Univerzita Komenského, Bratislava
Fakulta Matematiky, Fyziky a Informatiky

Konečné automaty so žetónmi

Bc. Pavol Panák

Bratislava, 2011

Univerzita Komenského, Bratislava
Fakulta Matematiky, Fyziky a Informatiky

Konečné automaty so žetónmi

(Diplomová práca)

Bc. Pavol Panák

Študijný program:	Informatika
Študijný odbor:	9.2.1. informatika
Školiace pracovisko:	Katedra informatiky
Vedúci práce:	prof. RNDr. Branislav Rovan, PhD.
Evidenčné číslo:	cbad4b61-4d7b-40df-87a2-817043e117e4

Bratislava, 2011

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Pavol Panák
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: Konečné automaty so žetónmi

Cieľ: Preskúmať výpočtovú silu konečných automatov so žetónmi pre rôzne počty žetónov a hláv.

Vedúci: prof. RNDr. Branislav Rovan, PhD.

Dátum zadania: 01.12.2009

Dátum schválenia: 18.02.2011

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 vedúcemu diplomovej práce prof. RNDr. Branislavovi Rovanovi, PhD. za pomoc pri výbere témy, jej vypracovávaní, za konzultácie a nemenej za každú radu, ktorú mi poskytol.

Abstrakt

V práci skúmame výpočtovú silu jednosmerných konečných automatov so žetónmi pre rôzne počty žetónov a hláv. Uvádzame dôkaz nárastu výpočtovej sily jednosmerných žetónových automatov v závislosti od pridávania hláv. Podobný nárast výpočtovej sily žetónových automatov sme ukázali aj pri zväčšovaní počtu žetónov. Dokázali sme rozdielnosť jednosmerných žetónových deterministických a nedeterministických automatov. Rozobrali sme aj tému, či je možné zameniť jeden žetón za jednu hlavu a naopak, v ktorej sme ukázali, že takéto zámeny nie sú možné bez určitej straty výpočtovej sily žetónového automatu. V neposlednom rade sme dokázali hornú hranicu počtu „sensing“ hláv, ktoré sú schopné kompenzovať stratu jedného žetóna.

Kľúčové slová: žetón, jednosmerné a dvojsmerné, žetónové automaty, výpočtová sila

Abstract

In the work we are exploring computing power of one-way finite-state automata with pebbles for different numbers of pebbles and heads. We are presenting proof of computing power increase for one-way pebble automata, which depends on number of heads. Similarly, we showed that, adding pebbles to pebble automata increases computing power. Besides that we proved non-equality of one-way pebble deterministic and non-deterministic automata. We discussed a topic, if it is possible to exchange one pebble for one head and vice versa. We presented that such exchanges are not possible without some lost of computing power. But we proved upper bound on number of sensing heads, which are capable to replace one lost pebble.

Keywords: pebble, one-way and two-way, pebble automata, computing power, sensing heads

Obsah

Úvod

Žetónové automaty („Pebble automata“) sú často skúmanou variantou rôznych výpočtových modelov. V literatúre je možné nájsť rôzne definície pre prácu so žetónami. Všeobecne žetónové automaty označujú výpočtový model, ktorý má schopnosť na časť vstupu položiť objekt alebo zdvihnút položené objekty, ktoré nazývame žetónmi („pebble“). Veľmi často sa pod pojmom žetónových automatov rozumejú automaty využívajúce žetóny pre prácu so stromami ako vstupmi. Blum a Hevitt ako prví využívali automaty s určitou formou žetónov. Ich automaty pracovali na dvoj-rozmerných vstupoch a používali namiesto žetónov rozličné značky („marker“). Postupom času ľudia začali vymýšlať rozličné výpočtové modely obohatené o žetóny. Samozrejme sa objavili konečné automaty so žetónmi, ktorých vstupy boli slová. Vplyvom nástupu pravdepodobnostných metód sa tieto modely rozšírili o pravdepodobnostné akceptovanie vstupných slov. Vo všeobecnosti každý výpočtový model, ktorému pridanie žetónov zvýši výpočtovú silu, bolo rozumné preskúmať.

Primárne v centre záujmu boli dvojsmerné konečné automaty so žetónmi, kvôli previazanosti s inými štandartnými výpočtovými modelmi. Nie je tažké ukázať ekvivalenciu spomínaných automatov a turingovych strojov s logaritmickým priestorom. Tiež ekvivalencia viac-hlavových „sensing“ dvojsmerných automatov a dvojsmerných automatov so žetónmi je ľahko dokázateľná. Nemenej sa skúmala výpočtová sila dvojsmerných konečných automatov so žetónmi v závislosti od počtu žetónov. Ukázalo sa, že pridávaním žetónov dostávame výpočtovo silnejšie automaty. Okrem výpočtovej sily sa tiež objavili výsledky z oblasti popisnej zložitosti dvojsmerných žetónových automatov. Konkrétne bolo ukázané, že odobratím jedného žetóna môže narásť počet stavov až exponenciálne.

Mimo záujmu avšak ostali jednosmerné konečné automaty so žetónmi. Lenže model jednosmerných konečných automatov má len jednu hlavu, preto

je prirodzene potrebné pridať viac hláv, aby sa prejavila využiteľnosť žetónov. Zjavne takýto model je závislý od počtu hláv a počtu žetónov. Cieľom našej práce je určiť ako sa mení výpočtová sila jednosmerných žetónových automatov od počtu hláv a počtu žetónov. Či je možné zameniť jednu hlavu za jeden žetón (ako pri dvojsmerných) alebo naopak jeden žetón za jednu hlavu.

Jednosmerné konečné automaty (aj so žetónmi) reprezentujú špeciálnu triedu jazykov, ktoré sú akceptovateľné veľmi efektívne v lineárnom čase. Štúdium výpočtovej sily takýchto žetónových automatov veľa vypovedá o využití žetónov pri výpočte ako aj o tom, či žetóny majú pre jednosmerné automaty zmysel. Tiež je zaujímavé ako sa zmenia niektoré už dokázané vlastnosti dvojsmerných žetónových automatov pre jednosmerný model žetónových automatov.

V práci prinášame viacero nových výsledkov ako napríklad, že pridanie hlavy pre jednosmerné žetónové automaty zvyšuje výpočtovú silu, čo je výsledok podobný ako pre viac-hlavové konečné automaty. Okrem spomínaného výsledku sme dokázali, že aj pridanie žetóna zväčšuje výpočtovú silu. Dokázali sme, že nedeterministické a deterministické žetónové automaty nie sú ekvivalentné, čo je pre dvojsmerné žetónové automaty stále otvorený problém. Tiež sme sa zaoberali otázkou, či je možné zameniť jeden žetón za jednu „sensing“ hlavu. Ako sme v texte práce dokázali takáto zámena nie je možná. Nakoniec sme zhora odhadli aj počet „sensing“ hláv, ktoré už sú postačujúce pri zámene za jeden žetón.

Táto práca je rozdelená nasledovne: V kapitole 1 na strane 3 uvádzame rozličné žetónové automaty s dôrazom na dvojsmerné konečné automaty. V nasledujúcej kapitole 2 na strane 8 poskytujeme prehľad základných faktov a dokázaných tvrdení pre dvojsmerné žetónové automaty. V poslednej kapitole 3 na strane 16 sú dokázané hlavné výsledky našej práce.

Kapitola 1

Žetónové automaty

V tejto kapitole uvedieme definíciu dvojsmerného konečného automatu so žetónmi (v literatúre známe ako „Pebble automata“ alebo tiež „Marker automata“). Kedže definícia daného výpočtového modelu nie je jednotná (každý zdroj má inú definíciu, ktorá je vhodná pre výsledky daného článku), spomenieme aj definíciu iných variantov tohto výpočtového modelu.

Väčšinu definícií z tejto časti vyžijeme v prehľadovej kapitole 2 na strane 8, ktorá sa zaoberá iba dvojsmernými žetónovými automatmi.

Okrem základných definícií z tejto kapitoly uvádzame v kapitole 3 (na strane 16) nové výsledky aj definície, týkajúce sa jednosmerných konečných automatov.

1.1 Základné definície

Nezačneme priamo s definíciou konečného automatu so žetónmi, ale najprv uvedieme definíciu dvoj-smerného nedeterministického konečného automatu. Následne jeho jednoduchým rozšíreným dostaneme žetónový automat.

Označenie. 2^A je potenčná množina množiny A.

Definícia 1.1.1. ***Dvoj-smerný nedeterministický konečný automat*** *(2NFA) A je pätica $(Q, \Sigma, \delta, g_0, F)$, kde :*

- *Q je konečná množina stavov*
- *Σ je konečná množina vstupnej abecedy, pričom $\{\vdash, \dashv\} \notin \Sigma$*
- *$g_0 \in Q$ je začiatočný stav*
- *$F \subseteq Q$ je množina akceptačných stavov*
- *$\delta : Q \times (\Sigma \cup \{\vdash, \dashv\}) \to 2^{Q \times \{-1,0,+1\}}$ je prechodová funkcia.*

Vstup pre 2NFA je uložený medzi end-mark-ami. Teda vstup w pre 2NFA vyzerá nasledovne: $\vdash w \dashv$. Prechodová funkcia určuje, do akého stavu prejde automat v jednom kroku a ako sa presunie vstupná hlava - v jednotlivých prípadoch -1, 0, $+1$: postupne sa presunie vstupná hlava o jedno políčko doľava, nechá sa hlava na mieste alebo sa presunie vstupná hlava o jedno políčko doprava. Pričom musí platiť, že $(q', -1) \notin \delta(q, \vdash)$ a $(q', +1) \notin \delta(q, \dashv)$, t.j. end-marker ohraničuje pozíciu vstupnej hlavy.

Ešte spomenieme, kedy vstupné slovo w 2NFA akceptuje: ak existuje postupnosť krokov zo začiatočného stavu g_0 so vstupnou hlavou na pozícii ľavého end-markera a končiaci v stave $q \in F$.

Formálnu definíciu konfigurácie, kroku výpočtu a jazyka neuvedieme, pretože sú zrejmé z predchádzajúceho popisu. Nasledujúcou definíciou zadefinujeme deterministický dvoj-smerný automat pomocou nedeterministického automatu.

Definícia 1.1.2. ***Dvoj-smerný deterministický konečný automat*** *(2DFA) A je 2NFA s vlastnosťou, že* $|\delta(q, a)| \leq 1$ *(*$q \in Q$ *,* $a \in \Sigma \cup \{\vdash, \dashv\}$*).*

V tomto momente už máme všetko potrebné pre pridanie žetónov a zadefinovanie dvoj-smerného nedeterministického žetónového automatu.

Definícia 1.1.3. ***K-žetónový nedeterministický konečný automat*** *(2NPA$_k$) A je šestica* $(Q, \Sigma, \delta, g_0, F, P)$*, kde :*

- Q *je konečná množina stavov*
- Σ *je konečná množina vstupnej abecedy, pričom* $\{\vdash, \dashv\} \notin \Sigma$
- $g_0 \in Q$ *je začiatočný stav*
- $F \subseteq Q$ *je množina akceptačných stavov*
- P *je množina žetónov a* $|P| = k$
- $\delta : Q \times 2^P \times (\Sigma \cup \{\vdash, \dashv\}, 2^P) \to 2^{Q \times \{-1,0,+1\} \times 2^P}$ *je prechodová funkcia.*

Žetónový nedeterministický automat funguje rovnako ako dvoj-smerný nedeterministický konečný automat, avšak navyše má schopnosť položiť na políčko (kde je vstupná hlava) žetón(y) a následne ho(ich) z políčka zdvihnúť.

2NPA$_k$ má zjavne iný tvar prechodovej funkcie oproti tej z definície pre 2NFA, pretože musí obsahovať navyše nejakú správu žetónov. Nech $(q', d, P_3) \in \delta(q, P_1, (a, P_2))$ je prechod z prechodovej funkcie, kde 2NPA$_k$ je v stave q, množina P_1 je množina zatiaľ nepoložených žetónov, $a \in \Sigma$ je symbol a P_2 je množina žetónov, ktoré sú položené na políčku (kde je

vstupná hlava), q' je stav v nasledujúcom kroku, $d \in \{-1, 0, +1\}$ označuje pohyb hlavy (doľava, žiadny, doprava), P_3 je množina žetónov zanechaných (položených) na políčku. Musí platiť $P_3 \subseteq P_1 \cup P_2$, t.j. na políčku je možné položiť len žetóny, ktoré sú nepoložené alebo sú už položené na aktuálnom políčku. Množina $(P_1 \cup P_2) - P_3$ je nová množina nepoložených žetónov po uskutočnený daného kroku výpočtu.

Podobne ako pri 2NFA neuvedieme definície jazyka, kroku výpočtu ani konfigurácie, pretože sú zrejmé a veľmi podobné s definíciami 2NFA(navyše obsahujú správu žetónov). Okrem toho ešte zadefinujeme prirodzene deterministickú verziu žetónového automatu.

Definícia 1.1.4. ***K*-žetónový deterministický konečný automat** *(2DPA$_k$) A je 2NPA$_k$ s vlastnosťou, že* $|\delta(q, P_1, (a, P_2))| \leq 1$ *(*$q \in Q$ *,* $a \in \Sigma \cup \{\vdash, \dashv\}$*,* $P_1, P_2 \subseteq P$*).*

Pre jednoduchšie označovanie, uvedieme nasledujúce označenie:

Poznámka 1.1.1. 2NPA $= \cup_i$ 2NPA$_i$. (t.j. zjednotenie cez všetky konečnožetónové nedeterministické konečné automaty)

Poznámka 1.1.2. 2DPA $= \cup_i$ 2DPA$_i$. (t.j. zjednotenie cez všetky konečnožetónové deterministické konečné automaty)

Poznámka 1.1.3. Rovnako, ako sme označili triedy automatov 2NPA$_k$, 2DPA$_k$, 2NPA a 2DPA, označíme aj triedy jazykov akceptovaných automatmi z príslušných tried. V ďalšom texte bude význam označenia zrejmý z kontextu.

1.2 Rôzne varianty žetónových automatov

Okrem štandardnej definícii žetóna pre nejaký automat, existujú aj trochu exotickejšie formy žetónov. Niektoré z nich spomenieme v nasledujúcej časti.

