UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY,
FYZIKY A INFORMATIKY

ROZŠÍRENÉ ZÁSOBNÍKOVÉ AUTOMATY

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	9.2.1. informatika
Školiace pracovisko:	Katedra informatiky
Vedúci práce:	prof. RNDr. Pavol Ďuriš, PhD.

Bratislava 2011

Bc. Marek Košta

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Marek Košta
Študijný program: informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský

Názov: Rozšírené zásobníkové automaty

Cieľ: Hlavným cieľom práce je je skúmať výpočtovú silu jednosmerných zásobníkových automatov s možnosťou k-krát obrátiť zásibník v priebehu výpočtu (k-flipové zásobníkové automaty). Ďalšími cieľmi práce sú: sumarizovať výsledky dosiahnuté v uvedenej oblasti, osvojiť si dôkazové techniky používané v danej oblasti, preskúmať niektoré uzáverové vlastnosti jazykov akceptovateľných takýmito deterministickými automatmi, preskúmať možnosť eliminácie epsilon prechodov pre takéto nedeterministické automaty a pokúsiť sa vyriešiť niektoré problémy formulované v predošlých prácach venovaných danej problematike.

Vedúci: doc. RNDr. Pavol Ďuriš, CSc.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 29.10.2010

Dátum schválenia: 03.11.2010

prof. RNDr. Branislav Rovan, PhD.
garant študijného programu

.....
študent

.....
vedúci

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne s použitím citovaných zdrojov.

.....

Chcel by som sa poďakovať môjmu vedúcemu prof. RNDr. Pavlovi Ďurišovi, PhD. za ochotu, pomoc, cenné rady a pripomienky pri písaní tejto diplomovej práce. Veľká vďaka patrí tiež mojej priateľke Tonke a mojej rodine za ich neustálu podporu. Bez ich podpory by táto práca nevznikla.

Abstrakt

V práci pojednávame o modifikácií klasického modelu zásobníkového automatu, presnejšie o flipových zásobníkových automatoch. Venujeme sa deterministickému aj nedeterministickému modelu. Skúmame silu epsilon krokov v nedeterministickom modeli. V deterministickom modeli sa venujeme najmä uzáverovým vlastnostiam. Snažíme sa ponúknuť nové idey a výsledky. Hlavným výsledkom práce je vyriešenie otvoreného problému epsilon krokov v nedeterministickom modeli.

Kľúčové slová:
Zásobníkový automat, flipový zásobníkový automat, sila epsilon krokov, Greibachovej normálny tvar, uzáverové vlastnosti

Obsah

Kapitola 1

Úvod

Model klasického zásobníkového automatu je jedným z prvých výpočtových modelov ktorý bol podrobený intenzívnemu teoretickému výskumu. Ako je známe definuje triedu bezkontextových jazykov, ktoré sú jednou zo štyroch hlavných tried Chomského hierarchie. Popri ohromnom využití zásobníkových automatov a ich variantov v praktických prípadoch (akými su teória parsovania, prekladu a podobne), sa štúdium uberá aj teoretickými smermi. Skúmajú sa rôzne varianty, obmeny, zosilnenia i zoslabania pôvodného modelu.

Práve pre dosť veľký teoretický záujem a (podľa nášho názoru) aj preto, že uvedený variant je „úplne intuitívny“ je istým paradoxom, že až v roku 2001 vychádza článok [11], ktorý definuje model flipového zásobníkového automatu. Uvádza i tvrdenia priamo vyplývajúce z tejto definície. Väčšie svetlo do problematiky vnášajú až následné články [5, 6]. V našej práci z nich významne čerpáme aj my. Hlavným objektom záujmu našej práce je ďalšie skúmanie tohto konkrétneho rozšírenia (variantu) klasického zásobníkového automatu.

Ako hlavné ciele práce sme si stanovili nasledovné. Podať istý prehľad známych výsledkov, poprípade ich vhodne okomentovať. Omnoho dôležitejším

cieľom ale je vyriešenie niektorých otvorených problémov v oblasti, ponúknutie nových ideí, resp. využitie a prispôsobenie už existujúcich za účelom čo najväčšieho rozšírenia poznatkov v oblasti.

Práca je rozčlenená nasledovne. V druhej kapitole uvádzame definície uvažovaného modelu. Uvedené sú aj definície štandardného zásobníkového automatu. V tretej kapitole spomíname hlavné známe techniky, tvrdenia a fakty. Najdôležitejšia pre nás bude technika Flip-Pushdown Input-Reversal. V štvrtej kapitole je uvedený a do podrobna rozvedený hlavný výsledok práce. Podávame najprv ideu dôkazu, potom isté formálne pojednanie. Na konci kapitoly je zhrnutie našej dôkazovej techniky a postupu pri dôkaze. V piatej kapitole sa venujeme uzáverovým vlastnostiam deterministických flipových zásobníkových automatov. V záverečnej šiestej kapitole zhrnieme výsledky práce. Podstatnejším obsahom tohto záveru je ale formulovanie zostávajúcich otvorených problémov v skúmanej oblasti.

Kapitola 2

Uvažovaný model

V tejto kapitole popíšeme modely zásobníkových automatov s ktorými budeme ďalej pracovať. Slúži ako istý formálny úvod do problematiky flipových zásobníkových automatov. Uvádzame tu definície a označenia, upozorňujeme na terminológiu a uvádzame fakty vyplývajúce priamo z týchto definícií.

2.1 Klasické zásobníkové automaty

Uvedieme definíciu a základné vlastnosti klasického modelu zásobníkových automatov. Dôvodom je, aby bolo vidieť z akého základu vychádza rozšírenie tohto klasického modelu ktorý budeme študovať. Téma zásobníkových automatov je veľmi rozsiahla, preto čitateľa odkazujeme na [10, 7, 8].

Pre porovnanie s definíciou flip-zásobníkového automatu uvedieme formálnu definíciu.

Definícia 2.1. Nedeterministický zásobníkový automat je sedmica $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$. Pričom K je konečná množina stavov automatu A, Σ je vstupná abeceda, Γ je zásobníková abeceda, δ prechodová funkcia, q_0 počiatočný stav, Z_0 počiatočný zásobníkový symbol a $F \subseteq K$ je množina akceptačných stavov. Funkcia δ: $K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \mapsto 2^{K \times \Gamma^*}_{fin}$ je prechodovou

funkciou automatu A.

Poznámka 2.2. Označením 2^X_{fin} sa myslí množina všetkých konečných podmnožín množiny X. Preto δ funkcia zobrazuje trojicu (q, a, Z) na konečnú množinu dvojíc (p, w).

Ďalšie štandardné definície ako konfigurácia, krok výpočtu a jazyk neuvádzame, sú priamočiare a čitateľ ich môže nájsť v [10]. Upozorňujeme ale, že ich budeme ďalej používať.

Definícia 2.3. Jazykom akceptovaným prázdnou pamäťou automatom A označujeme množinu:

$$N_A = \{w \mid (\exists q \in K)(q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon)\}$$

Triedu jazykov rozpoznávanú zásobníkovými automatmi budeme označovať, ako je zvykom, $\mathscr{L}$(CFL) alebo jednoducho CFL. Niekedy tiež použijeme označenie $\mathscr{L}$(PDA).

Deterministický zásobníkový automat dostaneme, ak pridáme dve obmedzenia na δ-funkciu zásobníkového automatu. Sú to:

1) ak $\delta(q, a, Z) \neq \emptyset$ pre nejaké $a \in \Sigma$ potom $\delta(q, \varepsilon, Z) = \emptyset$,

2) pre všetky $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $Z \in \Gamma$ obsahuje množina $\delta(q, a, Z)$ najviac jeden prvok.

Triedu jazykov rozpoznávanú deterministickými zásobníkovými automatmi označujeme $\mathscr{L}$(DCFL), DCFL alebo aj $\mathscr{L}$(DPDA). Vieme, že $\mathscr{L}$(DCFL) $\subset$ $\mathscr{L}$(CFL), [8].

Teória zásobníkových automatov je pomerne stará a prebádaná disciplína. Preto v ďalšom predpokladáme, že čitateľ má aspoň základný prehľad v tejto oblasti. V opačnom prípade odkazujeme čitateľa na [8] a [10]. Zaujímavou je aj prehľadová kapitola [7]. Potrebné výsledky z tejto oblasti budeme v ďalšom explicitne uvádzať, len ak to bude potrebné.

2.2 Nedeterministické flipové zásobníkové automaty

Hneď na začiatku upozorňujeme na terminológiu: vždy keď hovoríme o flipových (zásobníkových) automatoch máme na mysli automaty so schopnosťou obracať zásobník a naopak (a to či sa jedná o deterministický alebo o nedeterministický bude vždy jasné z kontextu). Podobne majú pre nás rovnaký význam slová „obrat zásobníka“ a „flip“.

Najprv formálne definujeme nedeterministický flipový automat

Definícia 2.4. Nedeterministický zásobníkový automat so schopnosťou obracať zásobník (flipový zásobníkový automat) je osmica $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$, kde Q je konečná množina stavov, Σ je vstupná abeceda, Γ je pracovná (zásobníková) abeceda, δ je zobrazenie z $Q \times (\Sigma \cup \{\varepsilon\} \times \Gamma$ do konečných podmnožín množiny $Q \times \Gamma^*$ nazývaná aj prechodová funkcia, Δ je zobrazenie z Q do 2^Q nazývaná aj veľká prechodová funkcia, $q_0 \in Q$ je počiatočný stav automatu, $Z_0 \in \Gamma$ je začiatočný zásobníkový symbol, ktorý je na začiatku výpočtu v zásobníku, voláme ho aj spodok zásobníka a nakoniec $F \subseteq Q$ je množina akceptačných (finálnych) stavov.

Definícia 2.5. Konfiguráciou flip-zásobníkového automatu $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$ nazývame usporiadanú trojicu (q, w, γ), kde $q \in Q$, $w \in \Sigma^*$ a $\gamma \in \Gamma^*$. Hovorí nám, že A je v stave q, zvyšok neprečítaného vstupu je w a obsah zásobníka je γ.

Definícia 2.6. Krokom výpočtu automatu A nazývame reláciu $\vdash_A$ (ak je z kontextu jasné o aký automat sa jedná budeme uvádzať iba $\vdash$) na konfiguráciách automatu A definovanú nasledovne. Ak $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $w \in \Sigma^*$, γ, $\beta \in \Gamma^*$, $Z \in \Gamma$ a súčasne $(p, \beta) \in \delta(q, a, Z)$ potom $(q, aw, \gamma Z) \vdash_A (p, w, \gamma\beta)$. Ak $p \in \Delta(q)$ tak $(q, w, Z_0\gamma) \vdash_A (p, w, Z_0\gamma^R)$. Pričom žiadne iné konfigurácie v relácií $\vdash_A$ nie sú.

Poznámka 2.7. Z tejto definície vidíme, že ak $\alpha = a_1 \dots a_n$ je slovo v zásobníku, tak a_1 je spodný symbol a a_n je vrchný symbol. Podobne pre vstupné slovo w, jeho prvý symbol je nasledujúci symbol na vstupe. Nasledujúca konfigurácia je vybratá nedeterministicky spomedzi všetkých, ktoré sú s ňou v relácií $\vdash_A$, pričom nastane práve jedna možnosť – krok výpočtu je buď štandardný alebo flipový. Pri štandardnom kroku výpočtu sa jedná o krok výpočtu ako v štandardnom zásobníkovom automate. Pri flipovom kroku výpočtu sa jedná o obrat (flip) zásobníka. V ďalšom pomerne často použijeme rozlišovanie týchto dvoch možností kroku výpočtu automatu A. Pri prvej možnosti hovoríme o obyčajných, štandardných, bezflipových krokoch výpočtu, alebo δ-krokoch. Pri druhej možnosti hovoríme o flipových krokoch, krokoch pri ktorých sa obracia zásobník, alebo o Δ-krokoch. Podobne budeme túto terminológiu uplatňovať na celé úseky výpočtov.

Definícia 2.8. Jazykom rozpoznávaným (akceptovaným) nedeterministickým flipovým zásobníkovým automatom A rozumieme množinu $L(A) = \{w \in \Sigma^* \mid (\exists q \in F)\ (\exists \gamma \in \Gamma^*)\ (q_0, w, Z_0) \vdash_A^* (q, \varepsilon, \gamma)\}$. Symbol $\vdash_A^*$ znamená reflexívno-tranzitívny uzáver relácie $\vdash_A$.

Takáto definícia nedáva žiadne obmedzenie na počet flipov zásobníka. Preto čitateľovi s istými znalosťami z formálnych jazykov a automatov pri troche zamyslenia isto napadne, že takýto model s neobmedzeným počtom flipov zásobníka je výpočtovou silou ekvivalentný turingovým strojom. Toto tvrdenie by sme pri označení ktoré budeme používať zapísali $\mathscr{L}(\text{NFPDA}) = \mathscr{L}(\text{NFPDA}(\infty)) = \mathscr{L}(\text{RE})$. My sa tomuto dôkazu nebudeme ďalej venovať, podrobnosti sa dajú nájsť v [12, 11].

Ak ale nepovolíme ľubovoľne veľa flipov, tak situácia začína byť zaujímavá v tom zmysle, že počet flipov sa stáva prirodzeným výpočtovým prostriedkom. Presnejšie, zápisom $\mathscr{L}(\text{NFPDA}(\leq k))$ označíme triedu jazykov, ktoré sa dajú rozpoznať flipovým zásobníkovým automatom za použitia najviac

k flipov. Formálnejšie by sme to mohli formulovať tak, že najviac k krokov výpočtu môže byť druhého druhu v zmysle horeuvedenej definície, resp. v každom akceptačnom výpočte sa vyskytuje najviac k Δ-krokov. Analogicky $\mathscr{L}(\mathrm{NFPDA}(= k))$ je trieda jazykov rozpoznávaných za použitia práve k obratov zásobníka. Ak položíme k rovné nule dostávame práve bezkontextové jazyky. Ďalej je zaujímavou triedou jazykov trieda $\mathscr{L}(\mathrm{NFPDA}(\mathrm{fin})) = \bigcup_{k=0}^{\infty} \mathscr{L}(\mathrm{NFPDA}(\leq k))$.

Tiež, ako sme zvyknutí definujeme akceptáciu prázdnou pamäťou.

Definícia 2.9. Pre flip-zásobníkový automat $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, \emptyset)$ definujeme jazyk

$$N_k(A) = \{w \mid (\exists q)(q_0, w, Z_0) \vdash_A^* (q, \varepsilon, \varepsilon) \text{ pričom } A \text{ použije práve } k \text{ obratov}\}$$

Uveďme pre úplnosť nejaký príklad, aby bola zrejmá naša notácia.

