I F I G
RESEARCH
REPORT

INSTITUT FÜR INFORMATIK

FLIP-PUSHDOWN AUTOMATA: NONDETERMINISM IS BETTER THAN DETERMINISM

Markus Holzer Martin Kutrib

IFIG RESEARCH REPORT 0301
FEBRUARY 2003

Institut für Informatik
JLU Gießen
Arndtstraße 2
D-35392 Giessen, Germany
Tel: +49-641-99-32141
Fax: +49-641-99-32149
mail@informatik.uni-giessen.de
www.informatik.uni-giessen.de

JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

IFIG Research Report
IFIG Research Report 0301, February 2003

Flip-Pushdown Automata: Nondeterminism is Better Than Determinism

Markus Holzer[1]

Institut für Informatik, Technische Universität München

Boltzmannstraße 3, D-85748 Garching bei München, Germany

Martin Kutrib[2]

Institute für Informatik, Universität Giessen

Arndtstr. 2, D-35392 Giessen, Germany

Abstract. Flip-pushdown automata are pushdown automata with the additional ability to flip or reverse its pushdown. We investigate deterministic and nondeterministic flip-pushdown automata accepting by final state or empty pushdown. In particular, for nondeterministic flip-pushdown automata both acceptance criterion are equally powerful, while for determinism, acceptance by empty pushdown is strictly weaker. This nicely fits into the well-known results on ordinary pushdown automata. Moreover, we consider hierarchies of flip-pushdown automata w.r.t. the number of pushdown reversals. There we show that nondeterminism is better than determinism. Moreover, since there are languages which can be recognized by a deterministic flip-pushdown automaton with $k+1$ pushdown reversals but which cannot be recognized by a k-flip-pushdown (deterministic or nondeterministic) as shown in [10] we are able to complete our investigations with incomparability results on different levels of the hierarchies under consideration.

CR Subject Classification (1998): F.1, F.4.3

[1]E-mail: holzer@in.tum.de
[2]E-mail: mk@ifig.de

1 Introduction

A pushdown automaton is a one-way finite automaton with a separate pushdown store, that is a last-in first-out (LIFO) storage structure, which is manipulated by pushing and popping. Probably, such machines are best known for capturing the family of context-free languages, which was independently established by Chomsky [3] and Evey [5]. The origin of the pushdown concept is not clear and is attributed by most to Burks *et al.* [2] and Newell and Shaw [11]. A little later the term LIFO storage was used explicitly in the literature, by Samelson and Bauer [13], who proposed it as an aide in the translation of ALGOL formulas into machine instructions. Pushdown automata have been extended in various ways. Examples of extensions are variants of stacks [7, 9], queues or dequeues, while restrictions are for instance counters or one-turn pushdowns [8]. The results obtained for these classes of machines hold for a large variety of formal language classes, when appropriately abstracted. This led to the rich theory of abstract families of automata, which is the equivalent of abstract families of languages theory. For the general treatment of machines and languages we refer to Ginsburg [6].

In this paper, we consider a recently introduced extension of pushdown automata, so-called flip-pushdown automata [14]. Basically, a flip-pushdown automaton is an ordinary pushdown automaton with the additional ability to flip its pushdown during the computation. This allows the machine to push and pop at both ends of the pushdown. Hence, a flip-pushdown is a form of a dequeue storage structure, and thus becomes equally powerful to Turing machines, since a dequeue automaton can simulate two pushdowns. On the other hand, if the number of pushdown flips or pushdown reversals is zero, obviously the family of context-free languages is characterized. Thus it remains to investigate the number of pushdown reversals as a natural computational resource.

By Sarkar [14] it was shown for nondeterministic computations that if the number of pushdown flips is bounded by a constant, then a nonempty hierarchy of language classes is introduced, and it was conjectured that the hierarchy is strict. In [10] it is shown that $k+1$ pushdown reversals are better than k. To this end, a technique is developed to decrease the number of pushdown reversals, which simply speaking shows that flipping the pushdown is equivalent to reverse part of the remaining input for nondeterministic automata. An immediate consequence is that every flip-pushdown language accepted by a flip-pushdown with a constant number of pushdown reversals obeys a semi-linear Parikh mapping. It turned out, that the family of nondeterministic flip-pushdown languages share similar closure and non-closure properties as the family of context-free languages like, e.g., closure under intersection with regular sets, or the non-closure under complementation. Nevertheless, there are some interesting differences as, e.g., the non-closure under concatenation and Kleene star.

Here mainly we investigate several variants of deterministic flip-pushdown automata, nondeterministic flip-pushdown automata, and their relationships among each other. In particular, we distinguish flip-pushdown automata where the number of pushdown reversals has to be exactly k or has to be at most k. For nondeterministic automata this distinction is shown to make no difference. This is not true for deterministic automata that accept by final state. Interestingly, it also makes no difference if deterministic automata are considered that accept by empty pushdown. The two modes of acceptance are the second considered distinction. Our main contribution are results yielding, for a fixed k, strict inclusions in between the different types of deterministic flip-pushdown automata and, furthermore, between deterministic and nondeterministic automata. By an adaption of the hierarchy result in [10], all considered classes are separated for k and $k+1$. The question, how the number of pushdown reversals relates to determinism/nondeterminism, at most k/exactly k reversals, or acceptance by final state/empty pushdown, is answered by deriving incomparability for all classes which are not related by a strict inclusion.

The paper is organized as follows: The next section contains preliminaries and basics on flip-pushdown automata. Then Section 3 is devoted to the separation of deterministic and nondeterministic flip-pushdown automata. The next section deals with the comparison between flip-pushdown automata making exactly k and at most k pushdown reversals. In Section 5 we consider the relationships caused by the acceptance modes empty pushdown and final state. The penultimate Section 6 is devoted to the separation of the flip-pushdown hierarchies for all considered classes. Finally, we summarize our results and pose a few open questions in Section 7.

2 Preliminaries

We denote the powerset of a set S by 2^S. The empty word is denoted by λ, the reversal of a word w by w^R, and for the length of w we write $|w|$. For the number of occurrences of a symbol a in w we use the notation $|w|_a$. In the following we consider pushdown automata with the ability to flip their pushdowns. These machines were recently introduced by Sarkar [14] and are defined as follows:

Definition 1 *A* nondeterministic flip-pushdown automaton (NFPDA) *is a system $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F\rangle$, where Q is a finite set of states, Σ is the finite input alphabet, Γ is a finite pushdown alphabet, δ is a mapping from $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ to finite subsets of $Q \times \Gamma^*$ called the transition function, Δ is a mapping from Q to 2^Q, $q_0 \in Q$ is the initial state, $Z_0 \in \Gamma$ is a particular pushdown symbol, called the bottom-of-pushdown symbol, which initially appears on the pushdown store, and $F \subseteq Q$ is the set of final states.*

A *configuration* or *instantaneous description* of a flip-pushdown automaton is a triple (q, w, γ), where q is a state in Q, w a string of input symbols, and γ is a string of pushdown symbols. A flip-pushdown automaton A is said to be in configuration (q, w, γ) if A is in state q with w as remaining input, and γ

on the pushdown store, the rightmost symbol of γ being the top symbol on the pushdown. If p, q are in Q, a is in $\Sigma \cup \{\lambda\}$, w in Σ^*, γ and β in Γ^*, and Z is in Γ, then we write $(q, aw, \gamma Z) \vdash_A (p, w, \gamma\beta)$, if the pair (p, β) is in $\delta(q, a, Z)$, for "ordinary" pushdown transitions and $(q, aw, Z_0\gamma) \vdash_A (p, aw, Z_0\gamma^R)$, if p is in $\Delta(q)$, for pushdown-flip or pushdown-reversal transitions. Whenever there is a choice between an ordinary pushdown transition or a pushdown reversal one, the automaton nondeterministically chooses the next move. Observe, that we do not want the flip-pushdown automaton to move the bottom-of-pushdown symbol when the pushdown is flipped. As usual, the reflexive transitive closure of $\vdash_A$ is denoted by $\vdash_A^*$. The subscript A will be dropped from $\vdash_A$ and $\vdash_A^*$ whenever the meaning remains clear.