1.2.1 Abstraktné a fyzické značky

V jednom z prvých článkov, ktoré sa týkali žetónov v nejakej forme, konkrétne [BH67], uviedli výpočtové modely na dvoj-rozmerných (2D) vstupoch, ktoré využívali abstraktne značky alebo fyzické značky („markers“).

Výpočtový model na 2D vstupoch využívajúci konečný počet abstraktných značiek môže abstraktnú značku položiť na políčko vstupu a v prípade,

že daná abstraktná značka je už niekde na vstupe položená, na tomto mieste daná značka zmizne. Taktiež samozrejme danú abstraktnú značku možno zdvihnúť.

Naopak, výpočtový model na 2D vstupoch využívajúci konečný počet fyzických značiek môže fyzickú značku položiť, len ak ju má k dispozícii. Preto danú značku musí najprv z políčka zdvihnúť a až následne ju môže na iné políčko položiť.

Nie je ťažké vidieť, že pomocou k-fyzických značiek možno jednoducho simulovať k-abstraktných značiek.

Obrátene simulácia k-abstraktných značiek pomocou k-fyzických značiek je trochu obtiažnejšia, ale o to zaujímavejšia. Problém nastáva, ak treba položiť značku m_d, ktorá už bola niekde položená. Riešením je pamätanie si stavu, v ktorom bol simulovaný automat, keď videl niektorú naposledy videnú značku m_i. Teda, ak chce automat položiť značku m_d, nájde značku m_d na vstupe a zdvihne ju. Následne nájde značku m_i, prejde do stavu, ktorý má zapamätaný a simuluje pokiaľ netreba položiť m_d. Značku m_d položí na políčko vstupu a pokračuje so simuláciou.

Ako vidno z predchádzajúceho popisu abstraktné a fyzické značky sú pre spomínaný model ekvivalentné.

1.2.2 Žetónový automat s obmedzením na regulárne jazyky

Definícia takto obmedzeného automatu sa objavila v článku [GH96]. Zámerom definovania nového modelu bolo obmedzenie viac-žetónových automatov iba na regulárne jazyky, pretože v článku sa zaujímali o efektívnosť simulácie jednotlivých modelov medzi sebou (ekvivalentných s regulárnymi jazykmi).

Definícia 1.2.1. ***K-žetónový nedeterministický konečný automat (LIFO)*** *je $2NPA_k$ so žetónmi $p_1 \dots p_k$ s nasledujúcimi obmedzeniami:*

1. *Žetón p_{i+1} je možné položiť iba ak žetón p_i je už položený na vstupe. Žetón p_i môže byt zdvihnutý iba ak p_{i+1} nie je položený na vstupe. Význam tohto pravidla je, že položenie a zdvihnutie žetóna je v štýle zásobníka (LIFO).*

2. *Medzi časom, keď je p_{i+1} položený a p_i je zdvihnutý alebo p_{i+2} je položený, automat môže pracovať iba na podslove medzi pozíciou p_i a*

koncom vstupu, ktorý je v smere žetónu p_{i+1}. Navyše na tomto podslove sa správa ako 1-žetónový automat so žetónom p_{i+1}. Teda nemôže zdvihnúť, položiť a ani cítiť prítomnosť žiadneho iného žetónu okrem žetóna p_{i+1}.

Predchádzajúca definícia je trochu ťažkopádna, preto neformálne je sa možné na tento model pozerať ako na zásobník 1-žetónových automatov. V každom momente výpočtu pracuje na vstupnom slove iba automat, ktorý je na vrchole zásobníka. Keď je na vstup vložený nový žetón, tiež je vložený na zásobník 1-žetónový automat. Naopak automat je zo zásobníka vybraný, ak je žetón zo vstupného slova zdvihnutý. Navyše automat pracuje iba na podslove vstupného slova, ktorý je medzi predchádzajúcim žetónom a príslušným koncom vstupného slova.

V článku [GH96] tiež dokázali, že takto definovaní žetónový automat akceptuje iba regulárne jazyky.

Kapitola 2

Dvojsmerné žetónové automaty

V tejto kapitole uvedieme základné výsledky, ktoré sú v súčastnosti známe o dvojsmerných žetónových automatoch. Najprv sa budeme venovať výpočtovej sile žetónových automatov. Spomenieme ekvivalentné výpočtové modely (s rovnakou výpočtovou silou), ktoré sú v rámci formálnych jazykov a výpočtovej zložitosti známejšie. Detailnejšie rozoberieme odpoveď na otázku, či sa zvyšuje výpočtová sila so zvyšovaním počtu žetónov. Okrem iného spomenieme otvorený problém ekvivalencie deterministických a nedeterministických žetónových automatov. Na koniec sa budeme zaoberať popisnou zložitosťou žetónových automatov.

2.1 Ekvivalencia triedy jazykov 2NPA$_1$ a regulárnych jazykov

Je známe, že jazyky (ne)deterministických konečných automatov s iba jedným žetónom sú ekvivalentné s regulárnymi jazykmi. Teda keď pridáme ku konečnému automatu iba jeden žetón, jeho výpočtová sila sa nezväčší.

Dôkaz možno nájsť v článku [GH96], preto uvedieme len hlavnú myšlienku dôkazu, pričom technické detaily vynecháme.

V dôkaze sa postupne transformuje daný 2DPA$_1$ P na nedeterministický konečný automat. Výsledný automat sa nevytvára priamo, ale vytvorí sa 2-smerný konečný automat P' bez žetónu. Ak automat P pracoval na vstupnej abecede Σ, potom automat P' pracuje na abecede $\Sigma' = \{(a, p_1, \ldots, p_n) | a \in$

$\Sigma, n = |P|, p_i$ sú stavy automatu P alebo 0 }. Význam Σ' je nasledovný: p_i je stav, do ktorého sa automat P dostane, ak naposledy dane políčko opustil v stave q_i a medzi tým nepoužil žetón. Ak taký stav p_i neexistuje, potom $p_i = 0$. Automat P' na rozšírenej abecede pracuje rovnako ako automat P, avšak pri manipulácii so žetónom, ktorý nemá, využije informáciu zo vstupného symbolu.

Následne sa skonštruuje 2-smerný automat P'', ktorý skontroluje na vstupnom slove z abecedy Σ', či stavy $p_1, \ldots, p_n$ v symboloch vstupného slova sú navzájom konzistentné pre automat P.

Spojením automatu P'' a automatu P' dostávame 2-smerný automat P''', ktorý akceptuje vstupné slovo z rozšírenej abecedy Σ' práve vtedy, keď automat P akceptuje vstupné slovo z abecedy Σ.

V predposlednom kroku P''' transformujeme na deterministický konečný automat D.

V poslednom kroku v automate D nahradíme každý symbol rozšírenej abecedy $(a, p_1, \ldots, p_n)$ symbolom a, čím dostaneme výsledný nedeterministický konečný automat N, ktorý nedeterministicky uhádne stavy $p_1, \ldots, p_n$ pre každý znak vstupného slova. Pričom platí, že P akceptuje práve vtedy, keď N akceptuje.

2.2 Ekvivalencia NSPACE($\log(n)$) a triedy jazykov 2NPA

Na začiatok uvedieme definície tried jazykov, ktoré budeme v tejto časti porovnávať s triedou jazykov akceptovanou žetónovými automatmi.

Poznámka 2.2.1. NSPACE($\log(n)$) je trieda jazykov, ku ktorým existuje nedeterministický turingov stroj s priestorovým obmedzením logaritmus veľkosti vstupu. Označenie tejto triedy je NL.

Poznámka 2.2.2. DSPACE($\log(n)$) je trieda jazykov (podobne ako pre NSPACE($\log(n)$)), ku ktorým existuje deterministický turingov stroj s priestorovým obmedzením logaritmus veľkosti vstupu. Označenie tejto triedy je tiež L.

Dá sa dokázať, že trieda jazykov NSPACE($\log(n)$) a trieda jazykov 2NPA sú ekvivalentné. Spomenieme len myšlienku dôkazu, zvyšné detaily vynecháme.

Ak chceme simulovať nejaký konkrétny k-žetónový 2NPA na turingovom stroji s pamäťou $O(\log(n))$, možno to urobiť nasledovne: Kedže každý žetón sa môže vyskytovať na vstupe v n rôznych pozíciach, stačí $\log(n)$ bitov vstupnej pásky na binárne zakódovanie pozície žetóna. Preto turingov stroj simulujúci k-žetónový 2NPA má v pamäti uložené pozície žetónov a pri každom kroku, zisťuje koľko a ktoré žetóny sú na súčasnej pozícii hlavy vo vstupe ($O(\log(n))$ pamäti je postačujúce). Následne podľa prechodovej funkcie 2NPA prejde do stavu a zmení pozície žetónov. Takto vie odsimulovať celý výpočet k-žetónového 2NPA.

Opačne, ak máme nedeterministický turingov stroj s pamäťou $O(\log(n))$ a chceme ho simulovať na žetónovom 2NPA, postup je nasledovný: Pomocou žetónov (konkrétne ich pozíciou na vstupnej páske) možno kódovať informáciu. Jeden žetón, môže zakódovať $\log(n)$-bitov, z čoho vyplýva, že nejaký konštantný počet žetónov vie zakódovať $O(\log(n))$ pamäte. Preto simulácia nedeterministického turingovho stroja je na žetónovom 2NPA priamočiara, stačí využiť žetóny na kódovanie pamäte a pomocou nejakého konštantného počtu žetónov zisťovať hodnotu bitu v zakódovanej pamäti. Takto vie žetónový 2NPA (s konštantným počtom žetónov) odsimulovať cely výpočet nedeterministického turingovho stroja.

Kedže dané automaty sú schopné sa navzájom simulovať, vyplýva z toho, že $\mathrm{NSPACE}(\log(n)) = \mathrm{2NPA}$.

Pri porovnaní $\mathrm{DSPACE}(\log(n))$ a 2DPA možno použiť rovnakú myšlienku dôkazu ako sme už vyššie spomínali a preto: $\mathrm{DSPACE}(\log(n)) = \mathrm{2DPA}$.

2.3 Ekvivalencia $\mathrm{2DPA}_k$ a 2DSeFA(k)

Pre poriadok najprv uvedieme definíciu výpočtového modelu s ktorým budeme porovnávať výpočtovú silu dvojsmerných žetónových automatov.

Definícia 2.3.1. ***2DSeFA(k)*** *je dvoj-smerný deterministický konečný automat s k-citlivými hlavami („two-way deterministic finite automaton with k-sensing heads“). Jeho hlavy sa môžu pohybovať nezávisle, avšak navyše sú schopné sa navzájom detektovať, keď sa stretnú na rovnakom políčku vo vstupe.*

Tento výpočtový model je spomínaný v článkoch [Pet95] a [Pet97].

Je zjavné, že automat 2DSeFA(k) možno simulovať na 2DPA$_k$. Stačí, aby pozícia každého žetónu označovala pozíciu hlavy pre automat 2DSeFA(k). Následne pre každý krok spomenutého automatu, automat 2DPA$_k$ prejde vstup, zistí vstupné symboly na pozíciach žetónov, vzájomné pozície hláv a prejde do ďalšieho stavu (tiež prislúchajúco posunie žetóny), ktorý prislúcha prechodovej funkcii pre automat 2DSeFA(k).

Opačne, simulácia k-žetónového DPA na k-hlavovom 2DSeFA je problematickejšia. Samozrejme aj pri tejto simulácii možno pozíciu žetóna reprezentovať ako pozíciu hlavy na tom istom políčku, avšak problém nastáva, ak na páske sú položené všetky žetóny. Vtedy sú všetky hlavy automatu 2DSeFA zafixované (kvôli zapamätaniu si pozície žetónov) a nemožno v simulácii pokračovať. V takejto situácii sa DPA správa iba ako dvojsmerný deterministický konečný automat, čiže je schopný akceptovať iba slová z regulárnych jazykov. Preto výsledný k-hlavový 2DSeFA bude pracovať v dvoch módoch. V prvom móde bude automat, pokiaľ na vstupe neleží všetkých k-žetónov. V tomto móde, ak simulovaný automat položí žetón na vstup, tak sa zafixuje jedna hlava (asociovaná s daným žetónom) na danej pozícii vo vstupe. Ak sa žiaden žetón nepoloží a hlava sa v simulovanom automate iba pohne, všetky nezafixované hlavy sa pohnú spolu na to isté políčko. V druhom móde sa výsledný automat bude nachádzať, keď chce simulovaný automat položiť posledný žetón. V tomto móde nastane situácia, že žetóny rozdelia vstupné slovo na $k + 1$ podslov. Ako som už vyššie spomínal, tieto podslová by mali byť akceptovateľné 2-smernými deterministickými konečnými automatmi. Preto je potrebné si vytvoriť predvypočítanú tabuľku s informáciou o tom, že keď je 2DFA na pozícii jedného žetóna v nejakom stave, do akého stavu prejde, keď sa dostane prvý krát na pozíciu susedného žetóna. Takúto tabuľku je neskoro vypĺňať v druhom móde automatu, ale je ju možné vypĺňať ešte v prvom móde výsledného automatu. Z uvedeného vyplýva, že v druhom móde stačí len pomocou tabuľky zistiť nasledujúci stav susednej hlavy a odfixovať ju.

Predchádzajúce myšlienky nie sú síce formálne dôkazy, ale je zrejmé, že 2DPA$_k$ a 2DSeFA(k) sú ekvivalentné. Bližšie informácie o detailoch v dôkazoch sú už vo vyššie spomínaných článkoch.

Podobne výsledky možno dokázať o triedach 2NPA$_k$ a 2NSeFA(k) („two-way nondeterministic finite automaton with k-sensing heads“).

2.4 Hierarchia žetónových automatov

Ďalšia prirodzená otázka sa týka závislosti výpočtovej sily od počtu žetónov, ktoré má automat k dispozícii. Je zrejmé, že pridaním jedného žetónu neklesne výpočtová sila modelu. Ako si je možné všimnúť už v prvej časti tejto kapitoly, pridanie iba jedného žetónu k nula žetónom, nezmení výpočtovú silu modelu (zostane ekvivalentné triede regulárnych jazykov).