Príklad 2.10. Ukážeme, ako jednoducho sa dá týmto flip-zásobníkovým automatom rozpoznávať jazyk $L = \{ww \mid w \in \Sigma^*\}$. Nech $A = (\{q_0, q_1\}, \{a, b\}, \{A, B, Z_0\}, \delta, \Delta, q_0, Z_0, \emptyset)$. Prechodovú funkciu definujeme nasledovne:

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. $\delta(q_1, a, A) = \{(q_1, \varepsilon)\}$
8. $\delta(q_1, b, B) = \{(q_1, \varepsilon)\}$
9. $\delta(q_1, \varepsilon, Z_0) = \{(q_1, \varepsilon)\}$

Δ-funkcia je definovaná jednoducho: $\Delta(q_0) = \{q_1\}$. Množina akceptačných stavov je prázdna z dôvodu akceptácie prázdnou pamäťou. Nie je zložité nahliadnuť že $N_1(A) = L$. Uveďme ale príklad výpočtu na nejakom slove:

$$\begin{aligned}
(q_0, abaababaab, Z_0) &\vdash (q_0, baababaab, Z_0A) \vdash^4 (q_0, abaab, Z_0ABAAB) \\
(q_0, abaab, Z_0ABAAB) &\vdash (q_1, abaab, Z_0BAABA) \\
(q_1, abaab, Z_0BAABA) &\vdash (q_1, baab, Z_0BAAB) \vdash^4 (q_1, \varepsilon, Z_0) \vdash (q_1, \varepsilon, \varepsilon)
\end{aligned}$$

Veríme, že čitateľ si bez problémov uvedomí v ktorom momente je aplikovaný ktorý riadok δ-funkcie. Podstatný je ale druhý riadok v uvedenom výpočte automatu A. Tu sa totiž udial flip zásobníka.

2.3 Deterministické flipové zásobníkové automaty

Analogicky ako pri štandardnom zásobníkovom automate definujeme deterministický variant pridaním obmedzení. Podstatou je, že automat má mať pre každú možnú konfiguráciu najviac jednu nasledujúcu konfiguráciu. Formálnejšie, ak $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$ je nedeterministický flipový automat a ak navyše platia nasledujúce podmienky

1) $(\forall a \in \Sigma \cup \{\varepsilon\})(\forall q \in Q)(\forall Z \in \Gamma)$ obsahuje $\delta(q, a, Z)$ najviac jeden prvok,

2) $(\forall a \in \Sigma)(\forall q \in Q)(\forall Z \in \Gamma)$ ak $\delta(q, \varepsilon, Z) \neq \emptyset$, potom $\delta(q, a, Z) = \emptyset$,

3) $(\forall q \in Q)$ obsahuje množina $\Delta(q)$ najviac jeden prvok,

4) $(\forall q \in Q)(\forall a \in \Sigma \cup \{\varepsilon\})(\forall Z \in \Gamma)$ ak $\Delta(q) \neq \emptyset$, potom $\delta(q, a, Z) = \emptyset$,

nazývame A deterministickým flipovým zásobníkovým automatom, označujeme DFPDA. Prvá a druhá podmienka sú rovnaké, ako pre obyčajné deterministické zásobníkové automaty, tretia a štvrtá zavádzajú determinizmus do možností obratov zásobníka.

Úplne analogicky sa zavedie aj akceptácia prázdnou pamäťou. Tu ale zavedieme označenie (lebo módy akceptácie majú rôznu popisnú silu, viď. ďalej v texte). $\mathscr{L}(\mathrm{DFPDA}(N_{=k}))$ bude trieda jazykov rozpoznávaná deterministickými flipovými zásobníkovými automatmi prázdnou pamäťou, pričom na každom akceptovanom slove použije *presne* k flipov zásobníka. Trieda

$\mathscr{L}(\text{NFPDA}(N_{\leq k}))$ je podobne ako v predošlom, avšak daný automat použije na každom akceptovanom slove *najviac* k flipov zásobníka.

Podobne definujeme triedy jazykov $\mathscr{L}(\text{DFPDA}) = \mathscr{L}(\text{DFPDA}(\infty))$, respektíve $\mathscr{L}(\text{DFPDA}(= k))$, $\mathscr{L}(\text{DFPDA}(\leq k))$ alebo $\mathscr{L}(\text{DFPDA}(\text{fin})) = \bigcup_{k=0}^{\infty} \mathscr{L}(\text{DFPDA}(\leq k))$.

2.4 Jednoduché tvrdenia

V tejto podkapitole uvádzame fakty, ktoré priamo vyplývajú z definícií. Formálne dôkazy sa dajú v každom prípade jednoducho doplniť, preto ich neuvádzame, spravidla načrtneme myšlienku.

Tvrdenie 2.11. *Ku každému nedeterministickému flipovému automatu A, ktorý na každom vstupe spraví najviac k flipov, vieme skonštruovať taký flipový automat B, že spraví na každom vstupe presne k flipov a $L(A) = L(B)$. Preto $\mathscr{L}(\text{NFPDA}(= k)) = \mathscr{L}(\text{NFPDA}(\leq k))$.*

Dôkaz. Automat B bude iba simulovať A a počítať počet flipov (a pamätať si samozrejme stav). Zapamätanie počtu flipov je realizovateľné, lebo k je konštanta. Ak bude simulovaný automat A v akceptačnom stave, tak potom B bude mať nedeterministický prechod na ε do nového špeciálneho akceptačného stavu, v ktorom dorobí zvyšný počet flipov a až potom akceptuje. □

Tvrdenie 2.12. *Ku každému nedeterministickému k-flipovému automatu A, ktorý akceptuje stavom jazyk $L(A)$, vieme zostrojiť nedeterministický k-flipový automat B, ktorý akceptuje prázdnou pamäťou a $N_k(B) = L(A)$ a naopak.*

Dôkaz. Princíp je úplne totožný s klasickými zásobníkovými automatmi. Ak máme daný A akceptujúci stavom, tak B ho bude iba simulovať a keď je

A v akceptačnom stave, tak bude môcť prejsť do stavu, v ktorom vyprázdni zásobník. Naopak, ak máme B akceptujúci prázdnou pamäťou, tak ho vieme simulovať tak, že simulujúci automat C iba na začiatku pridá na zásobník odrážku a keď sa na ňu opäť vráti, tak na ε prejde do akceptačného stavu.

□

Takže pri nedeterminizme máme podobnú robustnosť, ako pri klasických zásobníkových automatoch. Pri klasických totiž mód akceptácie (prázdnou pamäťou, či stavom) nemení výpočtovú silu modelu. Pri modeli nedeterministických flipových zásobníkových automatov je to analogické. Navyše nezáleží na tom, či automat spraví presne k obratov, alebo najviac k obratov. V kapitole 3 uvidíme, že v deterministickom variante (podobne ako pri klasických zásobníkových automatoch) obdobné tvrdenia neplatia.

Okamžite z definície tiež máme.

Tvrdenie 2.13 ([11])**.**

$$\begin{aligned}\mathscr{L}(\mathrm{CFL}) = \mathscr{L}(\mathrm{NFPDA}(\leq 0)) &\subseteq \mathscr{L}(\mathrm{NFPDA}(\leq 1)) \dots \\ &\subseteq \mathscr{L}(\mathrm{NFPDA}(\leq \mathrm{k})) \cdots \subseteq \mathscr{L}(\mathrm{NFPDA}(\mathrm{fin}))\end{aligned}$$

Omnoho zaujímavejšie je ale to, že práve táto hierarchia už pri zavedení modelu viedla k otvorenému problému, čí sú dané inklúzie ostré. Ako sa poukazuje v [11], medzi $\mathscr{L}(\mathrm{CFL})$ a $\mathscr{L}(\mathrm{NFPDA}(\leq 1))$ je ostrá inklúzia. Toto ihneď vyplýva aj z príkladu 2.10. V ďalšom uvedieme potvrdenie tejto domnienky a načrtneme aj techniku, ktorá bola použitá pri dôkaze.

V tejto kapitole sme formálne definovali model s ktorým budeme pracovať. Uviedli sme tiež niektoré tvrdenia, najmä o módoch akceptácie. Upozorňujeme hneď na začiatku na silnú analógiu s klasickými modelmi zásobníkových automatov: nedeterministické módy akceptácie sú ekvivalentne silné, deterministické nie. Ďalšou analógiou je to, že nedeterministický model je silnejší, ako deterministický (pre k flipov), [6].

Kapitola 3

Známe fakty

V tejto kapitole uvedieme už známe fakty a dokázané tvrdenia pre flipové zásobníkové automaty. Budeme sa snažiť podať ucelený obraz týchto výsledkov, ale hlavnú pozornosť budeme venovať práve tým, z ktorých budeme neskôr vychádzať a používať ich pri vlastných tvrdeniach. Technickejšie dôkazy nebudeme uvádzať. Pôjde nám skôr o to, aby sme podali ozrejmujúci komentár k daným tvrdeniam. Hlavným zdrojom sú články [5] a [6].

3.1 Flip-Pushdown Input-Reversal technika

Keďže dôkaz nasledujúceho tvrdenia je dosť technický, neuvádzame ho podrobne. Naznačíme iba jeho hlavné idey. Čitateľ ho môže nájsť v [5] kompletný. Vo výskumnej správe [4] možno nájsť ešte podrobnejšií výklad uvedeného dôkazu. Veľmi podobný výsledok je aj v diplomovej práci [12].

Veta 3.1 ([5])**.** *Nech k je prirodzené číslo (prípadne nula). Jazyk L je akceptovaný flip-zásobníkovým automatom $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$ prázdnym zásobníkom s $k+1$ flipmi, teda $L = N_{k+1}(A)$* **vtedy a len vtedy**

ak je jazyk

$$L_R = \{wv^R \mid (q_0, w, Z_0) \vdash_A^* (q_1, \varepsilon, Z_0\gamma) \text{ s k obratmi, } q_2 \in \Delta(q_1) \text{ a } (q_2, v, Z_0\gamma^R) \vdash_A^* (q_3, \varepsilon, \varepsilon) \text{ bez žiadneho obratu } \}$$

akceptovaný flip-zásobníkovým automatom B prázdnou pamäťou s k flipmi, to jest $L_R = N_k(B)$. Toto tvrdenie platí aj keď uvažujeme akceptáciu stavom.

Náčrt dôkazu. Hlavnou myšlienkou, ako už aj názov napovedá, je akási výmena obratu zásobníka za reverz konca akceptovaného slova. Majme výpočet A na slove wv, ako v znení vety. Potrebujeme automat B, ktorý bude až po posledný flip simulovať automat A. Namiesto toho, aby B odsimuloval aj posledný flip automatu A, chceme ale docieliť, aby z konfigurácie $(q_1, v^R, Z_0\gamma)$ akceptoval prázdnou pamäťou (bez obratu zásobníka) práve vtedy, keď $q_2 \in \Delta(q_2)$ a z konfigurácie $(q_2, v, Z_0\gamma^R)$ je A schopný prečítať v a vymazať zásobník, t. j. prejsť do konfigurácie $(q_3, \varepsilon, \varepsilon)$.

Podstatou je spätná simulácia automatu A. Automat B v momente, kedy by mal automat A urobiť posledný obrat ho neurobí, ale prejde do tohto simulačného módu. Očakáva, že má na vstupe slovo v^R s vyššie uvedenými vlastnosťami a toto ide overiť. Uloží na zásobník symbol Z_0 (lebo toto bol posledný symbol zmazaný zo zásobníka automatom A pri výpočte na v). Ďalej musí nedeterministicky simulovať pospiatky kroky automatu A na slove v^R. Toto sa docieli zámenou práce so zásobníkom a opačnou zmenou stavu. To jest, ak automat A niečo do zásobníka pridal, tak B toto odoberie a naopak. Analogicky, ak automat A prešiel zo stavu p_1 do stavu p_2, tak automat B bude simulovať opačný prechod, t. j. zo stavu p_2 do p_1. Práca so zásobníkom sa deje nad slovom γ.

Kebyže je $\gamma = \varepsilon$ tak by sa predošlý popis spätnej simulácie dal prakticky priamočiaro použiť. Čo ale robí komplikácie je práve to, že pri začatí tejto simulácie je už nejaké slovo na zásobníku. Preto musí B počas spätnej simulácie správne vymazávať zo zásobníka toto slovo (chceme aby B

tiež akceptoval prázdnou pamäťou). Toto robí nedeterministickým uhádnutím momentu (pre každý jeden symbol zrejme inokedy), kedy ho automat A poslednýkrát použil a v tomto momente ho zmaže. Tu je kriticky podstatné to, že spomenuté dva úseky na seba správne priliehajú vo význačných momentoch. V týchto momentoch je na zásobníku zvyšok slova γ a nad týmto jeden symbol z horného úseku, používaného na simuláciu. Preto je príhodné vymazanie realizovateľné. Mimo týchto význačných momentov vo výpočte môže samozrejme horný úsek na zásobníku rásť.

Táto idea je veľmi podobná konštrukcií v dôkaze uzavretosti bezkontextových jazykov na reverz cez zásobníkové automaty, ale ako sme spomenuli problém tu robí to, že $\gamma \neq \varepsilon$ a preto musíme kvôli kontrole zabezpečiť vhodné vymazávanie tohto slova γ.

□

Veta 3.1 je jednou z hlavných charakterizácií výpočtov daného flipového zásobníkového automatu A. Podstatou je, že jej dôkaz je **konštruktívny**, teda dáva algoritmický návod na to, ako k automatu A zostrojiť B.

Preto, ak je daný automat A_k a vieme, že na každom vstupnom slove použije práve k obratov zásobníka, existuje algoritmus, ktorý vie k tomuto automatu vyrobiť korešpondujúci automat A_{k-1} podľa vety 3.1, to jest taký, že jazyk ním rozpozávaný je jazyk rozpoznávaný automatom A_k len „s otočenými koncami slov“. Takto vieme ďalej konštruovať automaty $A_{k-2}, \ldots, A_0$, kde A_0 nepoužíva flipy, takže $L(A_0)$ je obyčajný bezkontextový jazyk, priamo z definície.

Poznámka 3.2. Preto vieme k danému k-flipovému zásobníkovému jazyku $L(A_k)$ priradiť nejaký bezkontextový jazyk $L(A_0)$, ktorý ho v istom zmysle charakterizuje. Touto charakterizáciou máme na mysli to, že slovo w je z jazyka $L(A_k)$ práve vtedy, ak jeho vhodným prevracaním koncov vieme dostať slovo z (bezkontextového) jazyka $L(A_0)$. Formálnejšie, slovo $w = v_0\ v_1\ v_2\ \ldots\ v_{k-1}\ v_k \in L(A_k)$ práve vtedy, keď slovo $v_0 v_1 \ldots v_{k-1} v_k^R \in L(A_{k-1})$. Podslovo

v_k je „koncové“, teda také, ktoré automat A_k prečítal po poslednom flipe (je to slovo v, ak sa na to pozeráme cez označenie vo vete 3.1). A takto ďalej pokračujeme postupným otáčaním, až dostaneme slovo podľa parity čísla k. Ak $k = 2m$, tak je tvaru $v_0 v_2 v_4 \ldots v_{2m} v_{2m-1}^R v_{2m-3}^R \ldots v_1^R$. Ak $k = 2m+1$, tak vyzerá nasledovne: $v_0 v_2 v_4 \ldots v_{2m} v_{2m+1}^R v_{2m-1}^R \ldots v_1^R$.

3.2 Využitie techniky Flip-Pushdown Input-Reversal

Na význam uvedenej techniky a na jej dôležitosť poukazuje to, že jej autori ju používajú na odvodenie prakticky všetkých známych zaujímavých výsledkov. Tu niektoré z týchto výsledkov uvádzame a komentujeme.

Najdôležitejšími výsledkami sú separácia hierarchie a porovnanie sily determinizmu a nedeterminizmu. Dôkazy sú pomerne technické. V ďalšom ich ideu nebudeme nijako potrebovať a preto ich neuvádzame.