A deterministic flip-pushdown automaton (DFPDA) is a flip-pushdown automaton for which there is at most one choice of action for any possible configuration. In particular, there must never be a choice of using an input symbol or of using λ input. Formally, a flip-pushdown automaton $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F\rangle$ is *deterministic* if:

1. $\delta(q, a, Z)$ contains at most one element, for all a in $\Sigma \cup \{\lambda\}$, q in Q, and Z in Γ.
2. If $\delta(q, \lambda, Z)$ is not empty, then $\delta(q, a, Z)$ is empty, for all a in Σ, q in Q, and Z in Γ.
3. $\Delta(q)$ contains at most one element, for all q in Q.
4. If $\Delta(q)$ is not empty, then $\delta(q, a, Z)$ is empty, for all a in $\Sigma \cup \{\lambda\}$, q in Q, and Z in Γ.

Let $k \geq 0$. For a (deterministic) flip-pushdown automaton A we define $T_{\leq k}(A)$, the language *accepted by final state and at most k pushdown reversals*, to be

$$T_{\leq k}(A) = \{\, w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_A^* (q, \lambda, \gamma) \text{ with at most } k \text{ pushdown reversals, for some } \gamma \in \Gamma^* \text{ and } q \in F \,\}.$$

Example 2 We define the deterministic flip-pushdown automaton
$$A = \langle \{q_0, q_1, q_2, q_3\}, \{a, b, \$\}, \{A, B, Z_0\}, \delta, \Delta, q_0, Z_0, \{q_3\}\rangle,$$
where

1. $\delta(q_0, a, Z_0) = \{(q_0, Z_0A)\}$
2. $\delta(q_0, b, Z_0) = \{(q_0, Z_0B)\}$
3. $\delta(q_0, a, A) = \{(q_0, AA)\}$
4. $\delta(q_0, b, A) = \{(q_0, AB)\}$
5. $\delta(q_0, a, B) = \{(q_0, BA)\}$
6. $\delta(q_0, b, B) = \{(q_0, BB)\}$
7. $\delta(q_0, \$, Z_0) = \{(q_1, Z_0)\}$
8. $\delta(q_0, \$, A) = \{(q_1, A)\}$
9. $\delta(q_0, \$, B) = \{(q_1, B)\}$
10. $\delta(q_2, a, A) = \{(q_2, \lambda)\}$
11. $\delta(q_2, b, B) = \{(q_2, \lambda)\}$
12. $\delta(q_2, \lambda, Z_0) = \{(q_3, \lambda)\}$

and $\Delta(q_1) = \{q_2\}$, that accepts by final state with one pushdown reversal the non-context-free language $\{\, w\$w \mid w \in \{a, b\}^* \}$.

The transitions (1) through (6) allow A to store the input on the pushdown. If A reads \$, then it moves from state q_0 to q_1 and directly applies the flip operation specified by $\Delta(q_1) = \{q_2\}$ afterwards, since this is the only choice the machine has. Then in state q_2 automaton A tries to match the remaining input symbols with the reversed pushdown content. This is done with the transitions (10) and (11). Thus, if the input is of the form $w\$w$, then all symbols will match, and A will change into the accepting state q_3 with transition (12). In addition, A will empty its pushdown with its last transition.

The family of languages accepted by nondeterministic resp. deterministic flip-pushdown automata by final state making at most k pushdown reversals is denoted by $\mathscr{L}(\mathrm{NFPDA}_{\leq k})$ resp. $\mathscr{L}(\mathrm{DFPDA}_{\leq k})$. Furthermore, let

$$\mathscr{L}(\mathrm{NFPDA}_{\leq \mathit{fin}}) = \bigcup_{k=0}^{\infty} (\mathrm{NFPDA}_{\leq k})$$

and similarly $\mathscr{L}(\mathrm{DFPDA}_{\leq \mathit{fin}})$.

An essential technique for flip-pushdown automata is the so-called "flip-pushdown input-reversal" technique, which has been developed and proved in [10]. It allows to simulate flipping the pushdown by reversing the (remaining) input, and reads as follows. For sake of completeness the proof from [10] is included.

Theorem 3 *Let $k \geq 0$. Language L is accepted by a nondeterministic flip-pushdown automaton $A_1 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F\rangle$ by final state with at most $k+1$ pushdown reversals, i.e., $L = T_{\leq (k+1)}(A_1)$, if and only if language*

$$\begin{aligned} L_R = \{\, wv^R \mid{}& (q_0, w, Z_0) \vdash^*_{A_1} (q_1, \lambda, Z_0\gamma) \text{ with at most } k \text{ reversals},\\ & q_2 \in \Delta(q_1), \text{ and } (q_2, v, Z_0\gamma^R) \vdash^*_{A_1} (q_3, \lambda, q_4) \text{ without any reversal}, q_4 \in F \,\} \end{aligned}$$

is accepted by a nondeterministic flip-pushdown automaton A_2 by final state with at most k pushdown reversals, i.e., $L_R = T_{\leq k}(A_2)$.

In order to simplify presentation, we introduce the notion of a generalized flip-pushdown automaton $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F\rangle$, where Q, Σ, Γ, Δ, $q_0 \in Q$, $Z_0 \in \Gamma$, and $F \subseteq Q$ are as in the case of ordinary flip-pushdown automata, and δ is a *finite domain* mapping from $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma^*$ to the finite subsets of $Q \times \Gamma^*$. With standard techniques one can construct an ordinary flip-pushdown automaton from a given generalized one, without increasing the number of pushdown-flips. Due to the ability to read words instead of symbols, the necessary checks, whether a push or pop action can be performed in the backward simulation becomes easier to describe.

The following proof has been presented in [10] in terms of nondeterministic flip-pushdown automata that make exactly k pushdown reversals and accept by empty pushdown. The equivalence of these models to the one of Theorem 3 is shown in Section 4 and Section 5.

Proof. [of Theorem 3] We only prove the direction from left to right. The converse implication can be shown by similar arguments.

Let $A_1 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, \emptyset \rangle$ be a flip-pushdown automaton satisfying $\gamma \in \{\lambda\} \cup \{ ZX \mid X \in \Gamma \}$ for all $(p, \gamma) \in \delta(q, a, Z)$, where $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$. This normal form can be easily achieved.

Then we define a generalized flip-pushdown automaton

$$A_2 = (Q \cup \bar{Q} \cup \{\bar{q}_f\}, \Sigma, \Gamma \cup \bar{\Gamma} \cup Q, \delta', \Delta', q_0, Z_0, \{\bar{q}_f\}),$$

where $\bar{Q} = \{ \bar{q} \mid q \in Q \}$, $\bar{\Gamma} = \{ \bar{Z} \mid Z \in \Gamma \}$, and δ' and Δ' are specified as follows:

1. For all $q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$, set $\delta'(q, a, Z)$ includes all elements of $\delta(q, a, Z)$ and
2. for all $q \in Q$, let $\Delta'(q)$ contain all elements of $\Delta(q)$.
3. For all $r \in Q$, if $\Delta(r) \neq \emptyset$, then $\delta'(r, a, Z)$ contains $(\bar{q}, ZZ_0 r \bar{Z}_0)$, where $q \in Q$ satisfies $(p, \lambda) \in \delta(q, a, Z_0)$ for some $p \in Q$ and $a \in \Sigma \cup \{\lambda\}$.
4. For all $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $X, Y \in \Gamma$, let $\delta'(\bar{q}, a, \bar{X}\bar{Y})$ contain $(\bar{p}, \bar{X})$ if $(q, XY) \in \delta(p, a, X)$.
5. For all $p, q, r \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $X, Y \in \Gamma$, then
   (a) let $\delta'(\bar{q}, a, \bar{X})$ contain $(\bar{p}, \bar{X}\bar{Y})$ if $(q, \lambda) \in \delta(p, a, Y)$ and
   (b) let $\delta'(\bar{q}, a, Xr\bar{X})$ contain $(\bar{p}, r\bar{Y})$ if $(q, \lambda) \in \delta(p, a, Y)$.
6. For all $X \in \Gamma$ and $p \in \Delta(r)$, for some $r \in Q$, let $\delta'(\bar{p}, \lambda, Z_0 X r \bar{X})$ contain $(\bar{q}_f, \lambda)$.

