Univerzita Komenského
Fakulta Matematiky, Fyziky a Informatiky
Katedra Informatiky

Diplomová práca

Triedy jazykov definované pomocou obmedzení LBA

Autor: Peter Orolín
Diplomový vedúci: prof. RNDr. Branislav Rovan, PhD.

Bratislava 2008

Týmto čestne vyhlasujem, že som diplomovú prácu vypracoval samostatne, s odbornou pomocou školiteľa a s použitím literatúry uvedenej v práci.

Bratislava, Apríl 2008 Peter Orolín

Poďakovanie

Touto cestou sa chcem poďakovať vedúcemu diplomovej práce prof. RNDr. B. Rovanovi, PhD., že ma viedol pri vypracovaní tejto práce, za jeho cenné rady a pripomienky.

Abstrakt

Diplomová práca sa zaoberá skúmaním tried jazykov, ktoré sú definované pomocou deterministických a deterministických lineárne ohraničenéhých automatov, pričom vo výpočte obmedzujeme počet zmien obsahu políčka.

Klúčové slová: LBA, DLBA, zmena obsahu políčka, sweep automaty

Contents

1 Úvod

Cieľom mojej diplomovej práce je skúmať triedy jazykov definovaných pomocou LBA, pričom obmedzujeme počet prepisov(zmena obsahu) na políčkach alebo globálne na celom slove.
V nasledujúcom odseku je notácia nasledovná: Obmedzenie robiť prepisy na políčkach a ďalšie podobné miery zložitosti sa skúmali na jednopáskových (nielen) Turingových strojoch(X $\in$\{D,N\},N znamená nedeterministicky, D deterministicky):

1. XREV - počet zmien smerov pohybu hlavy
2. XCROSS – veľkosť prechodových postupností medzi j a j+1 políčkom
3. XRET – počet návštev na políčku po prvej zmene obsahu
4. XDUR – počet návštev na políčku pred poslednou zmenou obsahu

U nich sa na jednopáskových TS dosiahli výsledky (uvádzam samozrejme podľa mňa tie najzaujímavejšie):

- Nech n je dĺžka vstupného slova. Ak DCROSS(1) = DCROSS(f(n)), potom log(n) je asymptoticky rýchlejšie rastúca funkcia ako f(n)
- Jazyk $\{0^n1^n|$ n$\in$ N$\}\in$ DCROSS(log(n))
- $\mathcal{L}_{CF}$ = NRET(t), $\forall$ t $\in$ N,t > 1
- Trieda jazykov akceptovaná deterministickým dvojsmerným PDA je obsiahnutá v DREV(n^2 log(n))
- Trieda jazykov akceptovaná nedeterministickým dvojsmerným PDA je obsiahnutá v DREV(n^4)
- NCROSS(f(n)) = NSPACE(f(n)), ak f(n)je asymtoticky väčšie alebo rovné ako n.

V tejto práci sme ukázali, že LBA, ktoré majú obmedzenú možnosť robiť prepisy globálne, sú rovnako silné ako LBA, ktoré majú obmedzenú možnosť robiť prepisy na políčku, ak celkový počet prepisov je asymptoticky rovnaký. Uvedená konštrukcia sa dá použiť aj na dôkaz ekvivalencie, keď LBA majú na výpočet časové obmedzenie f(n)$\cdot$ n, tak LBA s obmedzením NCROSS(f(n)) sú rovnako silné.
Ďalej sme ukázali, že ak DLBA majú obmedzene robiť prepisy globálne, ak im ešte pridáme obmedzenie sweep, tak je to ich normálny tvar. Z toho

vyplýva tiež vzťah, že existuje normálny tvar pre LBA – konštantne krát zmení smer hlavy, tak potom musí nastať prepis. Teda pre triedy jazykov nad LBA platí: {počet prepisov f(n)}⊆DDRET(c·f(n)). Tento vzťah sa dá priamočiaro zovšeobecniť na jednopáskové TS, ak veľkosť používanej pásky zostane rovnaká.
Pri spomínanom dôkaze, že sweep je normálny tvar, sa na jeden prepis DLBA muselo urobiť 2·|K| prepisov v sweep DLBA. Naša konštrukcia simulovala výpočet medzi dvoma prepismi a nevyužívala možnosť, že by prepisy navzájom mohli súvisieť, teda ku každej novej časti výpočtu medzi prepismi pristupoval LBA B nezávisle na ostatných. Ukázali sme, že ľubovoľná iná konštrukcia(za spomínaného predpokladu nezávislosti dvoch prepisov pri LBA) by musela využiť aspoň 2·|K|-2 prepisov.

Aplikácia výskumu: uvažujme o LBA s konštantným prepisom políčok. Algoritmy, ktoré by mohli byť s touto vlastnosťou navrhnuté, by mali nasledovnú vlastnosť: dané políčko predstavuje časť pamäte, bude meniť obsah maximálne konštatne krát. To sa dá ľahko pamätať. A keby sme o nejakom súvislom počte políčok vedeli povedať, že ho už nebudeme prepisovať, mohli by sme ho nahradiť prechodovou funkciou dĺžky O(|K|) stavov, čo by sme mohli využiť napríklad pri swapovaní.

2 Definície

V nasledujúcej kapitole definujeme základné pojmy, s ktorými budeme pracovať. Model dvojsmerných konečných automatov využijeme v dôkaze, že sweep DLBA je normálny tvar DLBA, ak obmedzíme počet prepisov vo výpočte. Potom definujeme lineárne ohraničené automaty a na záver definujeme triedy jazykov, ktorých vlastností budeme skúmať nasledujúcich kapitolách.

Definícia 2.1 *Dvojsmerným nedeterministickým konečným automatom (2NKA) nazývame 5-ticu* (K,Σ,δ,q_0,F)*, kde* K *je konečná množina stavov,* Σ *je konečná abeceda vstupných symbolov(pričom* $¢,\$ \notin\Sigma$*),* $q_0\in K$ *je začiatočný stav,* $F\subseteq K$ *je množina akceptačných stavov a* $\delta : K\times(\Sigma\cup\{¢,\$\}) \longrightarrow 2^{K\times\{-1,0,1\}}$ *je prechodová funkcia, pričom platí:*

$\forall q \in K;\ \delta(q,¢)\subseteq K \times\{¢\}\times \{0,1\}$

$\forall q \in K;\ \delta(q,\$)\subseteq K \times\{\$\} \times \{-1,0\}$

Definícia 2.2 *Konfiguráciou 2NKA je trojica* $(q,¢\, w\$,i)$*, kde* $q \in K$ *je aktuálny stav,* $w \in \Sigma^*$ *je vstupné slovo a* $i \in \{0,\ldots,|w|+1\}$ *pozícia hlavy.*

Definícia 2.3 *Nech* w_i *je i-ty znak slova* w*, pričom definujeme* $w_0 = ¢$ *a* $w_{|w|+1} = \$$*. Potom krokom výpočtu 2NKA* A *nazývame binárnu reláciu* $\vdash_A$ *na množine konfigurácií definovanú nasledovne:*

$$(q,¢w\$,i)\vdash(p,w,i+j)\Longleftrightarrow (p,j)\in \delta(q,w_i)$$

Definícia 2.4 *Jazyk akceptovaný 2NKA* A *je množina*
$L(A) = \{w \mid \exists\ q_F \in F,\ (q_0,¢w\$,1)\vdash^* (q_F,¢w\$,|w|+1)\}$

Definícia 2.5 *Deterministickým dvojsmerným konečným automatom (ozn. 2DKA) je taký 2NKA, kde* $\forall q\in K,\ x \in \Sigma;\ |\delta(q,x)| \leq 1$.

Definícia 2.6 $\mathcal{R} = \{\ L \mid$ *existuje 2NKA* A *také, že* $L = L(A)\}$

Definícia 2.7 *Nedeterministickým lineárne ohraničeným automatom(LBA, linear-bounded automata) nazveme 6-ticu* $(K,\Sigma,\Gamma,\delta,q_0,F)$*, kde* K *je konečná množina stavov,* Σ *je konečná abeceda vstupných symbolov,* $\Gamma\supseteq\Sigma$ *je konečná abeceda pracovných symbolov (pričom* $¢,\$ \notin\Gamma$*),* $q_0\in K$ *je začiatočný stav,* $F\subseteq$

K je množina akceptačných stavov a $\delta : K \times (\Gamma \cup \{¢,\$\}) \longrightarrow 2^{K \times \Gamma \cup \{¢,\$\} \times \{-1,0,1\}}$ *je prechodová funkcia, pričom platí:*

$\forall q \in K;\ \delta(q,¢) \subseteq K \times \{¢\} \times \{0,1\}$

$\forall q \in K;\ \delta(q,\$) \subseteq K \times \{\$\} \times \{-1,0\}$

Definícia 2.8 *Konfiguráciou LBA je trojica* $(q,¢w\$,i)$*, kde* $q \in K$ *je aktuálny stav,* $w \in \Gamma^*$ *aktuálny obsah pásky a* $i \in \{0,\ldots,|w|+1\}$ *pozícia hlavy.*

Definícia 2.9 *Nech* w_i *je i-ty znak slova w, pričom definujeme* $w_0 = ¢$ *a* $w_{|w|+1} = \$$. *Potom krokom výpočtu LBA A nazývame binárnu reláciu* $\vdash_A$ *na množine konfigurácií definovanú nasledovne:*

$$(q,w_0w_1\ldots w_{|w|+1},i) \vdash (p,w_0w_1\ldots w_{i-1}xw_{i+1}\ldots w_{|w|+1},i+j) \Longleftrightarrow (p,x,j) \in \delta(q,w_i)$$

Definícia 2.10 *Jazyk akceptovaný LBA A je množina*
$L(A) = \{w \mid \exists\, q_F \in F, x \in \Gamma^*, i \in N;\ (q_0,¢w\$,1) \vdash^* (q_F,¢x\$,i)\}$

Definícia 2.11 *Deterministickým LBA (ozn. DLBA) je taký LBA, kde* $\forall q \in K, x \in \Gamma;\ |\delta(q,x)| \leq 1$.

Definícia 2.12 *Sweep (D)LBA je taký (D)LBA, ktorý môže smer pohybu hlavy meniť len na ¢ alebo na \$.*

Definícia 2.13 $\mathcal{L}_{ECS} = \{ L \mid$ *existuje LBA A také, že* $L = L(A)\}$

Definícia 2.14 *Hovoríme, že počas výpočtu na Turingovom stroji alebo LBA nastal prepis, ak*
$(q,w_0w_1\ldots w_{i-1}xw_{i+1}\ldots w_{|w|+1},i) \vdash (p,w_0w_1\ldots w_{i-1}yw_{i+1}\ldots w_{|w|+1},i+j)$*, pričom* $x \neq y$.

Poznámka 2.15 *Predošlá definícia teda hovorí, že nastal prepis, ak sa zmení obsah políčka.*

Definícia 2.16 *Pre danú funkciu g(n) označujeme* $\mathcal{O}(g(n))$ *množinu funkcií*

$$\mathcal{O}(g(n)) = \{ f(n) \mid \exists\, c > 0,\ n_0 > 0, \text{ také že } 0 \leq f(n) \leq c \cdot g(n)\ \forall n \geq n_0\}$$

Definícia 2.17 $\mathcal{L}_{ecss,f(n),word}$=$\{L \mid \exists$ *LBA A, pre ktoré platí:* $\forall\ w \in L(A)$ *počet prepisov počas výpočtu na w je menší rovný ako O(f(n)) a zároveň L(A)=L* $\}$

Definícia 2.18 $\mathcal{L}_{ecss,f(n),square}$=$\{L \mid \exists$ *LBA A, pre ktoré platí:* $\forall\ w \in L(A)$ *počet prepisov na ľubovoľnom políčku na w je menší rovný ako O(f(n)) a zároveň L(A)=L* $\}$

Definícia 2.19 $\mathcal{L}_{decss,f(n),word}$=$\{L \mid \exists$ *DLBA A, pre ktoré platí:* $\forall\ w \in L(A)$ *počet prepisov počas výpočtu na w je menší rovný ako O(f(n)) a zároveň L(A)=L* $\}$

Definícia 2.20 $\mathcal{L}_{decss,f(n),square}$=$\{L \mid \exists$ *DLBA A, pre ktoré platí:* $\forall\ w \in$ *L(A) počet prepisov na ľubovoľnom políčku na w je menší rovný ako O(f(n)) a zároveň L(A)=L* $\}$

Poznámka 2.21 *V predchádzajúcich definíciach uvažujeme teda o najslabšej definicií– akceptácie z troch základných prístupov. Ostatné dve sú :*

(silná def.) *Každý výpočet na vstupnom slove w dĺžky n použije najviac f(n) prepisov na jednom políčku(celom slove)*

(stredná def.) *Každý výpočet na každom vstupnom slove w∈L(A) dĺžky n použije najviac f(n) prepisov na jednom políčku(celom slove)*

Neskôr ukážeme, že v nedeterministickom prípade sú definície ekvivalenté pre ,,slušné" f(n).