Veta 3.3 ([5])**.** *Nech*

$$L_k = \{\#w_1\#w_1\#w_2\#w_2\#\ldots\#w_k\#w_k\# \mid w_i \in \{a,b\}^*, i \in \{1,\ldots,k\}\}$$

Pre $k \geq 0$ sa jazyk L_{k+1} dá akceptovať (deterministickým) flipovým automatom za použitia $k+1$ flipov, ale nie za použitia k flipov. Preto $L_{k+1} \in \mathscr{L}(\text{NFPDA}(= k+1)) - \mathscr{L}(\text{NFPDA}(= k))$. A preto pre triedy jazykov platí

$$\begin{aligned} \mathscr{L}(\text{NFPDA}(\leq \text{k})) &\subset \mathscr{L}(\text{NFPDA}(\leq \text{k}+1)) \\ \mathscr{L}(\text{DFPDA}(\leq \text{k})) &\subset \mathscr{L}(\text{DFPDA}(\leq \text{k}+1)) \end{aligned}$$

Veta 3.4 ([6])**.** *Nech $k \geq 0$. Potom $\mathscr{L}(\text{DFPDA}(\leq k)) \subset \mathscr{L}(\text{NFPDA}(\leq k))$.*

Nasledujúce tvrdenie nám poskytuje konkrétny príklad jazyka, ktorý nevieme rozpoznať na ľubovoľne veľký ale konečný počet flipov. Tento jazyk

je veľmi zaujímavý a dôležitý, lebo mnohé zaujímavé vlastnosti ktoré flipové automaty nemajú alebo fakty, ktoré o nich neplatia, sa dokazujú (resp. vyvracajú) práve sporom ,ktorý bude práve s týmto tvrdením, teda že by sa nám podarilo akceptovať daný jazyk pre nejaké k. Aj my v ďalšom túto stratégiu použijeme (pozri kapitolu 5). Preto uvedieme aj dôkaz tejto vety. Poznamenajme iba, že idey na podobnom princípe (založené na variantoch Ogdenovej lemy [1]), aj keď zložitejšom, sa používajú aj v dôkazoch viet 3.3 a 3.4.

Veta 3.5 ([5])**.** *Jazyk* $L = \{a^n b^n c^n \mid n \geq 1\} \notin \mathscr{L}(\text{NFPDA}(\leq k))$ pre ľubovoľné $k \geq 0$.

Dôkaz. Sporom. Nech existuje k také, že $L \in \mathscr{L}(\text{NFPDA}(\leq k))$. Opäť sa ukáže dôležitosť vety 3.1. Aplikujme ju teda na jazyk L k-krát. Ako sme sa už zmieňovali, dostaname bezkontextový jazyk. Keďže sa otáčenie deje na koncoch vstupných slov, budú mať slová v tom jazyku tvar $w = a^{n_1} b^{m_1} c^n b^{m_2} a^{n_2}$, pričom musí platiť, že $n_1 + n_2 = m_1 + m_2 = n$. K ozrejmeniu stačí aplikovať myšlienky poznámky 3.2. O tomtom jazyku sa dá ukázať, že nie je bezkontextový, napríklad pomocou Ogdenovej lemy a jej variantov (pozri napríklad [1] a [8]). Týmto sporom sme ukázali, že $L \notin \mathscr{L}(\text{NFPDA}(\text{fin}))$.

□

Veta 3.6 ([5])**.** *Jazyk rozpoznávaný flip-zásobníkovým automatom* A_k *s* k *flipmi nad abecedou* $\Sigma = \{a\}$ *je regulárny.*

Dôkaz. Treba si všimnúť, že $L(A_k) = L(A_{k-1}) = \cdots = L(A_0)$, lebo pracujeme nad jednoznakovou abecedou, práve kvôli vete 3.1 a predošlému komentáru. Z Parikhovej vety [9] vieme, že bezkontextový jazyk nad jednoznakovou abecedou je regulárny. Preto je aj $L(A_k)$ regulárny.

□

Ďalším zaujímavým dôsledkom vety 3.1 je, že platí inklúzia $\mathscr{L}(\text{NFPDA}(= k)) \subseteq \mathscr{L}(\text{CS})$. Toto vidno z toho, že lineárne ohraničený automat nedeterministicky uhádne k otočení koncov vstupného slova a následne overí, či takto

vytvorené slovo patrí do príslušného bezkontextového jazyka. Práve veta 3.1 zaručuje správnosť tohto postupu.

V skutočnosti je ale práve spomínaná inklúzia ostrá. Ak totiž vezmeme jazyk $L = \{a^{n^2}\}$, $L \notin \mathscr{L}(\mathrm{CF})$, $L \in \mathscr{L}(\mathrm{CS})$. Kebyže je tento jazyk rozpoznávaný k-flipovým automatom A pre nejaké k, bol by regulárnym a to je spor s vetou 3.6. Spomínané fakty môžeme zhrnúť.

Veta 3.7 ([5])**.** *Nech $k \in N$, $k > 0$. Potom pre triedy jazykov platí:*

$$\mathscr{L}(\mathrm{CFL}) \subset \mathscr{L}(\mathrm{NFPDA}(\leq \mathrm{k})) \subset \mathscr{L}(\mathrm{CS})$$

Na záver uveďme tvrdenia, pomocou ktorých sa vysporiadame s otázkou módov akceptácie v deterministickom variante. Už v kapitole 2 sme spomenuli, že oproti nedeterministickému modelu tu isté problémy sú. Opäť pre ilustráciu použitých techník uvádzame aj dôkaz jedného z nasledujúcich výsledkov.

Veta 3.8 ([6])**.** *Pre deterministické flipové automaty je akceptácia prázdnou pamäťou slabšia ako akceptácia stavom. Presnejšie: pre ľubovoľné k platí, že $\mathscr{L}(\mathrm{DFPDA}(N_{=k})) \subset \mathscr{L}(\mathrm{DFPDA}(= k))$.*

Veta 3.9 ([6])**.** *Nech $k \geq 1$. Potom $\mathscr{L}(\mathrm{DFPDA}(= k)) \subset \mathscr{L}(\mathrm{DFPDA}(\leq k))$.*

Dôkaz. Inklúzia je zrejmá z definície. Sporom ukážeme, že sa jedná o ostrú inklúziu. Majme dané $L_1 = \#\{a, b\}^*\#$ a $L_2 = \{\#w_0\#w_1\$w_1\# \mid w_0, w_1 \in \{a, b\}^*\}$. Ukážeme, že $L = L_1 \cup L_2 \in \mathscr{L}(\mathrm{DFPDA}(\leq k)) - \mathscr{L}(\mathrm{DFPDA}(= k))$. Evidentne platí, že L vieme akceptovať deterministickým flipovým zásobníkovým automatom za použitia najviac jedného flipu. Automat iba číta w za prvým #, ak po druhom # príde a alebo b, dáva ich na zásobník a potom po symbole \$ kontroluje, či sú slová rovnaké. Preto $L \in \mathscr{L}(\mathrm{DFPDA}(\leq k))$. Predpokladajme ale, že je L akceptovaný deterministickým flipovým automatom $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$, pričom na každom akceptovanom slove spraví

presne k obratov. Platí teda, že $(\exists q \in Q)(\exists \gamma \in \Gamma^*)(q_0, \#w_0\#, Z_0) \vdash_A^* (q, \varepsilon, \gamma)$ a počas tohto úseku výpočtu spravil A presne k obratov. Keďže ale pre ľubovoľné $w_1 \in \{a, b\}^*$ patrí slovo do L (a aj preto, že A je deterministický), musí byť na slove v akceptačný výpočet, ktorého prvá časť je ako na slove w_0. Na nej ale A žiaden flip nepoužije. Tu už je vidno spor, lebo sme schopní skonštruovať DPDA B, ktorý iba najprv vloží γ na zásobník a simuluje výpočet A. Takto by platilo $L(B) = \{w\$w\# \mid w \in \{a, b\}^*\}$ Keďže ale tento jazyk nie je ani len bezkontextový, dospeli sme k sporu. Preto $L \notin \mathscr{L}(\text{DPFDA}(= k))$ pre *žiadne* k. Preto dokonca platí, že $\mathscr{L}(\text{DFPDA}(= \text{fin})) \subset \mathscr{L}(\text{DFPDA}(\leq \text{fin}))$. □

Hlavnú pozornosť v tejto kapitole sme venovali vete 3.1 a jej dôsledkom. Poukázali sme na jej dôležitosť pri riešení rozličných problémov vyvstávajúcich v oblasti flipových zásobníkových automatov. V ďalšom dôležitosť tejto vety ďalej potvrdíme, keď ju použijeme pri vlastnej konštrukcií.

Kapitola 4

Problém epsilon krokov

V tejto kapitole vyriešime otvorený problém formulovaný v [6].

Východiskom pre nás bude otázka: Ako sa zmení trieda jazykov rozpoznávaných klasickými zásobníkovými automatmi, ak v definícií automatu nepovolíme kroky na ε? Odpoveď je: nijako, ostane nezmenená, [8, 3]. Teda každý bezkontextový jazyk sme schopní akceptovať automatom, ktorý nepoužíva kroky na epsilon. Toto je pomerne zaujímavý fakt, lebo vieme každý jazyk rozpoznať v „reálnom čase“. Úvodzovky sme použili preto, lebo na slove dĺžky n musí zásobníkový automat urobiť najviac $O(n)$ (potenciálne) nedeterministických krokov. Tento fakt je pomerne nenáročným dôsledkom vety o Greibachovej normálnom tvare bezkontextovej gramatiky [3].

V tejto kapitole ukážeme, že podobnú vlastnosť majú aj nedeterministické flipové zásobníkové automaty. Konkrétne ukážeme, že $\mathscr{L}(\mathrm{NFPDA}(k))$ a trieda jazykov rozpoznávaná automatmi nepoužívajúcimi epsilon kroky s k flipmi sú totožné, čím vyriešime problém formulvaný v [6].

4.1 Hlavná idea

Definícia 4.1. Triedu jazykov rozpoznávanú (akceptačným stavom) nedeterministickými zásobníkovými automatmi najviac k obratmi, ktoré nepoužívajú δ-kroky na ε budeme označovať $\mathscr{L}(\text{NFPDA}(\leq k)_\varepsilon)$.

Majme jazyk $L = L(M_1)$, pričom M_1 je ľubovoľný NFPDA($\leq k$), ktorý použije presne k flipov a akceptuje prázdnym zásobníkom. Pre jednoduchosť predpokladajme, že $\varepsilon \notin L$. Chceme ukázať, ako zostojiť nejaký automat $\bar{M}$ nepoužívajúci kroky na ε, taký, že $L(M_1) = L(\bar{M})$. Chceme ukázať, že $L \in \mathscr{L}(\text{NFPDA}(\leq k)_\varepsilon)$. Chceli by sme použiť vetu 3.1, t.j. techniku flip-pushdown input-reversal. Za týmto účelom zavedieme do jazyka L značky takto:

$$L_k = \{v_0\#v_1 \ldots \#v_k\# \mid w = v_0v_1 \ldots v_k \in L\}$$

pričom jazyk $L_k = L(M')$ a platí, že M' je vhodný $\mathscr{L}(\text{NFPDA}(\leq k))$, ktorý hneď po každom flipe prečíta symbol # a na konci slova číta tiež tento symbol. Teda každé slovo z jazyka L_k má v sebe vsunutých presne $k+1$ mriežok, pričom slovo $v_i = \varepsilon$ vtedy a len vtedy, ak automat M_1 medzi i-tým a $i+1$-vým obratom zásobníka (pri akceptačnom výpočte na slove w) nečítal zo vstupu.

Chceme docieliť to, aby sme mohli skonštruovať postupnosť jazykov L_k, L_{k-1}, ..., L_0, tak, že $u\#v\# \in L_i \Leftrightarrow u\#v^R\# \in L_{i-1}$, kde u obsahuje práve i symbolov #. To znamená, že na jazyk L_i použijeme presne i flipov zásobníka a že L_{i-1} vzniká z L_i aplikáciou spomenutej techniky (veta 3.1). Takto budeme môcť problém príslušnosti slova w do jazyka L_k previesť na problém príslušnosti slova w_0 do jazyka L_0, ktorý je bezkotenxtový. Bude teda platiť, že $w \in L_k \Leftrightarrow w_0 \in L_0$, kde jazyk L_0 dostaneme k-násobnou aplikáciou techniky flip-pushdown input-reversal. Ako vyzerá jazyk L_0, presnejšie slová w_0? Závisí to od parity počtu flipov. Pozri tiež poznámku 3.2.

Ak $k = 2l$ potom $L_0 =$

$$\left\{v_0\#v_2\#v_4\ldots\#v_{2l}\#v_{2l-1}^R\ldots\#v_3^R\#v_1^R\# \mid v_0\#v_1\#\ldots\#v_{2l}\#\in L_k\right\}$$

Ak $k = 2l+1$ tak $L_0 =$

$$\left\{v_0\#v_2\#v_4\ldots\#v_{2l}\#v_{2l+1}^R\#v_{2l-1}^R\ldots\#v_3^R\#v_1^R\# \mid v_0\#v_1\#\ldots\#v_{2l+1}\#\in L_k\right\}$$

Z povedaného vyplýva, že slovo $w = w_k$ patrí do L_k práve vtedy keď w_0 (ktoré dostaneme postupným obracaním koncov slov $w_i \in L_i$ podľa vety 3.1) patrí do L_0.

Dôležitý fakt je, že L_0 je bezkontextový, teda máme gramatiku G_0 v Greibachovej normálnom tvare pre jazyk L_0. Ako nám toto pomôže? Predpokladajme kvôli výkladu, že $k = 2l$. (Pre nepárne k sa zmenia totiž len označenia.) Vezmime ľavé krajné odvodenie slova w_0 v gramatike $G_0 = (N, T, P, S)$:

$$\begin{aligned}
S &\Rightarrow^* v_0\alpha_0 \Rightarrow^* && (4.1)\\
&\Rightarrow^* v_0\#v_2\alpha_2 \Rightarrow^* && (4.2)\\
&\Rightarrow^* v_0\#v_2\#v_4\alpha_4 \Rightarrow^*\\
&\Rightarrow^* \cdots \Rightarrow^*\\
&\Rightarrow^* v_0\#v_2\#v_4\ldots\#v_{2l}\alpha_{2l} \Rightarrow^*\\
&\Rightarrow^* v_0\#v_2\#v_4\ldots\#v_{2l}\#v_{2l-1}^R\alpha_{2l-1}\\
&\Rightarrow^* v_0\#v_2\#v_4\ldots\#v_{2l}\#v_{2l-1}^R\#v_{2l-3}^R\alpha_{2l-3} \Rightarrow^*\\
&\Rightarrow^* \cdots \Rightarrow^*\\
&\Rightarrow^* v_0\#v_2\#v_4\ldots\#v_{2l}\#v_{2l-1}^R\#\ldots\#v_3^R\#v_1^R\# = w_0 \in L_0
\end{aligned}$$

pričom $\alpha_i \in N^+$. Z tohto vidno, že kebyže možeme odvodenie správne smerovať a obracať, tak by sme vedeli odvodiť aj slovo w_k. Toto znamená, že kebyže vezmeme časť odvodenia (4.1) a reťazec neterminálov α_0^R a z každého neterminálu odvodzujeme také slová, ktoré su zrkadlovými obrazmi

slov, ktoré vieme z daného neterminálu odvodiť v gramatike G_0 dostaneme „odvodenie“, ktoré by vyzeralo takto:

$$\begin{aligned} S \;&\Rightarrow^*_{G_0}\; v_0\alpha_0 \;\text{ obrátť neterminály} \\ &\qquad v_0\alpha_0^R \;\text{ teraz budeš odvodzovať v reverznej gramatike G'} \\ &\Rightarrow^*_{G'}\; v_0\#v_1\beta \;\text{ kde } \beta \text{ je reťazec neterminálov G'} \end{aligned}$$

Toto samozrejme nevieme zabezpečiť bezkontextovou gramatikou! Podstatu tohto procesu ale vieme odsimulovať za pomoci obratov (flipov) zásobníka. Ak by sme obrátili aj reťazec β a urobili podobný postup, vedeli by sme „odvodiť“ vetnú formu (v reverznej ku reverznej gramatike ku G_0) $v_0\#v_1\#v_2\gamma$. Ozrejmime si, čo presne máme na mysli presnejším označením:

$$\begin{aligned} \alpha_0 &= A_1 \dots A_n \\ A_i &\Rightarrow^*_{G_0} x_i \in T^* \\ u &= x_1 \dots x_n \\ u &= \#v_2\#v_4 \dots \#v_{2l}\#v^R_{2l-1}\# \dots \#v^R_3\#v^R_1\# \\ u^R &= \#v_1\#v_3\# \dots \#v_{2l-1}\#v^R_{2l}\# \dots \#v^R_4\#v^R_2\# \end{aligned} \tag{4.3}$$

Definujme tiež formálnejšie, čo máme na mysli pojmom reverzná gramatika.