We summarize the transitions for the backward simulation of A_2 in Table 1.

<table>
<tr><th colspan="4">Simulation</th></tr>
<tr><th colspan="2">forward</th><th colspan="2">backward</th></tr>
<tr><td>flip</td><td>$p \in \Delta(q)$</td><td>$(\bar{p}', ZZ_0 q \bar{Z}_0) \in \delta'(\bar{q}, a, Z)$
if $(p'', \lambda) \in \delta(p', a, Z_0)$</td><td>push</td></tr>
<tr><td>push</td><td>$(p, XY) \in \delta(q, a, X)$</td><td>$(\bar{q}, \bar{X}) \in \delta'(\bar{p}, a, \bar{X}\bar{Y})$</td><td>pop</td></tr>
<tr><td rowspan="2">pop</td><td rowspan="2">$(p, \lambda) \in \delta(q, a, Y)$</td><td>$(\bar{q}, \bar{X}\bar{Y}) \in \delta'(\bar{p}, a, \bar{X})$</td><td>push (a)</td></tr>
<tr><td>$(\bar{q}, rY) \in \delta'(\bar{p}, a, XrX)$</td><td>push (b)</td></tr>
<tr><td>accept</td><td>$(p, \lambda) \in \delta(q, a, Z_0)$</td><td colspan="2">—</td></tr>
<tr><td colspan="2">—</td><td>$(\bar{q}_f, \lambda) \in \delta'(\bar{p}, \lambda, Z_0 X q \bar{X})$
if $p \in \Delta(q)$</td><td>accept</td></tr>
</table>

Table 1: Transitions of A_2 for the backward simulation of A_1.

Transitions from (1) and (2) cause A_2 to simulate A_1 step-by-step until the $(k+1)$st pushdown reversal done by A_1 appears. All elements described in (3), (4), (5), and (6) allow A_2 to start a backward simulation of A_1 on the reverse remaining input. To be more precise, the transitions in (3) start the backward simulation of A_2 by undoing the very last step of A_1, i.e., by pushing $Z_0 r \bar{Z}_0$ onto the pushdown, reading symbol a, and continuing with state $\bar{q}$, whenever A_1 has used transition $(p, \lambda) \in \delta(q, a, Z_0)$, for some $p \in Q$, in its last computation step. Then in (4) push moves of A_1 are simulated as pop moves by A_2, always assuming to have a boldface symbol on top of the pushdown. Moreover, transitions specified in (5) simulate pop moves of A_1 by push moves of A_2. Here we have to consider two cases, namely starting a sub-computation which (a) comes back to the same pushdown height or (b) comes not back to the same pushdown height. In the latter case A_2 has to pop a compatible non-boldface symbol together with a boldface symbol in order to decrease the pushdown height. Finally, in (6) the termination of the computation is done, by checking that the pushdown contains a string of the form $Z_0 X r \bar{X}$ for some $X \in \Gamma$ and $r \in Q$, and has reached some state in $\Delta(r)$.

Now assume that $w \in N_{k+1}(A_1)$ such that $w = uva$ with

$$\begin{aligned}(q_0, uva, Z_0) \vdash^*_{A_1} (q_1, va, Z_0 X \gamma) \vdash_{A_1} (q_2, va, Z_0 \gamma^R X) \\ \vdash^*_{A_1} (q_3, a, Z_0) \vdash_{A_1} (q_4, \lambda, \lambda),\end{aligned}$$

where $u, v \in \Sigma^*$, $a \in \Sigma \cup \{\lambda\}$, $X \in \Gamma \cup \{\lambda\}$, $\gamma \in \Gamma^*$, $X = \lambda$ implies $\gamma = \lambda$, and the last pushdown reversal appears at $(q_1, va, Z_0 X \gamma) \vdash_{A_1} (q_2, va, Z_0 \gamma^R X)$. Thus, by our previous considerations we find the simulation

$$\begin{aligned}(q_0, uav^R, Z_0) \vdash^*_{A_2} (q_1, av^R, Z_0 X \gamma) \vdash_{A_2} (\bar{q}_3, v^R, Z_0 X \gamma Z_0 q_1 \bar{Z}_0) \\ \vdash^*_{A_2} (\bar{q}_2, \lambda, Z_0 X q_1 \bar{X}) \vdash_{A_2} (\bar{q}_f, \lambda, \lambda),\end{aligned}$$

and therefore $uav^R = u(va)^R$ belongs to $T_k(A_2)$, since the number of reversals was decreased by one. By similar reasoning, if $u(va)^R \in T_k(A_2)$, then $uva \in N_{k+1}(A_1)$. Since state acceptance and acceptance by empty pushdown coincides for flip-pushdown automata, the claim follows. $\square$

3 Determinism versus Nondeterminism

We show that nondeterminism is better than determinism. In particular, the family of languages accepted by deterministic flip-pushdown automata with at most k pushdown reversals is a strict subset of the family of languages accepted by nondeterministic flip-pushdown automata with at most k pushdown reversals. To this end, we need the closure of deterministic flip-pushdown languages under intersection with regular sets and under the prefix operation. The first property is straightforward by simulating a deterministic finite automaton in the control of the flip-pushdown automaton at the same time. For the second property let $w = a_1 a_2 \cdots a_n$ with $a_i \in \Sigma$, for $1 \leq i \leq n$, be some word over Σ. The *set of prefixes* of w is defined to be $\{\lambda, a_1, a_1 a_2, \ldots, a_1 \cdots a_n\}$. For a language $L \subseteq \Sigma^*$ and a natural number $i \geq 1$ let

$$P_i(L) = \{\, w \in L \mid \text{ exactly } i \text{ prefixes of } w \text{ are belonging to } L \,\}.$$

The following theorem shows that deterministic flip-pushdown languages are closed under the P_i operation.

Theorem 4 *Let $i \geq 1$ and $k \geq 0$. If $L \in \mathscr{L}(\mathrm{DFPDA}_{\leq k})$, then $P_i(L) \in \mathscr{L}(\mathrm{DFPDA}_{\leq k})$.*

Proof. Let $A_1 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F\rangle$ be a deterministic flip-pushdown automaton such that $L = T_{\leq k}(A_1)$. Then define

$$A_2 = \langle Q \times \{0, 1, \ldots, i+1\}, \Gamma, \delta', \Delta', q, Z_0, \{\, (q, i) \mid q \in F \,\}\rangle,$$

where q equals $(q_0, 0)$, if $\lambda \notin L$, and is $(q_0, 1)$, otherwise. The transition functions δ' and Δ' are defined as follows: For all $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$ let

1. $((p, j), \gamma) \in \delta'((q, j), a, Z)$, if $(p, \gamma) \in \delta(q, a, Z)$ and ($0 \leq j \leq i + 1$ and $p \in Q \setminus F$) or ($j = i + 1$ and $p \in F$),
2. $((p, j + 1), \gamma) \in \delta'((q, j), a, Z)$, if $(p, \gamma) \in \delta(q, a, Z)$, $0 \leq j \leq i$, and $p \in F$,
3. $\Delta'((q, j)) = \{\, (p, j) \mid p \in \Delta(q) \,\}$, for all $0 \leq j \leq i + 1$.