Poznámka k označeniu premennej n k diplomovej práci 2.22 *Často vo vetách používame premennú n. Ak nie je uvedené inak, máme na mysli, dĺžku vstupného slova.*

3 Základné výsledky

V nasledujúcej kapitole sa zaoberáme základnými vlastnosťami tried jazykov $\mathcal{L}_{ecss,f(n),square}$ a $\mathcal{L}_{decss,f(n),square}$. V časti 3.1 definujeme normálne tvary DLBA a LBA, ktoré neskôr využijeme, v časti 3.2 uvedieme časové obmedzenia daných tried jazykov, v časti 3.3 uzáverové vlastnosti a nakoniec v časti 3.4 porovnanie s triedou bezkontextových jazykov. Všetky uvedené vlastnosti tried $\mathcal{L}_{ecss,f(n),square}$ má aj $\mathcal{L}_{ecss,f(n)\cdot n,word}$ a podobne, uvedené vlastnosti $\mathcal{L}_{decss,f(n),square}$ a má aj $\mathcal{L}_{decss,f(n)\cdot n,word}$.

3.1 Normálne tvary LBA a DLBA

V tejto časti definujeme a dokážeme lemy o normálnych tvaroch LBA a DLBA, ktoré neskôr využijeme v ďalších častiach práce.

Lema 3.1.1 *Ku každému LBA A existuje sweep LBA B taký, že urobí maximálne $\mathcal{O}(n)$ krokov bez prepisu políčka.*

Dôkaz:
Neformálne: Uvažujme o výpočte LBA A, pričom nás zaujíma výpočet medzi prepismi symbolov. Predpokladajme, že v každom kroku výpočtu LBA A pohne hlavou, keď nenastal prepis. Ľahko vidieť, že ide o normálny tvar.
Základná myšlienka dôkazu: zaujíma nás najkratší výpočet na páske, ak sa neprepisuje symbol. Pri ňom musí platiť, že hlava môže byť na políčku maximálne |K| krát, pričom K je množina stavov. Definujeme prechodovú postupnosť. Je to postupnosť usporiadaných dvojíc (q,s) pričom q∈K, s∈{-1,1}. Ak s= -1, tak automat políčko pri stave q opúšťa zľavo, ak s = 1, tak zpravo. Pri najkratšom výpočte musí platiť, že na danom políčku nebola hlava s tým istým stavom dvakrát, preto daná prechodová postupnosť je konečná, a môžeme si ju pamätať v stave(je to rádovo $2^{|K|} \cdot |K|!$ stavov). Predpokladajme, že LBA B urobil m prepisov. Posledný z nich označil a' (LBA A by ho prepísalo na a), potom sa vráti bez prepisu na začiatok, pričom si pamätá stav, v ktorom bol LBA A, keď nastal prepis(pričom ak smer hlavy nebol predtým smerom na cent, tak pôjde najprv na \$, kde zmení smer, a vráti sa na cent),nedeterministicky si tipne prechodovú postupnosť, prejde na ďalší symbol, tiež si tipne prechodovú postupnosť, overí, či navzájom postupnosti nadväzujú, ak áno, tak takýmto spôsobom pokračuje ďalej, inak sa zasekne. Ak si nedeterministicky tipne, že sa na konci prechodovej postupnosti na nejakom políčku spraví prepis, (ak LBA A prepísal napríklad na b), tak LBA B ho prepíše na b"(predpokladajme, že b" LBA A nepoužíva).

Takto overovaním prechodovej postupnosti pokračuje ďalej, pričom ak narazí na políčko a', tak ho prepíše na a a overí, či na začiatku prechodovej postupnosti začína od stavu, v ktorom bol LBA A po prepise tohto políčka. Keď sa LBA B podarí prejsť celú pásku úspešne, tak sa vráti na políčko, ktoré LBA A, mal prepísať nakoniec, teda b" prepíše ho na b, a túto konštrukciu opakujeme medzi ďalšími prepismi.
Na celú túto konštrukciu potrebujeme celkovo O(n) krokov (hlavou prejdeme na začiatok, potom na koniec pásky a potom na prepisované políčko), teda naše tvrdenie platí. □

Lema 3.1.2 *Ku každému DLBA A existuje DLBA B také, že urobí maximálne $\mathcal{O}(n)$ krokov bez prepisu políčka.*

Dôkaz:
Dá sa urobiť jednoduchá úvaha, že medzi prepismi môže byť hlava na danom políčku maximálne |K| krát, kde K je počet stavov, inak by sa výpočet zacyklil. Z tejto úvahy vyplýva, keďže počet políčok je |w|, tak najdlhší možný výpočet medzi prepismi (po zarátaní ¢ a $) by bol $(|w|+2) \cdot |K|$. Teda musíme zostrojiť DLBA B, ktoré sa nemôže medzi prepismi zacykliť.
Konštrukcia sa podobá na konštrukciu z dôkazu predchádzajúcej lemy. Akurát nemôžeme nedeterministicky hádať prechodovú postupnosť, preto ich musíme v každom kroku vygenerovať všetky. Tiež si nemôžme nedeterministicky tipnúť, že na tomto políčku nastal prepis, len si zapamätáme, že pri tejto prechodovej postupnosti nastal, a teda túto informáciu si pamätáme, aj na daných nadväzujúcich prechodových postupnostiach. Prejdeme na $ a teda overíme, či existuje výpočet LBA A od daného prepisu k nejakému ďalšiemu. Ak nie, to znamená, že výpočet LBA A by sa buď zacyklil alebo zasekol, tak sa teda výpočet na LBA B tiež zasekne. Ak áno, tak sa LBA B vráti po políčko, kde nastal naposledy prepis, symbol na ňom prepíše z a' späť na a, a začne krok za krokom simulovať LBA A.□

Dôsledok 3.1.3 *Nech A je taký LBA, ktorý na prepis môže používať len konečne veľa políčok. Potom L(A)$\in \mathcal{R}$.*

Dôkaz: Z normálneho tvaru LBA z lemy 3.1 vyplýva, že na jeden prepis zmení LBA v danom normálnom tvare len konštantne veľa krát smer hlavy. Konečne veľa políčok na prepis, znamená, že počet rôznych obsahov pásky počas výpočtu je konečný. Z toho vyplýva, že celkový počet zmien na smer hlavy je tiež konštanta a v [1] je ukázané, že je to regulárny jazyk.□

3.2 Časové odhady

V tejto kapitole urobíme horné odhady pre zadefinované stroje s prepisom jedného políčka f(n) krát pomocou normálnych tvarov LBA a DLBA.

Veta o hornom časovom odhade

Veta 3.2.1 (o hornom časovom odhade) *Nech je daný* $\{D, N\}$*LBA A prepisujúci jedno políčko najviac f(n) krát, kde n je dĺžka vstupu. Ak* $f(n) \in \mathcal{O}(n^k)$*, pre* $k \in N$*. Potom* $L(A) \subseteq XTIME(n^{k+2})$*, pričom X patrí* $\{D, N\}$*.*

Dôkaz: Celkový počet prepisov f(|w|)· |w|. Využijúc časť 3.1, horný časový odhad je f(|w|)· |w| · O(|w|).□

Dôsledok 3.2.2 *Nech je daný* $\{D, N\}$*LBA A prepisujúci jedno políčko najviac f(n) krát, kde n je dĺžka vstupu. Ak* $f(n) = k$*, tak* $L(A) \in XTIME(n^2)$*, pričom X patrí D, N.*

Poznámka 3.2.3 *Prirodzená otázka je, či sa vo všeobecnosti dá zmenšiť horný odhad na čas. Odpoveď je záporná, lebo napríklad o jazyku* $L = \{ wcw^r | \ w \in \{a,b\}^* \}$ *sa dá ukázať, že dolný odhad je* $\Omega(DTIME(n^2))$ *a* $L \in \mathcal{L}_{decss,f(n),square}$*, kde f je konštantná funkcia, a teda horný odhad je tiež* $DTIME(n^2)$

3.3 Uzáverové vlastnosti

Homomorfizmus

Cieľom nasledujúcej podkapitoly je ukázať vzťah medzi uzavretosťou triedy na dĺžku zachovávajúci homomorfizmus pri triede $\mathcal{L}_{decss,f(n),square}$, kde f(n) je polynóm, a otvoreným problémom vzťahu P a NP.
Zaoberať sa vymazávajúcim homomorfizmom nemá veľký zmysel, lebo sa dá ľahko ukázať, že najmenšia trieda uzavretá na vymazávajúci homorfizmus a obsahujúca $\mathcal{L}_{decss,f(n),square}$, ak f(n) je konštantná funkcia, je $\mathcal{L}_{re}$.

Lema 3.3.1 *$\mathcal{L}_{ecss,f(n),square}$ je uzavretá na nevymazávajúci homomorfizmus, ak f(n) je polynóm.*

Definícia 3.3.2 *Nech SAT je množina všetkých splniteľných booleovských výrazov. SAT je jazyk nad abecedou $\{\vee,\wedge,not, \Rightarrow,\Leftrightarrow,(,),0,1\ \}$.*

Poznámka 3.3.3 *Je dobre známe, že SAT je NP-úplný jazyk.*

Lema 3.3.4 *Nech $L = \{\ p_1\ p_2 \ldots\ \ p_n\#\#w \mid w{\in}SAT$; obsahuje premenné $x_1,\ldots,x_n$; $p_1,..p_n \in \{0,1\}$; pri ohodnotení premenných $x_1{=}p_1;x_2{=}p_2;\ldots x_n{=}p_n$ dáva výsledok true$\}$. Potom jazyk $L \in \mathcal{L}_{decss,f(n),square}$, ak $f(n) = n$.*

Dôkaz:
Neformálne: namiesto premenných $x_1,\ldots,x_n$ v časti slova w∈SAT priradíme im prislúchajúce hodnoty. A nakoniec overíme, či je výraz pri danom ohodnotení premenných pravdivý. Konštrukcia sa dá uskutočniť, ak f(n) patrí do O(n), teda políčko sa môže prepísať O(n) krát. □

Lema 3.3.5 *Uvažujme dĺžku zachovávajúci homomorfizmus h, definovaný nasledovne: h(0)=y, h(1)=y, h(x)=x, ak x nepatrí do 0,1 (neformálne: správne ohodnotenie premenných, aby dal nám booleoský výraz true schováme).Nech $L_1 = h(L)$. Potom jazyk L_1 je NP-úplný.*

Dôkaz:
Neformálne -Stačí ukázať, že SAT sa dá polynomionálne transformavať na jazyk L1, čo je triviálne. □

Veta 3.3.6 *Ak trieda $\mathcal{L}_{decss,f(n),square}$, pričom f(n) je polynóm, je uzavretá na dĺžku zachovávajúci homomorfizmus, potom P = NP.*

Dôkaz: Jazyk L $\in \mathcal{L}_{decss,f(n),square}$. L_1 = h(L) je NP-úplný. Z toho vyplýva, že ak $\mathcal{L}_{decss,f(n),square}$, pričom f(n) je polynóm, je uzavretá na dĺžku zachovávajúci homomorfizmus, potom P = NP. □

Inverzný homomorfizmus

Veta 3.3.7 *Trieda dLecss je uzavretá na inverzný homomorfizmus.*

Dôkaz: Podobne, ako keď sa ukazuje, že LBA je uzavretá na inverzný homomorfizmus, s využitím viacstopovej pásky. Danú techniku možno nájsť v publikácií [2]. □

Komplement dLecss

Veta 3.3.8 *Trieda $\mathcal{L}_{decss,f(n),square}$, ak f(n) je konštanta, uzavretá na komplement.*

Dôkaz:
Majme DLBA A, ktorý môže políčko prepisovať c krát a akceptuje jazyk L(A). Predpokladajme, že je v normálnom tvare z lemy 3.2. Predpokladajme, že DLBA A upravíme tak, že na políčku si ukladá aj informáciu, koľkokrát sa dané políčko prepísalo a v prípade, že by ho chcel prepísať viac ako c -krát, výpočet sa zasekne. Keďže DLBA A je v normálnom tvare z lemy 3.2, nemôže sa zacykliť, pričom by nerobil prepisy, a keďže si kontroluje koľko prepisov spravil na políčku, tak sa nemôže zacykliť ani tak, že by robil prepisy. Teda po konečnom počte krát výpočet DLBA A skončí. Zostrojený DLBA B bude simulovať A, pričom po skončení výpočtu dané slovo akceptuje, ak DLBA A ho neakceptuje, a opačne.□

3.4 Porovnanie s Chomského hierarchiou

Intuitívne, ak f(n) je polynóm, tak $\mathcal{L}_{ecss,f(n),square}$ je ostro silnejšie než $\mathcal{L}_{CF}$. V nasledujúcom odseku to dokážeme.