Definícia 4.2. Gramatika $H = (N_H, T, P_H, S_H)$ sa nazýva reverznou ku gramatike $I = (N_I, T, P_I, S_I)$, ak platí:

- H je v Greibachovej normálnom tvare
- $N_H \cap N_I = N_I$
- $(\forall \xi \in N_I)(\forall z \in T^*)\ \xi \Rightarrow^*_H z^R \iff \xi \Rightarrow^*_I z$

Slovne: v gramatike H vieme generovať z každého neterminálu gramatiky I (tieto neterminály sú jediné spoločné pre obe gramatiky) iba také

terminálne slová, ktoré su reverzami slov odvoditeľných z daného neterminálu v gramatike I. Ak máme danú gramatiku H, vieme I algoritmicky bez problémov skonštruovať. Toto bude popísané nižšie. Podstatné je, že platí nasledovný fakt, ktorý v ďalšom použijeme.

Lema 4.3. *Nech $H = (N_H, T, P_H, S_H)$ je reverzná gramatika ku gramatike $I = (N_I, T, P_I, S_I)$. Potom pre ľubovoľný neprázdny reťazec neterminálov $\alpha \in N_I^+$ a pre ľubovoľné terminálne slovo v platí:*

$$\alpha \Rightarrow_I^* v \quad \text{vtedy a len vtedy ak} \quad \alpha^R \Rightarrow_H^* v^R$$

Dôkaz. Indukciou na $|\alpha|$. Báza vyplýva z definície 4.2. Indukčný krok je jednoduchý. Predpokladajme, že tvrdenie platí pre všetky α dĺžky k. Nech teraz platí $\alpha A \Rightarrow_I^* v_\alpha v_A$. Z indukčného predpokladu máme, že $\alpha^R \Rightarrow_H^* v_\alpha^R$. Z definície 4.2 máme $A \Rightarrow_H^* v_A^R$. A preto platí $A\alpha^R \Rightarrow_H^* v_A^R v_\alpha^R$. Ľahko vidno, že to platí aj naopak.

□

Chceme skonštruovať zásobníkový automat M_2 ktorý by simuloval kroky odvodenia gramatiky asi takto: Na vstupe má slovo w_k a ide overiť, či je to naozaj tak (či patrí do L_k). Časť odvodenia (4.1) vie simulovať štandardne, po prečítaní v_0 (a súčasnom odvodzovaní a overovaní v gramatike G_0) sa dostane do konfigurácie $(q, \#v_1\#v_2\#\ldots\#v_k\#, A_n \ldots A_1)$. Upozorňujeme, že vrch zásobníka je napravo. Potom spraví obrat (flip), teda prejde do konfigurácie $(p, \#v_1\#v_2\#\ \ldots\ \#v_k\#, A_1\ \ldots\ A_n)$. (Stavy teraz nie sú dôležité.) V tomto momente začne simulovať odvodenie už v rámci gramatiky G_1 (ktorá je reverznou ku gramatike G_0, v zmysle definície 4.2). Čo sa dá v gramatike G_1 odvodzovať z vetnej formy $A_n \ldots A_1$, ktorú máme reprezentovanú na

zásobníku? No podľa (4.3) a definície 4.2 platí:

$$\begin{aligned} A_i &\Rightarrow^*_{G_1} x_i^R \\ A_n \dots A_1 &\Rightarrow^*_{G_1} x_n^R \dots x_1^R \qquad (4.4) \\ u^R &= x_n^R \dots x_1^R \\ u^R &= \#v_1\#v_3\# \dots \#v_{2l-1}\#v_{2l}^R\# \dots \#v_4^R\#v_2^R\# \qquad (4.5) \end{aligned}$$

a potom zo (4.4) a (4.5) vyplýva:

$$A_n \dots A_1 \Rightarrow^*_{G_1} \#v_1 B_1 B_2 \dots B_m$$

kde B_i sú neterminály gramatiky G_1. Tu by opäť nasledoval obrat zásobníka a tak ďalej... To jest: simulovali by sme gramatiku G_2, reverznú ku G_1, použili predpoklady podobné (4.2) a definíciu 4.2. Presnejšie by sme simulovali tú časť odvodenia, ktorá by z vetnej formy $B_m \dots B_1$ odvodila $\#v_2 C_1 \dots C_r$ v rámci gramatiky G_2. Takéto „prepnutie módu simulácie“ by nastalo k-krát počas každého akceptačného výpočtu automatu M_2.

Neformálne sme popísali, ako by mal automat M_2 vyzerať, aké gramatiky a kedy bude simulovať a kedy obracať zásobník. Keďže bude simulovať iba gramatiky v Greibachovej normálnom tvare, nebude používať kroky na epsilon.

Z popísaného ale vyplýva, že automat M_2, ktorého konštrukciu sme celý čas neformálne popisovali, akceptuje „dobré slová“, t. j. také ktoré sú z jazyka L_k. V skutočnosti sme popisovali, ako pre tieto slová ukázať, že existuje nedeterministický akceptačný výpočet. Na prvý pohľad nie je jasné, či neakceptuje náhodou aj nejaké iné slová. V ďalšom ukážeme, že tomu tak nie je. Je tomu tak preto, lebo máme zavedené do slov jazyka L mriežky. Teda že sa nám nemôže stať, ak budeme simulovať a obracať odvodenia gramatiky G_0 (a ďalších k nej reverzných a k nim reverzných atď.), že by sa dala táto gramatika G_0 simuláciou „použiť nesprávne“. Uvedieme jednoduchý príklad, kedy by sa toto mohlo stať, ak nepoužijeme mriežky.

Vezmime jazyk $L = \{ww \mid w \in \{a,b\}^+\}$. Tento jazyk patrí do triedy jazykov $\mathscr{L}(\text{NFPDA}(= 1))$ a vieme ho akceptovať nejakým automatom E_1 prázdnym zásobníkom, pričom tento robí obrat vždy po slove w. Náročky nevložme mriežky a vezmime jazyk $L_0 = \{ww^R \mid w \in \{a,b\}^+\}$, ktorý dostaneme priamočiarou aplikáciou vety 3.1 na jazyk L. Majme gramatiku (v Greibachovej normálnom tvare) $G = (N, \{a,b\}, P, S)$ s pravidlami

$$\begin{aligned} S &\to aSA \mid bSB \\ S &\to aa \mid bb \\ A &\to a \\ B &\to b \end{aligned}$$

Evidentne $L(G) = L_0$. Kebyže chceme aplikovať vyššie naznačenú myšlienku, tak bezepsilonový automat E_2 by najprv simuloval gramatiku G. Preto by bolo možné, aby sa z konfigurácie (q_0, w, Z_0) dostal do konfigurácie $(q, \varepsilon, Z_0 W^R S)$, kde $W \in \{A,B\}^*$ a zodpovedá slovu w. Potom by sa mohol obratom dostať do $(p, \varepsilon, Z_0 SW)$. Simulovaním reverznej gramatiky (ktorá je v tomto prípade presne gramatika G) ďalej na slove w by sa mohol dostať do $(r, \varepsilon, Z_0 S)$. A tu je problém, lebo ďalšou priamočiarou simuláciou reverznej gramatiky by vedel akceptovať aj slová tvaru $wwvv^R$. Pričom ww je časť, ktorú v princípe odsimuloval správne, ale časť vv^R je tam zanesená, kvôli symbolu S, ktorý ostal na zásobníku. Tomuto chceme zabrániť presným vymedzením konca slova a miesta obratu pomocou značiek.

Tento príklad je síce triviálny a umelý, dal by sa dokonca aj opraviť, aby platilo nie iba $L \subseteq L(E_2)$, ale aj $L = L(E_2)$. Chceme ale poukázať na to, že nemôžeme aplikovať vetu 3.1 bez značiek, ktoré nám budú presne určovať kde v čítanom slove má nastať obrat zásobníka a kde je koniec slova. Tieto značky sa nám vnesú aj do gramatiky G_0 a následných k nej reverných gramatík a budú zamedzovať nesprávnym simuláciám (nesprávnym v práve uvedenom zmysle).

4.2 Dôkaz

Majme teda jazyk $L = L(M_1)$, pričom M_1 je NFPDA($\leq k$), ktorý použije presne k flipov a akceptuje prázdnym zásobníkom. Pre jednoduchosť predpokladajme, že $\varepsilon \notin L$ Nie je ťažké dokázať, že aj jazyk

$$L_k = \{v_0 \# v_1 \ldots \# v_k \# \mid v_0 v_1 \ldots v_k \in L\} \tag{4.6}$$

je akceptovaný nejakým flipovým automatom M', pričom tento automat použije presne k flipov, po flipe vždy prečíta symbol # (ak M_1 urobil flip), ktorý je nový a akceptuje prázdnym zásobníkom. Jednoducho vsunieme do slov mriežky tam, kde na ich výpočte automat M_1 spravil flip. Presnejšie bezprostredne za tento flip.

Pôjde nám o to dokázať, že jazyk L_k je akceptovaný automatom, ktorý nepoužíva $\delta(q, \varepsilon, Z)$, pre žiadne q ani Z. Ak budeme mať toto, potom sa ukáže, že vieme mriežky z jazyka L_k vymazať a dostať tak pôvodný jazyk L, za predpokladu, že týmto vymazaním dostaneme opäť jazyk, ktorý sa dá akceptovať bez epsilon krokov. Toto by malo byť len technické tvrdenie, stačí nám pamätať si istý úsek zásobníka a robiť viac krokov akoby naraz, pričom nebudeme musieť použiť epsilon krok. Navyše chceme z každého slova vymazať iba konštantne veľa symbolov, teda sa bude jednať o tzv. limited erasing.

Ukážeme konštrukciu odepsilonovania. V ďalšom predpokladáme, že symbol # je nový. Použijeme:

Veta 4.4. *Nech L_k je jazyk, ktorý vznikne vhodným vsunutím symbolov # do jazyka L, ako sme už popísali vyššie. Postupným aplikovaním techniky flip-pushdown input-reversal z neho zostrojíme jazyky L_{k-1}, ..., L_1, L_0. Potom*

pre tieto jazyky platí:

$$w = v_0\#v_1\#\ldots\#v_{i-1}\#v_i\#v\# \in L_i$$
$$\textit{vtedy a len vtedy ak}$$
$$w' = v_0\#v_1\#\ldots\#v_{i-1}\#v^R\#v_i^R\# \in L_{i-1}.$$

Tu $1 \leq i \leq k$*, pričom jazyk* $L_{i-1} = L(D)$*, pre nejaký NFPDA*$(\leq i-1)$ *automat* D*, je jazyk, ktorý dostaneme, ak na NFPDA*$(\leq i)$ *automat* C*, kde* $L(C) = L_i$ *aplikujeme spomínanú techniku.*

Dôkaz. Stačí si uvedomiť, čo presne znamená aplikovanie techniky na automat C akceptujúci jazyk L_i. Tu je podstatný predpoklad o tom, že máme miesta flipu označené symbolom #. Presnejšie:
$\boxed{\Rightarrow}$:
Z vety 3.1 vyplýva, že ak máme akceptačný výpočet automatu C na slove w vieme z tohto skonšturovať akceptačný výpočet na slove w' automatu D. (Presnejšie to vyplýva z konštrukcie v dôkaze vety 3.1 a ešte presnejšie zo spätnej simulácie, ktorú vykonáva automat D.) Dôležitý je samozrejme predpoklad, že automat C číta po každom flipe znak # a že veta 3.1 odstraňuje posledný flip.
$\boxed{\Leftarrow}$:
Majme akceptačný výpočet na w' automatu D. Potom výpočet po $i-1$-vú mriežku vieme zopakovať v automate C. Ale keďže po tomto poslednom $i-1$-vom flipe musí bezprostredne čítať mriežku, vieme, že sa jedná a koniec nejakého slova z jazyka L_i. Z konštrukcie jazykov $L_k, \ldots, L_1, L_0$ vyplýva, že koniec w' je obratom konca w.

□

Ak vetu 4.4 aplikujeme k-krát dostávame:

Veta 4.5. *Nech* $L_k, L_{k-1}, \ldots, L_1, L_0$ *sú jazyky, ktorých konštrukciu sme*

popísali vo vete 4.4. Potom pre $k = 2l$ *platí:*

$$v_0\#v_1\#\ldots\#v_k\# \in L_k$$
$$\textit{vtedy a len vtedy ak}$$
$$v_0\#v_2\#v_4\#\ldots\#v_{2l}\#v_{2l-1}^R\#\ldots\#v_3^R\#v_1^R\# \in L_0.$$

Podobne, pre $k = 2l + 1$ *platí:*

$$v_0\#v_1\#\ldots\#v_k\# \in L_k$$
$$\textit{vtedy a len vtedy ak}$$
$$v_0\#v_2\#v_4\#\ldots\#v_{2l}\#v_{2l+1}^R\#v_{2l-1}^R\#\ldots\#v_3^R\#v_1^R\# \in L_0.$$

Podstatná pre nás bude korešpondencia medzi slovami z L_0 a slovami z L_k, ktorá bola práve formulovaná vo vete 4.5. Istá nepríjemnosť je v parite čísla k, ktoré je počtom flipov, ako vidno zo znenia vety. Pre jednoduchosť výkladu pripomíname, že budeme v ďalšom predpokladať, že $k = 2l$. Pre prípad $k = 2l + 1$, ako sme už spomenuli, budú uvedené idey aplikovateľné tak isto.

Vieme, že z faktu $L_0 \in \mathscr{L}(\mathrm{CFL})$ vyplýva existencia bezkontextovej gramatiky v Greibachovej normálnom tvare $G_0 = (N_0, \Sigma, P_0, S_0)$, $L(G_0) = L_0$. Ukážeme, ako sme už naznačili, že vieme skonštruovať gramatiky G_i pre $i \in \{1, 2, \ldots, k\}$, tak že gramatika G_i bude reverznou ku gramatike G_{i-1} v zmysle definície 4.2.