Naopak pre viac-žetónové automaty pridanie ďalšieho žetónu zvýši výpočtovú silu. Konkrétne v už spomínanom článku [Pet95] je dôkaz pre unárnu abecedu, že 2DSeFA(k) $\subsetneq$ 2DSeFA($k+1$). V predchádzajúcej časti som naznačil, že trieda 2DSeFA(k) je ekvivalentná s 2DPA$_k$. Z týchto dvoch faktov vyplýva, že aj 2DPA$_k \subsetneq$ 2DPA$_{k+1}$. V tomto článku je možné nájsť aj referenciu na iný článok, v ktorom sú údajne hierarchické výsledky závislé od počtu žetónov nad väčšou ako unárnou abecedou. Avšak tiež sa možno dočítať, že dôkaz nie je celkom korektný, pretože uvádzajú príklad, kedy daná konštrukcia(dôkaz) nezafunguje.

Jednoznačne bola dokázaná hierarchia dvojsmerných žetónových automatov (aspoň na unárnej abecede), t.j. 2DPA$_k \subsetneq$ 2DPA$_{k+1}$.

Podobné výsledky platia aj pre nedeterminizmus: 2NPA$_k \subsetneq$ 2NPA$_{k+1}$.

2.5 Ekvivalencia deterministických a nedeterministických žetónových automatov

Problém, ktorý sa dá formálne zapísať ako 2NPA $\stackrel{?}{=}$ 2DPA, je stále otvorený problém. Veľmi neformálne je tento problém o tom, či k ľubovolnému nedeterministickému žetónovému automatu existuje nejaký deterministický žetónový automat (samozrejme s konečným počtom žetónov).

Ako sme už spomínali v jednej z predchádzajúcich častí, platí: 2NPA = NSPACE(log(n)) a 2DPA = DSPACE(log(n)). Primárne sa tento otvorený problém objavil v tvare NSPACE(n) $\stackrel{?}{=}$ DSPACE(n). Avšak keby sme vedeli rozhodnúť o probléme 2NPA $\stackrel{?}{=}$ 2DPA (alebo NSPACE(log(n)) $\stackrel{?}{=}$ DSPACE(log(n)), kedže sú ekvivalentné), potom by sme z translačnej vety vedeli rozhodnúť aj problém NSPACE(n) $\stackrel{?}{=}$ DSPACE(n) aspoň pre prípad, keby sa triedy rovnali.

Spomínaný otvorený problém sa primárne študuje na turingovom stroji s obmedzenou logaritmickou páskou. Preto sa objavili aj čiastočné výsledky

z tejto oblasti pre rozličné triedy definované pomocou turingovho stroja:

$$\mathrm{L} \subseteq \mathrm{SL} \subseteq \mathrm{NL} \subseteq \mathrm{L}^2.$$

Poznámka 2.5.1. SL je trieda problémov, ktoré sú redukovateľné v logaritmickom priestore na problém USTCON.

Poznámka 2.5.2. USTCON („undirected s-t connectivity“) je problém, či existuje cesta medzi dvoma vrcholmi vo vstupnom neorientovanom grafe. USTCON je SL-úplný problém.

Dokonca bolo dokázané, že L = SL, pretože sa ukázalo: USTCON $\in$ L.

Poznámka 2.5.3. STCON („directed s-t connectivity“) je problém, či existuje cesta medzi dvoma vrcholmi vo vstupnom orientovanom grafe. STCON je NL-úplný problém.

Problém STCON je nielen z triedy NL, ale dokonca aj NL-ťažký. Preto STCON je NL-úplný problém.

Poznámka 2.5.4. Označenie $\mathbf{L}^2$ je DSPACE($\log(n)^2$).

Vlastnosť NL $\subseteq \mathrm{L}^2$ vyplýva zo Savitchovej vety. Význam je, že nedeterministický turingov stroj s $\log(n)$ pamäťou možno simulovať na deterministickom turingovom stroji s $\log(n)^2$ pamäťou. Toto je najlepšie doteraz nám známe horné ohraničenie triedy NL.

2.6 Popisná zložitosť

Na meranie zložitosti modelov sa používa výpočtová zložitosť ako aj popisná zložitosť. V tejto časti sa budeme zaoberať hlavne druhou spomenutou - popisnou zložitosťou. V tomto prípade miera popisnej zložitosti jednotlivých automatov je meraná ako počet stavov automatov.

V článku [GH96] je čiastočné zhrnutie výsledkov o popisnej zložitosti niektorých výpočtových modelov. Spomenuté výpočtové modely mali schopnosť akceptovať iba regulárne jazyky. Ako výpočtové modely porovnávali konečné automaty, ktoré mali rozdielne vlastnosti ako napríklad: nedeterminizmus, alternovanie alebo dvoj-smernosť. Spomínané vlastnosti navzájom autori kombinovali, čim vznikali rozdielne výpočtové modely. Detailnejšie výsledky možno nájsť v už spomínanom článku.

2.6.1 Model žetónových automatov s obmedzením na regulárne jazyky

Model žetónových automatov s obmedzením na regulárne jazyky sú spomenuté v tom istom článku [GH96]. Definíciu takto upravených žetónových automatov sme tiež spomínali v časti 1.2.2. Dôvod pre také obmedzenie žetónových automatov je snaha o zafixovanie výpočtovej sily automatu len na regulárne jazyky, aby výpočtová sila nezávisela od počtu žetónov.

Nasledujúce tvrdenia sa týkajú transformácii žetónových automatov na konečné automaty, pričom horná hranica transformácie je nejaká funkcia $g(n)$ taká, že ak žetónový automat má n stavov potom ekvivalentný konečný automat má najviac $g(n)$ stavov.

Poznámka 2.6.1. exp[k] je trieda funkcii, ku ktorým existuje konštanta $c > 1$ a polynóm $p(n)$, pričom rýchlosť rastu je rádovo $c^{c^{\cdots^{p(n)}}}$ (k-zložená exponenciálna funkcia).

Veta 2.6.1. *Horná hranica transformácie jedno-žetónového deterministického konečného automatu na nedeterministický konečný automat je* $exp[1]$. *(*$P \xrightarrow{1} E$ *v označení so spomínaného článku)*

Veta 2.6.2. *Horná hranica transformácie jedno-žetónového nedeterministického konečného automatu na nedeterministický konečný automat je* $exp[1]$. *(*E, $P \xrightarrow{1} E$*)*

Veta 2.6.3. *Horná hranica transformácie jedno-žetónového nedeterministického alternujúceho konečného automatu na nedeterministický konečný automat je* $exp[2]$. *(*E, A, $P \xrightarrow{2} E$*)*

Pre viac žetónov platí (definícia viac-žetónového automatu je z časti 1.2.2):

Veta 2.6.4. *Horná hranica transformácie* k*-žetónového konečného automatu na nedeterministický konečný automat je* $exp[k]$. *(* $kP \xrightarrow{k} E$*) Preto horná hranica transformácie* k*-žetónového konečného automatu na deterministický konečný automat je* $exp[k+1]$. *(* $kP \xrightarrow{k+1} DFA$*)*

V tom istom článku bolo ukázané, že existuje množina jazykov L_n, ktoré danú hornú hranicu dosahujú. Nasledujúci výsledok je potrebné chápať nasledovne: pre každú dĺžku vstupných slov n, existuje jazyk L_n, ktorý ak-

ceptuje nejaký minimálny $f(n)$-stavový žetónový automat a k nemu ekvivalentný minimálny nedeterministický konečný automat má aspoň $g(f(n))$ stavov. V tomto prípade $g(n)$ je dolná hranica transformácie.

Veta 2.6.5. *Dolná hranica transformácie k-žetónového konečného automatu na nedeterministický konečný automat je $exp[k]$. ($kP \underset{k}{\rightarrow} E$). Podobne dolná hranica transformácie k-žetónového konečného automatu na deterministický konečný automat je $exp[k+1]$. ($kP \underset{k+1}{\longrightarrow} DFA$)*

Všetky predchádzajúce tvrdenia a k nim prislúchajúce dôkazy je možné nájsť v už spomínanom článku.

2.6.2 Vzťah 1-žetónových a dvoj-smerných automatov

V jednom článku [GI09] bola položená jedna zaujímavá otázka, či k ľubovolnému n-stavovému 1-žetónovému nedeterministickému automatu, existuje 1-žetónový deterministický automat s nanajvýš $p(n)$-stavmi, kde $p(n)$ je nejaký pevne zadaný polynóm.

Podobný otvorený problém je aj na dvoj-smerných automatoch a to, či k ľubovolnému nedeterministickému dvoj-smernému automatu s n-stavmi, existuje deterministický dvoj-smerný automat s nanajvýš $p'(n)$ stavmi.($p'(n)$ je pevne zadaný polynóm)

Obidva spomenuté problémy sú otvorené, avšak ako bolo v článku [GI09] dokázané, sú navzájom závislé. Autori článku dokázali, že ak existuje polynóm $p'(n)$ pre problém s dvojsmernými ne- a deterministickými automatmi, potom existuje aj polynóm $p(n)$ pre problém s 1-žetónovými ne- a deterministickými automatmi.

Kapitola 3

Jednosmerné konečné automaty so žetónmi

Táto kapitola prináša vlastné výsledky z analýzy výpočtovej sily jednosmerných konečných automatov so žetónmi. Na úvod formálne zadefinujeme výpočtový model jednosmerných žetónových automatov. Následne v ďalších častiach ukážeme hierarchiu jednosmerných žetónových automatov vzhľadom na počet hláv a žetónov. Ku koncu kapitoly sa budeme zaoberať otázkou, či je možné pre jednosmerné žetónové automaty kompenzovať stratu žetóna pridaním jednej hlavy alebo niekoľko hláv.

3.1 Definícia modelu

Na úvod zadefinujeme jednosmerný žetónový automat iba obmedzením dvojsmerného žetónového automatu z definície 1.1.3, kde obmedzíme pohyb hláv len na jeden smer.

Definícia 3.1.1. ***P-žetónový jednosmerný nedeterministický konečný automat*** *(p-NPA) A je šestica* $(Q, \Sigma, \delta, g_0, F, P)$*, kde :*

- Q *je konečná množina stavov*
- Σ *je konečná množina vstupnej abecedy, pričom* $\dashv \notin \Sigma$
- $g_0 \in Q$ *je začiatočný stav*
- $F \subseteq Q$ *je množina akceptačných stavov*
- $P = \{1, \ldots, p\}$ *je množina žetónov*
- $\delta : Q \times 2^P \times (\Sigma \cup \{\dashv\}, 2^P) \to 2^{Q \times \{0,+1\} \times 2^P}$ *je prechodová funkcia.*

Avšak predchádzajúca definícia definuje iba automat, ktorý má len jednu hlavu a preto, keď na symbole slova zanechá nejaký žetón, tento žetón už opätovne nemôže automat znovu identifikovať (kedže je jednosmerný). Preto je zrejmé, že každý automat z predchádzajúcej definície možno simulovať pomocou jednosmerného nedeterministického konečného automatu. Rozšírime predchádzajúcu definíciu pridaním hláv, aby automat bol schopný využiť silu žetónov.

Definícia 3.1.2. ***K-hlavový p-žetónový jednosmerný nedeterministický konečný automat*** *((k,p)-NPA) A je šestica* $(Q, \Sigma, \delta, g_0, F, P)$*, kde*

- Q *je konečná množina stavov*
- Σ *je konečná množina vstupnej abecedy, pričom* $\dashv \notin \Sigma$
- $g_0 \in Q$ *je začiatočný stav*
- $F \subseteq Q$ *je množina akceptačných stavov*
- $P = \{1, \ldots, p\}$ *je množina žetónov*
- $\delta : Q \times 2^P \times \left[(\Sigma \cup \{\dashv\}) \times 2^P\right]^k \to 2^{Q \times \left[\{0,+1\} \times 2^P\right]^k}$ *je prechodová funkcia.*

Definícia 3.1.3. ***K-hlavový p-žetónový jednosmerný deterministický konečný automat*** *((k,p)-PA) A je (k,p)-NPA s nasledujúcim obmedzením:* $|\delta(q, P_0, [(a_1, P_1), \ldots (a_k, P_k)])| \leq 1$ *, pre* $q \in Q \wedge \forall i (P_i \in 2^P \wedge a_i \in (\Sigma \cup \dashv))$.

Poznámka 3.1.1. K-hlavový p-žetónový jednosmerný nedeterministický konečný automat v ďalšom texte budeme často označovať ako jednosmerný žetónový automat.

Ako si je možné všimnúť v predchádzajúcich definíciách je prechodová funkcia definovaná veľmi všeobecne a je potrebné detailnejšie zadefinovať korektné prechodové funkcie. Korektnosť prechodovej funkcie bude závisiet od spôsobu práce so žetónmi. Intuitívne korektná prechodová funkcia bude taká, ktorá bude používať iba žetóny, ktoré sú k dispozícii.

Definícia 3.1.4. ***Korektná prechodová funkcia*** *je prechodová funkcia* δ *z definície 3.1.2 s nasledujúcimi podmienkami na prácu so žetónmi : Ak* $(q', [(d_1, P'_1), \ldots (d_k, P'_k)]) \in \delta(q, P_0, [(a_1, P_1), \ldots (a_k, P_k)])$ *potom*

- $\bigcup_{i=0}^{k} P_i \supseteq \bigcup_{i=1}^{k} P'_i$*, t.j. možno použiť len žetóny, ktoré sú k dispozícii.*
- $\forall i, j : P'_i \cap P'_j = \emptyset$*, t.j. nemožno položiť jeden žetón na viac políčok.*
- *technická podmienka:* $a_i = \dashv \Rightarrow d_i = 0$*, t.j. ak je hlava na konci slova, nemôže ísť ďalej.*

Ešte zadefinujeme konfiguráciu, krok výpočtu ako aj jazyk žetónového automatu. Tieto definície bližšie popíšu ako vyzerá akceptačný výpočet na nejakom slove.