Porovnanie tried jazykov $\mathcal{L}_{ecss,f(n),square}$ a $\mathcal{L}_{CF}$

Lema 3.4.1 *Jazyk $\{a^n b^n c^n | n \in N\} \in \mathcal{L}_{ecss,f(n),square}$, ak f(n)=c a zároveň nepatrí do $\mathcal{L}_{CF}$.*

Dôkaz: Prvá časť vety dokážeme priamočiarou konštrukciou, druhá časť napr. cez pumpovaciu lemu pre bezkontextové jazyky. Príslušný dôkaz možno nájsť v [2].□

Lema 3.4.2 *Ku každému bezkontextovému jazyku L existuje PDA B a konštanta k$\in$ N taká, že po najviac k krokoch prečíta vstupný symbol, pričom L = L(B).*

Myšlienka dôkazu:
Môžeme využiť Greibachovej normálny tvar pre bezkontextové jazyky a následne zostrojeniť štandardnou konštrukciou k nej prislúchajúce PDA B. Z vlastností Greibachovej normálneho tvaru a štandardnej konštrukcie je zrejmé, že PDA B má požadované vlastnosti. Bližšie informácie nájdeme napr. v [2].□

Lema 3.4.3 *$\mathcal{L}_{CF}$ je podmnožina $\mathcal{L}_{ecss,f(n),square}$, ak f(n)=c.*

Dôkaz:
Uvažujeme o PDA B =(K,Σ, Γ, δ_B,q_0,0,F), ktoré bude mať vlastnosť, že počet zmien obsahu zásobníka, a teda aj veľkosť zásobníka, je lineárny vzhľadom na veľkosť vstupného slova. Z predchádzajúcej lemy vyplýva, že je to normálny tvar PDA. Nech teda dĺžka slova je n a počet zmien v zásobníku maximálne k ·n. Zostrojím k nemu LBA A, pričom počet prepisov na políčku je maximálne 2 ·k + 2.
Použijeme viacstopovú pásku. Na prvej páske si zapíšeme vstupné slovo a ostatné využijeme na simuláciu zásobníka. Nech PDA B je v stave q. Odsimulejeme krok nasledovne: ak B načíta symbol zo vstupného slova, tak aj LBA A načíta vstupný symbol z viacstopovej pásky a označím ho symbolom, aby LBA A vedel, že ho čítal naposledy. Daný symbol si zapamätá v stave, nájde posledný symbol uložený v zásobníka, prečíta ho, a upraví zásobník tak, ako PDA B. Zmení stav podľa δ_B . Vráti sa k symbolu, ktoré je na rade na čítanie a opísaným postupom pokračuje ďalej, až kým výpočet PDA B neskončí a akceptuje slovo vtedy a len vtedy, keď ho B akceptuje.
Uvedená konštrukcia môže sposobiť, že políčka sa prepíšu viac než konštantne veľa krát(keď sa zväčšuje zásobník a potom zmenšuje, a opakujeme veľa krát). Na skončenie dôkazu stačí použiť lemu, ktorú dokážeme neskôr, a teda $\mathcal{L}_{decss,n,word} = \mathcal{L}_{decss,c,square}$.
Dá sa to samozrejme urobiť priamo, stopa, do ktorej už raz bol uložený symbol zo zásobníka a následne vyprázdnený, sa označí špeciálnym symbolom, aby sa už daná stopa neprepisovala a tým pádom bude jednotlivá stopa prepisovaná 2 krát a celkovo políčko 2·k + 1(1 kvôli načítaniu vstupného slova).

Iná možnosť:
Môžeme využiť článok [4] a ukázať, že najťažší bezkontextový jazyk definovaný v danom článku patrí do $\mathcal{L}_{ecss,f(n),square}$, ak f(n)je konštanta. A potom, že trieda jazykov akceptovaná $\mathcal{L}_{ecss,f(n),square}$, ak f(n)je konštanta, je uzavretá

na inverzný homomorfizmus, čo sme už dokázali vo Vete 3.2.7. □

Veta 3.4.4 *$\mathcal{L}_{ecss,f(n),square}$, ak f(n)je konštanta, je ostro silnejšie ako $\mathcal{L}_{CF}$.*

Dôkaz: Z predchádzajúcich liem.□

4 Porovnanie tried $\mathcal{L}_{ecss,f(n),square}$ a $\mathcal{L}_{ecss,f(n)\cdot n,word}$

4.1 Cieľ

V tejto kapitole ukážeme, že k ľubovolnému LBA A, ktorý môže počas výpočtu urobiť maximálne |w|· f(|w|) prepisov, existuje LBA B, pričom L(A) = L(B), ktorý na každom políčku urobí maximálne O(f(|w|)) prepisov.
V deterministikom prípade uvediem postačujúcu podmienku ekvivalencií týchto strojov.

4.2 Nedeterministická verzia

Lema 4.2.1 *Nech LBA A počas výpočtu na slove w urobí maximálne c·n prepisov, pričom $|w| = n$. Tvrdíme, že k nemu existuje LBA B, pričom L(A)=L(B), ktorý na každom políčku urobí konštantne veľa prepisov.*

Dôkaz:

Idea dôkazu:

predstavme si obrázok, na ktorom bude vstupné slovo w=$a_1a_2...a_n$, ale za každým symbolom a_i pridané symboly F_{prepis}. Ich počet za každým symbolom znamená, koľkokrát sa bude dané políčko pri slove w a pri výpočte LBA A prepisovať (viď obrázok)

$$a_1\overbrace{F_{prepis}...F_{prepis}}^{l_1}\ a_2\overbrace{F_{prepis}...F_{prepis}}^{l_2}...\overset{l_i}{a_i...F_{prepis}}...a_n\ \overbrace{F_{prepis}...F_{prepis}}^{l_n}$$

Figure 1: $a_1a_2...a_i...a_n$ je vstupné slovo. Za každý a_i sme pridali toľkokrát F_{prepis}, koľkokrát sa bude prepisovať.

Zrejme platí Σl_i = c·n(celkový počet prepisov). LBA B prepíše vstupné slovo w na slovo na obrázku nasledujúcim spôsobom: prvých c+1 symbolov(počítame medzi ne aj F_{prepis}) zapíše na 1. políčko, ďalších c+1 symbol na druhé políčko, atď.

Potom bude automat B simulovať LBA A nasledovne: LBA B používa viacstopovú pásku. V stave si pamätáme, v ktorej stope sa momentálne nachádza, a zároveň stav simulovaného LBA A. Načíta symbol na danej stope, ak sa hlava LBA A, pri danom symbole a stave pohlo doprava a prešlo

do iného stavu, ale nenastal prepis, tak aj náš LBA B sa pohne doprava a prejde do daného stavu, kde si uchová informáciu o novom stave LBA A, pričom postupuje nasledovne: ak nie je na poslednej stope daného políčka, tak sa presunie na nadchádzajúcu stopu doprava(v skutočnosti sa zmení len stav B, ktorý hovorí, na ktorej stope sa aktuálne nachádzame), pozrie sa, či sa tam je a$\in\Gamma_a$, ak nie, postupuje ďalej doprava po stopách, až kým nenájde hľadný symbol. Keď bude teda na poslednej stope a hľadaný symbol sa tam nevyskytuje, tak sa LBA B pohne na ďalšie políčko doprava a pokračuje v hľadaní (začne od prvej stopy). Ak symbol nájde, tak ho zase načítame, a zase zo stavu B si pamätáme, aký stav mal A, a podobným spôsobom postupujeme ďalej.
V prípade, že nastal prepis, tak za stopou, na ktorej sa daný symbol nachádzal (môže to byť aj prvá stopa nasledujúceho políčka, ak daný symbol bol na poslednej stope), musí nasledovať F_{prepis}. Ten teda prepíše na nový symbol, ktorý má nahradiť pôvodný. Na stope, kde bol pôvodný symbol, ktorý B načítal, napíšeme $F_{preskoc}$. V prípade, že sa F_{prepis} nenachádza za prepisovaným symbolom, tak sa výpočet zasekne.
Ako je vidieť, máme viacstopovú pásku, každá zo stôp bude maximálne dvakrát prepisovaná, jeden krát, keď nahradíme F_{prepis} na symbol z Γ_A, a druhýkrát zo symbolu z Γ_A na $F_{preskoc}$. Teda celkový počet prepisov je konštantne veľa.

Formálny dôkaz:

Máme dané LBA A = (K,Σ,Γ,δ,q_0,F), ktorý môže urobiť maximálne c$\cdot$|w| počet prepisov na slove w. K nemu skonštruujeme LBA B, ktorý bude môcť vo výpočte urobiť konštantný počet prepisov na políčku.

Štruktúra formálneho dôkazu je nasledovná: Zadefinujeme LBA B_1, pomocou ktorého vytvoríme konfiguráciu pre začiatok simulácie, potom LBA B_2, ktorý bude slúžiť na simuláciu LBA A. „Spojením" LBA B_1 a LBA B_2 vznikne LBA B.

Postup teda bude nasledovný:

1) vytvorenie konfigurácie, z ktorej budeme simulovať LBA A

2) simulácia LBA A pomocou LBA B z tejto konfigurácie

3) dôkaz, že počet prepisov na jednom políčku je ohraničený konštantou

4) záver

1)Vytvorenie konfigurácie, z ktorej budeme simulovať LBA A

Najprv zadefinujeme konfiguráciu, z ktorej budeme v časti b) simulovať LBA A, potom uvedieme definíciu B_1 a následne dokážem, že B_1 ukončí výpočet v spomínanej konfigurácií.

Definícia konfigurácie pre začiatok simulácie (až na detaily je to formalizácia obrázku 1 + stav DLBA B_1 a jeho pozícia hlavy):

Nech w = $a_1 a_2 \cdots a_n$ je vstupné slovo a l_1,...l_i,...,l_n je počet prepisov na i-tom políčku na w vo výpočte LBA A.
Konfigurácia má tvar (prvýkrát spomínané symboly sú zadefinované nižšie):

1. po výpočte LBA B_1 bude v stave q_{hotovo}
2. pozícia hlavy na páske je 1 (je na prvom políčku)
3. obsah pásky – páska obsahuje ,,viacstopové" symboly. Ich súvis s pôvodným vstupom w a počtom prepisov vo výpočte A na w vyjadríme pomocou homomorfizmu takto :
   - viactopové symboly, ktoré aplikujeme homomorfizmus: $\{F_{zjed}, \cdots, F_{zjed}|$ c+1-tica, pričom $F_{zjed} = \{F_{prepis}\} \cup \{a'|$ a$\in\Sigma\}$ }
   - h((F_{zjed},F_{zjed},...,F_{zjed}))= $b_1 b_2 \cdots b_{c+1}$
   - ak i-ty podsymbol $F_{zjed} = F_{prepis}$, tak $b_i = \epsilon$
   - ak i-ty podsymbol F_{zjed} = a'; a $\in\Sigma$, tak b_i = a
   - Nech h(w_a) = w
   - Nech l_i je počet podsymbolov F_{prepis} za a_i', teda za každým a_i', nech je l_i krát F_{prepis}. Nech $\sum_{0\leq i\leq n} l_i$ = c·n (počet prepisov).

Definícia LBA B_1: Nech $B_1 = (K_1, \Sigma_1, \Gamma_1, \delta_1, q_0, F_1)$

$K_1 = \{q_{zac}, q_{cent}, q_{prepisovanie}, q_{overit}, q_{naCent}, q_{prepisovanie}, q_{najdi\_prepis}, q_{na\_prve\_policko}, q_{na\_prve\_policko}, q_{hotovo}\} \cup \{q_{prepis\_a} |$ a $\in \Sigma_1\}$

$\Sigma_1 = \Sigma$

Γ_1 =$\{$(a,$F_{zjed}, \cdots, F_{zjed})|$ c+2-tica, pričom a$\in\Sigma$; $F_{zjed} = \{F_{prepis}\} \cup \Sigma_1$' } $\cup$ $\{(F_{zjed}, \cdots, F_{zjed})|$ c+1-tica, pričom $F_{zjed} = \{F_{prepis}\} \cup \Sigma_1$' } $\cup\ \Sigma_1$

$F_1 = \{q_{hotovo}\}$

Definícia δ_1:

1. $\delta_1(q_{zac}$, a$\in \Sigma_1) = (q_{zac}$, (a$\in \Sigma_1$,F_{prepis},...,F_{prepis}), 1)
2. $\delta_1(q_{zac}$, \$) = (q_{cent}, \$, -1)
3. $\delta_1(q_{cent}$, (a,F_{prepis},...,F_{prepis})) = (q_{cent}, (a,F_{prepis},...,F_{prepis}), -1)
4. $\delta_1(q_{cent}$, ¢) = ($q_{prepisovanie}$, ¢, 1)
   (1)-(4) Najprv každý symbol zo a$\in\Sigma_1$ LBA B_1 prepíše na tvar (a$\in \Sigma_1$,F_{prepis},...,F_{prepis}), a potom sa vráti na prvé políčko na páske v stave $q_{prepisovanie}$.

5. $\delta_1(q_{prepisovanie}$, (a,$F_{prepis}$,...,$F_{prepis}$)) = ($q_{prepis\_a}$, ($F_{prepis}$,...,$F_{prepis}$), 1)
6. $\delta_1(q_{prepis\_a}$, Γ_a) = (q_{prepis_a}, Γ_a, k) k$\in${-1,0,1}
7. $\delta_1(q_{prepis\_a}$,(F_{zjed},$\cdots$,F_{zjed}, F_{prepis},...F_{prepis})) = (q_{overit},(F_{zjed},...F_{zjed},a',...F_{prepis}),1)
   (5)-(7) LBA B_1 načíta prvý podsymbol z viacstopových symbolov (je na ňom uložený pôvodný symbol na políčku), pričom stopu ,,zruší", a niekde na páske zapíše tento podsymbol namiesto F_{prepis} (a sa zmení na a').