It is easy to see that A_2 accepts an input w if and only if exactly i prefixes of w are belonging to L. □

The following corollary is an immediate consequence of the above given theorem.

Corollary 5 *Let $i \geq 1$. If $L \in \mathscr{L}(\mathrm{DFPDA}_{\leq fin})$, then $P_i(L) \in \mathscr{L}(\mathrm{DFPDA}_{\leq fin})$.*

Another essential ingredient of the proof of the following theorem is a generalization of Ogden's lemma, which is due to Bader and Moura [1] and reads as follows: For any context-free language L, there exists a natural number n, such that for all words z in L, if d positions in z are "distinguished" and e positions are "excluded," with $d > n^{e+1}$, then there are words u, v, w, x, and y such that $z = uvwxy$ and (1) vx contains at least one distinguished position and no excluded positions, (2) if r is the number of distinguished positions and s is the number of excluded positions in vwx, then $r \leq n^{s+1}$, and (3) word uv^iwx^iy is in L for all $i \geq 0$.

Now we are ready to separate determinism from nondeterminism for flip-pushdown languages.

Theorem 6 *Let $k \geq 0$. Then $\mathscr{L}(\mathrm{DFPDA}_{\leq k}) \subset \mathscr{L}(\mathrm{NFPDA}_{\leq k})$.*

Proof. The inclusion immediately follows for structural reasons. For the strictness we argue as follows.

For $k = 0$ the theorem states the well-known result for deterministic context-free languages. So let $k \geq 1$ and $L = \{\, ww \mid w \in \{a, b\}^+ \,\}$. Since L belongs to $\mathscr{L}(\mathrm{NFPDA}_{\leq 1})$, it belongs to $\mathscr{L}(\mathrm{NFPDA}_{\leq k})$ as well. In the following we show that $L \notin \mathscr{L}(\text{DFPDA}_{\leq fin})$. This immediately implies that $L \notin \mathscr{L}(\text{DFPDA}_{\leq k})$ for any k.

Assume to the contrary, that $L \in \mathscr{L}(\text{DFPDA}_{\leq k})$, for some k. Then by Theorem 4 we obtain that $P_2(L) \in \mathscr{L}(\text{DFPDA}_{\leq k})$, and since this language family is closed under intersection with regular sets, also the language $L' = P_2(L) \cap ba^*ba^*ba^*ba^*$ is a deterministic k-flip language. Thus, language L' belongs to $\mathscr{L}(\text{NFPDA}_{\leq fin})$, which is closed under rational a-transduction as shown in [10]. Since L' contains exactly the words of the form $ba^nba^na^mba^nba^na^m$, this implies that the language $L'' = \{\, a^nb^{n+m}c^nd^{n+m} \mid m, n \geq 1 \,\}$ belongs also to $\mathscr{L}(\text{NFPDA}_{\leq fin})$. In order to obtain a contradiction, it remains to show that this is not the case. To this end, we use Theorem 3 and the generalization of Ogden's Lemma.

Assume to the contrary, that language $L'' = \{\, a^nb^{n+m}c^nd^{n+m} \mid m, n \geq 1 \,\}$ belongs to $\mathscr{L}(\text{NFPDA}_{\leq k})$ for some k. Then we apply k times the flip-pushdown input-reversal Theorem 3 to L obtaining a context-free language. Since we do the input reversal from right-to-left, the block of d's remains adjacent in all words. Hence a word w in the context-free language reads as $w = ud^{n+m}v$, where $|uv|_a = |uv|_c = n$ and $|uv|_b = n + m$. Then it is an easy exercise to show that this language cannot be context-free using the generalized version of Ogden's lemma. This contradicts our assumption on L'', and thus, it does not belong to $\mathscr{L}(\text{NFPDA}_{\leq k})$, for any $k \geq 0$. This shows that $L'' \notin \mathscr{L}(\text{NFPDA}_{\leq fin})$.

Hence, we can conclude that L does not belong to $\mathscr{L}(\text{DFPDA}_{\leq k})$ for any k. $\square$

Corollary 7 $\mathscr{L}(\text{DFPDA}_{\leq fin}) \subset \mathscr{L}(\text{NFPDA}_{\leq fin})$.

4 Exactly *k* Flips versus At Most *k* Flips

Let $k \geq 0$. For a flip-pushdown automaton $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F \rangle$ we define $T_{=k}(A)$, the language *accepted by final state and exactly k pushdown reversals*, to be

$$T_{=k}(A) = \{\, w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_A^* (q, \lambda, \gamma) \text{ with exactly } k \text{ pushdown reversals, for some } \gamma \in \Gamma^* \text{ and } q \in F \,\}.$$

Similarly, we use the notations $\mathscr{L}(\text{DFPDA}_{=k})$, $\mathscr{L}(\text{DFPDA}_{=fin})$ etc. The next result shows that for nondeterministic computations the classes for exactly k flips and at most k flips coincide. It turns out that this is not true for deterministic computations, but the result shows also that exactly k flips are included in at most k flips for deterministic computations. This does not follow for structural reasons since the automaton with at most k flips may accept some input with strictly less than k flips. By definition these inputs do not belong to the accepted language of the automaton with exactly k flips.

The idea for both directions of the proof relies on the simple fact that a flip-pushdown automaton can count the number of reversals performed during its computation in its finite control. Moreover, the construction must ensure that it is possible to increase the number of pushdown flips without changing the accepted language.

Theorem 8 *Let $k \geq 0$. Then*

$$\mathscr{L}(\mathrm{NFPDA}_{=k}) = \mathscr{L}(\mathrm{NFPDA}_{\leq k}) \text{ and } \mathscr{L}(\mathrm{DFPDA}_{=k}) \subseteq \mathscr{L}(\mathrm{DFPDA}_{\leq k}).$$

Proof. Let $A_1 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F \rangle$ be a (deterministic) flip-pushdown automaton. Then we define

$$A_2 = \langle Q \times \{0, 1, \ldots, k\}, \Sigma, \Gamma, \delta', \Delta', (q_0, 0), Z_0, \{\, (q, k) \mid q \in F \,\} \rangle,$$

where δ' and Δ' are defined as follows: For all $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$ let

1. $((p, i), \gamma) \in \delta'((q, i), a, Z)$, if $(p, \gamma) \in \delta(q, a, Z)$, for $0 \leq i \leq k$,
2. $\Delta'((q, i)) = \{\, (p, i + 1) \mid p \in \Delta(q) \,\}$, for $0 \leq i < k$,
3. $\Delta'((q, k)) = \{\, (p, k) \mid p \in \Delta(q) \,\}$.

By construction one sees, that whenever A_1 is a deterministic flip-pushdown automaton, then A_2 is a deterministic machine, too.

Observe, that each state in A_2 is a tuple, where in the second component the number of pushdown reversals performed so far is stored. The transitions from (1) cause A_2 to simulate the original flip-pushdown A_1. If a pushdown flip is performed, the state's second component is increased by one, which is seen in (2). Since only the states of the form (q, k), for $q \in F$, are accepting states, the automaton A_2 must have done at least k pushdown reversals in order to reach a state of this form.

Assume that $w \in T_{=k}(A_1)$. Then $(q_0, w, Z_0) \vdash^*_{A_1} (q, \lambda, \gamma)$ for some $q \in F$ and $\gamma \in \Gamma^*$ with exactly k reversals. Thus, $((q_0, 0), w, Z_0) \vdash^*_{A_2} ((q, k), \lambda, \gamma)$. Therefore, $w \in T_{\leq k}(A_2)$, since the number of reversals is at most k now and no state of the form (q, i) with $0 \leq i < k$ is an accepting one. By similar reasoning, if $w \in T_{\leq k}(A_2)$, then $w \in T_{=k}(A_1)$.