Definícia 3.1.5. ***Konfigurácia*** *(k,p)-NPA je* $(q, k_1, \ldots, k_k, p_1, \ldots, p_p)$*, kde* q *je stav automatu,* k_i *je index symbolu vstupného slova, ktorý i-ta hlava číta a* p_i *je index symbolu na ktorom je položený i-ty žetón(ak nie je položený, potom* $p_i = 0$*).*

Definícia 3.1.6. ***Krok výpočtu*** *(k,p)-NPA A na slove* $w = a_1 \ldots a_n$ *je relácia* $\vdash_{A,w}$ *na konfiguráciách definovaná nasledovne:*
$(q, k_1, \ldots, k_k, p_1, \ldots, p_p) \vdash_{A,w} (q', k'_1, \ldots, k'_k, p'_1, \ldots, p'_p) \iff$
$(q', [(d_1, P'_1), \ldots (d_k, P'_k)]) \in \delta(q, P_0, [(a_{k_1}, P_1), \ldots (a_{k_k}, P_k)])$*, pričom:*

- $k'_i = k_i + d_i$*, pre všetky* $i \in \{1, \ldots, k\}$
- $P_i = \{j | p_j = k_i\}$*, pre všetky* $i \in \{1, \ldots, k\}$ *a* $P_0 = \{j | p_j = 0\}$
- $P'_i = \{j | p'_j = k_i\}$*, pre všetky* $i \in \{1, \ldots, k\}$
- *ak* $p_i \neq p'_i$ *potom* $(\exists j) k_j = p_i$ *(ak sa žetón pohne, niektorá hlava musela byť na žetóne)*

Definícia 3.1.7. ***Jazyk*** *akceptovaný (k,p)-NPA A je množina slov* $L(A) = \{w | w' = w\dashv = a_1 \ldots a_n\dashv \wedge \exists q_F \in F_A \wedge \exists p_1, \ldots, p_p \in \{0, \ldots, n+1\} : (q_0, 0, \ldots, 0) \vdash^*_{A,w'} (q_F, n+1, \ldots, n+1, p_1, \ldots, p_p)\}$

Výpočtový model jednosmerných žetónových automatov už máme presne definovaný, preto tiež môžme zadefinovať množiny jazykov, ktoré sú schopné akceptovať žetónové automaty.

Definícia 3.1.8. ***Trieda jazykov (k,p)-NPA*** *je množina jazykov, ku ktorým existuje nejaký k-hlavový p-žetónový jednosmerný nedeterministický automat, ktorého prechodová funkcia je v zmysle definície 3.1.4 korektná.*

Definícia 3.1.9. ***Trieda jazykov (k,p)-PA*** *je množina jazykov, ku ktorým existuje nejaký k-hlavový p-žetónový jednosmerný deterministický automat, ktorého prechodová funkcia je v zmysle definície 3.1.4 korektná.*

Poznámka 3.1.2. Definíciu 3.1.2 je možné rozšíriť na konečný automat ktorý má „sensing heads“, t.j. ak dve hlavy sú na jednom symbole vstupu, potom automat vie o tom a tiež vie ktoré dve hlavy to sú. Podobne ako v predchádzajúcich definíciách možno zadefinovať triedu jazykov takýchto automatov, ktoré označíme **(k,p)-NPA-Sens**.

Poznámka 3.1.3. V definícii 3.1.2 je množina P množinou žetónov, ktoré sú navzájom rôzne. Avšak bolo by možné definíciu zmeniť tak, že počet žetónov by bol p, ale žetóny by boli rovnaké. Ak by automat mal navyše „sensing heads“, potom pomocou rovnakých žetónov a pomocou stavu vieme simulovať navzájom rôzne žetóny a teda pôvodný model žetónových automatov.

3.2 Hierarchie tried jednosmerných automatov

V predchádzajúcej časti zadefinované triedy jazykov žetónových automatov závisia od počtu žetónov a počtu hláv. Preto v tejto časti dokážeme, že pridanie hlavy jednosmernému žetónovému automatu zvýši výpočtovú silu. Takýto výsledok vlastne dokazuje existenciu hierarchií tried jazykov definovaných žetónovými automatmi s rovnakým počtom žetónov, ale s rôznym počtom hláv.

Podobne ukážeme, že pridanie žetóna jednosmernému žetónovému automatu zväčší výpočtovú silu. Z toho tiež bude vyplývať existencia hierarchií tried jazykov žetónových automatov s rovnakým počtom hláv, ale s rôznym počtom žetónov.

3.2.1 Hierarchia automatov vzhľadom na počet hláv

Ak chceme dokázať, že pridanie hlavy žetónovému automatu zväčší výpočtovú silu, potrebujeme nájsť vhodný jazyk. Takýto jazyk by mal mať vlastnosť, že je ho možné akceptovať k-hlavovým žetónovým automatom, ale ekvivalentný $(k-1)$-hlavový žetónový automat k nemu neexistuje. Ako neskôr ukážeme, nasledujúca množina jazykov bude spĺňať túto vlastnosť.

Definícia 3.2.1. *Jazyk* $L_k = \{w_1\#w_2\#\ldots\#w_s\#w_s\#\ldots\#w_2\#w_1 | (\forall i) w_i \in \{0,1\}^* \wedge s = \binom{k}{2}\}$ *,pre* $k \in \{1,2,\ldots\}$.

Lema 3.2.1. *Existuje (k,p)-NPA A, ktorý akceptuje jazyk* L_k.

Dôkaz. Uvedieme len neformálnu konštrukciu automatu A. Jazyk L_k pozostáva zo slov tvaru $w_1\#w_2\#\ldots\#w_{2\binom{k}{2}}$, ktorých podslová w_i a $w_{2\binom{k}{2}-i+1}$ sú rovnaké. Na začiatok si treba všimnúť, že k je konštanta, preto je možné si v stave pamätať na ktorých podslovách w_i sú hlavy vo vstupnom slove. Automat A potrebuje overiť, či vstupné slovo má daný tvar a či dvojice podslov

sú rovnaké. Overenie tvaru je zrejmé a porovnávanie podslov sa bude vykonávať v $k-1$ fázach. V prvej fáze sa pošle posledná hlava na $w_{2\binom{k}{2}-(k-1)+1}$ (t.j. $(k-1)$-te podslovo od konca) a ostatné hlavy sa postupne zoradia na $w_1, \ldots, w_{k-1}$. Následne sa porovná prvých $k-1$ podslov s poslednými $k-1$ podslovami. Prvých $k-1$ hláv posunieme na podslovo w_k. Tým sa skončí prvá fáza a začne ďalšia fáza, v ktorej nám stačí porovnať dvojice podslov okrem prvých $k-1$. Ďalšia fáza bude pokračovať podobne ako predchádzajúca avšak prvých a posledných $k-1$ podslov z pôvodného slova si už automat nebude všímať. V tejto fáze porovnáme $k-2$ podslov, v ďalšej fáze $k-3$ atď. Vcelku porovnáme $(k-1)+(k-2)+\ldots+1 = \frac{k(k-1)}{2} = \binom{k}{2}$. Takýto automat je schopný akceptovať jazyk L_k (aj bez použitia žetónov). □

Veta 3.2.1. *Trieda jazykov (k,p)-NPA $\subsetneq$ trieda jazykov (k+1,p)-NPA (napríklad jazyk L_{k+1}).*

Dôkaz. Myšlienka nasledovného dôkazu je z článku [YR78], v ktorom autori dokázali vetu pre jednosmerné konečné automaty. Preto nám stačí rozšíriť dôkaz a ukázať, že pre spomínaný jazyk pridanie nejakých žetónov konečnému automatu nepomôžu.

Zo štruktúry jazyka L_{k+1} vidno, že na vstupnom slove je potrebné porovnať $\binom{k+1}{2}$ dvojíc podslov w_i a $w_{2\binom{k+1}{2}-i+1}$. Avšak nejaké porovnanie môže spraviť len niektorá dvojica hláv automatu. Tiež z poradia porovnávaných podslov w_i vyplýva, že ak niektorá dvojica hláv automatu porovná dvojicu podslov w_i a $w_{2\binom{k+1}{2}-i+1}$, potom táto istá dvojica hláv už nemôže porovnať žiadnu inú dvojicu podslov. Kedže k-hlavový žetónový automat má len $\binom{k}{2}$ dvojíc hláv, nie je schopný porovnať všetky dvojice podslov, čo je problém. Pridaním žetónov sa len veľmi málo zmení, pretože spomínaný problém sa neodstráni. Stále bude menej dvojíc hláv ako je potrebné na akceptáciu jazyka L_{k+1}.

Najprv si treba uvedomiť, že jazyk $L_{k+1} \in$ (k+1,p)-NPA, čo je priamy dôsledok lemy 3.2.1.

V druhej časti dokážeme, že $L_{k+1} \notin$ (k,p)-NPA. Dôkaz bude postupovať sporom, teda nech existuje nejaký (k,p)-NPA A, ktorý akceptuje jazyk L_{k+1}. Spor sa nám podarí ukázať na podmnožine jazyka L_{k+1}, konkrétne množine $L^m_{k+1} = \{w_1\#w_2\#\ldots\#w_s\#w_{s+1}\#\ldots\#w_{2s-1}\#w_{2s} | (\forall i) w_i \in \{0,1\}^m \wedge w_i = w_{2s-i+1} \wedge s = \binom{k+1}{2}\}$. Kedže L^m_{k+1} je špeciálna podmnožina L_{k+1}, NPA

A musí akceptovať každé slovo z jazyka L_{k+1}^m. Z toho ukážeme, že existuje nejaké slovo, ktoré nie je v jazyku L_{k+1}, ale NPA A ho akceptuje (čo bude želaný spor).

Konfigurácia NPA A je $(q, k_1, \ldots, k_k, p_1, \ldots, p_p)$ jednoznačne určená stavom automatu, k-pozíciami hláv automatu a p pozíciami žetónov na vstupnej páske. Pre danú konfiguráciu budeme hovoriť, že *typ konfigurácie* na slovách z L_{k+1}^m je usporiadaná k-tica $(\lceil k_1/(m+1) \rceil, \ldots, \lceil k_k/(m+1) \rceil)$. Typ konfigurácie vlastne vyjadruje v ktorom podslove $w_i\#$ je každá hlava automatu.

Vyberme si ľubovolné slovo u z jazyka L_{k+1}^m, k nemu existuje postupnosť konfigurácii (výpočet) pre slovo $u : a_1(u), \ldots, a_{d_u}(u)$. Takýchto výpočtov na slove u môže existovať viacero (kedže NPA A je nedeterministický), nám avšak bude postačovať, že existuje aspoň jeden a tento výpočet budeme bez ujmy na všeobecnosti považovať za jediný akceptačný výpočet na slove u. Z už spomínanej postupnosti konfigurácii pre slovo u vytvoríme podpostupnosť konfigurácii $b_1(u), \ldots, b_{d'_u}(u)$ nasledovne: vyberieme prvú konfiguráciu $a_1(u)$ a každú konfiguráciu $a_{i+1}(u)$, pre ktorú platí, že typ konfigurácie $a_i(u)$ je rôzny od typu konfigurácie $a_{i+1}(u)$. Takúto podpostupnosť konfigurácii nazveme *pattern slova* u. Význam patternu slova u je zrejmý z konštrukcie, t.j. popisuje postupnosť konfigurácii NPA A na slove u, avšak nie cely výpočet, ale iba miesta výpočtu, kde sa niektorá hlava posunie z jedného podslova $w_i\#$ na nasledujúce podslovo $w_{i+1}\#$. Treba si všimnúť, že dĺžka patternu d'_u na slovách z L_{k+1}^m je najviac $k \cdot 2\binom{k+1}{2} + 1$. Kedže sme ohraničili zhora dĺžku patternu, môžme ohraničiť zhora aj počet rôznych patternov:

$$P \leq \underbrace{\left(\underbrace{|K|}_{\text{(počet stavov)}} \cdot \underbrace{\left(2\binom{k+1}{2}(m+1) \right)^{k+p}}_{\text{(rôzne pozície hláv a žetónov)}} \right)^{k \cdot 2\binom{k+1}{2}+1}}_{\text{všetky možné konfigurácie}}$$

Slová z jazyka L_{k+1}^m rozdelíme do množín $M_1, M_2, \ldots$ podľa toho, aký pattern im prislúcha. Nech M_1 je najpočetnejšia množina slov s rovnakým patternom a kedže vieme, že počet rôznych patternov slov je P, množina M_1 obsahuje aspoň $\dfrac{2^{\binom{k+1}{2}m}}{P}$ slov.

Keď sa zamyslíme nad štruktúrou jazyka L_{k+1}^m všimneme si, že je potrebné v ňom porovnať $s = \binom{k+1}{2}$ podslov w_i a w_{2s-i+1}. Avšak ak nejaký NPA automat porovnáva dve podslová w_i a w_{2s-i+1} s dvomi konkrétnymi hlavami k_1 a k_2, táto istá dvojica hláv k_1 a k_2 už nemôže porovnať nijakú inú dvojicu podslov w_j a w_{2s-j+1} (vyplýva zo štruktúry slov). Význam porovnávať dve slová sa rozumie v zmysle, že súčasne nejaká hlava je na podslove w_i a nejaká iná hlava je na podslove w_{2s-i+1}. Jazyk L_{k+1}^m vynucuje až $\binom{k+1}{2}$ porovnaní podslov. NPA A má len k hláv, pričom počet všetkých dvojíc hláv NPA A je len $\binom{k}{2}$, t.j. maximálne je schopný porovnať $\binom{k}{2}$ dvojíc podslov. Z toho vyplýva, že existuje aspoň jedna dvojica podslov w_i a w_{2s-i+1}, ktoré NPA A neporovná.

Slová v množine M_1 majú rovnaké prislúchajúce patterny, t.j. neporovnávajú rovnaké dvojice podslov w_j a w_{2s-j+1} (kde j je možné zistiť z prislúchajúceho patternu). Takýchto neporovnávaných dvojíc podslov môže byť aj viac, preto za j je možné zvoliť ľubovolnú neporovnávanú dvojicu podslov w_j a w_{2s-j+1}. Opätovne slová z množiny M_1 prerozdelíme do množín $N_1, N_2, \ldots$ podľa toho, aké podslová $w_1, \ldots, w_{j-1}, w_{j+1}, \ldots, w_s$ slovo obsahuje (podslovo w_j môže byt rôzne). Podobne bez ujmy na všeobecnosti, nech N_1 je najpočetnejšia množina, ktorej veľkosť je aspoň $\frac{|M_1|}{2^{(s-1)m}}$, čo je aspoň $\frac{2^m}{P}$. Pre dostatočne veľké m je veľkosť množiny N_1 aspoň 2.