Majme teda gramatiku $G_{i-1} = (N_{i-1}, \Sigma, P_{i-1}, S_{i-1})$. Pre všetky $A \in N_{i-1}$ vezmime jazyk $L_A = \{w^R \mid A \Rightarrow^*_{G_{i-1}} w\}$. Tieto jazyky sú bezkontextové, lebo platí že $L_A = L_{G_A}$, kde $G_A = (N_A, \Sigma, P_A, A)$. (Vo všeobecnosti ak vezmeme bezkontextovú gramatiku a v nej ľubovoľný jej neterminál definujeme ako počiatočný, tak dostávame množinu slov z neho odvoditeľnú v rámci tejto gramatiky. Evidentne je táto množina bezkontextovým jazykom.) $N_A = N_{i-1}$ a platí, že $A \to \alpha^R$, $\alpha \in (N_A \cup \Sigma)^*$ je pravidlom gramatiky G_A práve vtedy, keď

pravidlo $A \to \alpha$ je pravidlom gramatiky G_{i-1}. Z bezkontextovosti vyplýva, že existuje gramatika $G'_A = (N'_A, \Sigma, P'_A, A)$ v Greibachovej normálnom tvare (vieme ju skonštruovať algoritmicky, viď napr. [8]) taká, že $L(G'_A) = L_A$. Ďalej môžeme predpokladať, že pre $A \neq B$ je $N'_A \cap N'_B = \emptyset$ a tiež, že $N'_A \cap N_{i-1} = A$. Gramatiku $G_i = (N_i, \Sigma, P_i, S_i)$ dostaneme ako „zjednotenie“ gramatík G'_A, t. j.

$$\begin{aligned} N_i &= \bigcup_{A \in N_{i-1}} N'_A \\ P_i &= \bigcup_{A \in N_{i-1}} P'_A \\ S_i &= S_{i-1} \end{aligned}$$

Z toho ako sme to definovali, je zrejmé, že je splnená definícia 4.2, to jest G_i je reverznou gramatikou ku gramatike G_{i-1}.

Jednoduchšie a menej formálne vyjadrenie predošlej formálnej konštrukcie je takéto. Máme gramatiku G_i. Pre každý jej neterminál A vezmeme gramatiku vzniknutú z G_i tak, že A bude počiatočný a pravé strany pravidiel reverzneme. Túto gramatiku prevedieme do Greibachovej normálneho tvaru. Toto bude gramatika G_A. Ešte treba premenovaním zabezpečiť to, aby pre rôzne neterminály $A \neq B$ mali gramatiky G_A a G_B rôzne neterminály a to kvôli tomu, aby sme nemohli na neterminál jednej použiť odvodzovacie pravidlo tej druhej. Tieto gramatiky G_A pre všetky $A \in N(G_i)$ vezmeme a spojíme do jednej jednoduchým zjednotením ich neterminálov a pravidiel. Takto dostávame gramatiku G_{i+1}.

Prikročme k zostrojeniu bezepsilonového NFPDA($\leq k$) automatu M_2 pre jazyk L_k. Tento bude iba postupne simulovať gramatiky G_i a to tak, že po i-tom flipe bude simulovať gramatiku G_i. Po flipe bude vždy na vstupe nasledovať mriežka. Automat bude akceptovať *stavom* a *iba vtedy*, ak je na zásobníku *jediný* symbol Z_0. Toto robíme preto, aby sme mohli priamo použiť vetu 3.1. Je to iba technické, ale stačí nám pamätať si v zásobníkovom

symbole, že pod ním je už iba symbol Z_0. (Treba si všimnúť, že v podkapitole 4.1 sme uvažovali tak, ako keby mal zásobníkový automat na zásobníku iba vetnú formu. Formálne musí mať na spodku aj symbol Z_0, ktorý nie je súčasťou tejto vetnej formy. A toto musíme formálne odsimulovať, aby sme mohli aplikovať myšlienky z podkapitoly 4.1, použiť vetu 3.1 a súčasne aby konštruovaný automat M_2 nerobil kroky na ε.)

Pre jednoduchosť môžeme predpokladať, že v_0 v (4.6) je rôzne od ε. Ak nie, vieme pridať bezprostredne pred prvý flip automatu M_1 prečítanie nejakého nového symbolu, napr. \$. Tento predpoklad si môžeme dovoliť z doho dôvodu, že vsuviek (teda nových symbolov vhodne vsunutých do slov jazyka L) môže byť konštantne veľa. Preto pridanie tohto ďalšieho symbolu \$ nijako neuškodí. Lebo v ďalšom ukážeme, že pre „vsuvkovaný“ jazyk L_k vieme spraviť bezepsilonový automat a že trieda jazykov rozpoznávaná týmito automatmi je uzavretá na operáciu výmazu jedného symbolu. Keďže ale potrebujeme vymazať konštantne veľa symbolov, tak našej konštrukcií nijako neuškodí, že táto konštanta (počet vsunutých symbolov) bude o jedna väčšia.

Formálne bude $M_2 = (K, \Sigma \cup \{\#\}, \Gamma, \delta, \Delta, \overline{q_0}, Z_0, \{q_F\})$,

$$\begin{aligned} K &= \{q_F\} \cup \{\overline{q_0}, \overline{q_1}, \ldots, \overline{q_k}\} \cup \{q_0, q_1, \ldots, q_k\}, \\ \Gamma &= \{A, A^E \mid A \in N_k\} \cup \{Z_0\}, \\ \Delta(q_i) &= \{\overline{q_{i+1}}\} \ \wedge \ i \in \{0, 1, \ldots, k-1\}. \end{aligned}$$

Symbol A^E máme iba kvôli tomu, aby sme vedeli, že pod týmto symbolom, je už iba Z_0 (E je zo slova end). Pripomíname, že $\# \notin \Sigma$. Tak spravíme aj δ-funkciu, aby sme toto pri simulácií gramatík G_i zohľadnili.

Nasleduje výpis δ-funkcie:

$$\begin{aligned}
\delta(\overline{q_0}, a, Z_0) &\ni (q_0, Z_0 A^E \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ S_0 \to a\beta A \in P_0 \\
\delta(q_0, a, Z) &\ni (q_0, \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta \in P_0 \\
\delta(q_0, a, Z^E) &\ni (q_0, A^E \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta A \in P_0 \\
\delta(\overline{q_i}, \#, Z) &\ni (q_i, \beta^R) \quad \Leftrightarrow \quad Z \to \#\beta \in P_i \\
\delta(\overline{q_i}, \#, Z^E) &\ni (q_i, A^E \beta^R) \quad \Leftrightarrow \quad Z \to \#\beta A \in P_i \\
\delta(q_i, a, Z) &\ni (q_i, \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta \in P_i \\
\delta(q_i, a, Z^E) &\ni (q_i, A^E \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta A \in P_i \\
\delta(\overline{q_k}, \#, Z) &\ni (q_k, \beta^R) \quad \Leftrightarrow \quad Z \to \#\beta \in P_k \\
\delta(\overline{q_k}, \#, Z^E) &\ni (q_i, A^E \beta^R) \quad \Leftrightarrow \quad Z \to \#\beta A \in P_k \\
\delta(q_k, a, Z) &\ni (q_k, \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta \in P_k \\
\delta(q_k, a, Z^E) &\ni (q_k, A^E \beta^R) \quad \Leftrightarrow \quad a \in \Sigma \ \wedge \ Z \to a\beta A \in P_k \\
\delta(q_k, \#, Z^E) &\ni (q_F, \varepsilon) \quad \Leftrightarrow \quad Z \to \# \in P_k
\end{aligned}$$

Tu je $i \in \{1, 2, \ldots, k-1\}$, teda pre každé takéto i máme príslušné riadky δ-funkcie. P_m sú pravidlá gramatiky G_m. Okrem uvedených nie sú v delta funkcií žiadne ďalšie položky. Podstatné je, aby platila nasledujúca lema.

Lema 4.6. *Pre skonštruovaný automat M_2 platí:*
$(\forall v \in \Sigma^*)(\forall \alpha \in \Gamma^*)(\forall \beta \in \Gamma^*)$

$$\begin{aligned}
(\overline{q_0}, v, Z_0) \vdash^* (q_0, \varepsilon, Z_0 \alpha^R) &\Leftrightarrow S_0 \Rightarrow^*_{G_0} v\alpha \\
(\overline{q_i}, \#v, Z_0 \alpha^R) \vdash^* (q_i, \varepsilon, Z_0 \beta^R) &\Leftrightarrow \alpha \Rightarrow^*_{G_i} \#v\beta \\
(\overline{q_k}, \#v\#, Z_0 \alpha^R) \vdash^* (q_F, \varepsilon, Z_0) &\Leftrightarrow \alpha \Rightarrow^*_{G_k} \#v\#
\end{aligned}$$

kde $i \in \{1, 2, \ldots, k-1\}$ $a \vdash$ kroky výpočtu sú iba za použitia δ-funkcie (teda bez obratov zásobníka).

Dôkaz. Vyplýva priamo z definície automatu M_2, lebo sme ho konštruovali na základe gramatík G_i tak, aby sme zabezpečili ich správnu simuláciu. Tro-

chu znásilňujeme notáciu v tom, že symboly zásobníkovej abecedy sú bohatšie (máme symboly A^E), ale v tvrdení lemy stotožnujeme A^E so symbolom A, aj keď tak vyslovene neuvádzame.

□

Veta 4.7. *Pre skonštruovaný automat M_2 platí, že $L(M_2) = L_k$, pričom automat M_2 nepoužíva kroky na epsilon.*

Dôkaz. To, že M_2 je bezepsilonový, je zrejmé z konštrukcie a z toho, že gramatiky G_i majú všetky pravidlá tvaru $N_i \to \Sigma N_i^*$ (sú v Greibachovej normálnom tvare). Podstatné je dokázať rovnosť. Pripomíname predpoklad, že $k = 2l$, pre nepárne k je dôkaz analogický.

$\boxed{L_k \subseteq L(M_2)}$:

Nech $w \in L_k$. Z vety 4.5 vieme, že

$$\begin{aligned} w &= v_0\#v_1\#\ldots\#v_{2l-1}\#v_{2l}\# \in L_k \Leftrightarrow \\ w_0 &= v_0\#v_2\#\ldots\#v_{2l}\#v_{2l-1}^R\#\ldots\#v_1^R\# \in L_0 \end{aligned}$$

Preto z predpokladu $w \in L_k$ vyplýva, že $w_0 \in L_0$, t.j. existuje ľavé krajné odvodenie slova w_0 v gramatike G_0. Na základe tohto odvodenia a skonštruovaných gramatík G_i zostrojíme platný výpočet automatu M_2 na slove w. Predpokladáme teda, že:

$$\begin{aligned} S_0 &\Rightarrow_{G_0}^* v_0\alpha \qquad (4.7)\\ v_0\alpha &\Rightarrow_{G_0}^* w_0 \\ \alpha &\Rightarrow_{G_0}^* \#v_2\#\ldots\#v_{2l}\#v_{2l-1}^R\#\ldots\#v_1^R\# \end{aligned}$$

Ďalej platí:

$$\begin{aligned} (\overline{q_0}, v_0, Z_0) &\vdash^* (q_0, \varepsilon, Z_0\alpha^R) \qquad (4.8)\\ (q_0, \varepsilon, Z_0\alpha^R) &\vdash (\overline{q_1}, \varepsilon, Z_0\alpha) \qquad (4.9)\\ \alpha^R &\Rightarrow_{G_1}^* \#v_1\beta \qquad (4.10)\\ \beta &\Rightarrow_{G_1}^* \#v_3\ldots\#v_{2l-1}\#v_{2l}^R\#\ldots\#v_2^R\# \qquad (4.11) \end{aligned}$$

Fakt (4.8) vyplýva z lemy 4.6 a z faktu (4.7). Fakt (4.9) je iba využitím Δ-kroku v automate M_2, lebo sme ho tak definovali, čím sme tento prechod umožnili. Fakty (4.10) a (4.11) vyplývajú z faktu (4.8) a z lemy 4.3. Tu β je reťazec neterminálov gramatiky G_1. Teraz opäť použijeme lemy 4.6 a 4.3 a dostávame:

$$(\overline{q_1}, \#v_1, Z_0\alpha) \quad \vdash^* \quad (q_1, \varepsilon, Z_0\beta^R) \tag{4.12}$$

Fakt (4.12) vyplýva opäť z lemy 4.6 a z faktu (4.10). Veríme, že čitateľ vie k sebe pripojiť časti výpočtu (4.8), (4.9) a (4.12). Toto spojenie nám dá časť výpočtu na prefixe slova w. Tento prefix je v tomto prípade $v_0\#v_1$. Takto vieme indukciou ďalej pokračovať. Postupne vieme skonštruovať časti výpočtov na prefixe $v_0\#v_1\#v_2$ aj $v_0\#v_1\#v_2\#v_3$ a tak ďalej. Tento postup budeme opakovať, až sa nám podarí zostrojiť platný výpočet na celom slove w. Preto $w \in L(M_2)$.

$\boxed{L(M_2) \subseteq L_k}$:

Predpokladajme, že $w = v_0\#v_1\#\ldots\#v_{2l-1}\#v_{2l}\# \in L(M_2)$. Rozpísaním:

$$\begin{aligned}
(\overline{q_0}, v_0\#v_1\#\ldots\#v_{2l}\#, Z_0) \quad &\vdash^* \quad (q_0, \#v_1\#\ldots\#v_{2l}\#, Z_0\alpha_0) \\
(\overline{q_i}, \#v_i\#\ldots\#v_{2l}\#, Z_0\alpha_{i-1}^R) \quad &\vdash^* \quad (\overline{q_{i+1}}, \#v_{i+1}\#\ldots\#v_{2l}\#, Z_0\alpha_i) \qquad (4.13) \\
(\overline{q_k}, \#v_{2l}\#, Z_0\alpha_{k-1}^R) \quad &\vdash^* \quad (q_F, \varepsilon, Z_0) \qquad (4.14)
\end{aligned}$$

Tu $i \in \{1, \ldots, k-1\}$. Toto je temer kompletný výpis výpočtu M_2 na slove w (chýbajú len Δ-kroky), ktoré nastanú medzi uvedenými riadkami, t. j. zo stavu q_i do stavu $\overline{q_{i+1}}$. To, že tento výpočet musí vyzerať takto, vyplýva z definície automatu M_2.

Ideme ukázať, že vieme skonštruovať odvodenie slova $w_0 = v_0\#v_2\# \ldots \#v_{2l}\ \#v_{2l-1}^R\ \#\ldots\#\ v_1^R\#$ v gramatike G_0. Potom už bude w patriť do L_k vďaka vete 4.5. Odvodenie v G_0 budeme z akceptačného výpočtu konštruovať odzadu. Z lemy 4.6 a z predpokladu (4.14) máme:

$$\alpha_{k-1} \quad \Rightarrow_{G_k}^* \quad \#v_{2l}\# \tag{4.15}$$

Z predpokladu (4.13) pre $i = k - 1$ a z lemy 4.6 máme:

$$\alpha_{k-2} \quad \Rightarrow^*_{G_{k-1}} \quad \#v_{2l-1}\alpha^R_{k-1} \tag{4.16}$$

Potom z toho, že G_k je reverzná ku G_{k-1}, lemy 4.3 a z faktov (4.15) a (4.16) máme:

$$\alpha_{k-2} \quad \Rightarrow^*_{G_{k-1}} \quad \#v_{2l-1}\#v^R_{2l}\#$$

To, že gramatika G_k je reverzná ku G_{k-1} je tu kľúčová záležitosť. Práve vďaka tejto skutočnosti môžeme použiť lemu 4.3 a máme zaručenú existenciu odvodenia $\alpha_{k-1} \Rightarrow^*_{G_{k-1}} \#v^R_{2l}\#$. Predpoklad lemy 4.3, že α_{k-1} musí byť reťazec neterminálov gramatiky G_{k-1} vyplýva z konštrukcie automatu M_2. Preto ju môžeme použiť. Týmto postupom vieme vždy zostúpiť do gramatiky s nižším indexom. Reverznosti gramatík, lemy 4.3 aj 4.6 a to ako máme rozpísaný výpočet M_2 na slove w nám nakoniec umožní ukázať, že existuje odvodenie slova w_0 v gramatike G_0, t. j. podarí sa nám zostúpiť až k odvodeniu slova w_0 v gramatike G_0. Všimnime si, že k obracaniu častí slov v_i dochádza pri tomto zostupovaní do gramatiky s nižším indexom.