8. $\delta_1(q_{overit}$,(a,F_{prepis},...F_{prepis})) = (q_{overit}, (a,F_{prepis},...F_{prepis}), 1)
9. $\delta_1(q_{overit}$,(F_{prepis},...F_{prepis}) = (q_{overit}, (F_{prepis},...F_{prepis}), 1)
10. $\delta_1(q_{overit}$, \$) = (q_{naCent}, \$, -1)
    (9)-(10) Skontroluje sa, či stopy napravo neobsahujú iné b'$\in\Sigma_1$, ak áno, výpočet sa zasekne. Políčka sa zo (a$\in \Sigma_1$,F_{prepis},...,F_{prepis}) prepisujú na (F_{prepis},...,F_{prepis}) zľava doprava, a tým pádom sa a_{i-1}' symbol nemôže nachádzať viac napravo ako a_i (viď definíciu homomorfizmu).

11. $\delta_1(q_{naCent}$, Γ_a) = (q_{naCent}, Γ_a, -1)
12. $\delta_1(q_{naCent}$, ¢) = (q_{najdi_prep}, ¢, 1)
13. $\delta_1(q_{najdi\_prep}$,(a,F_{zjed},...F_{zjed})) = ($q_{prepisovanie}$,(a,F_{zjed},...F_{zjed}), 0)

14. $\delta_1(q_{najdi\_prep},(F_{zjed},...F_{zjed})) = (q_{najdi\_prep},(F_{zjed},...F_{zjed}), 1\ )$
   (11)-(14) Nájde prvé zľava políčko tvaru (a,F_{zjed},...F_{zjed}), a keď ho nájde, postup sa od (5) opakuje.

15. $\delta_1(q_{najdi\_prep}, \$) = (q_{na\_prve\_policko},\$, -1\ )$

16. $\delta_1(q_{na\_prve\_policko},\Gamma_a) = (q_{na\_prve\_policko},\Gamma_a, -1\ )$

17. $\delta_1(q_{na\_prve\_policko}, ¢) = (q_{hotovo},¢, 1\ )$
   (15)-(17) Keď nenájde políčko tvaru (a,F_{zjed},...F_{zjed}), to znamená, že všetky podsymboly a$\in\Sigma_1$ sa prepísali, LBA B_1 prejde na prvé políčko na páske a skončí výpočet v stave q_{hotovo}.

Dôkaz správnosti výpočtu LBA B_1 Dokážeme, že sa automat dostane do ľubovolnej a len takej konfigurácie, ktorá spĺňa podmienky definované vyššie.

1. Zrejme n-krát aplikáciou pravidla (1) dostaneme z počiatočnej konfigurácie (q_{zac},¢$a_1a_2...a_n$\$,1) konfiguráciu

   (q_{zac},¢ $b_1b_2...b_n$\$,n+1) (1,1),

   pričom b_i=(a_i,F_{prepis},F_{prepis},...,F_{prepis}) a počet F_{prepis} je na c.

2. Vykonaním pravidiel (2) a (3) sa z konfigurácie (1,1) LBA B dostane do konfigurácie

   ($q_{prepisovanie}$,¢$b_1b_2...b_n$\$,1) (2,1),

   b_i je definované vyššie. Obsah pásky sa nezmení, len hlava sa presunie na prvý symbol.

3. Majme konfiguráciu (2,1). Vykonaním zvyšných pravidiel dostaneme hľadanú konfiguráciu. V časti 3.1 ukážeme, že ľubovoľnú konfiguráciu spĺňajúcu podmienky vygenerujeme a v časti 3.2, že nevygenerujeme žiadnu konfiguráciu, ktorá podmienky nespĺňa.

3.1) Vykonaním nasledujúceho cyklu sa najprv a_1 ,,umiestni" správne, ďalší cyklus a_2, atď, až po a_{n-1}, pričom i-tom cykle hlava skonči na i+1 políčku. n-ty symbol po vykonaní príkazov (5)-(14) sa tiež správne ,,umiestni", ale potom narazí na \$ a vďaka (15)-(17) sa dostane na prvé políčko (čísla napravo znamenajú, ktorá časť δ_a sa použije, a k_i je počet $\{F_{prepis} \cup a_j\text{'}\}$ pred a_i', teda platí $k_i = \sum_{0 \leq j \leq i-1} \{F_{prepis} \cup a_j\}$').

(5) $(q_{prepisovanie}$,(a,F_{prepis},...F_{prepis}),0) $\vdash$ $(q_{prepis\_a}$, (F_{prepis},...F_{prepis}))

(6) $(q_{prepis\_a}$, Γ_a,-1)$\vdash^{i-1}$ $(q_{prepis\_a}$, Γ_a), hlava je teda na prvom symbole

(6) $(q_{prepis\_a}$, Γ_a,1)$\vdash^{(k_i div(n+1))}$ $(q_{prepis\_a}$, Γ_a)

(7) $(q_{prepis\_a}$, (b,F_{zjed}...F_{prepis},...,F_{prepis}),1) $\vdash$ $(q_{overit}$, (b,F_{zjed}...a',...,F_{prepis})) podsymbol na (k_i mod(n+1)). mieste sa prepíše na a' alebo

(7)$(q_{prepis\_a}$, (F_{zjed}...,F_{prepis},...F_{prepis}),1) $\vdash$ $(q_{overit}$, (F_{zjed},...a',...F_{prepis})), podsymbol na (k_i mod(n+1)). mieste sa prepíše na a'

(8)$(q_{overit}$, (b,F$_{zjed}$, ...F$_{prepis}$,...F_{prepis}),1) $\vdash$*$(q_{overit}$, (b,F$_{zjed}$, ...F$_{prepis}$,...F_{prepis})) alebo
(9)$(q_{overit}$, (F$_{zjed}$, ...F$_{prepis}$,...F_{prepis}),1) $\vdash$* $(q_{overit}$, (F_{zjed}...F_{prepis},...F_{prepis}))

(10)$(q_{overit}$, \$,-1)$\vdash$ $(q_{naCent}$,\$)

(11)$(q_{naCent}$,Γ_a,-1)$\vdash$ $(q_{naCent}$,Γ_a)

(12)$(q_{naCent}$,¢,1)$\vdash$ $(q_{najdi\_prep}$,¢)

(14)$(q_{najdi\_prep}$, (F_{zjed},...,F_{zjed}) ,1)$\vdash$ $(q_{najdi\_prep}$, (F_{zjed},...,F_{zjed}))

(13)$(q_{najdi\_prep}$, (b,F_{zjed},...,F_{zjed}) ,0)$\vdash$ $(q_{prepisovanie}$, (b,F_{zjed},...,F_{zjed}))

ak je n-ty krát vykonaný cyklus, nevykoná sa (13), lebo už symbol (b,F_{zjed},...F_{zjed}) nie je na páske, ale n-krát (14) a potom
(15)$(q_{najdi\_prep}$, \$,1)$\vdash$ $(q_{na\_prve\_policko}$, \$)

(16)$(q_{na\_prve\_policko}$, Γ_a ,-1) $\vdash$ n-krát $(q_{na\_prve\_policko}$, Γ_a)

(17)$(q_{na\_prve\_policko}$,¢ ,1) $\vdash$ 1-krát $(q_{hotovo}$, ¢)

3.2) Treba ukázať, že po skončení výpočtu nevznikne žiadna iná konfigurácia. Majme teda konfiguráciu (2.1). Z definície δ_a (5) načítame symbol z prvého políčka a zmeníme ho z tvaru (a $\in \Sigma$,F_{Prepis},...F_{prepis}) na tvar (F_{Prepis},...,F_{prepis}), pričom LBA B zmení stav na q_{prepis_a}. Zo (6) vyplýva, že sa hlava bude po páske pohybovať, pričom nakoniec kvôli (7) musí spraviť prepis namiesto podsymbolu F_{prepis} napíše a', pamätaného v stave q_{prepis_a} a prejde do stavu q_{overit}. Potom musí kvôli (8) a (9) ísť na \$, kde kvôli (10) zmení stav na q_{naCent}, a preto vďaka (11) musí ísť na ¢. Na ¢ kvôli (12) zmení stav na q_{najdi_prep} a potom musí ísť doprava kvôli (14), až kým nenarazí na políčko (a,F_{zjed},$F_{zjed}\cdots$,F_{zjed}). Potom musí prejsť do stavu $q_{prepisovanie}$ kvôli (13). A tým pádom celý cyklus sa opakuje. V prípade, že by na symbol (a,F_{zjed},$F_{zjed}\cdots$,F_{zjed}) nenarazil, tak kvôli (14) prejde na \$, kde kvôli (15) prejde do stavu $q_{na\_prve\_policko}$ a tam vďaka (16) a (17) skončí hlava na prvom políčku.

Ukázali sme teda, že symboly a $\in \Sigma$ sa odstránia a za každé a$\in \Sigma$ sa napíše namiesto F_{prepis} a', pričom postup nastáva od ľavého políčka k pravému. Stačí ešte ukázať, že a_i' je naľavo od a_{i+1}'. Urobí sa prepis na a_{i+1}' a to vďaka (7). To zabezpečí, aby danom políčku nebol a_i' napravo od a_{i+1}'(na stope viac doľava). Potom LBA B prejde o pozíciu doprava a (8) a (9) skontroluje, že žiadne b' na týchto políčkach nie je a takto postupuje až na \$.

2)Simulácia LBA A pomocou LBA B_2 z tejto konfigurácie

Najprv zadefinujeme LBA B_2 a potom ukážeme, že w$\in$ L(A) $\leftrightarrow \exists$ w'$\in$ L(B_2), pričom w' vzniklo z w pomocu výpočtu LBA B_1.

Definícia LBA B_2 LBA $B_2 = (K_2,\Sigma_2,\Gamma_2,\delta_2,q_{hotovo},F_2)$

$K_2 = \{q_{hotovo},\}\cup\{$(q,Cislo), (p,Cislo,SmerHladania), (q1,SmerHladania,Zapis_b')$|$ $q\in$ K (stavy LBA A); cislo$\in\{$1, ..c+2$\}$; SmerHladania$\in\{$-1,0,1$\}$; $\{$Zapis_b'$|$b$\in\Gamma$ (LBA A)$\}$ $\}$

Pomocné množiny na symboly:

- F_{zjed}=$\{F_{prepis}\}\cup$ $\{$a'; a$\in\Sigma\}$
- $F_{zjednot}$=$\{F_{prepis}$, $F_{preskoc}\}$
- F_{vset}= $F_{zjed}\cup F_{preskoc}$

Σ_2 = { (F_{zjed},...,F_{zjed})}; počet podsymbolov F_{zjed} je c+1

Γ_2 = { ($F_{zjed}\cup F_{preskoc}$,...,$F_{zjed}\cup F_{preskoc}$; počet podsymbolov $F_{zjed}\cup F_{preskoc}$ je c+1

F_2 = { (q_F,Cislo)|$q_F\in$ F (LBA A); Cislo $\in$ {0,1,...,c+2}

Definícia δ_2:

1. δ_2(q_{hotovo},(F_{prepis},...,F_{prepis})) = (q_{hotovo},(F_{prepis},...,F_{prepis}),1)
2. δ_2(q_{hotovo},(F_{prepis},...a'...,F_{prepis})) = ((q_0,Cislo),(F_{prepis},...a'...,F_{prepis}),0), pričom podsymbol a' je prvé naľavo z množiny {b'; b$\in\Sigma$} a celkovo odľava v poradí Cislo podsymbol; q_0 je začiatočný stav LBA A
   (1)-(2) LBA B_2 sa dostane na prvý podsymbol množiny {a'|a$\in\Sigma$} a začne simuláciu LBA A, v Cislo si uchováva informáciu, na ktorej stope sa nachádza.
3. δ_2((q,Cislo),(F_{vset},$\cdots$ a',$\cdots$ F_{vset})) = ((p,Cislo,SmerHladania),(F_{vset},$\cdots$ a',$\cdots$ F_{vset}),0)

   $\Leftrightarrow$(q,a) = (p,a,SmerHladania) pre LBA A
4. δ_2((q,Cislo),(F_{vset},$\cdots$ a',F_{prepis},$\cdots$ F_{vset})) = ((p,Cislo+1,SmerHladania),(F_{vset},$\cdots$ $F_{preskoc}$,b',$\cdots$ F_{vset}),0)

   $\Leftrightarrow$(q,a) = (p,b,SmerHladania); a$\neq$ b; cislo < c+1 pre LBA A
5. δ_2((q,Cislo),(F_{vset},$\cdots$ a')) = ((p,1,SmerHladania,Zapis_b'),(F_{vset},$\cdots$ $F_{preskoc}$),1)