Potom ale existujú dve rozdielne slová $x, y \in N_1$, ktoré sa líšia iba na i-tych podslovách od začiatku a od konca slova:

$$x = x_1\# \ldots \#x_i\# \ldots \#x_{2s-i+1}\# \ldots \#x_{2s}$$

$$y = y_1\# \ldots \#y_i\# \ldots \#y_{2s-i+1}\# \ldots \#y_{2s}$$

Následne môžeme vytvoriť slovo z so slova x substituovaním podslova y_{2s-i+1} za x_{2s-i+1}:

$$z = x_1\# \ldots \#x_i\# \ldots \#y_{2s-i+1}\# \ldots \#x_{2s}$$

Slovo z zjavne nepatrí do jazyka L_{k+1}, pretože slová x a y sú rôzne slová. Kedže NPA A akceptuje slová x aj y, pričom patterny oboch slova sú rovnaké, dá sa ukázať, že akceptuje aj slovo z, čo je hľadaný spor.

Dôkaz, že NPA A akceptuje aj slovo z pozostáva z konštrukcie postupnosti konfigurácii (výpočtu), ktorý bude korektný a zároveň akceptačný.

Jediné, čo máme k dispozícii sú slová x, y a fakt, že prislúchajúce patterny sú totožné. Akceptačný výpočet vieme poskladať z akceptačného výpočtu prislúchajúceho slovu x a keď sa nejaká hlava dostane na podslovo y_{2s-i+1}, výpočet bude pokračovať ako na akceptačnom výpočte slova y. Keď hlava opustí podslovo y_{2s-i+1}, opäť sa pokračuje na akceptačnom výpočte pre slovo x. Kedže podslová x_i a y_{2s-i+1} sa neporovnávajú, týmto spôsobom možno vytvoriť korektný akceptačný výpočet pre z. □

Dôsledok 3.2.1. *Trieda jazykov (k,p)-PA $\subsetneq$ trieda jazykov (k+1,p)-PA (príklad jazyk L_{k+1}).*

Dôkaz. Z dôkazu lemy 3.2.1 vyplýva, že jazyk $L_{k+1} \in$ (k+1,p)-PA, pretože v konštrukcií automatu z lemy sme nevyužívali nedeterminizmus.

Z dôkazu vety 3.2.1 je zrejmé, že $L_{k+1} \notin$ (k,p)-NPA a teda aj $L_{k+1} \notin$ (k,p)-PA, čo bolo treba dokázať. □

Poznámka 3.2.1. Predchádzajúce tvrdenia môžeme prirodzene rozšíriť aj na žetónové automaty so „sensing heads“, konkrétne:

$$\text{(k,p)-NPA-Sens} \subsetneq \text{(k+1,p)-NPA-Sens}$$

Zdôvodnenie je zrejmé z vety 3.2.1 a jej dôkazu, v ktorom nezáležalo, či hlavy sú alebo nie sú „sensing“.

Veta 3.2.2. *Jazyk L_{k+1} nepatrí do triedy jazykov $\bigcup_{p}^{\infty}(k,p)$-NPA.*

Dôkaz. Sporom. Nech existuje (k,p)-NPA, ktorý akceptuje jazyk L_{k+1}. Kedže p je konštanta, možno zopakovať dôkaz z vety 3.2.1, čím dostaneme spor. □

Dôsledok 3.2.2. *Existuje jazyk, ktorý je v triede jazykov $(k+1,p)$-NPA, avšak nie je v triede jazykov $\bigcup_{p=0}^{\infty}(k,p)$-NPA.*

Predchádzajúci výsledok vlastne znamená, že jednu hlavu automatu nevieme nahradiť ľubovolným konštantným počtom žetónov.

3.2.2 Hierarchia automatov vzhľadom na počet žetónov

V tejto časti sa budeme zaoberať otázkou, či pridanie jedného žetóna zväčší výpočtovú silu automatu. Najskôr ukážeme pre 2-hlavové žetónové automaty, že pridaním žetónu nastane zväčšenie výpočtovej sily. Následne tento výsledok rozšírime aj na viac-hlavové automaty.

3.2.2.1 Hierarchia dvoj-hlavových automatov vzhľadom na počet žetónov

Na začiatok, podobne ako pri hlavách, je potrebné nájsť vhodný jazyk. Jazyk by mal byť akceptovateľný nejakým 2-hlavovým p-žetónovým automatom, avšak nemal by existovať ekvivalentný 2-hlavový $(p-1)$-žetónový automat. Jazyk s takou vlastnosťou naozaj existuje, ako ukážeme v tejto časti.

Označenie. Ak $w \in \{0,1\}^*$, potom $\overline{w}$ je bitová negácia slova w.

Definícia 3.2.2. *jazyk* $L_p = \{w_1 a_1 w_2 a_2 \dots a_p w_{p+1} \# w_1 \overline{a_1} w_2 \overline{a_2} \dots \overline{a_p} w_{p+1} \# w_1 * w_2 * \dots * w_{p+1} | (\forall i)(w_i \in \{0,1\}^* \wedge a_i \in \{0,1\})\}$ *,pre* $p \in \{0,1,\dots\}$.

Lema 3.2.2. *Existuje (2,p)-NPA A, ktorý akceptuje jazyk* L_p.

Dôkaz. Automat A bude na vstupných slovách pracovať nasledovne: Prvá hlava automatu zostane na začiatku slova, pričom druhá hlava sa presunie za prvý symbol # zo vstupného slova. Následne sa budu postupne porovnávať dvojice symbolov pod hlavami, pričom ak nastane nezhoda symbolov, druhá hlava položí žetón na vstupné políčko. Tento proces sa bude opakovať pokým obidve hlavy súčasne neprečítajú #. Automat A pokračuje overovaním rovnosti symbolov vo vstupom slove pod hlavami, pokým prvá hlava nedetekuje prítomnosť žetónu. V tom prípade sa skontroluje prítomnosť $*$ pod druhou hlavou a pokračuje ako predtým až pokým nedočíta slovo. Takýto automat A akceptuje jazyk L_p, pričom mu postačujú dve hlavy a p žetónov. □

Veta 3.2.3. *Trieda jazykov (2,p)-NPA* $\subsetneq$ *trieda jazykov(2,p+1)-NPA (príklad jazyk* L_{p+1}*).*

Dôkaz. Pozostáva z dvoch časti. V prvej je potrebné ukázať $L_{p+1} \in$ (2,p+1)-NPA, čo je zrejmé z lemy 3.2.2.

V druhej časti treba ukázať, že $L_{p+1} \notin$ (2,p)-NPA :

Ak máme slovo $w \in L_{p+1}$, stačí si všimnúť, že pozostáva z troch časti $w = s_1 \# s_2 \# s_3$ a $|s_1| = |s_2| = |s_3|$. Pričom, k zadanému podslovu s_2 a každému výberu (p+1) pozícii (kde nastanú rozdiely medzi s_1 a s_2) prislúcha jednoznačne slovo $s1$ aj $s3$.

Pre jednoduchosť označíme $m = |s_2|$. Pri NPA na jednom slove môže existovať viacero akceptačných výpočtov, preto ďalej budeme uvažovať, že na každom slove z jazyka vyberieme presne jeden akceptačný výpočet. V podstate si pre každé slovo zvolíme kandidáta z akceptačných výpočtov a

ostatné akceptačné výpočty ignorujeme. Takéto zjednodušenie si môžeme v nasledujúcom dôkaze dovoliť, lebo budeme vyžadovať iba existenciu nejakého výpočtu, nie množstvo akceptačných výpočtov na konkrétnom slove z jazyka.

Pri dôkaze, že $L_{p+1} \notin$ (2,p)-NPA, budeme postupovať sporom: nech existuje (2,p)-NPA A, ktorý akceptuje jazyk L_{p+1}. Môžeme predpokladať bez ujmy na všeobecnosti, že automat A s hlavami vždy skončí na konci vstupného slova.

Nech W_m je množina všetkých slov $w \in L_{p+1}$, ktorých dĺžka stredných slov s_2 je m. Zrejme platí, že $|W_m| = 2^m \cdot \binom{m}{p+1}$.

Pre každé slovo $u = \{0,1\}^*$, za V_u označme množinu slov tvaru $s_1\#u\#s_3$ z množiny W_m. Ďalej za V_u^a (kde $a \in <0,1>$) označíme množinu slov $w = s_1\#u\#s_3$ z V_u, ktorých rozdiely medzi podslovami s_1 a u nenastavajú na okrajoch dĺžky $\lceil a \cdot m \rceil$, t.j. $s_1 = u_p s_{1,j} u_s$ a $u = u_p u_j u_s$, kde $|u_p| = |u_s| = \lceil a \cdot |u| \rceil$.

Na každom slove w z W_m nastane konfigurácia, kedy prvý krát prvá hlava ukazuje na druhý # a druhá hlava je niekde medzi začiatkom slova a druhým # , t.j. $\underbrace{s_1\#s_2\#}_{\uparrow_2}\overset{\uparrow_1}{s_3}$. Takúto konfiguráciu pre slovo w nazveme *prvou-konfiguráciou* (prislúchajúcou slovu w). Ku každému slovu w z W_m jednoznačne prislúcha prvá konfigurácia, pretože pre každé slovo uvažujeme jeden pevne zvolený akceptačný výpočet.

V nasledujúcej leme dokážeme, že na väčšine slov musí byť v prislúchajúcich prvých konfiguráciách druhá hlava „blízko“ prvého #.

Lema 3.2.3. *Nech* $a \in \left(0, \frac{1}{4}\right)$, *potom platí:* $(\exists m_a \in \mathbb{N})(\forall m \geq m_a)(\exists u \in \{0,1\}^m)$: *aspoň* $\frac{1}{2} \cdot V_u^a$ *slov bude mať v prvých konfiguráciách druhú hlavu vzdialenú najviac* $\lceil a \cdot m \rceil$ *od prvého #.*

Lema vlastne tvrdí, že existuje pattern (stredná časť slova u) taký, že aspoň na $1/2 \cdot V_u^a$ slov bude v prvých konfiguráciách druhá hlava vzdialená najviac $\lceil a \cdot m \rceil$ od prvého #. Intuitívne, keby táto lema neplatila, tak by sa pre väčšinu slov neporovnávali väčšie časti podslov, čím by bolo možné akceptovať slová, ktoré nie sú z jazyka.

Zatiaľ predpokladajme, že lema platí, t.j. existuje nejaký dostatočne veľký pattern u pre ktorý aspoň na $\frac{1}{2} \cdot V_u^a$ slovách $w = s_1\#s_2\#s_3$ sú prvé

konfigurácie nasledovného tvaru: $s'_1 \underbrace{\overbrace{s''_1}^{\lceil a\cdot m\rceil} \# \overbrace{s'_2}^{\lceil a\cdot m\rceil}}_{\uparrow_2} s''_2 \# s_3$ $\uparrow_1$

Avšak všetkých možných prvých konfigurácií automatu A predošlého tvaru môže byť iba $\underbrace{(2m+2)^p}_{\text{pozície p-žetónov}} \cdot \underbrace{K}_{\text{počet stavov}} \cdot \underbrace{(2\cdot\lceil a\cdot m\rceil+1)}_{\text{možné pozície 2.hlavy}} \cdot \underbrace{2}_{\text{poradie hláv}}$. Kedže existuje aspoň $1/2\cdot V_u^a = 1/2\cdot\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}$ slov, tak pre dostatočne veľké m a veľmi malé a, existujú aspoň dve slová z množiny slov V_u^a, $s_1\#s_2\#s_3$ a $s'_1\#s_2\#s'_3$, ktorým prislúchajúce prvé konfigurácie sú totožné. Potom ale pre slovo $w = s_1\#s_2\#s'_3$ vieme poskladať akceptačný výpočet z dvoch akceptačných výpočtov a to tak, že výpočet začne akoby na slove $s_1\#s_2\#s_3$ až do momentu prvej konfigurácie, v ktorom výpočet bude pokračovať od prvej konfigurácie ako na akceptačnom výpočte slova $s'_1\#s_2\#s'_3$. Čím dostaneme korektný akceptačný výpočet na slove w. Lenže $w \notin L_{p+1}$, čo je spor.

Dôkaz lemy 3.2.3. sporom: $(\forall m_a)(\exists m \geq m_a)(\forall u \in \{0,1\}^m)$: aspoň $\frac{1}{2}\cdot V_u^a$ slov bude mať v prvých konfiguráciách druhú hlavu vzdialenú **viac ako** $\lceil a\cdot m\rceil$ od prvého #.

V nasledujúcich úvahách zvolíme m dostatočne veľké, aby všetky argumenty boli platné.

V slove (patterne) $u = u_p p u_s$ označíme jeho jednotlivé časti: u_p bude prefix patternu, p bude jadro patternu a u_s bude sufix patternu. Za X_p označíme množinu slov vytvorenú zjednotením množín $V^a_{u_p p u_s}$ (cez všetky prefixy a sufixy patternu), kde jadro patternu je p a prefixy u_p a sufixy u_s patternu majú dĺžky $\lceil a\cdot m\rceil$. Formálne: $X_p = \bigcup_{u=u_p p u_s} V_u^a$, kde $|u_p| = |u_s| = \lceil a\cdot m\rceil$.

Zvoľme si nejaké jadro patternu p, k nemu prislúchajú slová z X_p. Zrejme platí: $|X_p| = 2^{2\cdot\lceil a\cdot m\rceil}\cdot\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}$.

Kedže pre všetky patterny bude mať automat v prvých konfiguráciách druhú hlavu vzdialenú **viac ako** $\lceil a\cdot m\rceil$ od prvého #, tak aj pre všetky jadrá patternov to bude platiť. Teda v aspoň $1/2\cdot|X_p|$ slov bude v prislúchajúcich prvých konfiguráciách druhá hlava vzdialená **viac ako** $\lceil a\cdot m\rceil$ od prvého #, takéto slová zaradíme do množiny M. Množinu slov $w = s_1\#s_2\#s_3 \in M$ možno rozdeliť do množín $M_1, M_2\ldots$ podľa toho, kde nastávajú rozdiely

medzi slovami s_1 a s_2 - možných pozícii je $\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}$. Tvrdíme, že existuje najpočetnejšia množina (bez ujmy na všeobecnosti) M_1, ktorej mohutnosť je aspoň $\frac{1/2\cdot|X_p|}{\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}} = \frac{1/2\cdot 2^{2\cdot\lceil a\cdot m\rceil}\cdot\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}}{\binom{m-2\cdot\lceil a\cdot m\rceil}{p+1}} = \frac{1}{2}\cdot 2^{2\cdot\lceil a\cdot m\rceil} = 2^{2\cdot\lceil a\cdot m\rceil-1}$.