Conversely we argue as follows. Let $A_2 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F \rangle$ be a flip pushdown automaton. Then we define

$$A_1 = \langle (Q \cup \{q_0'\}) \times \{0, 1, \ldots, k\}, \Sigma, \Gamma, \delta', \Delta', (q_0', 0), Z_0, \{\, (q, k) \mid q \in F \,\} \rangle,$$

where δ' and Δ' are defined as follows: For all $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$ let

1. $((q_0, i), Z_0) \in \delta((q_0', i), \lambda, Z_0)$, for $0 \leq i \leq k$,
2. $((p, i), \gamma) \in \delta'((q, i), a, Z)$, if $(p, \gamma) \in \delta(q, a, Z)$,

3. $\Delta'((q_0', i)) = \{(q_0', i+1)\}$, for $0 \leq i < k$,
4. $\Delta'((q, i)) = \{ (p, i+1) \mid p \in \Delta(q) \}$, for $0 \leq i < k$.

The transitions from (1) cause A_1 to enter the initial configuration of A_2, while those transitions in (2) simulate the original flip-pushdown A_2. If a pushdown flip is performed, the state's second component is increased by one, which is seen in (4). In order to obtain exactly k flips, additional pushdown reversals can be performed using (3) at the very beginning of the computation.

Assume that $w \in T_{\leq k}(A_2)$. Then $(q_0, w, Z_0) \vdash_{A_2}^{*} (q, \lambda, \gamma)$ for some $q \in F$ and $\gamma \in \Gamma^*$ with ℓ reversals, for some $0 \leq \ell \leq k$. Thus,

$$((q_0', 0), w, Z_0) \vdash_{A_1}^{k-\ell} ((q_0', k-\ell), w, Z_0) \vdash_{A_1} ((q_0, k-\ell), w, Z_0) \vdash_{A_1}^{*} ((q, k), \lambda, \gamma),$$

where the first $k-\ell$ steps are pushdown flips only. Therefore, $w \in T_{=k}(A_1)$, since the number of reversals is exactly k now. By similar reasoning, if $w \in T_{=k}(A_1)$, then $w \in T_{\leq k}(A_2)$. □

Due to the equality, in the sequel we can simplify the notation to $\mathscr{L}(\text{NFPDA}_k)$ etc. The next step is to separate the families $\mathscr{L}(\text{DFPDA}_{=k})$ and $\mathscr{L}(\text{DFPDA}_{\leq k})$ for $k \geq 1$. Trivially, they coincide for $k = 0$.

Theorem 9 *Let $k \geq 1$. Then $\mathscr{L}(\text{DFPDA}_{=k}) \subset \mathscr{L}(\text{DFPDA}_{\leq k})$.*

Proof. The inclusion has been shown in the previous theorem. The strictness is seen as follows. Define $L = L' \cup L''$, where

$$L' = \{ \texttt{\#}w\texttt{\#} \mid w \in \{a,b\}^* \} \quad \text{and} \quad L'' = \{ \texttt{\#}w_0\texttt{\#}w_1\texttt{\$}w_1\texttt{\#} \mid w_0, w_1 \in \{a,b\}^* \}.$$

Obviously, language L is accepted by some deterministic flip-pushdown automaton making at most one flip. So, L is accepted by some $\text{DFPDA}_{\leq k}$ for any $k \geq 1$.

Assume to the contrary that language L is accepted by some deterministic flip-pushdown automaton $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F \rangle$ with exactly k pushdown reversals. Consider an arbitrary word $\texttt{\#}w_0\texttt{\#}$ that belongs to L' and, hence, also to L. There is an accepting computation of A which reads as $(q_0, \texttt{\#}w_0\texttt{\#}, Z_0) \vdash_A^* (q, \lambda, \gamma)$, where $q \in F$ and $\gamma \in \Gamma^*$. Since for every $w_1 \in \{a,b\}^*$ we find that $v = \texttt{\#}w_0\texttt{\#}w_1\texttt{\$}w_1\texttt{\#}$ belongs to L, there must be also an accepting computation on v. Since A is deterministic and, thus, has already performed k pushdown reversals on the prefix $\texttt{\#}w_0\texttt{\#}$ of v, we can construct an ordinary deterministic pushdown automaton accepting the language $\tilde{L} = \{ w\texttt{\$}w\texttt{\#} \mid w \in \{a,b\}^* \}$. The deterministic pushdown automaton accepting $\tilde{L}$ initially creates the pushdown content γ with a series of λ-moves, changes to state q, simulates A step-by-step, and accepts if A does.

Since $\tilde{L}$ is not even context free, we obtain a contradiction to our assumption. So $L \notin \mathscr{L}(\text{DFPDA}_{=k})$, and the assertion follows. □

Corollary 10 $\mathscr{L}(\mathrm{DFPDA}_{=fin}) \subset \mathscr{L}(\mathrm{DFPDA}_{\leq fin})$.

Proof. In the proof of the previous theorem we obtained for the witness language $L \notin \mathscr{L}(\mathrm{DFPDA}_{=k})$. Since k was choosen arbitrarily, it follows $L \notin \mathscr{L}(\mathrm{DFPDA}_{=fin})$. $\square$

So far we have the strict inclusions

$$\mathscr{L}(\mathrm{DFPDA}_{=k}) \subset \mathscr{L}(\mathrm{DFPDA}_{\leq k}) \subset \mathscr{L}(\mathrm{NFPDA}_{k}),$$

for all $k \geq 1$, and

$$\mathscr{L}(\mathrm{DFPDA}_{=fin}) \subset \mathscr{L}(\mathrm{DFPDA}_{\leq fin}) \subset \mathscr{L}(\mathrm{NFPDA}_{fin}).$$

5 Empty Pushdown versus Final State

Now we are going to consider flip-pushdown automata with a different mode of acceptance. For ordinary deterministic pushdown automata it is well known that acceptance by empty pushdown yields strictly weaker devices than acceptance by final state. In the following we distinguish for both modes also acceptance with at most and with exactly k pushdown reversals, respectively.

Let $k \geq 0$. For a flip-pushdown automaton $A = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F \rangle$ we define $N_{=k}(A)$, the language *accepted by empty pushdown and exactly k pushdown reversals*, to be

$$N_{\leq k}(A) = \{\, w \in \Sigma^* \mid (q_0, w, Z_0) \vdash_A^* (q, \lambda, \lambda) \text{ with at most } k \text{ pushdown reversals, for some } q \in Q \,\}.$$

As before we define $N_{=k}(A)$, and use the notations $\mathscr{L}_N(\mathrm{DFPDA}_{\leq k})$ and $\mathscr{L}_N(\mathrm{DFPDA}_{=k})$ etc.

The first result in this section concerns nondeterministic automata. It generalizes the theorem on ordinary pushdown automata, that languages accepted by nondeterministic flip-pushdown automata by final state are exactly those languages accepted by nondeterministic flip-pushdown automata by empty pushdown. We state the theorem without proof since it is a simple adaption of the proof for ordinary pushdown automata.

Theorem 11 *Let $k \geq 0$. Then*

$$\mathscr{L}_N(\mathrm{NFPDA}_{\leq k}) = \mathscr{L}(\mathrm{NFPDA}_{\leq k}) \text{ and } \mathscr{L}_N(\mathrm{NFPDA}_{=k}) = \mathscr{L}(\mathrm{NFPDA}_{=k}).$$

In particular, the theorem together with Theorem 8 shows

$$\mathscr{L}_N(\mathrm{NFPDA}_{\leq k}) = \mathscr{L}(\mathrm{NFPDA}_{\leq k}) = \mathscr{L}(\mathrm{NFPDA}_{=k}) = \mathscr{L}_N(\mathrm{NFPDA}_{=k})$$

and $\mathscr{L}_N(\text{NFPDA}_{\mathit{fin}}) = \mathscr{L}(\text{NFPDA}_{\mathit{fin}})$. So we have the robust situation that for the acceptance power of flip-pushdown automata it makes neither a difference whether the mode is by empty pushdown or by final state nor whether at most k or exactly k flips are considered. For deterministic classes it does make a difference, but interestingly, we can show that in case of acceptance by empty pushdown the families with at most k and exactly k pushdown reversals coincide. In case of acceptance by final state the classes have already been separated.