   $\Leftrightarrow$(q,a) = (p,b,SmerHladania); a$\neq$ b; cislo = c+1 pre LBA A;SmerHladania =1
   (3)-(5)Načíta symbol, prejde do stavu (p,Cislo,SmerHladania). V prípade, že nastal prepis, tak B_2 načítaný podsymbol a' prepiše na F_{prepis} a ďalšiu stopu napíše symbol, za ktorý LBA A prepísal a (v našom výpočte to predstavuje a').
6. δ_2((q,1,SmerHladania,Zapis_b'),(F_{prepis},$\cdots$)) = ((q,1,SmerHladania),(b',$\cdots$,0))
7. δ_2((q,Cislo,SmerHladania),($F_{zjednot}$,$\cdots$ a',$\cdots$)$\cup${ ¢,\$}) =

   ((q,Cislo+SmerHladania),($F_{zjednot}$,$\cdots$ a',$\cdots$)$\cup${ ¢,\$},0);

$\Leftrightarrow$ ak Cislo+SmerHladania podsymbol je a'; 1$\leq$ Cislo+SmerHladania $\leq$ c+1

8. δ_2((q,Cislo,SmerHladania),(F_{vset},$\cdots$a',$\cdots$))=
((q,Cislo+SmerHladania,SmerHladania),(F_{vset},$\cdots$a',$\cdots$),0)

   $\Leftrightarrow$ ak Cislo+SmerHladania podsymbol nie je a'; 1$\leq$ Cislo+SmerHladania $\leq$ c+1

9. δ_2((q,Cislo,SmerHladania),($\cdots$,$\cdots F_{zjednot}$)) = ((q,0,SmerHladania),($\cdots$,$\cdots F_{zjednot}$),1)

   $\Leftrightarrow$ teda ak c+1. podsymbol nie je a'; c+1 $<$ SmerHladania+cislo;

10. δ_2((q,Cislo,SmerHladania),($F_{zjednot}$,$\cdots$)) =
((q,c+2,SmerHladania),($F_{zjednot}\cdots$, -1))

    $\Leftrightarrow$ teda ak 1. podsymbol nie je a'; 1 $>$ SmerHladania+cislo;
    (6)-(10) LBA B_2 prechádza po stopách, pričom ignoruje F_{prepis} a $F_{preskoc}$, až do kým nenájde a'$\in\Sigma$'$\cup${¢,\$}.

Dôkaz správnosti výpočtu LBA B_2

Lema 4.2.2 *Tvrdíme, že keď najprv použijeme pravidlá (1),(2), a po l krokoch použití pravidiel (3)-(6) a pričom za každým takým pravidlom vykonáme pravidlá (7)-(10)(ich tranzitívny uzáver), bude LBA B_2 v konfigurácií:*

1. *Nech w_B je obsah pásky LBA B_2 po l krokoch použití pravidiel (3)-(6) a po každom z nich následne pravidlá (7)-(10)(ich tranzitívny uzáver). Nech w_l je obsah pásky LBA A po l krokoch. Nech h je homomorfizmus def. vyššie h. Potom platí h(w_b) = w_l.*

2. *pozícia hlavy: na j-tom políčku, pričom obsahuje podsymbol a';a$\in\Sigma$, počet podsymbolov b'; b$\in\Sigma$ naľavo od neho(aj na stopách naľavo na políčku aj na páske) je i, ak LBA A bude na i-tom políčku*

3. *v stave: (p,Cislo) na podsymbole a' v poradi Cislo, a';a$\in\Sigma$, počet podsymbolov b'; b$\in\Sigma$ naľavo od neho i, pričom LBA A bude v stave p, na i-tom políčku*

Dôkaz: matematickou indukciou na l

1$\circ$ Pre l = 0. Chceme teda ukázať, že LBA B bude v stave (q,Cislo) a na podsymbole a' Cislo. stope (poradie stopy), neobsahuje podsymboly b'; b$\in\Sigma$ naľavo od neho ani na stopách na políčku ani na stopách iných políčok naľavo (reprezentuje LBA A v začiatočnej konfigurácií).
Pravidlá (3)-(6) nepoužijeme(l = 0), len pravidlá (1),(2).

- Ak políčka neobsahujú a', tak: (1)(q_{hotovo},(F_{prepis},...F_{prepis}),1)$\vdash$* (q_{hotovo},(F_{prepis},...F_{prepis}))
- Ak políčko obsahuje a', tak: (2)(q_{hotovo},(F_{prepis},...a'...,F_{prepis}),0)$\vdash$ ((q,Cislo), F_{prepis},(...a'...,F_{prepis})), pričom podsymbol a' je prvé naľavo z množiny {b';b$\in\Sigma$} a celkovo od ľava Cislo. podsymbol, čo sme chceli dokázať

2$\circ$ Predpokladajme, tvrdenie platí pre l-1 krokoch.
Nech sme teda v konfigurácií (q,Cislo), ¢w_b \$, k -pozícia hlavy)
Kedže sme v stave (q,Cislo) (nie (q,Cislo,SmerHladania)), tak δ_b vyplýva, že LBA B načíta z viacstopovej pásky cislo v poradí symbol a'; a$\in\Sigma$. Nastávajú dva prípady:

- a)ak nenastane prepis
- b)ak nastane prepis

a) Z indukčného predpokladu a δ_2 vyplýva, ak B_2 načíta a' tak LBA A musí načítať a. (3)((q,Cislo),(F_{vset},...a',...F_{vset}), 0)$\vdash$ ((p,Cislo,SmerHladania),(F_{vset},...a',...F_{vset})) $\Leftrightarrow$(q,a) = (p,a,SmerHladania) pre LBA A

Teraz sa LBA B musí dostať na prvý podsymbol b' smerom SmerHladania. Pri tom môžu nastať prípady:

1. SmerHladania = 0
2. SmerHladania = 1
3. SmerHladania =-1

- 1) potom zo (7) vyplýva, že ((q,Cislo,0),($F_{zjednot}$,$\cdots$ a',$\cdots$))$\vdash$ ((q,Cislo+0),($F_{zjednot}$,$\cdots$ a',$\cdots$),0);
- 2) SmerHladania = 1 potom, (8)((p,Cislo,1),($F_{zjednot}$,...a',...$F_{zjednot}$))$\vdash$* ((p,Cislo+1,SmerHladania),($F_{zjednot}$,...a',...$F_{zjednot}$))

Ak c+1= Cislo, tak
((q,Cislo,1),(...,...$F_{zjednot}$, 1))⊢ ((q,0,1),(...,...$F_{zjednot}$)).

Z δ_2 vyplýva, že ,,príkazy" sa opakujú, dokým, nenájde prvý podsymbol a' alebo \$. A potom (7),
((q,Cislo,1),(($F_{zjednot}$,$\cdots$ a',$\cdots$)∪{\$}))⊢ ((q,Cislo+1),(($F_{zjednot}$,$\cdots$ a',$\cdots$)∪{\$})),0);

- 3) SmerHladania =-1 potom,
  (8)((p,Cislo,-1),($F_{zjednot}$,...a',...$F_{zjednot}$))⊢* ((p,Cislo-1,-1),($F_{zjednot}$,...a',...$F_{zjednot}$))
  Ak Cislo = 1, tak
  ((q,Cislo,-1),(...,...$F_{zjednot}$, 1))⊢ ((q,0,-1),(...,...$F_{zjednot}$)).

  Z δ_2 vyplýva, že ,,príkazy" sa opakujú, dokým, nenájde prvý podsymbol a' alebo ¢. A potom (7),
  ((q,Cislo,-1),($F_{zjednot}$,$\cdots$ a',$\cdots$)∪{¢})⊢ ((q,Cislo-1),($F_{zjednot}$,$\cdots$ a',$\cdots$∪{¢}),0);

b) Z indukčného predpokladu: taký istý symbol načíta aj LBA A. Ak LBA A prejde do stavu p, urobí prepis na symbol b a urobí krok na páske smerHladania, tak pre LBA B môžu nastať dve možnosti:

b1) ak v pôvodnom stave (q,Cislo) platilo: cislo < c+1, tak B vykoná kvôli (4), teda
((q,Cislo),(F_{vset},...a',F_{prepis},...F_{vset}), 0)⊢
((p,Cislo+1,SmerHladania),(F_{vset},...$F_{preskoc}$b',..F_{vset}))

b2) ak v pôvodnom stave (q,Cislo) platilo: cislo = c+1, tak B prejde do stavu (p,1,SmerHladania) a urobí krok napravo(5), teda
((q,Cislo),(F_{vset},$\cdots$ a'), 1)⊢ ((p1,SmerHladania,Zapis_b'),(F_{vset},$\cdots$ $F_{preskoc}$))
a potom
(6)((q,1,SmerHladania,Zapis_b'),(F_{prepis},$\cdots$), 0)⊢ (q1,SmerHladania,(b',$\cdots$))

Ako je vidieť, v oboch prípadoch b1) aj b2) LBA B prejde do (q,Cislo,SmerHladania) a musí si nájsť ďalší najbližší podsymbol zo smeru SmerHladania. Teraz už stačí urobiť tie isté úvahy ako v a).

3)Počet prepisov na jednom políčku je ohraničená konštantou

1. V a) časti nastáva prepis z a ∈ Σ ↦ (a,F_{prepis},...,F_{prepis}). (1c)
   Potom (a,F_{prepis},...,F_{prepis}) ↦ (F_{prepis},...,F_{prepis}) (2c)
   (F_{prepis},...,F_{prepis}) ↦ (F_{prepis}∪{a_1∈ Σ},...,F_{prepis}∪{a_n∈ Σ}) (3c)

   - Počet prepisov v (1c) jeden.

- Počet prepisov v (2c) jeden.
- Počet prepisov v (3c) maximálne c+1, lebo (F_{prepis}∪{a_1∈ Σ},...,F_{prepis}∪{a_n∈ Σ}) je c+1.

2. V b) časti nastáva prepis v δ_2 v (1)-(4). Vždy keď nastane prepis a'↦$F_{preskoc}$ a F_{prepis}↦ b'. Keď za a' nie je F_{prepis}, tak a' nemožno prepisovať, resp. ak sa to nastane, tak sa výpočet zasekne. Maximálny počet prepisov určite nie je väčší ako 2(c+1).

d)Záver

Ako je vidieť, v časti a) aj časti b) sme prepisovali len konečne veľa krát na políčku ⇒ celkovo sme na políčku prepisovali len konečne veľa krát. Hľadaný LBA B vznikne teda „spojením LBA B_1 a LBA B_2, teda B = (K_1∪K_2,Σ_1,Γ_1∪Γ_2,δ_1∪δ_2,q_{zac},F_2). Z a), b) a c) vyplýva, že LBA B bude mať požadované vlastnosti.□

Veta 4.2.3 *Nech f(n) je rastúca funkcia N ⇒ N a LBA A, ktoré môže urobiť maximálne f(|w|)·(|w|) prepisov na páske. Ak ∃ LBA B, ktoré môže prepisovať maximálne f(n)krát na políčku, také, že ∀w∈L(A) vie f(|w|) zapísať v binárnom tvare na páske, tak* $\mathcal{L}_{ecss,f(n)\cdot n,word} = \mathcal{L}_{ecss,f(n),square}$.

Dôkaz: Nech w je vstupné slovo. Zapíšeme f(|w|) na pásku. Nedeterministicky si tipneme, koľkokrát budú dané políčka $w_1 \cdots w_n$ prepisované počas prvých n krokov. Odsimulujeme prvých n-krokov rovnakou technikou ako v predošlej leme, čo nám zaručí, že jednotlivé políčka budú konštantne prepisované. Takto postupujeme ďalej, odsimulujeme ďalších n krokov, pričom si ukladáme informáciu, koľkokrát sme takýchto krokov robili. Ak ich je f(n) krát, tak výpočet skončí podľa toho, či sa dané slovo dovtedy akceptovalo. Na kontrolovanie počtu „n-krokov" sa políčka(stačí len jedno) maximálne f(n) krát prepísali a maximálny počet prepisov na políčku kvôli simulácii výpočtu je f(n)·c∈N krát, pre vhodnú konštantu c, preto $\mathcal{L}_{ecss,f(n),square} = \mathcal{L}_{ecss,f(n)\cdot n,word}$.□

Poznámka 4.2.4 *Uvedená konštrukcia sa dá použiť aj na dôkaz ekvivalencie, keď LBA má obmedzenie počtu návštev na políčku (visits, resp. cross) a LBA s globálnym obmedzením návštev políčok, teda NTIME.*

Dôsledok 4.2.5 *Všetky tri definície uvádzané v poznámke 2.14 sú ekvivalentné, ak f(n) je vyjadriteľná v binárnom tvare, kde n je dĺžka slova.*

Dôkaz: Konštrukciou z predchádzajúcej vety

4.3 Deterministická verzia

Dôsledok 4.3.1 *Nech* $L \in \mathcal{L}_{decss,n,word}$*, LBA A, také, že* $L(A) = L$ *a urobí maximálne* $\mathcal{O}(n)$ *prepisov vo výpočte, pričom* $\# \notin \Sigma_L$*. Definujme* $L_1 = \{w\#\#u|$ $w \in L$*;* $u = u_0\#\ldots\#u_n$*;* $u_i \subseteq \{0\}^*$*;* u_i *je počet prepisov na i-tom políčku vo výpočte na slove w LBA A(zapísané unárne)*$\}$*. Potom* $L_1 \in \mathcal{L}_{decss,c,square}$*, kde c je konštantná funkcia.*

Dôkaz: podobne ako v nedeterministickej verzii, len v tomto prípade DLBA neháda počet prepisov, ale ich má napísané na páske. □

5 Sweep automaty

5.1 Cieľ

V nasledujúcej kapitole ukážeme, že ak obmedzíme prepis políčok globálne na celom slove v deterministickom prípade, pridaním sweep obmedzenia sa sila strojov nezmenší (počet prepisov zostane asymptoticky rovnaký).