□

Dokázali sme, že vieme „odepsilonovať“ jazyk L_k, do ktorého sme vsunuli $k+1$ mriežok a ešte možno jeden symbol (tesne pred prvú mriežku — aby sme zabezpečili hladký štart simulácie automatom M_2). Podstatné je, že musíme z každého slova vymazať tieto nové symboly (je ich len konštantne veľa) a dostať tak jazyk L. Samozrejme potrebujeme to urobiť tak, aby sme vedeli garantovať, že akceptor pre jazyk L nebude používať kroky na ε.

4.3 Vymazanie značiek

Ukážeme, ako vymazať vložené značky, resp. ako vymazať konštantne veľa symbolov, ktoré nie sú v abecede Σ.

Veta 4.8. *Nech* $L \in \mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$ a $L = L(C)$ pre nejaký NFPDA$(= k)_\varepsilon$ automat C. Nech jazyk L obsahuje iba slová dĺžky aspoň dva. Potom platí, že jazyk

$$L' = \{uv \mid u, v \in \Sigma^* \ \wedge \ (\exists a \in \Sigma) uav \in L\}$$

je tiež z triedy $\mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$ a je akceptovaný nejakým NFPDA$(= k)_\varepsilon$ automatom D.

Dôkaz. Automat D bude iba simulovať automat C, pričom sa počas každého výpočtu musí práve raz rozhodnúť, že vloží nejaký znak a z abecedy Σ do čítaného slova. Ak sa mu podarí takto akceptovať, tak akceptuje. Zachovanie počtu obratov je zrejmé, lebo pôvodný automat robil najviac k obratov. Zachovanie bezepsilonovosti je technické, ale jedná sa iba o aplikovanie dobre známej idey, ktorá spojí dva kroky do jedného. Automat D bude mať možnosť naraz odsimulovať dva δ-kroky automatu C, pričom prvý z nich bude na symbole a. Presnejšie ak automat D číta b, odsimuluje naraz kroky automatu C na ab. Ak je ale a na konci (teda na vstupe už nenasleduje žiaden symbol), tak na vstupe b odsimuluje kroky automatu C na ba. Takýto automat sa dá priamočiaro skonštruovať, len je to dosť technické. Treba dbať pritom na to, že automat C mohol urobiť obrat po symbole a. Z tohto vidno, že D si bude musieť pamätať istú konštantnú časť zásobníka v stavoch – to čo je na spodku a čo je na vrchu a prispôsobovať výpočet. Alebo obrat odloží o jeden krok neskôr aby mohol spávne modifikovať zásobník. Predpoklad, že L obsahuje len slová dĺžky aspoň 2 nám zabezpečí bezepsilonovosť automatu D, lebo „virtuálny“ krok na symbole a vieme vždy spojiť s krokom na nejakom skutočnom čítanom vstupnom symbole. Jazyk L' ako vidno nebude obsahovať ε, ale bude už môcť obsahovať slová dĺžky 1.

□

Veta 4.9. *Trieda jazykov* $\mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$ je uzavretá na prienik s regulárnym jazykom.

Dôkaz. Triviálny. Automat iba v stavoch simuluje počas výpočtu konečnostavový automat pre daný regulárny jazyk a akceptuje iba vtedy, ak je tento konečnostavový automat v akceptačnom stave. Je zrejmé, že počet flipov aj vlastnosť nepoužívania ε-krokov sa zachová.

□

Vráťme sa späť k nášmu problému. Zobrali sme jazyk L, akceptovaný automatom M_1 s presne k obrami, o ktorom sme predpokladali, že neobsahuje ε. Skonštruovali sme jazyk L_k z jazyka L vsunutím značiek na miesta vo vstupnom slove, kde sa udial obrat počas výpočtu (v automatu M_1) na tomto slove. Ukázali sme, že jazyk L_k vieme akceptovať automatom M_2, pričom tento nepoužíva δ-kroky na epsilon. Teraz $k+1$ násobným použitím vety 4.8 a prienikom s regulárnym jazykom $(\Sigma - \{\#\})^*$ vieme docieliť vymazanie $k+1$ znakov # zo slov v jazyku L_k a dostať tak jazyk L. Veta 4.9 nám pritom zaručí, že $L \in \mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$. Keďže vety 4.8 a 4.9 sú konštruktívne, tak ich aplikovaním na automat M_2 dostávame finálny NFPDA$(= k)_\varepsilon$ automat $\bar{M}$ akceptujúci jazyk L.

4.4 Zhrnutie

Veta 4.10. *Nech* $k \geq 0$. *Potom* $\mathscr{L}(\text{NFPDA}(= k)) = \mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$.

Dôkaz. Pre $k = 0$ dostávame všeobecne známe tvrdenie pre klasické zásobníkové automaty. Ak je $k > 0$ vyplýva tvrdenie z predošlej konštrukcie. Jej postup bol takýto:

1. Máme daný jazyk $L \in \mathscr{L}(\text{NFPDA}(= k))$ akceptovaný automatom M_1.
2. Umelo doň vsunieme na správne miesta symboly #, čím dostaneme jazyk L_k akceptovaný automatom M'.
3. O jazyku L_k dokážeme, že patrí do $\mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$. Tento dôkaz je v

podstate konštrukcia automatu M_2. Toto je najdôležitejšia a podstatná časť dôkazu.

4. Z jazyka L_k následne pomocou uzáverových vlastností, respektíve viet 4.8 a 4.9, odstránime vsunuté symboly, čím dostaneme NFPDA$(= k)_\varepsilon$ automat $\bar{M}$.

Zosumarizujeme ešte raz každý krok osobitne.

1. V kapitole 2 sme ukázali, že nezáleží na spôsobe akceptácie, preto predpoklad, že $L \in \mathscr{L}(\text{NFPDA}(= k))$ nie je nijako obmedzujúci. Jediný predpoklad, ktorý sme počas predošlého výkladu urobili bol, že $\varepsilon \notin L$. Toto tiež nie je problém, lebo ε sa dá pridať do ľubovoľného jazyka z $\mathscr{L}(\text{NFPDA}(= k)_\varepsilon)$ tým spôsobom, že akceptoru tohto jazyka pridáme nový počiatočný stav a tento bude akceptačný a nikdy sa doň nebude dať dostať, ak ho počas výpočtu tento akceptor už raz opustí. Preto je aj tento predpoklad neobmedzujúci.

2. Vsúvanie symbolov robíme podľa popisu. Bezprostredne po každom flipe ide symbol # a na úplnom konci slova tiež. Technický predpoklad, ktorý sme vyššie uviedli, aby počiatočná časť slova (konkrétne v_0) bola rôzna od ε sa dá vyriešiť vsunutím iného špeciálneho symbolu. Jeho výmaz prebehne tiež za použitia vety 4.8. Upozorňujeme, že kvôli predpokladu $\varepsilon \notin L$ a práve povedanému má jazyk L_k iba slová dĺžky aspoň dva.

3. Obšírne sme sa tomuto kroku venovali v podkapitolách 4.1 a 4.2 – pri konštrukcií automatu M_2 a dokazovaní toho, že má požadované vlastnosti.

4. Tu už iba pripomenieme, že vetu 4.8 môžeme použiť práve kvôli faktu, že L_k neobsahuje ε ani slová dĺžky jedna.

□

V tejto kapitole sme dokázali, že δ-kroky na ε môžu byť z ľubovoľného automatu eliminované, bez ovplyvnenia výpočtovej sily modelu. Dá sa hovoriť o normálnom tvare pre nedeterministické flipové zásobníkové automaty. Jedná sa o rozšírenie dobre známeho normálneho tvaru (ktorý sme spomínali v úvode tejto kapitoly) pre klasické zásobníkové automaty. Tento normálny tvar tiež stojí na Greibachovej normálnom tvare pre bezkontextové gramatiky, lenže je aplikovaný v širšom kontexte úvah ako pri klasických zásobníkových automatoch. Týmto sme kladne odpovedali na nezodpovedanú otázku z článku [6].

Kapitola 5

Uzáverové vlastnosti

V tejto kapitole sa venujeme uzáverovým vlastnostiam (najmä) deterministického variantu flipového zásobníkového automatu. Týmto sa pokúšame aspoň o čiastočné vyriešenie ďalšej otázky položenej v článku [6] – „Aké sú uzáverové vlastnosti tried jazykov ktoré vznikajú keď uvažujeme deterministické flipové zásobníkové automaty?“.

5.1 Niektoré vlastnosti

V nasledujúcom budeme pracovať (ak nebude uvedené ináč) výhradne s deterministickými najviac k-flipovými automatmi akceptujúcimi stavom. Formálne povedané nás budú zaujímať uzáverové vlastnosti triedy $\mathscr{L}(\mathrm{DFPDA}(\leq k))$. Uvidíme, že nasledujúce tvrdenia budú dosť kopírovať známe vlastnosti obyčajných deterministických automatov. Pripomeňme a označme si najprv jazyky, ktoré budeme v nasledujúcich úvahách používať:

$$\begin{aligned} J_k &= \{w_1\#w_1\#w_2\#w_2\#\ldots\#w_k\#w_k \mid w_i \in \{a,b\}^*, i \in \{1,\ldots,k\}\}\,, \\ L_{abc} &= \{a^n b^n c^n \mid n \geq 1\}\,. \end{aligned}$$

Veta 5.1 ([5, 6])**.** *Nech k je prirodzené číslo väčšie ako nula. Potom*

$$J_k \in \mathscr{L}(\text{DFPDA}(= \text{k})) \subset \mathscr{L}(\text{DFPDA}(\leq \text{k})) \subset \mathscr{L}(\text{NFPDA}(= \text{k}))$$
$$J_k \notin \mathscr{L}(\text{NFPDA}(= \text{k} - 1))$$

Dôkaz. To že J_k je v $\mathscr{L}(\text{DFPDA}(= k))$ je jednoduché. Automat pre tento jazyk bude robiť nasledovné: na začiatku bude kopírovať symboly zo vstupu do zásobníka. Ak prečíta #, tak obráti zásobník a začne kontrolovať, či to, čo je na vstupe sa zhoduje s tým, čo číta zo zásobníka. Ak po dočítaní druhého w_i v poradí má na zásobníku iba Z_0, tak pokračuje, inak sa zasekne. Následne robí podobne pre w_{i+1}.

Obidve uvedené inklúzie vyplývajú priamo z definícií daných tried.

Tvrdenie $J_k \notin \mathscr{L}(\text{NFPDA}(= k - 1))$ vyplýva okamžite z vety 3.3.

□

Tvrdenie 5.2. *Trieda* $\mathscr{L}(\text{DFPDA}(\leq k))$ je uzavretá na prienik s regulárnym jazykom.

Dôkaz. Jednoduchý. Iba simulujeme konečný automat v stavoch riadiacej jednotky flipového zásobníkového automatu a akceptujeme iba vtedy, ak akceptuje aj tento konečný automat.

□

Ďalej bude pre nás veľmi užitočné tvrdenie vety 3.5. Zostrojíme pomocou neho jazyk s podobnou vlastnosťou ako má jazyk L_{abc}, ale s tým rozdielom, že L_{abc} je druh jazyka, ktorý sa nedá akceptovať nedeterministickými flipovými automatmi a tento sa nebude dať akceptovať flipovými deterministickými automatmi.

Veta 5.3. *Nech* $k \geq 0$*. Trieda* $\mathscr{L}(\text{DFPDA}(\leq k))$ nie je uzavretá na zjednotenie.

Dôkaz. Majme jazyky

$$\begin{aligned} L_1 &= \{a^n b^n \mid n \geq 1\} \in \mathscr{L}(\text{DFPDA}(\leq \text{k})), \\ L_2 &= \left\{a^n b^{2n} \mid n \geq 1\right\} \in \mathscr{L}(\text{DFPDA}(\leq \text{k})). \end{aligned}$$

Oba sú jednoduché deterministické zásobníkové jazyky (bezflipové) a preto patria aj do $\mathscr{L}(\text{DFPDA}(\leq k))$. Sporom ukážeme, že ich zjednotenie nemôže patriť do $\mathscr{L}(\text{DFPDA}(\leq k))$. Nech by $L_1 \cup L_2 \in \mathscr{L}(\text{DFPDA}(\leq k))$. Majme automat $A = (Q,\ \{a, b\},\ \Gamma,\ \delta,\ \Delta,\ q_0,\ Z_0,\ F)$, taký, že $L(A) = L_1 \cup L_2$. Z automatu A skonštruujeme iný flipový zásobníkový automat B, pričom tento bude nedeterministický a na vstupe bude robiť najviac $2k$ flipov a bude akceptovať jazyk L_{abc}. Toto bude žiadaný spor s vetou 3.5.

Najprv na základe automatu A zostrojme automat $B = (Q \cup \overline{Q},\ \{a, b, c\},\ \Gamma,\ \delta_B,\ \Delta_B,\ q_0,\ Z_0,\ F \cup \overline{F})$ takto:

$$\begin{aligned} \overline{Q} &= \{\overline{q} \mid q \in Q\} \\ \overline{F} &= \{\overline{q_F} \mid q_F \in F\} \\ \delta_B(q, a, Z) &\ni (p, \beta) \Leftrightarrow \delta(q, a, Z) \ni (p, \beta) \\ \delta_B(q, b, Z) &\ni (p, \beta) \Leftrightarrow \delta(q, b, Z) \ni (p, \beta) \\ \delta_B(\overline{q}, a, Z) &\ni (\overline{p}, \beta) \Leftrightarrow \delta(q, a, Z) \ni (p, \beta) \\ \delta_B(\overline{q}, c, Z) &\ni (\overline{p}, \beta) \Leftrightarrow \delta(q, b, Z) \ni (p, \beta) \\ \delta_B(q, \varepsilon, Z) &\ni (\overline{q}, Z) \Leftrightarrow q \in F \\ \Delta_B(q) &= \Delta(q) \\ \Delta_B(\overline{q}) &= \{\overline{p} \mid p \in \Delta(q)\} \end{aligned}$$

Toto je dobre definovaný (nedeterministický) flipový zásobníkový automat. Intuitívne sme urobili toto: zobrali sme dve kópie automatu A (v jednej sú stavy q, v druhej sú stavy označené – $\overline{q}$), pričom sme definovali v rámci prvej kópie δ_B aj Δ_B prechody úplne totožne ako v A a v druhej kópií sme urobili podobne s tým rozdielom, že namiesto čítania znaku b sme do novej

δ_B-funkcie zapísali čítanie znaku c. Prechod medzi kópiami je možný iba z prvej kópie do druhej a to iba medzi akceptačným stavom q_F v prvej kópií a jemu zodpovedajúcim $\overline{q_F}$ v druhej kópií na ε bez zmeny zásobníka.

Ako vyzerá $L(B)$? Sú dve možnosti, kedy mohol automat B akceptovať: buď v nejakom akceptačnom stave q_F z prvej kópie, alebo v nejakom akceptačnom stave $\overline{q_F}$ z druhej kópie. Ak nastala prvá možnosť, tak B nepoužil δ_B-prechod na ε z prvej do druhej kópie, teda výpočet musel vyzerať totožne ako v automate A (tak sme to definovali). Preto slová ktoré akceptuje touto možnosťou sú presne slová z jazyka $L(A)$.