Theorem 12 *Let $k \geq 0$. Then $\mathscr{L}_N(\text{DFPDA}_{\leq k}) = \mathscr{L}_N(\text{DFPDA}_{=k})$.*

Proof. In order to show the inclusion $\mathscr{L}_N(\text{DFPDA}_{\leq k}) \subseteq \mathscr{L}_N(\text{DFPDA}_{=k})$ let $A_1 = \langle Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, \emptyset\rangle$ be a deterministic flip-pushdown automaton. Basically, the idea is to simulate A_1 until it empties its pushdown and then to perform the missing number of flips. To this end, the simulating automaton A_2 has to count the number of simulated flips in its finite control, and has to provide a new bottom-of-pushdown symbol Z_0'. The latter is necessary to continue the computation when A_1 has emptied its pushdown. Since now the old bottom-of-pushdown symbol Z_0 is object of pushdown reversals, special attention has to be paid. Every time A_2 wishes to simulate a pushdown reversal of A_1, it has to push Z_0, to perform the reversal, and to pop Z_0.

Let Q' and Q'' be two disjoint copies of the set Q, and q_0' and q_e' and Z_0' be three new symbols. We define

$$A_2 = \langle \{q_0'\} \cup ((Q \cup Q' \cup Q'' \cup \{q_e'\}) \times \{1, 2, \ldots, k\}), \Sigma, \Gamma \cup \{Z_0'\}, \delta', \Delta', q_0', Z_0', \emptyset\rangle,$$

where δ' and Δ' are defined as follows. For all $p, q \in Q$, $a \in \Sigma \cup \{\lambda\}$, and $Z \in \Gamma$:

1. $\delta'(q_0', \lambda, Z_0') = \{((q_0, 0), Z_0'Z_0)\}$,
2. $\delta'((q, i), a, Z) = \{((p, i), \gamma)\}$, if $\delta(q, a, Z) = \{(p, \gamma)\}$, for $0 \leq i \leq k$,
3. $\delta'((q, i), \lambda, Z) = \{((p', i), Z_0)\}$, if $\Delta(q) = \{p\}$, for $0 \leq i < k$,
4. $\Delta'((p', i)) = \{(p'', i + 1)\}$, for $0 \leq i < k$,
5. $\delta'((p'', i), \lambda, Z_0) = \{((p, i), \lambda)\}$, for $0 \leq i \leq k$,
6. $\delta'((q, i), \lambda, Z_0') = \{((q_e', i), Z_0')\}$, for $0 \leq i \leq k$,
7. $\Delta'((q_e', i)) = \{(q_e', i + 1)\}$, for $0 \leq i < k$,
8. $\delta'((q_e', k), \lambda, Z_0') = \{((q_e', k), \lambda)\}$.

The automaton A_2 initially makes a setup for the simulation (1), simulates ordinary transitions of A_1 (2), prepares the simulation of a pushdown reversal (3), simulates the pushdown reversal (4), completes the simulation of a pushdown reversal (5), prepares the final pushdown reversals (6), performs the final pushdown reversals (7), and halts by popping the bottom-of-pushdown symbol (8).

It remains to show the inclusion $\mathscr{L}_N(\text{DFPDA}_{=k}) \subseteq \mathscr{L}_N(\text{DFPDA}_{\leq k})$. The construction is similar to the corresponding one given in the proof of Theorem 8. The main difference is, that now the simulating automaton is prevented from popping the bottom-of-pushdown symbol whenever the number of performed pushdown reversals is not equal to k. $\square$

Corollary 13 $\mathscr{L}_N(\mathrm{DFPDA}_{\leq fin}) = \mathscr{L}_N(\mathrm{DFPDA}_{=fin})$.

Due to the equalities, again, in the sequel we can simplify the notation to $\mathscr{L}_N(\mathrm{DFPDA}_k)$ and $\mathscr{L}_N(\mathrm{DFPDA}_{fin})$. Now the question for the relationships between $\mathscr{L}_N(\mathrm{DFPDA}_k)$ and the other classes under consideration arises. The next result nicely extends the chain of strict inclusions known so far.

Theorem 14 *Let $k \geq 0$. Then $\mathscr{L}_N(\mathrm{DFPDA}_k) \subset \mathscr{L}(\mathrm{DFPDA}_{=k})$.*

Proof. Due to Theorem 12 it suffices to show $\mathscr{L}_N(\mathrm{DFPDA}_{=k}) \subset \mathscr{L}(\mathrm{DFPDA}_{=k})$. Both, for inclusion and for its strictness we can adapt the proofs for ordinary deterministic pushdown automata.

For inclusion a new bottom-of-pushdown symbol is provided in order to detect when the simulated automaton empties its pushdown. If and only if this occurs, the simulating automaton enters an accepting state.

For strictness of the inclusion it is easily seen that a language $L \in \mathscr{L}_N(\mathrm{DFPDA}_{=k})$ must have the prefix property. That is, no word in L is a proper prefix of another word in L. Since there exist deterministic context-free languages that do not have the prefix property, there exist languages which do not belong to $\mathscr{L}_N(\mathrm{DFPDA}_{=k})$, but which do belong to $\mathscr{L}(\mathrm{DFPDA}_{=k})$, for all $k \geq 0$. $\square$

Corollary 15 $\mathscr{L}_N(\mathrm{DFPDA}_{fin}) \subset \mathscr{L}(\mathrm{DFPDA}_{=fin})$.

So far we have shown the strict inclusions

$$\mathscr{L}_N(\mathrm{DFPDA}_k) \subset \mathscr{L}(\mathrm{DFPDA}_{=k}) \subset \mathscr{L}(\mathrm{DFPDA}_{\leq k}) \subset \mathscr{L}(\mathrm{NFPDA}_k),$$

for all $k \geq 1$, and correspondingly for "fin."

6 Flip-Pushdown Hierarchies

In this section we consider hierarchies on flip-pushdown automata induced by the number of pushdown reversals. Recently, it was shown shown in [10] that for nondeterministic flip-pushdown automata $k+1$ pushdown reversals are better than k. The proof of this result is based on the "flip-pushdown input-reversal" technique. The witness language for the hierarchy has shown to belong to $\mathscr{L}(\mathrm{DFPDA}_{=(k+1)})$ but not to $\mathscr{L}(\mathrm{NFPDA}_k)$. Actually, the language is also accepted by some deterministic flip-pushdown automaton by empty pushdown making $k+1$ pushdown reversals. For the convenience of the reader we recall the proof given in [10].

Theorem 16 *Let $k \geq 0$. Then $\mathscr{L}_N(\mathrm{DFPDA}_{k+1}) \setminus \mathscr{L}(\mathrm{NFPDA}_k) \neq \emptyset$.*

Proof. Define, for $k \geq 1$, the language

$$L_k = \{\, \texttt{\#}w_1\texttt{\$}w_1\texttt{\#}w_2\texttt{\$}w_2\texttt{\#}\cdots\texttt{\#}w_k\texttt{\$}w_k\texttt{\#} \mid w_i \in \{a,b\}^* \text{ for } 1 \leq i \leq k \,\}.$$

Obviously, language L_{k+1} is accepted by a deterministic flip-pushdown automaton by empty pushdown making exactly $k+1$ pushdown reversals. So $L_{k+1} \in \mathscr{L}(\mathrm{DFPDA}_{k+1})$.