Množina M_1 je množina slov $w = s_1\#s_2\#s_3$, ktorých prefixy a sufixy patternu môžu byt rôzne, ale jadro patternu majú rovnaké ako aj rovnaké pozície rozdielov medzi s_1 a s_2 (rozdiely nastávajú iba v jadre patternu). Pre všetky slová z M_1 v prvej konfigurácii je druhá hlava vzdialená viac ako $\lceil a\cdot m\rceil$ od prvého # a jej pozícia je buď napravo alebo naľavo od prvého #: $s_1\overbrace{s_1'}^{\lceil a\cdot m\rceil}\#\overbrace{u_p}^{\lceil a\cdot m\rceil}\underbrace{pu_s\#}_{\uparrow_2}{}^{\uparrow_1}s_3$ alebo $\underbrace{s_1}_{\uparrow_2}\overbrace{s_1'}^{\lceil a\cdot m\rceil}\#\overbrace{u_p}^{\lceil a\cdot m\rceil}pu_s\#{}^{\uparrow_1}s_3$

Slová $w\in M_1$ rozdelíme do množín $S_1, S_2\ldots$ podľa toho v akej prislúchajúcej prvej konfigurácii sa automat ocitne. Možných prvých konfigurácii je menej ako $(2\cdot m+2)^p\cdot K\cdot(2\cdot m+2)\cdot 2$. Nech najpočetnejšia množina (bez ujmy na všeobecnosti) je S_1 a preto jej veľkosť je aspoň

$$\frac{|M_1|}{(2\cdot m+2)^p\cdot K\cdot(2\cdot m+2)\cdot 2} = \frac{2^{2\cdot\lceil a\cdot m\rceil-1}}{(2\cdot m+2)^{p+1}\cdot K\cdot 2}$$

Môžu nastať práve dve možnosti podľa pozície druhej hlavy v prvých konfiguráciách prislúchajúcich slovám z množiny S_1:

1. druhá hlava v prvej konfigurácii ľubovolného slova $w\in S_1$ je napravo od prvého #: $s_1\overbrace{s_1'}^{\lceil a\cdot m\rceil}\#\overbrace{u_p}^{\lceil a\cdot m\rceil}\underbrace{pu_s\#}_{\uparrow_2}{}^{\uparrow_1}s_3$

   Sufixov patternu môže byť až $2^{\lceil a\cdot m\rceil}$, preto v množine S_1 existujú aspoň dve slová w_1 a w_2 (ak $|S_1|>2^{\lceil a\cdot m\rceil}$), ktorých sufixy patternu sú rovnaké (ako aj prislúchajúce prvé konfigurácie) a prefixy patternu sú rôzne. Takže slová w_1 a w_2 majú tvar:

   $$w_1 = u_ps_1\#u_ppu_s\#u_ps_3$$

   $$w_2 = u_p's_1\#u_p'pu_s\#u_p's_3$$

Z týchto dvoch slov w_1 a w_2 poskladáme slovo:

$$w_3 = u_p s_1 \# u_p p u_s \# u'_p s_3$$

pre ktoré bude existovať akceptačný výpočet, ale $w_3 \notin L_{p+1}$. Čo je hľadaný spor.

2. druhá hlava v prvej konfigurácii ľubovolného slova $w \in S_1$ je naľavo od prvého #: $\underbrace{s_1}_{\uparrow_2} \overbrace{s'_1}^{\lceil a \cdot m \rceil} \# \overbrace{u_p}^{\lceil a \cdot m \rceil} \underset{\uparrow_1}{p u_s \# s_3}$

   V S_1 sú slová tvaru $u_p s_1 u_s \# u_p p u_s \# u_p s_3 u_s$, ktorých prislúchajúce prvé konfigurácie vyzerajú nasledovne $\underbrace{u_p s_1}_{\uparrow_2} u_s \# u_p p \underset{\uparrow_1}{u_s} \# u_p s_3 u_s$. Tiež si je možné všimnúť, že dve slová tvaru $u_p s_1 u_s \# u_p p u_s \# u_p s_3 u_s$ z S_1 sa líšia iba v podslovách u_p a u_s, pričom podslová p, s_1 a s_3 sú pre všetky slová z S_1 rovnaké.

   Podobne ako sme zadefinovali prvú-konfiguráciu prislúchajúcu nejakému slovu, v nasledovnej časti ešte zadefinujeme pojmy ako *druhá-konfigurácia*, *tretia-konfigurácia* a *nultá-konfigurácia*. Na každom slove w z S_1 nastane konfigurácia, kedy prvý krát druhá hlava ukazuje na prvý # a prvá hlava je niekde medzi prvým # a koncom vstupného slova, t.j. $\underbrace{s_1 \# s_2 \# s_3}_{\uparrow_1}$. Takúto konfiguráciu pre slovo w nazveme druhou-konfiguráciou (prislúchajúcou slovu w).

   Podobne na každom slove w z S_1 nastane konfigurácia, kedy prvý krát druhá hlava ukazuje na druhý # a prvá hlava je niekde medzi druhým # a koncom slova, t.j. $s_1 \# \underbrace{s_2 \# s_3}_{\uparrow_1}$. Takúto konfiguráciu pre slovo w nazveme treťou-konfiguráciou (prislúchajúcou slovu w).

   Tiež podobne zadefinujeme nultú-konfiguráciu prislúchajúcu nejakému slovu w. Na každom slove w z S_1 nastane konfigurácia, kedy prvý krát prvá hlava ukazuje na prvý # a druhá hlava je niekde medzi začiatkom slova a prvým #, t.j. $\underbrace{s_1 \# s_2 \# s_3}_{\uparrow_2}$. Takúto konfiguráciu pre slovo w nazveme nultou-konfiguráciou (prislúchajúcou slovu w).

Ku každému slovu w z W_m jednoznačne prislúcha druhá konfigurácia, tretia konfigurácia ako aj nultá konfigurácia.

Pre všetky slová z S_1 platí, kedže prislúchajúce prvé konfigurácie sú tvaru $\underset{\uparrow_2}{\underbrace{u_p s_1}}\, u_s \# u_p p \underset{\uparrow_1}{u_s \# u_p s_3 u_s}$, prislúchajúce druhé konfigurácie musia byť tvaru $u_p s_1 \underset{\uparrow_2}{u_s \# u_p p} u_s \underset{\uparrow_1}{\underbrace{\# u_p s_3 u_s}}$ a prislúchajúce nulté konfigurácie musia byť tvaru $\underset{\uparrow_2}{\underbrace{u_p s_1}}\, \underset{\uparrow_1}{u_s \# u_p p u_s \# u_p s_3 u_s}$.

Slová $w \in S_1$ rozdelíme do množín $N_1, N_2 \ldots$ podľa toho v akej prislúchajúcej druhej konfigurácii, tretej konfigurácii a nultej konfigurácii sa automat počas výpočtu ocitne. Možných konfigurácii (druhých, tretích ako aj nultých) je menej ako $(3 \cdot m + 3)^p \cdot K \cdot (3 \cdot m + 3)$. Nech najpočetnejšia množina je N_1 a jej veľkosť je aspoň $\frac{|S_1|}{((3 \cdot m + 3)^{p+1} \cdot K)^3}$.

Prefixov patternu môže byť až $2^{\lceil a \cdot m \rceil}$, preto slová množiny N_1 rozdelíme do množín $R_1, R_2, \ldots$ podľa prefixu patternu. Bez ujmy na všeobecnosti, nech R_1 je najpočetnejšia z nich. Preto veľkosť množiny R_1 je aspoň $\frac{|N_1|}{2^{\lceil a \cdot m \rceil}}$. Slová z R_1 sa líšia iba v sufixoch patternu.

Pre všetky slová z R_1 platí, že prislúchajúce prvé, druhé, tretie ako aj nulté konfigurácie a tiež prefixy patternu sú rovnaké. Slová z R_1 majú tvar: $s_1 u_{s1} u_{s2} \# s_2 u_{s1} u_{s2} \# s_3 u_{s1} u_{s2}$, pričom $|u_{s1}| = |u_{s2}| = \frac{\lceil a \cdot |u| \rceil}{2}$ (pre jednoduchosť zápisu predpokladáme, že $\lceil a \cdot |u| \rceil$ je párne číslo). Podslová s_1, s_2 aj s_3 sú pre všetky slová z R_1 rovnaké (líšia sa iba v podslovách u_{s1} a u_{s2}).

Keď sa pozrieme na ľubovolné slovo z množiny R_1 a k nemu prislúchajúcu druhú konfiguráciu, môžu nastať iba dve možnosti podľa pozície prvej hlavy na slove:

(a) prvá hlava je v poslednej časti slova na podslove $\# s_3 u_{s1}$:

$$s_1 u_{s1} u_{s2} \underset{\uparrow_2}{\# s_2 u_{s1} u_{s2}} \underset{\uparrow_1}{\underbrace{\# s_3 u_{s1}}}\, u_{s2}$$

Podslov u_{s1} môže byť až $2^{\frac{\lceil a \cdot m \rceil}{2}}$, preto v množine R_1 existujú aspoň dve slová w_1 a w_2 (ak $|R_1| > 2^{\frac{\lceil a \cdot m \rceil}{2}}$), ktorých podslová u_{s1} sú rovnaké a podslová u_{s2} sú rôzne. Slová w_1 a w_2 majú tvar:

$$w_1 = s_1 u_{s1} u_{s2} \# s_2 u_{s1} u_{s2} \# s_3 u_{s1} u_{s2}$$

$$w_2 = s_1 u_{s1} u'_{s2} \# s_2 u_{s1} u'_{s2} \# s_3 u_{s1} u'_{s2}$$

Z týchto dvoch slov w_1 a w_2 zostojíme slovo

$$w_3 = s_1 u_{s1} u'_{s2} \# s_2 u_{s1} u_{s2} \# s_3 u_{s1} u_{s2}$$

pre ktoré bude existovať akceptačný výpočet, ale $w_3 \notin L_{p+1}$. Čo je spor.

(b) prvá hlava je v poslednej časti slova na podslove u_{s2}:

$$s_1 u_{s1} \underset{\uparrow_2}{u_{s2}} \# s_2 u_{s1} u_{s2} \# s_3 u_{s1} \underbrace{u_{s2}}_{\uparrow_1}$$

Podslov u_{s2} môže byt až $2^{\frac{\lceil a \cdot m \rceil}{2}}$, preto v množine R_1 existujú aspoň dve slová w_1 a w_2 (ak $|R_1| > 2^{\frac{\lceil a \cdot m \rceil}{2}}$), ktorých podslová u_{s2} sú rovnaké a podslová u_{s1} sú rôzne. Takže podobne ako v predchádzajúcich prípadoch:

$$w_1 = s_1 u_{s1} u_{s2} \# s_2 u_{s1} u_{s2} \# s_3 u_{s1} u_{s2}$$

$$w_2 = s_1 u'_{s1} u_{s2} \# s_2 u'_{s1} u_{s2} \# s_3 u'_{s1} u_{s2}$$

Z týchto dvoch slov w_1 a w_2 zostrojíme slovo

$$w_3 = s_1 u_{s1} u_{s2} \# s_2 u'_{s1} u_{s2} \# s_3 u_{s1} u_{s2}$$

pre ktoré bude existovať akceptačný výpočet, ale $w_3 \notin L_{p+1}$. Čo je spor.

Popis v jednotlivých prípadoch ako skonštruovať sporný akceptačný výpočet je zrejmý z dôkazu a preto detaily nechávame na čitateľa.

□

Dokázaním lemy sme tiež dokončili dôkaz vety.

□

Podobné tvrdenie platí aj pre deterministické automaty.

Dôsledok 3.2.3. *Trieda jazykov (2,p)-PA $\subsetneq$ trieda jazykov (2,p+1)-PA (príklad jazyk L_{p+1}).*

Dôkaz. Z dôkazu lemy 3.2.2 je zrejmé, že jazyk $L_{p+1} \in$ (2,p+1)-PA, pretože skonštruovaný automat A je deterministický.

Naopak z vety 3.2.3 platí, kedže jazyk $L_{p+1} \notin$ (2,p)-NPA, tak aj $L_{p+1} \notin$ (2,p)-PA. □

Lema 3.2.4. *Jazyk* $L_\infty = \bigcup_{p=1}^{\infty} L_p$ *patrí do triedy (3,0)-PA.*

Dôkaz lemy. Automat akceptujúci jazyk L_∞ bude mať iba tri hlavy a nula žetónov. Na začiatku prvá hlava zostane na začiatku slova, druhá hlava sa posunie za prvý # vstupného slova a tretia hlava sa posunie za druhý # vstupného slova. Následne sa postupne overuje rovnosť symbolov pod hlavami vo vstupnom slove. Ak nastane situácia, že prvá a druhá hlava prečítajú rôzne symboly, posunú sa všetky hlavy o jedno políčko doprava a tretia hlava pred posunom navyše overí výskyt $*$. Pokračuje sa overovaním rovnosti/nerovnosti symbolov až pokým všetky hlavy naraz prečítajú #. Zjavne automat akceptuje jazyk L_∞. □

3.2.2.2 Hierarchia viac-hlavových automatov vzhľadom na počet žetónov

Výsledok z predchádzajúcej časti týkajúci sa 2-hlavových žetónových automatov dokážeme rozšíriť aj na viac-hlavové žetónové automaty. Podobne ukážeme, že pre viac-hlavové žetónové automaty pridanie čo i len jedného žetóna zväčší výpočtovú silu. Jazyk, pomocou ktorého to dokážeme, zadefinujeme pomocou predošlých definícií jazykov 3.2.1 a 3.2.2.