5.2 O normálnom tvare

Definícia 5.2.1 *Prechodová postupnosť* $A=(K,\Sigma,\delta,q_0,F)\in$ *2-DKA je postupnosť dvojíc (s,d), kde (s,d) s*$\in K$ *a d*$\in\{-1,1\}$

Poznámka 5.2.2 *Prechodová postupnosť sa dá samozrejme zadefinovať aj pre iné modely, najmä sa využíva v súvislosti s jednopáskovými Turingovými strojmi.*

Definícia 5.2.3 *Definujeme right-matching a left-matching dvojíc prechodových postupností na dvojsmerných konečných automatov na políčku so symbolom a rekurzívne v (1)-(5):*

1. *nulová postupnosť left a right match(ujú) nulovú postupnosť.*

2. *Ak* $q_3 ... q_k$ *right match* $p_1 ... p_l$ *a* $\delta(q_1,a)=(q_2,L)$*, potom* $q_1,q_2,...q_k$ *right-match* $p_1,p_2,...p_l$.

3. *Ak* $q_2 ... q_k$ *left match* $p_2 ... p_l$ *a* $\delta(q_1,a)=(p_1,R)$*, potom* $q_1,q_2,...q_k$ *right-match* $p_1,p_2,...p_l$.

4. *Ak* $q_1,...q_k$ *left match* $p_3 ... p_l$ *a* $\delta(p_1,a)=(p_2,R)$*, potom* $q_1,...q_k$ *left match* $p_1 ... p_l$

5. *Ak* $q_2,...q_k$ *right match* $p_2 ... p_l$ *a* $\delta(p_1,a)=(p_2,L)$*, potom* $q_1,...q_k$ *left match* $p_1 ... p_l$

Lema 5.2.4 *Uvažujme o 2-DKA A, ktorý v každom kroku výpočtu hýbe hlavou. Potom k nemu štandardnou konštrukciou z [1] zostrojíme 1-NKA, následne k nemu štandardnou konštrukciou z [1] 1-DKA. Ak 2-DKA neprečítal nejaké políčko a dané slovo akceptoval, potom práve jedna z prechodových postupností je tvaru* $(q_j,-1)$ $(q_i,-1)...(q_F,0)$ *alebo* $(q_j,-1)$ $(q_i,-1)...(q_k,-1)$.

Dôkaz: v štandardnej konštrukcii z 2-DKA na 1-NKA sa nederministicky tipne prechodová postupnosť, v deterministickej verzii si musíme všetky prechodové postupnosti vygenerovať. Keďže A je deterministický automat, tak pre dané w∈ L(A) existuje práve jeden akceptačný výpočet. Ak A neprečítal všetky symboly, tak uvažujme o políčko, ktoré najviac vpravo načítal. Potom prechodová postupnosť musí mať tvar buď (q_j,-1)... (q_i,-1)...(q_F,0)–posledné načítané políčko a na ňom prejde do akceptačného stavu alebo (q_j,-1)...(q_i,-1)...(q_k,-1) –posledné načítané políčko a na ňom neprejde do akceptačného stavu. V prípade, že by existoval člen tvaru (q_j,1), tak výpočet by pokračoval napravo, to je spor s tým, že sme na poslednom načítanom políčku. Na iných políčkach naľavo prechodové postupnosti nemôžu mať daný tvar ako na poslednom vpravo načítanom políčku, lebo to by znamenalo, výpočet na posledné načítané políčko neprejde (kedže od políčka ide len doľava).□

Veta 5.2.5 *K ľubovolnému DLBA A = (K,Σ,Γ,δ,q_0,F), ktorý počas výpočtu urobí f(n) prepisov, existuje sweep DLBA B taký, že urobí počas výpočtu maximálne O(f(n)) prepisov,pričom L(A)=L(B).*

Dôkaz:

Neformálne:

Skôr než prídeme priamo k dôkazu, skúsme sa zamyslieť, kde je problém. Keby sme použili priamu konštrukciu, simulujeme výpočet a vždy, keď sa zmení smer hlavy, označíme si príslušné políčko, ideme na koniec pásky a potom sa vrátime na políčko a pokračujeme simuláciou, tak musíme spraviť potencionálne O(n) prepisov medzi dvoma prepismi. No my si ich môžeme dovoliť len konštantne. Rozdeľme si výpočet DLBA A medzi dvoma prepismi-to vieme simulovať po technickej úprave 2-DKA. A tento 2-DKA budeme simulovať sweep DKA, pričom urobí maximálne |K| prepisov, kde K je množina stavov daného 2-DKA.
Predpokladajme, že LBA A spravil prepis na symbol a. My si symbol označíme a'(aby sme vedeli, kde sa stal naposledy prepis, nikde inde sa symbol s' nenachádza) a zapamätáme si stav, v ktorom sa LBA A nachádza. Prejdeme hlavou na začiatok pásky(na ¢). Zostrojíme dvojsmerný konečný deterministický automat nasledovne: začiatočný stav q_{zac}, ktorý vždy keď načíta symbol, hlavou sa posunie doprava, stav sa nemení, až kým nenačíta a'. Po načítaní tohto symbolu sa hlava nepohne, ale automat prejde do stavu, aký mal LBA A, keď spravil prepis. Teraz sa automat bude správať rovnako ako LBA A s tým, že keď LBA A spraví prepis, tak konečný automat prejde do akceptačného stavu. Týmto sme dostali dvojsmerný automat, ktorý

simuluje LBA A medzi prepismi. Na dokázanie lemy nám stačí ukázať, že
tento dvojsmerný konečný autom
tomatom(smer môže meniť len n
počtom prepisov, pričom konfigu
(okrem iného hlava musí byť presi
Teda zostrojme k tomuto dvojsm
na to štandardnú konštrukciu z [2
políčko najviac vpravo sa načítalo
pri výpočte nenarazil na $ (čitate
keď narazil na $). Simulácia: na p
automat prečítal, urobíme prepis
merný automat pri načítaní dan
si zapamätáme ten, pri ktorom h
sweep automat prejde na políčko
prepisom, a začne s využitím šta
jednosmerný (začiatočný stav bu
A zase s pomocou lema 5.2.4 spra
políčko čítal zľava. Takto to opak

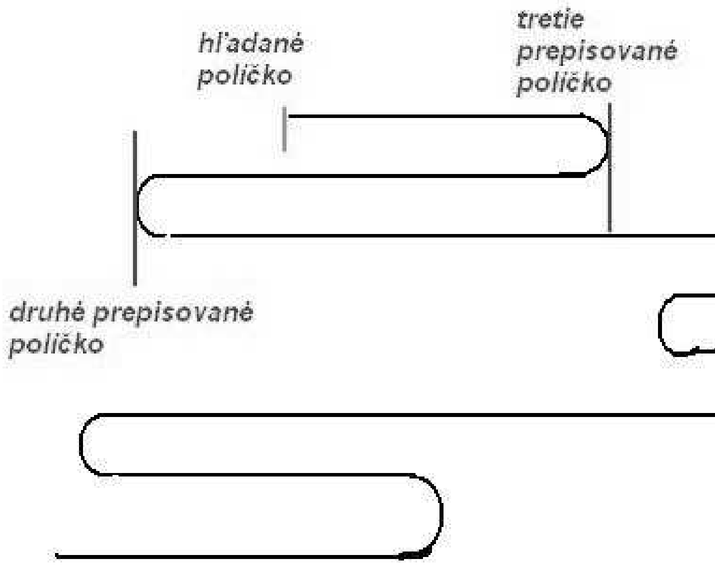

Figure 2: výpočet 2-DKA A automatom B

Keď na poslednom políčku, ktoré prečítal, sa dostane do akceptačného stavu, tak sa výpočet skončí. Na koniec dôkazu stačí ukázať, že týmto spôsobom potrebujeme maximálne k prepisov. Ako je z obrázku vidieť, všetky pomocné prepisy sú robené medzi políčkom, do ktorého sa automat dostane do akceptačného stavu. To značí, že dvojsmerný automat musel aspoň toľkokrát prejsť cez toto, políčko koľko náš sweep automat robil prepisov. No aby sa dvojsmerný deterministický automat nezacyklil, musí na danom políčku byť vždy v inom stave. Teda počet prepisov nemôže byť viac ako počet stavov dvojsmerného automatu.

Formálne:

Štruktúra formálneho dôkazu je nasledovná:

a) definícia sweep DLBA B

b) ukážeme, že sa DLBA B dostane do hľadanej konfigurácie(do konfigurácie, ktorá nastane po ďalšom prepise DLBA A)

c) ukážeme, že počet prepisov sa dá ohraničiť konštantou

d) záver

a) Definícia sweep DLBA B

Majme teda DLBA A = $(K_A, \Sigma_A, \Gamma_A, \delta_A, q_0$,F). Najprv zadefinujeme 2-DKA A, ktorý bude simulovať výpočet medzi dvoma prepismi DLBA A, a pomocou neho sweep DLBA B.

Definícia 2-DKA A = $(K_{DKA}, \Sigma_{DKA}, \delta_{DKA}, p_0$,F)

- $K_{DKA} = K_A \cup \{p_0, q_{akc}\}$
- $\Sigma_{DKA} = \Sigma_A \cup \{(a',q) | a \in \Sigma_A; q \in K_A\}$
- δ_{DKA}
  - $\delta_{DKA}(p_0, a \in \Sigma_A \cup \{¢\}) = (p_0, 1)$
  - $\delta_{DKA}(p_0, \$) = (p_{naZaciatok}, -1)$
  - $\delta_{DKA}(p_{naZaciatok}, a \in \Sigma_A) = (p_{naZaciatok}, -1)$
  - $\delta_{DKA}(p_{naZaciatok}, ¢) = (q_0, 1)$
  - $\delta_{DKA}(p_0, (a', q)) = (q, 0)$
  - $\delta_{DKA}(q, a \in \Sigma_A) = (p, x) \leftrightarrow \delta_A(q, a) = (p, a, x)$
  - $\delta_{DKA}(q, (a', r) | a \in \Sigma_A; r \in K_A) = (p, x) \leftrightarrow \delta_A(q, a) = (p, a, x)$
  - $\delta_{DKA}(q, a) = (q_{akc}) \Leftrightarrow \delta_A(q, a) = (p, b, x); a \neq b$
- F = $\{q_{akc}\}$

2-DKA A pracuje nasledovne: nájde políčko, ktoré označuje, kde sa urobil prepis (a',q), a potom simuluje výpočet DLBA A od tohto políčka zo stavu q. V prípade, že by hľadané políčko nenašiel, tak prejde na prvé políčko a simuluje DLBA A z jeho počiatočnej konfigurácie. 2-DKA A sa dostane do akceptačného stavu, ak DLBA A bude chcieť urobiť prepis.