Next we prove that $L_{k+1} \notin \mathscr{L}(\mathrm{NFPDA}_k)$. Assume to the contrary, language L_{k+1} is accepted by some flip-pushdown automaton A with exactly k pushdown reversals. Then applying the flip-pushdown input-reversal Theorem 3 exactly k times, results in a context-free language L. Now the idea is to pump an appropriate word from the context-free language and to undo the flip-pushdown input-reversals, in order to obtain a word that must be in L_{k+1}. If the pumping is done such that no input reversal boundaries in the word are pumped, then the flip-pushdown input-reversals can be undone. Therefore, we need the generalization of Ogden's lemma [1]. The principle idea of the proof is shown in Figure 1. Let n be the constant in the generalization of Ogden's

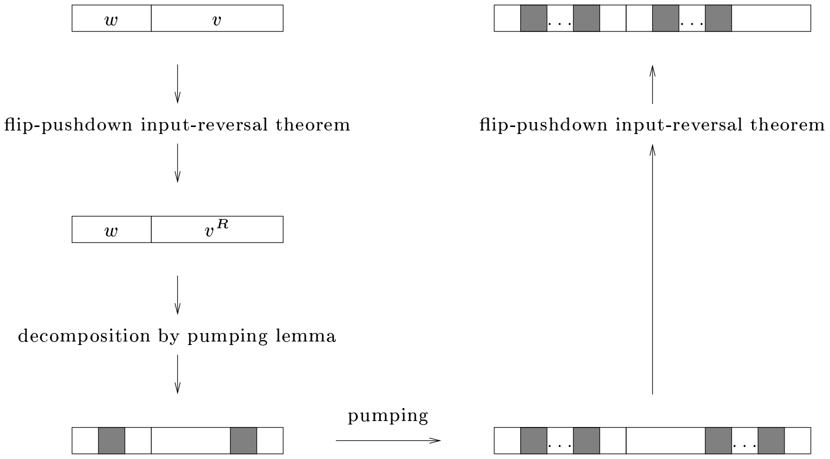

Figure 1: Principle idea of the proof.

lemma for L and $z = (\texttt{\#}a^{n^{2k+1}}b^{n^{2k+1}}\texttt{\$}a^{n^{2k+1}}b^{n^{2k+1}})^{k+1}\texttt{\#}$ be in L_{k+1}. Consider the word z when transformed into an instance z' of the context-free language L. When applying Theorem 3 to a word wv it becomes wv^R, then we mark the last position of w and the first position of v^R as excluded. Hence, after k applications of Theorem 3 word z' in L contains at most $2k$ excluded positions e. Moreover, since only k flip-pushdown input-reversals are allowed, and $k+1$ blocks $\texttt{\#}a^{n^{2k+1}}b^{n^{2k+1}}\texttt{\$}a^{n^{2k+1}}b^{n^{2k+1}}\texttt{\#}$ exist, due to the pigeon-hole principle there must be at least one block, which was not cut and (its remaining input) reversed. We pick one of these intact blocks in z' and mark all its positions as distinguished. Thus, there are $d = 4 \cdot n^{2k+1} + 2$ distinguished positions in z', with $d > n^{e+1}$.

Now assume that words u, v, w, x, and y satisfy the properties of the generalization of Ogden's lemma. First, we can easily see that if either v or x contains symbols $ or #, then we obtain a contradiction by considering word uv^2wx^2y, since every word in L (L_{k+1}, respectively) contains exactly $k+1$ symbols $ and exactly $k+2$ symbols #. Second, we know that because vx contains at least one distinguished position, word v or x lies completely within our chosen intact block $\#a^{n^{2k+1}}b^{n^{2k+1}}\$a^{n^{2k+1}}b^{n^{2k+1}}\#$ (excluding the symbols $ and #). Then we distinguish three cases:

1. Both words v and x are within the block under consideration. Then the number of excluded positions in vwx equals zero, and hence $|vwx| \leq n$. Then we obtain, that the block under consideration looses its "copy" form in the word $\tilde{z}' = uv^2wx^2y$, i.e., the block we are looking at is not of the form $\#w\$w\#$, for some w, anymore.
2. Word v is within the block under consideration, but x is not. Then the number of excluded positions in vwx is at most $2k$, and hence $|v| \leq n^{2k+1}$. Again, the block under consideration looses its form in the word $\tilde{z}' = uv^2wx^2y$.
3. Word v is not within the block under consideration, but x is. Then a similar reasoning as in the case above applies.

Since we know little about the context-free language L, we now transform our pumped string $\tilde{z}'$ back towards language L_{k+1}, according to Theorem 3. Now the advantage of the excluded positions comes into play. Since we have never pumped on excluded positions, the pushdown reversal move is still valid. Hence, word $\tilde{z}'$ leads us to a word $\tilde{z}$, where the original intact block considered so far is now not of the form $\#w\$w\#$, for some w anymore. Observe, that the application of Theorem 3 is done exactly in the reverse order as above. This means, that an input reversal appears only at excluded positions (or in-between two excluded ones). In particular, the block considered so far remains untouched during this process. Therefore, word $\tilde{z}$ is not a member of language L_{k+1}. This contradicts our assumption, and thus $L_{k+1} \notin \mathscr{L}(\mathrm{NFPDA}_k)$. □

So we have four hierarchies with separated levels and, furthermore, separated families on every level. In order to explore the relationships between any two families somewhere in the inclusion structure (cf. Figure 2), we have to answer the question how the number of pushdown reversals relates to determinism/nondeterminism, at most k/exactly k reversals, or acceptance by final state/empty pushdown. In principle, these answers have already been given. Either two families are related by a strict inclusion or they are incomparable. One direction of the following theorems is an immediate consequence of the proof of the hierarchy Theorem 16.

Theorem 17 *Let $\ell > k \geq 0$. The family $\mathscr{L}(\mathrm{NFPDA}_k)$ is incomparable with each of the families $\mathscr{L}(\mathrm{DFPDA}_{\leq\ell})$, $\mathscr{L}(\mathrm{DFPDA}_{=\ell})$, and $\mathscr{L}_N(\mathrm{DFPDA}_\ell)$.*

Proof. By the proof of Theorem 6 the language $\{ ww \mid w \in \{a,b\}^+ \}$ belongs to $\mathscr{L}(\mathrm{NFPDA}_1)$ but does not belong to $\mathscr{L}(\mathrm{DFPDA}_{\leq fin})$. This proves the assertion for $k \geq 1$. The case $k = 0$ is seen by the language $\{ ww^R \mid w \in \{a,b\}^+ \}$ which belongs to $\mathscr{L}(\mathrm{NFPDA}_0)$ but does not belong to $\mathscr{L}(\mathrm{DFPDA}_{\leq fin})$. The latter can be shown by the same argumentation used to show $\{ ww \mid w \in \{a,b\}^+ \} \notin \mathscr{L}(\mathrm{DFPDA}_{\leq fin})$. □

The next result holds for $k \geq 1$ only, since trivially the family $\mathscr{L}(\mathrm{DFPDA}_{\leq 0})$ equals $\mathscr{L}(\mathrm{DFPDA}_{=0})$.

Theorem 18 *Let $\ell > k \geq 1$. The family $\mathscr{L}(\mathrm{DFPDA}_{\leq k})$ is incomparable with each of the families $\mathscr{L}(\mathrm{DFPDA}_{=\ell})$ and $\mathscr{L}_N(\mathrm{DFPDA}_\ell)$.*

Proof. By the proof of Theorem 9 the language

$$\{ \mathtt{\#}w\mathtt{\#} \mid w \in \{a,b\}^* \} \cup \{ \mathtt{\#}w_0\mathtt{\#}w_1\mathtt{\$}w_1\mathtt{\#} \mid w_0, w_1 \in \{a,b\}^* \}$$

belongs to $\mathscr{L}(\mathrm{DFPDA}_{\leq 1})$ but does not belong to $\mathscr{L}(\mathrm{DFPDA}_{=fin})$. □

Theorem 19 *Let $\ell > k \geq 0$. The family $\mathscr{L}(\mathrm{DFPDA}_{=k})$ is incomparable with the family $\mathscr{L}_N(\mathrm{DFPDA}_\ell)$.*

Proof. As witness we may take any deterministic context-free language L that does not have the prefix property. Trivially, $L \in \mathscr{L}(\mathrm{DFPDA}_{=0})$ but $L \notin \mathscr{L}_N(\mathrm{DFPDA}_{fin})$. □

7 Conclusions

We have investigated (deterministic) flip-pushdown automata with a constant number of pushdown reversals, which were recently introduced by Sarkar [14]. The main interest was on deterministic computations. We distinguished deterministic automata, that accept with at most k pushdown reversals or with exactly k pushdown reversals, that accept by final state or by empty pushdown. For all models we showed a strict hierarchy induced by the number of pushdown reversals. Furthermore, we have separated deterministic from nondeterministic automata and have considered the relationships between the distinguished deterministic classes. The major technique is the “flip-pushdown input-reversal” technique developed in [10].