Definícia 3.2.3. *Jazyk* $L_p^k = \{w_1\#w_2\#\ldots\#w_{s-1}\#w_s\#w_{s-1}\#\ldots\#w_2\#w_1 |$ $s = \binom{k}{2} \wedge w_s \in L_p \wedge (\forall i{<}s)\, w_i \in \{0,1\}^*\}$, *pre* $p \in \{0,1,\ldots\}$ *a* $k \in \{3,4,\ldots\}$

Špeciálny prípad pre dve hlavy: jazyk $L_p^2 = L_p$, *pre* $p \in \{0,1,\ldots\}$

Lema 3.2.5. *Existuje (k,p)-NPA A, ktorý akceptuje jazyk* L_p^k.

Dôkaz. Automat A na začiatku vykonáva rovnaké kroky ako automat z lemy 3.2.1. Avšak keď by mal s poslednými dvoma hlavami porovnávať dve prostredné podslová vstupného slova, pokračuje ako automat z lemy 3.2.2. Takto popísaný automat akceptuje presne jazyk L_p^k. □

Veta 3.2.4. *Trieda jazykov (k,p)-NPA* $\subsetneq$ *trieda jazykov (k,p+1)-NPA (napríklad jazyk* L_{p+1}^k*).*

Dôkaz. Dôkaz bude pozostávať z dvoch častí. V prvej je potrebné ukázať, že $L_{p+1}^k \in$ (k,p+1)-NPA, čo je zrejmé z lemy 3.2.5.

V druhej časti ukážeme, že $L_{p+1}^k \notin$ (k,p)-NPA. Budeme postupovať podobne ako v predchádzajúcich dôkazoch sporom. Preto predpokladajme, že existuje (k,p)-NPA A taký, že jazyk akceptovaný automatom A je L_{p+1}^k. Zvolíme nasledovnú podmnožinu jazyka L_{p+1}^k:

$L_{n,m} = \{w_1\#w_2\#\ldots\#w_{s-1}\#w\#w_{s+1}\#\ldots\#w_{2s-2}\#w_{2s-1} |\ s = \binom{k}{2} \wedge (\forall i)w_i = w_{2s-i} \in \{0,1\}^n \wedge w \in (L_p \cap \{0,1,\#,*\}^m)\}$ Kedže automat A akceptuje jazyk L_{p+1}^k, musí akceptovať aj jeho podmnožinu $L_{n,m}$ pre každé zvolené n a m. Z toho na každom slove z jazyka $L_{n,m}$ existuje aspoň jeden akceptačný výpočet. Pre jednoduchosť môžme uvažovať, že na každom slove z jazyka existuje práve jeden akceptačný výpočet. Predchádzajúce zjednodušenie si môžme dovoliť, lebo nám bude postačovať len existencia nejakého akceptačného výpočtu na konkrétnom slove, nie počet akceptačných výpočtov na slove.

Tvrdíme, že počas výpočtu automatu A na väčšine slov z $L_{n,m}$ (pre dosť veľké n a m) by malo platiť, že len jedna dvojica hláv automatu je súčasne na strednom podslove w ($\in L_p$).

Ak by predchádzajúce tvrdenie neplatilo, tak na väčšine slov by existovali aspoň dve dvojice hláv automatu, ktoré by sa súčasne vyskytli na podslove w. Avšak treba si všimnúť, že ak nejaká dvojica hláv sa počas výpočtu vyskytne súčasne na podslove w, potom táto istá dvojica hláv predtým a ani potom vo výpočte nemôže porovnať nejaké podslová w_i a w_{2s-i}. Podobne ak nejaká dvojica hláv porovnáva podslová w_i a w_{2s-i}, potom táto istá dvojica hláv už nemôže porovnať žiadne iné podslová w_j a w_{2s-j}. Tiež vidno, že v slovách z jazyka $L_{n,m}$ je potrebné porovnať $\binom{k}{2} - 1$ dvojíc podslov. Avšak dvojíc hláv je $\binom{k}{2}$, pričom na väčšine slov aspoň dve dvojice hláv nemôžu porovnávať (lebo sa súčasne vyskytnú na podslove w). Teda existuje $\leq \binom{k}{2} - 2$ dvojíc hláv, ktoré môžu porovnávať, ale je potrebne porovnať až $\binom{k}{2} - 1$ podslov. Z toho na väčšine slov existuje podslovo, ktoré sa neporovná. Mohli by sme pokračovať podobne ako pri dôkaze vety 3.2.1 na strane 20, čím dostaneme spor.

Teda vieme, že pre väčšinu slov je počas výpočtu súčasne len jedna dvojica hláv automatu na podslove w ($\in L_p$). Keď zvolíme m veľmi veľké oproti n, potom väčšina vstupného slova je podslovo w, ostatné časti slova sú oproti podslovu w skoro zanedbateľné. Avšak takéto slová sa veľmi podobajú slo-

vám z jazyka L_{p+1}, o ktorých sme už dokázali, že pomocou (2,p)-NPA je ich nemožno akceptovať. Automat A má pre väčšinu slov na podslove w súčasne iba jednu dvojicu hláv, preto jeho správanie na podslove w musí by veľmi podobné (2,p)-NPA. Preto by sme mohli pokračovať veľmi podobným dôkazom ako pri vete 3.2.3 na strane 24, čím by sme nakoniec dostali spor.

Tým sme dokázali vetu, pretože jazyk $L^k_{p+1} \in$ (k,p+1)-NPA a $L^k_{p+1} \notin$ (k,p)-NPA. □

Dôsledok 3.2.4. *Trieda jazykov (k,p)-PA $\subsetneq$ trieda jazykov (k,p+1)-PA (príklad jazyk L^k_{p+1}).*

Dôkaz. Výsledok vyplýva z lemy 3.2.5 ($L^k_{p+1} \in$ (k,p+1)-PA), pretože výsledný žetónový automat je deterministický a z vety 3.2.4 ($L^k_{p+1} \notin$ (k,p)-PA). □

Poznámka 3.2.2. Predchádzajúce tvrdenia, konkrétne veta 3.2.3 ako aj jej dôsledky a tiež veta 3.2.4 s jej dôsledkami, je možné zovšeobecniť aj na žetónové automaty so „sensing heads“. Preto tiež platí:

$$\text{(k,p)-NPA-Sens} \subsetneq \text{(k,p+1)-NPA-Sens}$$

Zdôvodnenie vychádza z dôkazov spomínaných viet, ktoré sa nezmenia ak by sme uvažovali žetónové automaty so „sensing heads“.

Lema 3.2.6. *Jazyk $L^k_\infty = \bigcup_{p=1}^{\infty} L^k_p$ patrí do triedy (k+1,0)-PA.*

Dôkaz. Stačí skombinovať automat z lemy 3.2.1 (ktorý porovná $\binom{k}{2} - 1$ prvých a posledných podslov vstupného slova) s automatom z lemy 3.2.4 (ktorý overí príslušnosť prostredného podslova k jazyku L_∞ pomocou troch hláv). Týmto dostaneme automat, ktorý zjavne akceptuje jazyk L^k_∞ □

3.3 Rozdielnosť deterministických a nedeterministických žetónových automatov

Vplyv nedeterminizmu na výpočtový model žetónových automatov sme analyzovali v tejto časti. Budeme uvažovať len dvoj- a viac-hlavové žetónové automaty, pretože jedno-hlavové žetónové automaty sú ekvivalentné konečným automatom. Ukážeme, že k-hlavový p-žetónový deterministický automat je výpočtovo slabší ako jeho nedeterministická verzia. Podobný nový výsledok

dokážeme všeobecne aj pre konečno-hlavový konečno-žetónový deterministický automat.

Pri dokazovaní využijeme jazyky z predchádzajúcich častí a ich uzáverové vlastnosti, konkrétne uzavretosť na komplement.

Veta 3.3.1. *Jazyk* $(L_{p+1}^k)^{\complement}$ *patrí do triedy (k,0)-NPA.*

Dôkaz. Slová w z jazyka $L = (L_{p+1}^k)^{\complement}$ rozdelíme do množín podľa toho, ktoré vlastnosti jazyka L_{p+1}^k porušujú:

1. slovo w nie je tvaru $w_1\#w_2\#\ldots\#w_{s-1}\#w_s\#w_{s-1}\#\ldots\#w_2\#w_1 | s = \binom{k}{2} \wedge (\forall i < s) w_i \in \{0,1\}^* \wedge w_s = \{0,1\}^n\#\{0,1\}^n\#\{0,1,*\}^n$ pre $n \geq 0$. Ak slovo nie je predošlého tvaru je to možné zistiť pomocou nedeterministického k-hlavového automatu.

2. slovo w je predošlého tvaru, ale obsahuje podslovo $w_s = u_1\#u_2\#u_3$, ktoré nie je z jazyka L_{p+1}:

   (a) podslová u_1 a u_2 sú na viac (alebo menej) ako $p+1$ pozíciách rozdielne. Túto vlastnosť sme schopný overiť pomocou jednej hlavy a stavu.

   (b) podslovo u_3 obsahuje iný počet $*$ ako $p+1$, čo je overiteľné pomocou 1 hlavy.

   (c) v podslove u_3 na pozícii i-tej nezhody medzi podslovami u_1 a u_2 nie je hviezdička. Je možné overiť pomocou dvoch hláv a nedeterminizmu.

Všetky predchádzajúce vlastnosti možno overiť pomocou k hláv v nedeterministických vetvách výpočtu.

□

Dôsledok 3.3.1. *Jazyk* $(L_{p+1}^k)^{\complement}$ *patrí do triedy (k,p)-NPA.*

Dôsledok 3.3.2. *Trieda (k,p)-NPA nie je uzavretá na komplement.*

Dôkaz. Sporom. Ak by trieda bola uzavretá na komplement, potom jazyk L_{p+1}^k patrí do triedy (k,p)-NPA, čo je spor s vetou 3.2.4. □

Lema 3.3.1. *Trieda (k,p)-PA je uzavretá na komplement.*

Dôkaz. Je potrebné ukázať: Ak $L \in$ (k,p)-PA, potom aj $L^{\complement} \in$ (k,p)-PA.

Kedže jazyk $L \in$ (k,p)-PA, z toho existuje deterministický k-hlavový p-žetónový automat A, ktorý ho akceptuje (t.j. $L(A) = L$). Zostrojíme deterministický k-hlavový p-žetónový automat A' s vlastnosťou $L(A') = L^{\complement}$. Automat A' bude simulovať automat A tak, že bude počítať kroky automatu A, ktoré urobí na mieste. Ak automat A spraví na mieste viac krokov ako je počet stavov autmatu A, musel sa zacykliť. V tomto prípade automat A' vie, že automat A už neakceptuje vstupné slovo (musel by dočítať slovo) a teda automat A' môže akceptovať. Opačne ak sa automat A dostane všetkými hlavami na koniec vstupného slova (t.j. na symbol $\dashv$) a je v akceptačnom stave, automat A' už vstupné slovo neakceptuje. Z toho je zrejmé, že $L(A') = L^{\complement}$ a teda $L^{\complement} \in$ (k,p)-PA. □

Lema 3.3.2. *Jazyk* $(L^k_{p+1})^{\complement}$ *nepatrí do triedy (k,p)-PA.*

Dôkaz. Sporom. Ak by jazyk $(L^k_{p+1})^{\complement}$ patril do triedy (k,p)-PA, potom jazyk L^k_{p+1} tiež patrí do triedy (kedže trieda je uzavretá na komplement podľa 3.3.1). Čo je avšak spor s vetou 3.2.4. □

Dôsledok 3.3.3. *Trieda jazykov (k,p)-PA* $\subsetneq$ *trieda jazykov (k,p)-NPA (napríklad jazyk* $(L^k_{p+1})^{\complement}$*).*

Podobný výsledok platí aj pre konečno-žetónový a konečno-hlavový automat, ale najprv uvedieme označenie.

Poznámka 3.3.1. NPA $= \bigcup_{k,p}(k,p)$-NPA. (t.j. zjednotenie cez všetky triedy jazykov konečno-žetónových a konečno-hlavových nedeterministických konečných automatov)

Poznámka 3.3.2. PA $= \bigcup_{k,p}(k,p)$-PA. (t.j. zjednotenie cez všetky triedy jazykov konečno-žetónových a konečno-hlavových deterministických konečných automatov)

Zadefinujeme jazyk, ktorý je veľmi podobný jazyku z definície 3.2.1.

Definícia 3.3.1. $L^{\infty} = \{w_1\# \ldots \#w_s\#w_s\# \ldots \#w_1 | (\forall i) w_i \in \{0,1\}^*\}$

Veta 3.3.2. *Jazyk* $(L^{\infty})^{\complement}$ *patrí do triedy (3,0)-NPA.*

Dôkaz. Slová w z jazyka $L = (L^{\infty})^{\complement}$ porušujú niektorú vlastnosť jazyka L^{∞}. Ak vstupné slovo w nemá tvar $w{=}w_1\#w_2\#\ldots\#w_s\#w_{s+1}\#\ldots\#w_{2s-1}\#w_{2s}$ s vlastnosťou podslov $(\forall i)w_i{\in}\{0,1\}^*$, t.j. podslová w_i neobsahujú len $0,1$ alebo podslov je nepárny počet, potom slovo w vieme akceptovať len pomocou jednej hlavy. Na takéto vlastnosti je postačujúci deterministický konečný automat. Slovo w z jazyka $L = (L^{\infty})^{\complement}$, ktoré síce má predchádzajúci tvar, musí porušovať inú vlastnosť jazyka L^{∞} a preto musí existovať dvojica podslov w_i a w_{2s-i+1} (i-te podslovo slova w a i-te podslovo od konca slova w) také, že sú rôzne. Takúto vlastnosť vieme nedeterministicky zistiť pomocou troch hláv. Na začiatku sa jedna dvojica hláv nedeterministicky umiestni na podslove w_i (jednoducho si tipne pozíciu podslova w_i). Následne potrebujeme zvyšnú tretiu hlavu umiestniť na podslove w_{2s-i+1}, čo je možné pomocou prvej hlavy z dvojice hláv. Stačí aby prvá hlava prechádzala podslová $w_{i+1}, w_{i+2}, \ldots$ (ktorých je $2s-i$), pričom zvyšná tretia hlava zároveň prechádzala podslová $w_1, w_2, \ldots$, pokým prvá hlava neskončí na konci slova. Týmto spôsobom sa tretia hlava dostane na podslovo w_{2s-i+1}, kde už druhá a tretia hlava môžu overiť rôznosť podslov w_i a w_{2s-i+1}. Zjavne pre jazyk $(L^{\infty})^{\complement}$ existuje (3,0)-NPA, ktorý ho akceptuje. □

Dôsledok 3.3.4. *Jazyk $(L^{\infty})^{\complement}$ patrí do NPA.*

Dôsledok 3.3.5. *Trieda NPA nie je uzavretá na komplement.*

Dôkaz. Sporom. Ak by trieda bola uzavretá na komplement, potom jazyk L^{∞} patrí do triedy NPA, čo je nepriamo spor s vetou 3.2.1. □

Lema 3.3.3. *Jazyk $(L^{\infty})^{\complement}$ nepatrí do triedy PA.*

Dôkaz. Sporom. Ak by jazyk $(L^{\infty})^{\complement}$ patril do triedy PA, potom jazyk L^{∞} tiež patrí do triedy (kedže trieda je uzavretá na komplement podľa 3.3.1). Čo je avšak spor s dôsledkom vety 3.2.1. □

Dôsledok 3.3.6. *Trieda jazykov PA $\subsetneq$ trieda jazykov NPA (napríklad jazyk $(L^{\infty})^{\complement}$).*

Z predchádzajúcich výsledkov vyplýva, že nedeterministické a deterministické žetónové automaty nie sú ekvivalentné.