B = ($K_B, \Sigma_B, \Gamma_B, \delta_B, q_{zaciatok}$**,F)**

- K_B =
  - $\{q_{zaciatok}$,$q_{akceptacne\_policko}\}\cup$
  - $\{(q_{zapam\ddot{a}taj\_p_s,x})$, $(q_{najdiOznacene\_p_s}$,x)| x∈{-1,1}}∪
  - $\{(q_{x_1,x_2,,x_o}$,x)| x∈{-1,1}; x_i prechodová postupnosť 2-DKA A o najviac | K_A | stavov}
- $\Sigma_B = \Sigma_{DKA}$
- $\Gamma_B = \Sigma_{DKA}$
- F = $\{q_{akceptacne\_policko}\}$
- *Definícia* δ_B*:*
  1. $\delta_B(q_{zaciatok}$,¢) = (($q_{x_1,x_2,,x_o}$,1),¢, 0) (p_0,1) je 1. člen prechodovej postupnosti x_j; p_0 začiatočný stav 2-DKA A
  2. $\delta_B((q_{x_1,x_2,,x_o}$,1),a∪{(a',p)}) = (($q_{y_1,y_2,,y_r}$,1),a, 1); a∈ Σ; $\forall y_j \exists x_i$ (x_i,a,1) right matching y_i a v prípade, že načíta políčko (a',p), tak prvý člen prechodovej postupnosti musí mať tvar (p,k∈{-1,0,1})
  3. $\delta_B((q_{x_1,x_2,,x_o}$,1),a∪{(a',p)}) = (($q_{zapamataj\_p_s}$, 1),$\overline{a}$, 1); a∈ Σ; $\exists x_i$ je tvaru (p_1,-1), (p_s,-1) a posledný člen nie je tvaru (p_s,0) ∈ F v 2-DKA A
  4. $\delta_B((q_{x_1,x_2,,x_o}$,1),\$) = (($q_{y_1,y_2,,y_r}$,-1),\$, -1); Z vlastností $\exists x_i$ je tvaru (p_1,-1), (p_s,-1)
     a∈ Σ; $\forall y_j \exists x_i$ (x_i,a,1) left matching y_i a zároveň p_s je 1. časťou prechodovej postupnosti y_j
  5. $\delta_B((q_{x_1,x_2,,x_o}$,1),a∪{(a',p)}) = ($q_{akceptacne\_policko}$,a, 0); a∈ Σ; $\exists x_i$ je tvaru (p_1,-1), (p_s,-1) a posledný člen je tvaru (p_s,0) ∈ F v 2-DKA A
  6. $\delta_B((q_{x_1,x_2,,x_o}$,1),\$) = ($q_{akceptacne\_policko}$,\$, 0); $\exists x_i$ je tvaru (p_1,-1), (p_s,-1) a posledný člen je tvaru (p_s,0) ∈ F v 2-DKA A
  7. $\delta_B((q_{zapam\ddot{a}taj\_p_s}$,1),a) = (($q_{zapam\ddot{a}taj\_p_s}$,1),a, 1)
  8. $\delta_B((q_{zapam\ddot{a}taj\_p_s}$,1),\$) = (($q_{najdiOznacene\_p_s}$,-1),\$, -1)
  9. $\delta_B((q_{najdiOznacene\_p_s}$,-1),a) = (($q_{najdiOznacene\_p_s}$,-1),a, 0)
  10. $\delta_B((q_{najdiOznacene\_p_s}$,-1),$\overline{a}$) = (($q_{x_1,x_2,,x_o}$,-1),a, 0); $\forall x_i$ p_s je 1. časťou prechodovej postupnosti

11. $\delta_B((q_{x_1,x_2,,x_o},\text{-}1),\text{a}) = ((q_{y_1,y_2,,y_r},\text{-}1),\text{a}, \text{-}1)$; $\forall$ a$\in \Sigma$; $\forall y_j \exists x_i$ $(x_i,\text{a},1)$ left matching y_i
12. $\delta_B((q_{x_1,x_2,,x_o},\text{-}1),\text{a}) = (q_{akceptacne\_policko},\text{a}, 0)$; a$\in \Sigma$; $\exists x_i$ je tvaru $(p_1,1)$, $(p_s,1)$ a posledný člen je tvaru $(p_s,0) \in$ F v 2-DKA A
13. $\delta_B((q_{x_1,x_2,,x_o},\text{-}1),\text{¢}) = (q_{akceptacne\_policko},\text{¢}, 0)$; $\exists x_i$ je tvaru $(p_1,1)$, $(p_s,1)$ a posledný člen je tvaru $(p_s,0) \in$ F v 2-DKA A
14. $\delta_B((q_{x_1,x_2,,x_o},\text{-}1),\text{a}) = ((q_{zapamätaj\_p_s},\text{-}1),\overline{a}, \text{-}1)$; $\forall$ a$\in \Sigma$; $\exists x_i$ je tvaru $(p_1,1)$, $(p_s,1)$ a posledný člen nie je tvaru $(p_s,0) \in$ F v 2-DKA A
15. $\delta_B((q_{x_1,x_2,,x_o},\text{-}1),\text{¢}) = ((q_{y_1,y_2,,y_r}, \text{-}1),\text{a}, 1)$; $\exists x_i$ je tvaru $(p_1,1)$, $(p_s,1)$ a$\in \Sigma$; $\forall y_j \exists x_i$ $(x_i,\text{a},1)$ righh matching y_i a zároveň p_s je 1. časťou prechodovej postupnosti y_j
16. $\delta_B((q_{zapamätaj\_p_s},\text{-}1),\text{a}) = ((q_{zapamätaj\_p_s},\text{-}1),\text{a}, \text{-}1)$
17. $\delta_B((q_{zapamätaj\_p_s},1),\text{¢}) = ((q_{najdiOznacene\_p_s},1),\text{¢}, 1)$
18. $\delta_B((q_{najdiOznacene\_p_s},1),\text{a}) = ((q_{najdiOznacene\_p_s},1),\text{a}, 1)$
19. $\delta_B((q_{najdiOznacene\_p_s},1),\overline{a}) = ((q_{x_1,x_2,,x_o},1),\text{a}, 0)$; $\forall x_i$ p_s je 1. časťou prechodovej postupnosti

b)Ukážeme, že sa DLBA B dostane do konfigurácie, ktorá nastane po ďalšom prepise DLBA A

Definícia 5.2.6 *Uvažujeme o dvojsmernom automate A, ktorý nemusí prečítať celý vstup a slove w∈L(A). Definujeme výpočtovú postupnosť políčok nasledovne:*

1. *a_1 člen - políčko, ktoré A pri danom výpočte slova načítal najviac vpravo(môže to byť aj \$).*
2. *a_{i+1}:*
   - *Ak (i+1) mod 2 = 0 políčko, ktoré A pri danom výpočte slova načítal najviac vľavo od a_i člena postupnosti.*
   - *Ak (i+1) mod 2 = 1 políčko, ktoré A pri danom výpočte slova načítal najviac vpravo od a_i člena postupnosti.*
3. *Posledný člen postupnosti je políčko, do ktorého sa 2-DKA A dostal do akceptačného stavu.*
4. *Každé políčko može byť členom postupnosti najviac raz.*

Poznámka 5.2.7 *Členy postupnosti môžeme vidieť na spomínanom obrázku(Obrázok 2). Posledný bod nám zaručí na danom slove w jednoznačnosť postupnosti.*

Lema 5.2.8 *Uvažujme o sweep DLBA B a daného 2-DKA A so slovom w, w ∈ L(A), s výpočtovou prechodovou postupnosťou s počtom členov n. Tvrdíme, že po k - prechodoch, $k \leq n$, na páske urobí prepis na k-tom člene prechodovej výpočtovej postupnosti 2-DKA A a v stave si zapamätá stav A na danom políčku, keď ho čítal naposledy, alebo, ak $n = k$, výpočet skončí na poslednom člene prechodovej výpočtovej postupnosti.*

Dôkaz: MI na počet prechodov

1$\circ$ m = 1
Sme na začiatku výpočtu. Predpokladáme, že hlava stojí na ¢ a w ∈ L(A).
Z δ: ($q_{zaciatok}$,¢,0)⊢(($q_{x_1,x_2,,x_o}$,1),¢); p_0 je 1. časťou prechodovej postupnosti x_j; p_0 začiatočný stav 2-DKA A

Možu nastať dve možnosti:

1. Na páske sa nenachádza políčko tvaru (a',q). V tom prípade z δ_B vyplýva, (($q_{x_1,x_2,,x_o}$,1),a,1) (2)⊢* (($q_{y_1,y_2,,y_p}$,1),a,1) (2)⊢
(($q_{y_1,y_2,,y_p}$,1),\$,-1) (4)⊢
(($q_{y_1,y_2,,y_r}$,-1),\$,)
Musíme prejsť velú pásku, lebo aj 2-DKA A prejde celú pásku

2. Na páske sa nachádza políčko tvaru (a',q).

2) Výpočet teda prebieha: (2)(($q_{x_1,x_2,,x_o}$,1),a,1) ⊢* (($q_{y_1,y_2,,y_p}$,1),(a',q),1) (2)⊢* (($q_{s_1,s_2,,s_r}$,1),a)

Môžu nastať dva prípady (z δ_B však vyplýva, že (a',q) sa prepíše buď na symbol a alebo $\overline{a}$):

1. ak najviac vpravo načítané políčko DLBA A pri danom výpočte \$
Buď sa vykoná
   1a) (4)(($q_{x_1,x_2,,x_o}$,1),\$,-1)⊢ (($q_{y_1,y_2,,y_r}$,-1),\$) alebo
   1b) (6)(($q_{x_1,x_2,,x_o}$,1),\$,0)⊢ ($q_{akceptacne\_policko}$,\$)
   V tomto prípade, hľadané políčko je \$ a výpočet skončil. Korektnosť vyplýva z predchádzajúcej lemy, kedže len jedna prechodová postupnosť môže byť akceptačná.

2. ak najviac vpravo načítané políčko DLBA A pri danom výpočte nie je \$. V tomto prípade nastanú opäť 2 možnosti:

2a) (3)$((q_{x_1,x_2,,x_o},1)$,a,1)$\vdash$ $((q_{zapamataj\_p_s},1)$,$\overline{a}$,1)$\vdash$
(7)$((q_{zapamätaj\_p_s},1)$,a,1)$\vdash^*$ $((q_{zapamätaj\_p_s},1)$,a,1)$\vdash$
(8)$((q_{zapamätaj\_p_s},1)$,\$,-1)$\vdash$ $((q_{najdiOznacene\_p_s},-1)$,\$)

Označené políčko je najviac sprava načítaný symbol a v stave si pamätá stav, aký mal DKA A naposledy, keď ho čítal. Potom prejde na \$.

2b) (5)$((q_{x_1,x_2,,x_o},1)$,a,0)$\vdash$ $(q_{akceptacne\_policko}$,a)
V tomto prípade, hlava je na hľadanom políčku a výpočet skončil. Korektnosť vyplýva z predchádzajúcej lemy, keďže len jedna prechodová postupnosť môže byť akceptačná.

2$\circ$ Predpokladajme, že tvrdenie platí pre m - 1. Chceme to dokázať pre m. Ak k > ako m-1, tak výpočet skončil a nie je čo dokazovať. Predpokladajme teda, že k m-1. Môžu nastať dva prípady:

1. (n-1) mod 2 = 0
2. (n-1) mod 2 = 1

1. To znamená, že hlava je 1. symbol pred ¢ (nepárnym prechodom B ide smerom na \$, párnym na ¢). Sme v stave $(q_{najdiOznacene\_p_s},1)$.
(18)$((q_{najdiOznacene\_p_s},1)$,a,1)$\vdash^*$ $((q_{najdiOznacene\_p_s},1)$,a,1)$\vdash$

(19)$((q_{najdiOznacene\_p_s},1)$,$\overline{a}$,1)$\vdash$ $((q_{x_1,x_2,,x_o},1)$,a,1)

Môžu nastať dve možnosti:

1a) (3)$((q_{x_1,x_2,,x_o},1)$,a,1)$\vdash$ $((q_{zapamataj\_p_s},1)$,$\overline{a}$,1)$\vdash$
(7)$((q_{zapamätaj\_p_s},1)$,a,1)$\vdash^*$ $((q_{zapamätaj\_p_s},1)$,a,1)$\vdash$
(8)$((q_{zapamätaj\_p_s},1)$,\$,-1)$\vdash$ $((q_{najdiOznacene\_p_s},-1)$,\$)

Označené políčko je najviac sprava načítaný symbol a v stave si pamätáme stav, aký mal DKA A naposledy, keď ho čítal. Potom prejde na \$.

1b) (5)$((q_{x_1,x_2,,x_o},1)$,a,0)$\vdash$ $(q_{akceptacne\_policko}$,a)
V tomto prípade, hlava je na hľadanom políčku a výpočet skončil. Korektnosť vyplýva z predchádzajúcej lemy, keďže len jedna prechodová postupnosť môže byť akceptačná.

Poznámka: nemôže sa stať, že by najviac vpravo políčko bol \$, lebo na to by sme už narazili v indukčnom kroku.

2. To znamená, že hlava je na \$(nepárnym prechodom B ide smerom na \$, párnym na ¢). Sme v stave ($q_{najdiOznacene\_p_s}$,-1).
(9)(($q_{najdiOznacene\_p_s}$,-1),a,-1)⊢* (($q_{najdiOznacene\_p_s}$-1),a,-1)⊢

(10)(($q_{najdiOznacene\_p_s}$-1),$\overline{a}$,-1)⊢ (($q_{x_1,x_2,,x_o}$,-1),a,-1)⊢

(11)(($q_{x_1,x_2,,x_o}$,-1),a,-1)⊢*
(($q_{y_1,y_2,,y_r}$,-1),a) –left matching

Môžu nastať dve možnosti:

2a) ak a_m člen nie je ¢

2b) ak a_m člen je ¢

2a) (14)(($q_{x_1,x_2,,x_o}$,-1),a,-1)⊢ (($q_{zapamataj\_p_s}$,-1),$\overline{a}$,-1)⊢
(16)(($q_{zapamätaj\_p_s}$,-1),a,-1)⊢* (($q_{zapamätaj\_p_s}$,-1),a,-1)⊢
(17)(($q_{zapamätaj\_p_s}$,1),¢,-1)⊢ (($q_{najdiOznacene\_p_s}$,1),¢)

Označené políčko je najviac sprava načítaný symbol a v stave si pamätáme stav, aký mal DKA A naposledy, keď ho čítal. Potom prejde na ¢.

(12)(($q_{x_1,x_2,,x_o}$,-1),a,0)⊢ ($q_{akceptacne\_policko}$,a)
V tomto prípade, hlava je na hľadanom políčku a výpočet skončil. Korektnosť vyplýva z predchádzajúcej lemy, keďže len jedna prechodová postupnosť môže byť akceptačná.