A natural property for further distinctions is whether ordinary λ-moves are allowed or not. Here we assume that a pushdown reversal never consumes an input symbol. A well-known language which separates deterministic context-free languages without λ-moves from deterministic context-free languages with λ-moves is $\{ a^m b^n x c^m \mid m,n \geq 1 \} \cup \{ a^m b^n y c^n \mid m,n \geq 1 \}$ as shown in [4]. Obviously, this language is accepted by some deterministic flip-pushdown automaton without λ-moves by empty pushdown making one pushdown reversal.

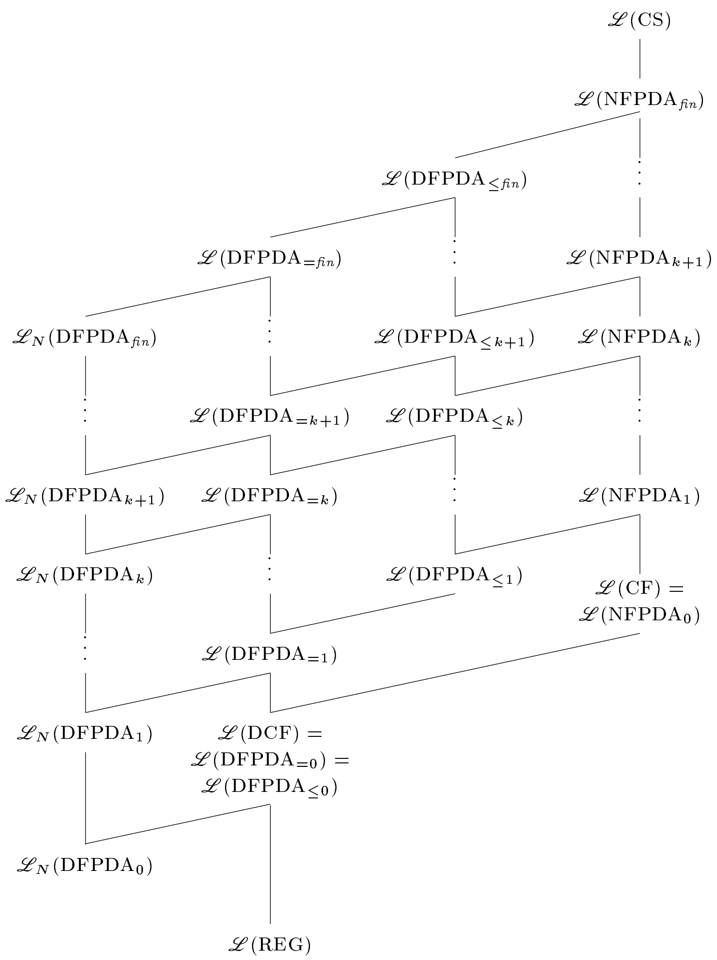

Figure 2: Inclusion structure.

Nevertheless, we can generalize the language to $L = \{\, a^\ell b^m c^n x d^\ell \mid \ell, m, n \geq 1 \,\} \cup \{\, a^\ell b^m c^n y d^m \mid \ell, m, n \geq 1 \,\} \cup \{\, a^\ell b^m c^n z d^n \mid \ell, m, n \geq 1 \,\}$. It is easy to see that L is accepted by an ordinary deterministic pushdown automaton by empty pushdown or by final state that makes some λ-moves. On the other hand, language L cannot be accepted by any deterministic flip-pushdown automaton not allowed to make λ-moves.

Conversely, the language L_k of the hierarchy Theorem 16 is accepted by some deterministic flip-pushdown automaton without λ-moves by empty pushdown making k pushdown reversals. But L_k is not accepted by any deterministic flip-pushdown automaton making strictly less than k pushdown reversals, even if λ-moves are allowed. This shows incomparability between each two of the families.

Nevertheless, several questions for flip-pushdown languages remain unanswered. We mention a few of them: (1) What is the power of λ-moves for nondeterministic flip-pushdown automata? Can they be removed without affecting the computational capacity? (2) What are the closure properties of the deterministic families? (3) Which properties are decidable? (4) What are the relationships between these language families and other well-known formal language classes? Especially, the latter question is of some interest, because not even the relationship between the family of flip-pushdown languages and some Lindenmayer families like, e.g. E0L or ET0L languages is known. For more on Lindenmayer languages we refer to Rozenberg and Salomaa [12]. We conjecture incomparability, but have no proof yet. Obviously, $\{ a^n b^n c^n \mid n \geq 0 \}$ is an E0L language which does not belong to $\mathscr{L}(\mathrm{NFPDA}_{\mathit{fin}})$ [10], but for the other way around we need a language with a semi-linear Parikh mapping which is not an ET0L language.

References

[1] Ch. Bader and A. Moura. A generalization of Ogden's lemma. *Journal of the ACM*, 29(2):404–407, 1982.

[2] A. W. Burks, D. W. Warren, and J. B. Wright. An analysis of a logical machine using parenthesis-free notation. *Mathematical Tables and Other Aids to Computation*, 8(46):53–57, April 1954.

[3] N. Chomsky. *Handbook of Mathematic Psychology*, volume 2, chapter Formal Properties of Grammars, pages 323–418. Wiley & Sons, New York, 1962.

[4] S. N. Cole. Deterministic pushdown store machines and real-time computation. *J. Assoc. Comput. Mach.*, 18:306–328, 1971.

[5] R. J. Evey. *The Theory and Applications of Pushdown Store Machines.* Ph.D thesis, Harvard University, Massachusetts, May 1963.

[6] S. Ginsburg. *Algebraic and Automata-Theoretic Properties of Formal Languages.* North-Holland, Amsterdam, 1975.

[7] S. Ginsburg, S. A. Greibach, and M. A. Harrison. One-way stack automata. *Journal of the ACM*, 14(2):389–418, April 1967.

[8] S. Ginsburg and E. H. Spanier. Finite-turn pushdown automata. *SIAM Journal on Computing*, 4(3):429–453, 1966.

[9] S. A. Greibach. An infinite hierarchy of context-free languages. *Journal of the ACM*, 16(1):91–106, January 1969.

[10] M. Holzer and M. Kutrib. *Flip-pushdown automata: $k+1$ pushdown reversals are better than k.* IFIG Research Report 0206, Institute of Informatics, University of Giessen, 2002. `http://www.ifig.de/staff/kutrib/papers.html`

[11] A. Newell and J. C. Shaw. Programming the logic theory machine. In *Proceedings of the 1957 Western Joint Computer Conference*, pages 230–240, Institute of Radio Engineers, New York, February 1957.

[12] G. Rozenberg and A. Salomaa. *The Mathematical Theory of L Systems*, volume 90 of *Pure and Applied Mathematics*. Academic Press, 1980.

[13] K. Samelson and F. L. Bauer. Sequential formula translation. *Communications of the ACM*, 3(2):76–83, February 1960.

[14] P. Sarkar. Pushdown automaton with the ability to flip its stack. Report TR01-081, Electronic Colloquium on Computational Complexity (ECCC), November 2001.