computability*, page 224. Springer, New York, 2007. DOI: 10.1007/978-1-4612-1844-9.

[11] A. Meduna. Simultaneously one-turn two-pushdown automata. *Int. J. Comput. Math.*, 2003(80):679–687, 2003. DOI: 10.1080/0020716031000070616.

[12] D. Wood. *Theory of Computation*, page 274. Longman Higher Education, 1986.

[13] L. Breveglieri, A. Cherubini, C. Citrini, and S.C. Reghizzi. Multi-push-down languages and grammars. *Int. J. Found. Comput. Sci.*, 7(3):253–292, 1996. DOI: 10.1142/S0129054196000191.

[14] M. Atig. From multi to single stack automata. In *CONCUR 2010 - Concurrency Theory*, volume 6269 of *Lecture Notes in Computer Science*, pages 117–131. 2010. DOI: 10.1007/978-3-642-15375-4_9.

[15] M. Atig, B. Bollig, and P. Habermehl. Emptiness of multi-pushdown automata is 2etime-complete. In *Developments in Language Theory*, volume 5257 of *Lecture Notes in Computer Science*, pages 121–133. 2008. DOI: 10.1007/978-3-540-85780-8_9.

[16] N. Limaye and M. Mahajan. Membership testing: Removing extra stacks from multi-stack pushdown automata. In *Language and Automata Theory and Applications*, volume 5457 of *Lecture Notes in Computer Science*, pages 493–504. 2009. DOI: 10.1007/978-3-642-00982-2_42.

[17] A. Seth. Global reachability in bounded phase multi-stack pushdown systems. In *Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 615–628. 2010. DOI: 10.1007/978-3-642-14295-6_53.

[18] A. Meduna. *Automata and Languages: Theory and Applications*. Springer, London, 2000. DOI: 10.1007/978-1-4471-0501-5.

[19] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages, Vol. 1: Word, Language, Grammar*. Springer, New York, 1997.

[20] A. Salomaa. *Formal Languages*. Academic Press, London, 1973.

[21] G. Pighizzini, J. Shallit, and M. Wang. Unary context-free grammars and pushdown automata, descriptional complexity and auxiliary space lower bounds. *J. Comput. Syst. Sci.*, 65:393–414, 2002. DOI: 10.1006/jcss.2002.1855.