2b) Buď sa vykoná
(15)(($q_{x_1,x_2,,x_o}$,-1),¢,1)⊢ (($q_{y_1,y_2,,y_r}$,1),¢), ak najviac vpravo načítané políčko DLBA A pri danom výpočte nie je ¢
alebo
(13)(($q_{x_1,x_2,,x_o}$,-1),¢,0)⊢ ($q_{akceptacne\_policko}$,¢)
V tomto prípade, hľadané políčko je ¢ a výpočet skončil. Korektnosť vyplýva z predchádzajúcej lemy, keďže len jedna prechodová postupnosť môže byť akceptačná.

c)Ukážeme, že počet prepisov sa dá ohraničiť konštantou

Priamo z popisu δ' vyplýva, že ak B urobil prepis, keď išiel na páske smerom doprava, tak už neskoršie viac doprava prepis nespravil (prepis sa uskutočňuje na najviac vpravom políčku, ktoré A čítal v danom výpočte). Podobne sa dá tvrdenie sformulovať, ak hlava išla doľava. To znamená, že simulovaný výpočet automatu A, vždy išiel medzi hľadaným políčkom(kde sa dostane do akceptačného stavu) v nejakom stave q. Automat A sa nemohol nachádzať viac ako |K| krát, lebo z Dirichletovho princípu by sa museli aspoň 2 stavy opakovať a výpočet by sa zacyklil, čo je spor predpokladom, že 2-DKA A dané slovo akceptuje. To znamená, že celkový počet prepisov je konštantný.

d)Záver

Ukázali sme, že výpočet medzi dvoma prepismi DLBA sa dá simulovať sweep DLBA, pričom sa urobí konštatne veľa prepisov. Z toho vyplýva, že obmedzenie sweep je normálny tvar DLBA, kde obmedzíme počet prepisov.□

Dôsledok 5.2.9 *K ľubovolnému výpočtu na DLBA A(jednopáskovému TS A) existuje taký DLBA A(jednopáskový TS A), že po 2·|K| zmene smeru pohybu hlavy urobí prepis, pričom počet prepisov zostane rovnaký a veľkosť používanej pásky sa nezmení.*
Dôkaz je rovnaký ako pri predošlej vete, len namiesto pomocných prepisov hlava zmení smer.□

5.3 O dolnom odhade počtu prepisov

Cieľ

V predchádzajúcom odseku sme ukázali štandardnú konštrukciu ako z dvojsmerného LBA vytvoriť sweep LBA, pričom počet prepisov narástol 2· |K| krát. Naša konštrukcia nevyužívala možnosť, že by prepisy navzájom mohli súvisieť, teda ku každej novej časti výpočtu medzi prepismi pristupoval LBA B nezávisle na ostatných. Otázka je, do akej miery bola naša konštrukcia optimálna za spomínaného predpokladu. Medzi dvoma prepismi sme museli urobiť maximálne 2·|K|, kde K je množina stavov, v nasledujúcom odseku dokážeme, že ľubovoľná iná konštrukcia(za spomínaného predpokladu nezávislosti dvoch prepisov pri LBA) by museli využiť aspoň 2·|K|-2 prepisov.

O dolnom odhade počtu prepisov

Veta 5.3.1 *Uvažujme o k-stavovom deterministickom konečnom 2-smernom automate. Nahradením sweep deterministickým konečným automatom(viackrát môže prepisovať), pričom chceme, čo aby skončil v rovnakej akceptačnej konfigurácií(hlava musí byť na rovnakom políčku a obsah pásky musí byť rovnaký) potrebujeme sweep DKA urobiť aspoň 2k-2 prepisov, pričom k je počet stavov 2-DKA.*

Poznámka 5.3.2 *Je zrejmé, že počet prepisov musí byť párne číslo, lebo keď urobíme prepis na nejakom políčku, aby sme mohli mať hľadanú konfiguráciu pri danom 2-DKA, musíme na políčku urobiť prepis na pôvodný symbol.*

Dôkaz:

Uvažujme o 2-DKA A=(K,Σ,δ,q_0), pričom
K=$\{q_1,q_2,...,q_k,q_{akc}\}$
$\Sigma = \{a_1,a_2,...,a_k\}$

Definícia δ:
$\delta(q_k, a_l) = (q_k, a_l, 1)$ ak k mod 2 = 1;k$\neq$ l
$\delta(q_k, a_l) = (q_k, a_l, -1)$ ak k mod 2 = 0;k$\neq$ l
$\delta(q_{k+1}, a_k) = (q_k, a_k, 0)$ ak $\exists\ q_{k+1}$
$\delta(q_{akc}, a_k) = (q_k, a_k, 0)$ ak $\nexists\ q_{k+1}$

Na dokončenie dôkazu stačí dokázať nasledujúcu lemu:

Lema 1 *Tvrdíme, že sweep DLBA sa nedostane do rovnakej akceptačnej konfigurácii ako 2-DKA A, ak nepoužije aspoň 2k-2 prepisov.*

Neformálne: Skôr než začneme formálne dokazovať, skúsme sa zamyslieť ako automat funguje. Na začiatku je na prvom symbole, začne postupovať vpravo, dokým nenarazí na symbol a_1. Potom hľadá a_2 smerom doľava, dokým nenatrafí na tento symbol. Teraz zase inkrementuje hľadaný symbol a zmení smer napravo, dokým nenájde daný symbol. Takto pokračuje až kým nenarazí na a_k. Potom prejde do akceptačného stavu a výpočet skončí. V prípade, že by počas výpočtu narazil na ¢ alebo \$, tak sa výpočet zasekne. Uvažujme o slovách, na ktorých by výpočet 2-DKA A vyzeral nasledovne: a_1 bude symbol najviac načítaný vpravo pri ktorom sa menil smer, a_2 symbol najviac vľavo, a_3 symbol druhý najviac načítaný vpravo, kde 2-DKA menil smer a tak ďalej. Zaoberáme sa slovami, pri ktorých platí, že keď pri nejakom symbole sa zmenil smer hlavy(sprava doľava alebo naopak), tak už

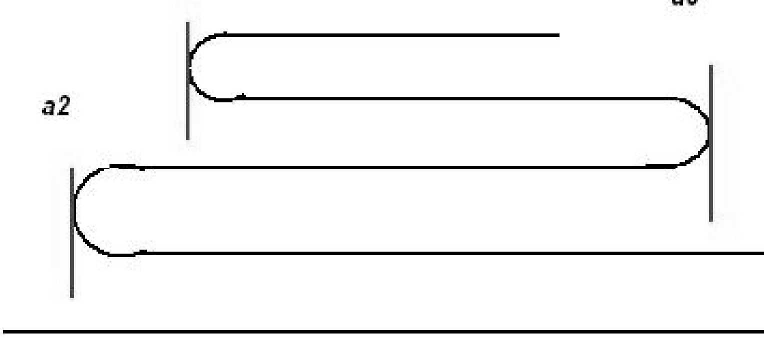

Figure 3: výpočet 2-DKA A

neskôr výpočet na danom symbole nepokračoval. Viď obrázok

Tvrdím, že za každé a_i, pri ktorom sa mení smer hlavy, treba urobiť prepis, ak v ňom automat neprešiel do akceptačného stavu. Predpokladajme, že by sme napr. za a_i prepis nemuseli urobiť. Výpočet by nemohol pokračovať od \$ smerom k ¢, lebo by sme nevedeli, ktoré a_i je prvé zľava, lebo ich môže pred posledným a_i potencionálne O(|w|) a to si nemôžeme k-stavovým automatom pamätať. Výpočet by tiež nemohol pokračovať od ¢, lebo by sme nevedeli, ktoré a_{i+1} je prvé sprava od hľadaného a_i, lebo ich môže pred posledným a_{i+1} potencionálne O(|w|) a to si teda nemôžeme k-stavovým automatom pamätať.

Formálne: Dôkaz matematickou indukciou:
1∘ k=1.
Z vety vyplýva, že sweep DLBA nemusí urobiť žiaden prepis. Preto nie je, čo dokazovať.
2∘
Predpokladajme, že veta platí pre k-1. Chceme dokázať, že platí aj pre k. Nech w je reťazec splňujúci podmienku pre k-1, teda do rovnakej akceptačnej konfigurácie sa sweep DLBA ako daný 2-smerný DKA aspoň na 2k-4 prepisov. Chceme nájsť reťazec spĺňajúci podmienku pre k. Dôkaz sporom. Predpokladajme, že stačí použiť menej 2k-2 prepisov, teda najviac 2k-4(viď poznámka). Nech sweep DLBA, ktoré to splňa, má m stavov.
Zostrojme homomorfizmus h: h(a_i)=a_{i+1} a nech h(w)=w'(len sme preznačili symboly). Nech $(w')^r$=u. Uvažujme o slove $u^{3m}a_1(ua_1)^{3m}$. Môžu nastať dva prípady:

1. sweep DLBA neurobí prepis mimo podslova u, ktoré je prvé zľava od prvého sprava a_1(hľadané u). Predpokladajme, že sweep DLBA prešlo n-krát pásku a na n+1-krát urobí prvý prepis mimo hľadaného u.

   a) hlava sa nachádza na ¢.
   Hlava nemôže načítať prvé a_1, lebo potom by výpočet nepokračoval

na hľadanom u. To znamená, že si musí pamätať v stave posledné u pred a_1. Keďže však počet u je 3m a počet stavov sweep DLBA je m, aby sa dostal na hľadané u, tak sa stavy musia opakovať, to znamená, že sa zacyklí alebo sa zasekne.

b) hlava sa nachádza na \$.
Hlava sa musí dostať na posledné a_1 sprava. Keďže však počet stavov je m a sweep DLBA musí 3m -krát $(\mathrm{u}a_1)^{3m}$, to znamená, že stavy sa musia opakovať, aby došiel na pozíciu prvej sprava a_1, preto sa zacyklí alebo sa zasekne.

2. sweep DLBA urobí prepis prvého u zľava od prvého sprava a_1(hľadané u)
To znamená, že má k dispozícií 2m-6 prepisov, keďže predpokladáme, že to zvládne menej než 2m-4 prepismi celú pásku. No to je spor s indukčným predpokladom, lebo potom by sme mohli u spraviť menej než 2m-4 prepismi. □

6 Otvorené problémy

Najzaujímavejšia otázka aj z pohľadu praktického využitia je vzťah medzi $\mathcal{L}_{decss,f(n)\cdot n,word}$ a $\mathcal{L}_{decss,f(n),square}$.

Uvažujme o triede jazykov $\mathcal{L}_{decss,f(n),square}$, a LBA automatov, ktoré danú triedu akceptujú. Keď im dáme sweep obmedzenie, je to normálny tvar týchto automatov? Pre triedu LBA akceptujúcu triedu $\mathcal{L}_{decss,f(n),word}$ sme dokázali kladný výsledok.

Definujme obmedzenie LBA na visits, teda počet návštev na danom políčku bez ohľadu, či na ňom nastane prepis. Prislúchajúcou triedu jazykov označme $\mathcal{L}_{decss,f(n),square,visits}$. Aký je vzťah medzi $\mathcal{L}_{decss,f(n),square,visits}$ a $\mathcal{L}_{decss,f(n)n,word}$, resp. $\mathcal{L}_{decss,f(n)n,square}$? Triviálne platí $\mathcal{L}_{decss,f(n),square} \subseteq \mathcal{L}_{decss,f(n)\cdot n,square,visits}$, ale aj $\mathcal{L}_{decss,f(n)\cdot n,word} \subseteq \mathcal{L}_{decss,f(n)n,square,visits}$

Definujme jazyk $L_1 = \{w_1\#w_2\#...\#w_n|\ w_i\in\{0{,}1\}^*$; $\forall$ i,j i$\neq$j, w$_i\neq w_j\}$.
Myslíme si, že $L_1 \in \mathcal{L}_{decss,n^2,square,visits}$, ale $L_1 \notin \mathcal{L}_{decss,n,word}$, resp. $L_1 \notin \mathcal{L}_{decss,c,square}$.

Triedy $\mathcal{L}_{ecss,f(n),word}$, $\mathcal{L}_{ecss,f(n),square}$, $\mathcal{L}_{decss,f(n),word}$, $\mathcal{L}_{decss,f(n),square}$ sú definované pomocou O(f(n)). Ak by sme definovali triedy jazykov len s f(n), aký je medzi nimi vzťah?

Bibliography

[1] John E. Hopcroft, Jeffrey D. Ullman. *Introduction to automata theory, languages and computation*, Addison-Wesley Publishing Company, Inc, 1979.

[2] Branislav Rovan. *Prednášky Formálne jazyky a automaty* (spísal Michal Foríšek, 2001.)

[3] Pavol Duriš. *skriptá k predmetu Výpočtová zložitosť.* 2003.

[4] Sheila A. Greibach. *The hardest context-free language.*, SIAM Journal of Computing, 1973.

[5] Klaus Wagner, Gerd Wechsung. *Computional complexivity.*, Berlin, VEB Deutcher Verlag der Wissenschaften, 1986.