Druhá možnosť je, že B použil (iba raz, lebo je to tak definované), ε prechod medzi akceptačnými stavmi q_F a $\overline{q_F}$. Teraz si treba uvedomiť, že v čase tohto ε-prechodu je v konfigurácií, v ktorej automat A akceptuje. Preto slovo ktoré B prečítal pred tým ako sa do tejto konfigurácie dostal je tvaru buď $a^n b^n$ alebo $a^n b^{2n}$. Lenže B je tiež v takej istej konfigurácií v akej by bola druhá kópia automatu po prečítaní $a^n c^n$ alebo $a^n c^{2n}$, kebyže je začiatočný stav $\overline{q_0}$! Ak prečítané slovo bolo $a^n c^{2n}$, tak po prečítaní zvyšku vstupu B neakceptuje, lebo slovo $a^n c^{2n}$ nie je prefixom žiadneho slova ktoré akceptuje kópia A. Zaujímavé je keď prečítané slovo bolo $a^n b^n$. V tomto prípade B prečítal v prvej kópií $a^n b^n$ a bol v stave q_F, presunul sa na ε do $\overline{q_F}$ a teraz ak bude na vstupe c^n, tak musí akceptovať, lebo táto konfigurácia v ktorej sa nachádza je totožná s konfiguráciou v ktorej by bola kópia automatu A po prečítaní $a^n c^n$. Preto je B schopný akceptovať slovo tvaru $a^n b^n c^n$.

Ako je to s počtom flipov? Keďže A na ľubovoľnom vstupe nepoužije viac ako k flipov, tak potom B nemôže použiť viac ako $2k$. Lebo ak máme akceptačný výpočet automatu B, tak sa dá rozdeliť na dve časti: v rámci prvej a v rámci druhej kópie automatu A. Každá táto časť zodpovedá (pod)časti akceptačného výpočtu automatu A na nejakom slove. Preto v každej časti môže B použiť najviac k flipov. Teda dokopy najviac $2k$ flipov.

Preto

$$L(B) = \{a^n b^n \mid n \geq 1\} \cup \{a^n b^{2n} \mid n \geq 1\} \cup \{a^n b^n c^n \mid n \geq 1\}.$$

Prienikom jazyka $L(B)$ s regulárnym jazykom $a^+b^+c^+$ dostaneme práve jazyk L_{abc}. Ale trieda $\mathscr{L}(\text{NFPDA}(\leq k))$ je uzavretá na prienik s regulárnym jazykom, preto musí byť aj $L_{abc} \in \mathscr{L}(\text{NFPDA}(\leq 2k))$ a toto je spor s vetou 3.5.

□

Tvrdenie 5.4. *Trieda* $\mathscr{L}(\text{DFPDA}(\leq k))$ nie je uzavretá na prienik.

Dôkaz. Majme jazyky L_1 a L_2 akceptovateľné obyčajnými deterministickými zásobníkovými automatmi.

$$\begin{aligned} L_1 &= \left\{a^n b^n c^l \mid n, l \geq 1\right\}, \\ L_2 &= \left\{a^l b^n c^n \mid n, l \geq 1\right\}. \end{aligned}$$

Potom $L_1 \cap L_2$ je jazyk L_{abc}. Podľa vety 3.5 tento jazyk nemôže patriť do $\mathscr{L}(\text{DFPDA}(\leq k))$.

□

Venujme sa teraz uzavretosti triedy $\mathscr{L}(\text{DFPDA}(\leq k))$ na komplement. Ako je známe, trieda $\mathscr{L}(\text{DCFL})$ je na túto operáciu uzavretá. Toto je práve dosť zásadný dôsledok determinizmu zavedeného do modelu DPDA. Konštrukcia ako k danému DPDA A skonštruovať DPDA B tak, že $L(B) = L(A)^C$ je zaujímavá, ale dosť technická. Jej dôkaz aj s výkladom sa dá nájsť v [10, 8]. Pozitívne ale je, že ju môžeme bez väčšších problémov prispôsobiť modelu DFPDA.

Lema 5.5. *Ku každému najviac k-flipovému DFPDA A existuje DFPDA B taký, že B na každom vstupnom slove zastane a prečíta celý vstup. Pritom* $L(A) = L(B)$ *a B použije najviac k flipov (tak isto ako A).*

Dôkaz. Cieľ je taký istý ako pri DPDA. Chceme predísť tomu, aby daný automat A nemohol pokračovať vo výpočte. Chceme, aby pre každú konfiguráciu B (ktorý bude vhodne simulovať A) platilo, že sa z nej dostane do takej konfigurácie, z ktorej bude definovaný prechod na vstupný symbol z abecedy Σ. Sú tri možnosti kedy toto nie je splnené:

1. A úplne vyprázdni zásobník (zasekne sa)
2. A nemá definovaný ďalší prechod, t.j. v situácií keď je v stave p a na zásobníku je Z platí, že pre všetky $a \in \Sigma \cup \{\varepsilon\}$ je $\delta(p, a, Z) = \emptyset$ a tiež $\Delta(p) = \emptyset$ (tiež sa zasekne)
3. A sa dostane do konfigurácie, z ktorej je možný nekonečný počet δ-krokov na ε (zacyklí sa)

Tu je veľmi dôležité, že A sa nemôže zacykliť tak, že by robil nekonečný počet Δ-krokov. Práve vďaka tomu sú príčiny nedočítania vstupu úplne rovnaké, ako pri klasickom DPDA. Preto naplno využijeme konštrukciu uvedenú v [10].

Majme daný k-flipový DFPDA $A = (Q, \Sigma, \Gamma, \delta, \Delta, q_0, Z_0, F)$. Bez ujmy na všeobecnosti môžeme predpokladať, že A má počas celého výpočtu na dne zásobníka symbol Z_0. Predpokladáme, že k je nám známe. Pre názornosť skonštruujeme automat A', na ktorý budeme môcť pohodlnejšie aplikovať spomínanú konštrukciu. Nech $A' = (Q', \Sigma, \Gamma, \delta', \Delta', q_0', Z_0, F')$, kde

$$\begin{aligned}
Q' &= \{[q, i] \mid q \in Q, i \in \{0, 1, \ldots, k\}\} \\
\delta'([q, i], a, Z) &= ([p, i], \gamma) \Leftrightarrow \delta(q, a, Z) = (p, \gamma) \\
\Delta'([q, i]) &= [p, i + 1] \Leftrightarrow \Delta(q) = p \wedge i \in \{0, 1, \ldots, k - 1\} \\
q_0' &= [q_0, 0] \\
F' &= [q, i]\ q \in F \wedge i \in \{0, 1, \ldots, k\}
\end{aligned}$$

Spravili sme iba $k+1$ rovnakých kópií automatu A, pričom v rámci každej kópie sú možné iba δ-prechody a Δ-prechody vedú z i-tej do $i+1$-vej kópie. Každá kópia osve je klasickým DPDA. Je zrejmé, že $L(A) = L(A')$. Cieľom je aplikovať techniku z [10] vhodne na každú z kópií.

Stručne zhrňme spomínanú techniku. Ide v nej o to, predísť všetkým trom nepriaznivým prípadom menovaným vyššie. Prvé dva sa dajú vyriešiť ľahko. V našom prípade, ak by sa mal vymazať symbol Z_0, tak stačí dodefinovať prechod do nového mŕtveho stavu (neakceptačného), v ktorom sa nebude pracovať so zásobníkom, iba čítať vstup. Čo sa týka toho, že niektoré možnosti môžu mať nedefinované δ-prechody, sa dá tiež jednoducho vyriešiť dodefinovaním týchto prechodov (do mŕtveho stavu). A nakoniec jediná zaujímavá možnosť je detekovanie zacyklenia na ε. V tomto prípade sa pre daný DPDA M dá nájsť konštanta (závislá iba od M), ktorá ohraničuje počet ε-krokov tak, že ak automat spraví viac krokov, ako je táto konštanta, tak sa zacyklil. Preto vieme túto detekciu realizovať iba pamätaním si prídavnej informácie v stave. Jedná sa o niekoľko kombinatorických argumentov, ktorými sa dajú elegantne tieto problémy vyriešiť.

Teraz si už len treba uvedomiť tieto fakty: V každej kópií vieme dodefinovať nový mŕtvy neakceptačný stav a nedefinované prechody nasmerovat doň. Tiež vieme ošetriť zaseknutie sa na prázdnom zásobníku. Aplikáciou predstavenej techniky (počítanie počtu ε-krokov a detekovanie zacyklenia) vieme vyriešiť aj tretí zo spomínaných problémov. Tu len treba podotknúť, že pri prechode z jednej kópie do inej (pri flipe) treba počítadlo ε-krokov vynulovať (resp. to tak formálne definovať). Formálnu konštrukciu robiť nebudeme. Naozaj najdôležitejší je fakt, že dôvody nedočítania vstupu sú rovnaké, ako pri obyčajnom DPDA a že ich môžeme aj rovnakým spôsobom vyriešiť (v rámci každej kópie, ktorá je osve štandardným DPDA).

□

Veta 5.6. *Trieda jazykov* $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ je uzavretá na komplement.

Dôkaz. Opäť obdobne ako pri DPDA. Majme k-flipový DFPDA A akceptujúci $L(A)$. Vezmime z neho skonštruovaný A' podľa lemy 5.5. Chceme z neho skonštruovať B, akceptujúci $L(A)^C$. Opäť postretávame rovnaký problém, ako pri DPDA. To čo potrebujeme je nasledovné: po prečítaní slova zistiť, či A' akceptoval a ak áno, tak B zamietne a ak nie, tak B akceptuje. Z lemy 5.5 vieme, že z ľubovoľnej konfigurácie sa po konečnom počte δ-krokov na ε alebo Δ-krokov dostane A' do konfigurácie, z ktorej je možné spraviť krok na symbol zo Σ (to jest A' sa nezasekne a nezacyklí na ε). Preto pre každé w nutne bude existovať nejaký výpočet $(q_0, w, Z_0) \vdash^*_{A'} (p, \varepsilon, Z_0\gamma)$. Problémom ale je, že z konfigurácie $(p, \varepsilon, Z_0\gamma)$ môžu ešte existovať δ-kroky na ε, alebo Δ-kroky, ktoré môžu viesť do nejakého akceptačného stavu. Toto potrebujeme detekovať. Potrebujeme zabezpečiť, aby po každom symbole z abecedy Σ sme overili, či sa na takýto druh krokov vieme dostať do akceptačného stavu a akceptovať len vtedy, ak tomu tak nie je (lebo chceme komplement jazyka $L(A)$). Toto si opäť vieme vhodne pamätať a detekovať v stave. Zase sa jedná o známu konštrukciu, ktorá sa dá priamočiaro použiť. Vhodne je opísaná v [8].

□

Poznámka 5.7. Iný, štandardnejší dôkaz vety 5.3 by priamo vyplynul z práve dokázaných faktov. Máme totiž uzavretosť na komplement a neuzavretosť na prienik. Preto hneď z De Morganových zákonov máme, že $(L(A)^C \cup L(B)^C)^C = L(A) \cap L(B)$. Potom, kebyže pripustíme uzavretosť na zjednotenie, tak sa okamžite dostaneme do sporu s tvrdením 5.4. Naschvál sme ale robili dôkazy v tomto poradí, lebo vo vete 5.3 sme ukázali aj konkrétny jazyk, ktorý nepatrí do $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ pre žiadne k. Na toto sme použili rovnakú dôkazovú techniku, aká sa použila pri klasických DPDA. Práve táto konštrukcia z vety 5.3 dáva zaujímavý pohľad na porovnanie: nedeterminizmus bez možnosti flipov *verzus* determinizmus s konečným počtom flipov. Z konštrukcie vyplýva, že žiadny konečný počet obratov v kombinácií s de-

terminizmom nestačí na odsimulovanie jediného nedeterministického kroku štandardného zásobníkového automatu. Navyše, myšlienku konštrukcie v ďalšom ešte podstatne využijeme.

Tvrdenie 5.8. *Trieda jazykov* $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ nie je uzavretá na homomorfizmus (ani nevymazávajúci).

Dôkaz. Stačí opäť zobrať

$$\begin{aligned} L_1 &= \{a^n b^n \mid n \geq 1\}, \\ L_2 &= \left\{a^n b^{2n} \mid n \geq 1\right\}. \end{aligned}$$

Jazyk $cL_1 \cup dL_2$ je z $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ pre každé k. K tomu vezmime homomorfizmus h taký, že $h(c) = h(d) = h(a) = a$, $h(b) = b$. Preto $h(cL_1 \cup dL_2) = a(L_1 \cup L_2)$. Kebyže je tento posledný jazyk z triedy $\mathscr{L}(\mathrm{DFPDA}(\leq k))$, potom by aj jazyk $L_1 \cup L_2$ bol z tejto triedy. To je ale spor s faktami uvedenými na začiatku dôkazu vety 5.3.

□

Tvrdenie 5.9. *Trieda jazykov* $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ je uzavretá na inverzný homomorfizmus.

Dôkaz. Opäť jednoduché. Použijeme štandarnú ideu, ako pre obyčajné deterministické zásobníkové automaty. Vstup zobrazujeme po jednotlivých symboloch homomorfizmom a ukladáme obrazy do konečného buffra. Na týchto symboloch simulujeme pôvodný automat.

□

Tvrdenie 5.10. *Trieda jazykov* $\mathscr{L}(\mathrm{DFPDA}(\leq k))$, pre $k \geq 1$ nie je uzavretá na zreťazenie ani (kladnú) iteráciu.

Dôkaz. Triviálne použitie vety 5.1. Stačí zobrať jazyk $J_k.J_k$, resp. J_k^+. Potom spraviť prienik s regulárnym jazykom $\{\Sigma^*\#\Sigma^*\}^{k+1}$, kde $\Sigma = \{a, b\}$. V oboch

prípadoch dostaneme L_{k+1}. Ak by bola trieda $\mathscr{L}(\text{DFPDA}(\leq k))$ uzavretá na zreťazenie alebo iteráciu, dostali by sme spor s vetou 5.1.

□

Veta 5.11. *Trieda jazykov* $\mathscr{L}(\text{DFPDA}(\leq k))$, pre $k \geq 0$ nie je uzavretá na reverz.

Dôkaz. Aplikujeme vhodne techniku z dôkazu vety 5.3. Postupujeme sporom, veľmi podobne ako v uvedenom dôkaze. Nech je teda trieda jazykov $\mathscr{L}(\text{DFPDA}(\leq k))$ uzavretá na reverz. Nech

$$\begin{aligned} L &= \#\{b^{2n}a^n \mid n \geq 1\} \cup \{b^n a^n \mid n \geq 1\}, \\ L^R &= \{a^n b^{2n} \mid n \geq 1\}\# \cup \{a^n b^n \mid n \geq 1\}. \end{aligned}$$

Jazyk L sa dá akceptovať nejakým DPDA, čiže patrí do $\mathscr{L}(\text{DFPDA}(\leq k))$ pre ľubovoľné k. Z predpokladu vyplýva, že aj L^R je z triedy $\mathscr{L}(\text{DFPDA}(\leq k))$ pre nejaké k. Nech je L^R akceptovaný nejakým DFPDA automatom A. Potom spomenutým postupom z dôkazu vety 5.3 vieme z tohto automatu skonštruovať iný, nedeterministický a najviac $2k$ flipový automat B, ktorý bude akceptovať jazyk L_B, kde

$$L_B = \{a^n b^n \mid n \geq 1\} \cup \{a^n b^{2n} \mid n \geq 1\}\# \cup \{a^n b^n c^n \mid n \geq 1\}\#.$$

Z uzáverových vlastností triedy NFPDA($\leq 2k$) potom vyplynie, že aj L_{abc} je z tejto triedy a dostaneme spor s vetou 3.5.

□

5.2 Kvocienty

Definícia 5.12. Majme jazyky L_1 a L_2. Pravý kvocient jazyka L_1 vzhľadom na jazyk L_2, je jazyk:

$$L_1/L_2 = \{x \mid (\exists w \in L_2) : xw \in L_1\}$$

Ľavý kvocient jazyka L_1 vzhľadom na jazyk L_2 je jazyk:

$$L_2 \backslash L_1 = \{x \mid (\exists w \in L_2) : wx \in L_1\}$$

Tvrdenie 5.13. *Trieda jazykov* $\mathscr{L}(\mathrm{DFPDA}(\leq k))$, pre ľubovoľné k, nie je uzavretá na ľavý kvocient s regulárnym jazykom. Presnejšie ak máme daný regulárny jazyk R a $L \in \mathscr{L}(\mathrm{DFPDA}(\leq k))$, tak potom jazyk $R \backslash L$ nemusí patriť do $\mathscr{L}(\mathrm{DFPDA}(\leq k))$.

Dôkaz. Stačí zobrať jazyky

$$\begin{aligned} L &= c\left\{a^n b^{2n} \mid n \geq 1\right\} \cup d.\left\{a^n b^n \mid n \geq 1\right\}, \\ R &= \{c, d\}. \end{aligned}$$

Je hneď z definície vidno, že jazyk $R \setminus L$ je náš notoricky známy jazyk, ktorý nepatrí do $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ na základe dôkazu vety 5.3.

□

Venujme sa teraz otázke, či je trieda $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ uzavretá na pravý kvocient s regulárnym jazykom. Hneď na úvod musíme uviesť, že na túto otázku, bohužiaľ, nepoznáme odpoveď. Jedná sa snáď o poslednú z bežných uzáverových vlastností deterministického variantu, kde túto odpoveď nepoznáme. Napriek tomu uveďme aspoň krátky komentár.

Je známe, že $\mathscr{L}(\mathrm{DPDA})$ nie je uzavretá na reverz ani ľavý kvocient s regulárnym jazykom. Pomerne zaujímavým výsledkom je, že sú uzavreté na pravý kvocient s reg. jazykom. Dôkaz sa opiera o konštrukciu tzv. predicting machine k danému DPDA, viď. napríklad [8]. Snažili sme sa túto techniku zovšeobecniť a použiť na DFPDA, ale narážali sme na rôzne problémy. Napriek tomu sa nám zdá, že by sa táto idea dala použiť na veľmi špecifický prípad: jednoflipové automaty a regulárny jazyk by musel byť Σ^*. Už len takéto náznaky a prvotná intuícia nás vedú k domnienke, že trieda $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ možno je uzavretá na pravý kvocient s regulárnym jazykom.

Iný prístup, ktorý by možno mohol vniesť trocha svetla do problematiky je študovať najmenšiu triedu jazykov obsahujúcu $\mathscr{L}(\text{DFPDA}(\leq k))$ a uzavretú na pravý kvocient s regulárnym jazykom.

Ako príklad uveďme nasledovnú úvahu: Ak sa pozrieme na najmenšiu triedu jazykov obsahujúcu $\mathscr{L}(\text{DFPDA}(\leq k))$ a uzavretú na nevymazávajúci homomorfizmus, tak akú triedu jazykov dostaneme? Označme si ju $\mathscr{L}$. Je zrejmé, že $\mathscr{L} \subseteq \mathscr{L}(\text{NFPDA}(\leq k))$, lebo $\mathscr{L}(\text{NFPDA}(\leq k))$ je na túto operáciu uzavretá. Ukážme ale, že platí aj opačná inklúzia. Vezmime jazyk L z triedy $\mathscr{L}(\text{NFPDA}(\leq k))$ a automat A, ktorý ho rozpoznáva. Podstatou je, že nedeterministické kroky (rozhodnutia) vieme kódovať do symbolov. Presnejšie, pre každú trojicu (q, a, Z) máme len konečne veľa možností na krok ktorý A spraví, lebo množina $\delta(q, a, Z)$ je konečná. Očíslujeme jej prvky číslami 1, $\ldots$, n a zavedieme symboly $[a, i]$, pre $i \in \{1, \ldots, n\}$. Takto budeme vedieť akceptovať jazyk L' (pomocou deterministického automatu) nad takouto rozšírenou abecedou. Jednoduchou projekciou (ktorá je homomorfizmom) na prvé súradnice potom dostaneme jazyk L. Spomínaný homomorfizmus nemusí byť ani vymazávací, lebo (ako sme ukázali v kapitole 4) môžeme predpokladať, že automat A pre jazyk L nepoužíva kroky na ε. Preto je ľubovoľný jazyk z $\mathscr{L}(\text{NFPDA}(\leq k))$ v triede jazykov $\mathscr{L}$ a keďže $\mathscr{L} \subseteq \mathscr{L}(\text{NFPDA}(\leq k))$ je jednoduchým dôsledkom definícií, tak dostávame, že $\mathscr{L} = \mathscr{L}(\text{NFPDA}(\leq k))$. Tohto okamžitým dôsledkom je, že $\mathscr{L}(\text{DFPDA}(\leq k))$ nemôže byť uzavretá na (nevymazávajúci) homomorfizmus, lebo platí, že $\mathscr{L}(\text{DFPDA}(\leq k)) \subset \mathscr{L}(\text{NFPDA}(\leq k))$.

Práve z tohto dôvodu je zaujímavé pozrieť sa na najmenšiu triedu obsahujúcu $\mathscr{L}(\text{DFPDA}(\leq k))$ a uzavretú na nejakú operáciu (v našom prípade pravý kvocient s regulárnym jazykom), pričom na túto operáciu je trieda $\mathscr{L}(\text{NFPDA}(\leq k))$ uzavretá. Ak by sa táto trieda už rovnala $\mathscr{L}(\text{NFPDA}(\leq k))$, tak z toho vyplýva, že $\mathscr{L}(\text{DFPDA}(\leq k))$ na túto operáciu uzavretá nie je.

Samozrejme tento postup nám nemusí nič povedať. Toto nastane, ak takáto trieda generovaná $\mathscr{L}(\text{DFPDA}(\leq k))$ a uzavretosťou na nejakú operáciu nie je rovná $\mathscr{L}(\text{NFPDA}(\leq k))$. Ako príklad vezmime triedu $\mathscr{L}$, definovanú opäť ako najmenšiu triedu jazykov obsahujúcu $\mathscr{L}(\text{DFPDA}(\leq k))$ a uzavretú na reverz. Výsledok z [6] nám hovorí, že jazyk $L = \{ww \mid w \in \{a, b\}^+\}$ nepatrí do $\mathscr{L}(\text{DFPDA}(\leq k))$ pre žiadne k. Keby ale jazyk L patril do $\mathscr{L}$, tak potom by musel byť aj v $\mathscr{L}(\text{DFPDA}(\leq k))$, pre nejaké k (lebo $L^R = L$) a toto by bol spor. Na druhej strane jazyk L patrí do $\mathscr{L}(\text{NFPDA}(\leq k))$, pre $k \geq 1$. Preto sa ale triedy $\mathscr{L}$ a $\mathscr{L}(\text{NFPDA}(\leq k))$ nemôžu rovnať. Toto je práve ten nepoužiteľný prípad, lebo nám nehovorí nič o uzavretosti triedy $\mathscr{L}(\text{DFPDA}(\leq k))$ na reverz. Túto otázku sme ale v predošlom už úspešne zodpovedali pomocou iných prostriedkov.

V tejto kapitole sme zodpovedali väčšinu otázok týkajúcich sa uzáverových vlastností deterministického modelu. Ako hlavný nedostatok vidíme nevyriešenie otázky ohľadom pravého kvocientu s regulárnym jazykom. Pri tejto otázke sme sa pristavili iba istými špekuláciami a domniekami.

Kapitola 6

Záver

V práci sme sa venovali modelu flipového zásobníkového automatu. Pojednali sme o nedeterministickom i deterministickom variante modelu. Hlavným cieľom bolo pokúsiť sa priniesť nové nápady, techniky a výsledky týkajúce sa tohto výpočtového modelu. Môžeme prehlásiť, že sa nám to z veľkej miery podarilo. Hlavným výsledkom práce je vyriešenie otvoreného problému týkajúceho sa sily epsilon krokov v nedeterministickom modeli. Ďalšími nápadmi a aplikáciami známych faktov sme sa úspešne pokúsili o zodpovedanie otázok uzavretosti deterministického modelu na niektoré uzáverové vlastnosti. Týmto sme značnou mierou priespeli k zodpovedaniu ďalšej otvorenej otázky.

Veríme, že sme aspoň malou mierou prispeli k už známym poznatkom a tak isto dúfame, že počet známych faktov a zodpovedaných otázok bude narastať. Preto pokladáme za dôležité uviesť otvorené problémy, na ktoré sme pri písaní práce narazili, bez ohľadu na to, či sme sa neúspešne pokúšali o ich riešenie, alebo sme ich len formulovali.

- Je trieda $\mathscr{L}(\mathrm{DFPDA}(\leq k))$ uzavretá na pravý kvocient s regulárnym jazykom? Istý komentár k tomuto problému sme podali na konci kapitoly 5. Domnievame sa že áno, aj keď argumentov za je málo ...

- Akú triedu jazykov dostaneme, ak budeme predpokladať, že do nej patria jazyky z $\mathscr{L}(\text{DFPDA}(\leq k))$ a je uzavretá na nejakú uzáverovú vlastnosť? Toto je druh otázok hojne skúmaných (pre iné triedy jazykov a uzáverové vlastnosti a s omnoho väčšou mierou abstrakcie) v monografii [2]. Motivácia súvisí s predošlou otázkou a komentármi na konci kapitoly 5.

- Čo sa stane, ak aplikujeme techniku Flip-Pushdown Input-Reversal na deterministický najviac k-flipový automat? Dostaneme najviac $k-1$-flipový deterministický jazyk? Technika sa využívala zatiaľ len na nedeterministické automaty. Dá sa rozšíriť aj na deterministické? Alebo existuje deteterministický jazyk, na ktorý keď techniku aplikujeme dostaneme jazyk ktorý deterministický nie je?

- Vieme ohraničiť počet stavov nedeterministického k-flipového automatu bez ovplyvnenia výpočtovej sily? O klasických zásobníkových automatoch (PDA) je známe, [8, 10], že k ľubovoľnému PDA A vieme skonštruovať iný, akceptujúci rovnaký jazyk, pričom tento má jeden stav (a akceptuje prázdnou pamäťou). Z výsledkov v kapitole 4 vyplýva, že pre k-flipový nedeterministický automat stačí použiť najviac $O(k)$ stavov. Nedá sa tento odhad stlačiť nižšie?

- Aké vlastnosti má takzvaný zásobníkový jazyk L_z?

$$L_z = \{\alpha \in \Gamma^* \mid (\exists w \in \Sigma^*)(\exists q \in Q)\ (q_0, w, \alpha) \vdash^* (q, \varepsilon, \varepsilon)\}$$

  L_z je jazyk slov zo zásobníkovej abecedy, ktoré sú „vymazateľné“. Je regulárny? Pri klasických zásobníkových automatoch sa vie, že je regulárny, [7].

- Ktoré vlastnosti sú pre flipové zásobníkové automaty rozhodnuteľné a ktoré nie sú? Táto otázka bola formulovaná už v článku [6]. Tu je zaujímavé pristaviť sa pri vlastnostiach, pre ktoré má vôbec zmysel si túto

otázku klásť.

Pre nedeterministické flipové automaty je zmysluplné pýtať sa iba na problém príslušnosti a prázdnosti (lebo ostatné z bežných vlastností, pri ktorých nás zaujíma ich rozhodnuteľnosť sú nerozhodnuteľné už pre PDA). Ale z techniky Flip-Pushdown Input-Reversal vyplýva, že obe tieto vlastnosti sú rozhodnuteľné, [5, 6].

Pri deterministických flipových zásobníkových automatoch je situácia omnoho menej zrejmá. Vlastnosť $L = R$, kde L je jazyk z triedy $\mathscr{L}(\text{DFPDA}(\leq k))$ zadaný pomocou automatu a R je zadaný pomocou konečného automatu, je rozhodnuteľná. Toto vyplýva z uzavretosti DFPDA na komplement a z rozhodnuteľnosti problému prázdnosti pre NFPDA. Vidno to z De Morgranových zákonov:

$$L = R \text{ vtedy a len vtedy keď } L' = \big((L^C \cap R) \cup (L \cap R)\big) = \emptyset.$$

Z uzáverových vlastností vyplýva, že jazyk L' je z $\mathscr{L}(\text{NFPDA}(\leq k))$, takže otázku prázdnosti tohto jazyka (a vďaka ekvivalencií aj otázku či $L = R$) vieme rozhodnúť algoritmicky. Dve vlastnosti sú ale zrejme omnoho zložitejšie: zistiť, či je daný jazyk (zadaný pomocou DFPDA) regulárny, resp. zistiť pre dva dané jazyky, či sa rovnajú. Oba tieto problémy sú pre DPDA rozhodnuteľné, [8]. Problém rovnosti bol ale dlhé roky otvorený a nevieme, či by sa technika použitá pri dôkaze dala nejako použiť pre DFPDA. Podobne ani pri probléme regularity nevieme, či a ako by sa dal adaptovať dôkaz spravený pre DPDA. Teoreticky je možné že by sa dôkazy dali bez razantnejšej zmeny použiť aj pre DFPDA. Toto ale nevieme posúdiť, lebo sme ani jeden z týchto dôkazov nijako podrobnejšie neštudovali.

Literatúra

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

[2] Seymour Ginsburg. *Algebraic and Automata Theoretic Properties of Formal Languages*. North-Holland, 1975.

[3] Sheila A. Greibach. A new normal-form theorem for context-free phrase structure grammars. *Journal of ACM*, 12:42–52, 1965.

[4] Markus Holzer and Martin Kutrib. Flip-pushdown automata: $k + 1$ pushdown reversals are better than k. Technical report, Institut für Informatik Universität Giessen, November 2002.

[5] Markus Holzer and Martin Kutrib. Flip-pushdown automata: $k+1$ pushdown reversals are better than k. *Lecture Notes in Computer Science*, 2719/2003:490–501, 2003.

[6] Markus Holzer and Martin Kutrib. Flip-pushdown automata: Nondeterminism is better than determinism. *Lecture Notes in Computer Science*, 2710/2003:361–372, 2003.

[7] Hendrik Jan Hoogeboom and Joost Engelfriet. Pushdown automata. In C. Martín-Vide, V. Mitrana, and G. Paun, editors, *Formal Languages and Applications*, pages 117–138. Springer, 2004.

[8] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[9] Rohit J. Parikh. On context-free languages. *Journal of the ACM*, 13:570–581, 1966.

[10] Branislav Rovan and Michal Forišek. Skriptá k prednáške z predmetu formálne jazyky a automaty, 2011. Aktuálna verzia je dostupná na http://foja.dcs.fmph.uniba.sk/materialy/skripta.pdf.

[11] Palash Sarkar. Pushdown automaton with the ability to flip its stack. *Electronic Colloquium on Computational Complexity*, Report No. 81, 2001.

[12] Martin Širáň. Zásobníkové automaty so schopnosťou obracať zásobník, 2004. Diplomová práca, FMFI UK Bratislava.