3.4 Vzťah medzi žetónmi a hlavami

V tejto časti prinášame odpoveď na otázku, či je možné vymeniť niekoľko žetónov za niekoľko hláv (možno aj „sensing heads“) bez straty výpočtovej sily.

Jeden z prvých problémov, ktorý nás napadne je, či je bez straty výpočtovej sily možné zameniť iba jeden žetón za jednu „sensing“ hlavu. Riešenie tohto problému závisí od počtu hláv jednosmerného žetónového automatu, ktorému odoberieme jeden žetón a pridáme jednu „sensing“ hlavu.

Začneme analýzou pre dvoj- a troj-hlavové žetónové automaty, pre ktoré ukážeme, že strata výpočtovej sily nenastáva. Následne budeme pokračovať štvor-hlavovými žetónovými automatmi, pri ktorých sa už strata výpočtovej sily objaví. Tento náš výsledok rozšírime aj na viac-hlavové žetónové automaty. Nakoniec ukážeme triviálny odhad počtu hláv, ktoré kompenzujú stratu žetóna(žetónov) nejakému žetónovému automatu.

3.4.1 Dvoj-hlavové a troj-hlavové automaty

Prvý prípad, ktorý detailnejšie rozoberieme, bude jednosmerný žetónový automat A s dvomi hlavami a p žetónmi. Ako sa neskôr ukáže ľubovolný automat A je možné simulovať pomocou troch „sensing“ hláv a $p-1$ žetónov. Myšlienka simulácie je priamočiare simulovanie jedného žetónu pomocou jednej „sensing“ hlavy. Preto ak v poradí prvá hlava od konca vstupného slova automatu A chce položiť žetón, ktorý nemá k dispozícii, stačí na to isté miesto umiestniť pridanú tretiu hlavu. Následne ak v poradí druhá hlava od konca slova sa dostane na miesto spomínaného žetóna, automat je to schopný zistiť, pretože tretia hlava označuje miesto položeného žetóna. V prípade, že by v automate A chcela druhá hlava v poradí položiť žetón, ktorý nemá k dispozícii, žetón už nemusíme simulovať, pretože žiadna iná hlava ho už kvôli jednosmerným hlavám nedetekuje. Preto platí:

$$\text{(2,p+1)-NPA-Sens} \subset \text{(3,p)-NPA-Sens}$$

Druhý prípad je jednosmerný žetónový automat A s tromi hlavami a p žetónmi. Ukážeme, že ľubovolný automat A je možné simulovať pomocou štyroch „sensing“ hláv a $p-1$ žetónov. Myšlienka simulácie je rovnaká ako v predchádzajúcom prípade, budeme simulovať jeden chýbajúci žetón pomocou jednej „sensing“ hlavy. Na začiatok si zoradíme hlavy v poradí od konca slova kvôli jednoduchosti popisu. Ak prvá hlava automatu A chce

položiť žetón, môžme využiť štvrtú hlavu, podobným spôsobom ako v predchádzajúcom prípade. V prípade ak druhá hlava automatu A chce položiť žetón, tiež je možné využiť štvrtú hlavu. Avšak v tomto prípade sa môže zdanlivo objaviť problém, keby spomínaná štvrtá (pomocná) hlava nebola k dispozícii, t.j. bola by niekde medzi prvou a druhou hlavou. Lenže takáto situácia môže nastať iba v prípade, keby bol žetón už položený prvou hlavou. Čo nie je možné, pretože ten istý žetón chceme znovu položiť druhou hlavou. Posledný prípad v ktorom chce posledná tretia hlava položiť žetón je triviálny, pretože daný žetón už netreba ďalej simulovať. Z toho vyplýva:

$$(3,p+1)\text{-NPA-Sens} \subset (4,p)\text{-NPA-Sens}$$

3.4.2 Štvor-hlavové automaty

Ďalej ukážeme, že pre viac-hlavové automaty podobné výsledky ako tie predošlé neexistujú. V predchádzajúcich dvoch prípadoch mali automaty spoločnú vlastnosť a to, že žetóny sa pohybovali len smerom ku koncu slova, čo bolo možné simulovať pomocou jednej jednosmernej hlavy. Preto sa nemohlo stať, že by sa žetón zdvihol a presunul niekde smerom k začiatku slova,t.j. spätne (iba ak by ho položila prvá hlava od začiatku slova, čo je iba triviálny prípad). Avšak pre viac hlavové automaty (aspoň 4 hlavy) táto vlastnosť nemusí platiť. Už pre štyri hlavy môže nastať problém, že žetón bude treba presunúť spätne, čo ako ukážeme v nasledujúcej časti, nebude riešiteľné len pomocou jednej hlavy. K dosiahnutiu spomínaného výsledku potrebujeme zadefinovať nasledujúci jazyk.

Definícia 3.4.1. *Jazyk* $\widetilde{L_p} = \{w_1\&w_2\&\ldots\&w_s@w_{s+1}\&w_{s+2}\&\ldots\&w_{2s} | s \geq 0 \wedge (\forall i)(w_i \in L_p \Leftrightarrow w_{i+s} \in L_p)\}$*, kde* L_p *je jazyk z definície 3.2.2, pre* $p \in \{1, 2, \ldots\}$.

Výsledky neskôr rozšírime aj pre viac ako štvor-hlavové žetónové automaty. Dôvod prečo sme skúmali štvor-hlavové žetónové automaty oddelene je, že práve pri nich sa objavuje problém spätného pohybu žetóna v najelementárnejšej forme.

Lema 3.4.1. *Existuje (4,p)-NPA A, ktorý akceptuje jazyk* $\widetilde{L_p}$.

Dôkaz. Automat A bude postupne overovať všetky vlastnosti jazyka $w_i \in L_p \Leftrightarrow w_{i+s} \in L_p$. Na začiatku pošle prvú dvojicu hláv za oddeľovač @ a druhá dvojica hláv zostane na začiatku slova. Následne sa v cykle overí, či

$w_i \in L_p \Leftrightarrow w_{i+s} \in L_p$ pre všetky i. Jedna iterácia cyklu pozostáva zo zistení, či $w_i \in L_p$ a $w_{i+s} \in L_p$. Takéto overenie je na základe lemy 3.2.2 možné pomocou dvoch hláv a p žetónov. Preto stačí ak prvá dvojica hláv prejde slovo w_{i+s} použijúc pri tom p žetónov a následne druhá dvojica hláv prejde slovo w_i s použitím p dostupných žetónov. Následne je možné vyhodnotiť, či vstupné slovo dodržuje vlastnosť jazyka $w_i \in L_p \Leftrightarrow w_{i+s} \in L_p$. Takto skonštruovaný žetónový automat zjavne akceptuje jazyk $\widetilde{L_p}$ so štyrmi hlavami a p žetónmi. □

Veta 3.4.1. *Jazyk $\widetilde{L_{p+1}}$ nepatrí do triedy (5,p)-NPA.*

Dôkaz. sporom. Nech teda existuje jednosmerný žetónový automat A s p žetónmi a piatimi hlavami, ktorý akceptuje jazyk $\widetilde{L_{p+1}}$.

Hlavná myšlienka dôkazu je postavená na tom, že automat A musí postupne overovať každú i-tu vlastnosť jazyka $\widetilde{L_{p+1}}$ ($w_i \in L_{p+1} \Leftrightarrow w_{i+s} \in L_{p+1}$). Bez ujmy na všeobecnosti vzhľadom na pozície troch hláv, nech pre nejaké i sú tri hlavy na podslove w_i a zvyšné dve hlavy na w_{i+s}, potom podľa vety 3.2.3 dve hlavy s p žetónmi nie sú schopné overiť vlastnosť $w_{i+s} \in L_{p+1}$. Čím pre automat A vzniká potreba overovať dané podslovo w_{i+s} neskôr a keď s je dostatočne veľké, automat A neskôr už nie je schopný overiť všetky vlastnosti a teda niektorá j-ta vlasnosť zostane neoverená. Následne stačí pozmeniť vstupné slovo tak, že j-tu vlasnosť nespĺňa a automat A túto zmenu nepostrehne, pretože ju neoveril.

Začnime tým, ako sa musia hlavy automatu A na vstupnom slove pohybovať, aby mohli akceptovať jazyk $\widetilde{L_{p+1}}$. Slová spomínaného jazyka môžme pretransformovať tak, že jednotlivé podslová w_i substituujeme symbolmi $c_i \in \{0,1\}$, podľa toho, či w_i je(nie je) z jazyka L_{p+1}. Cieľom tejto transformácie je jednoduchšia analýza pohybu hláv na novovzniknutom jazyku $\widetilde{L}$, ktorého formálna definícia je:
$\widetilde{L} = \{c_1\&c_2\&\ldots\&c_s@c_{s+1}\&c_{s+2}\&\ldots\&c_{2s} | s \geq 0 \wedge (\forall i)(c_i \in \{0,1\}) \wedge (c_i = 0 \Leftrightarrow c_{i+s} = 0)\}$. Takto zadefinovaný jazyk vystihuje podstatu pôvodného jazyka. Jediný spôsob ako by vo všeobecnosti nejaký žetónový automat mohol akceptovať jazyk $\widetilde{L}$ je, že prvá hlava prechádza symboly c_i a súčasne druhá hlava prechádza symboly c_{i+s}, pričom ich navzájom porovnáva. Treba si všimnúť, že keď prvá hlava prečíta @, automat už nemôže veľmi porovnávať, lebo si môže pamätať len veľmi obmedzený počet informácií z prvej polovice vstupného slova. Z jednoduchého pozorovania tiež vyplýva, že ak je prvá hlava na symbole c_i, potom vo všeobecnosti druhá hlava je „veľmi blízko“

symbolu c_{i+s}. Ak by to vo všeobecnosti na väčšine c_i neplatilo, automat by musel strácať informáciu o obsahu symbolu c_i alebo c_{i+s}, čo nie je možné. Z tohto všetkého vyplýva, že najvšeobecnejší postup pre akceptáciu jazyka $\widetilde{L}$ je postupné „synchrónne“ porovnávanie symbolov c_i a c_{i+s}.

Podobne ako musí prebiehať výpočet na jazyku $\widetilde{L}$, rovnako musí prebiehať výpočet aj automatu A na $\widetilde{L_{p+1}}$. Avšak na jazyku $\widetilde{L_{p+1}}$ automat s piatimi hlavami a p žetónmi nemôže pracovať len s dvoma hlavami, ale musí používať dve skupiny hláv. Prvá skupina hláv bude prechádzať podslová w_i a druhá skupina hláv bude prechádzať podslová w_{i+s}, pričom sa bude overovať príslušnosť podslov do jazyka. Problém nastane, keď v jednej skupine hláv sú tri hlavy, pomocou ktorých nie je problém overovať príslušnosť podslov do jazyka, avšak zvyšná druhá skupina hláv má iba dve hlavy a p žetónov, čo je podla vety 3.2.3 nedostatočný počet žetónov pre overenie príslušnosti podslov do jazyka.

Skúmaní jazyk $\widetilde{L_{p+1}}$ je na analýzu príliš všeobecný, preto spor ukážeme na slovách špeciálneho tvaru z jazyka L:
$\widetilde{L_{p+1}} \supseteq L = \{w_1\& \ldots \&w_s@w_{s+1}\& \ldots \&w_{2s} | (\forall i)(w_i \in L_{p+1} \Leftrightarrow w_{i+s} \in L_{p+1}) \wedge (\forall i \leq \frac{s}{2})(|w_i| = g(n) \wedge |w_{i+s}| = f(n)) \wedge (\forall i; s \geq i > \frac{s}{2})(|w_i| = f(n) \wedge |w_{i+s}| = g(n))\}$, kde n je dĺžka slova a $f(n)$, $g(n)$ sú vhodne zvolené funkcie, pričom $f(n) \gg g(n)$. Zmysel týchto funkcii uvedieme v dôkaze neskôr (konkrétne v 3.1 a 3.2). Predpokladáme pre jednoduchosť, že s je párne číslo.

Ak automat A akceptuje jazyk $\widetilde{L_{p+1}}$, potom musí akceptovať aj slová z jazyka L. Ďalej môžme predpokladať, že skupina hláv, ktorá obsahuje tri hlavy je tá, ktorá je v ľavej časti slova. V tomto prípade si budeme všímať podslová $w_1, \ldots, w_{\frac{s}{2}}$ a podslová $w_{1+s}, \ldots, w_{\frac{s}{2}+s}$. V opačnom prípade ak by bola skupina troch hláv v pravej časti slova je potrebné si všímať podslová $w_{\frac{s}{2}+1}, \ldots, w_s$ a podslová $w_{\frac{s}{2}+1+s}, \ldots, w_{s+s}$. Ďalej teda bez ujmy na všeobecnosti budeme predpokladať, že skupina troch hláv je v ľavej časti slova na podslove w_i a zvyšné dve hlavy sú v pravej časti slova na podslove w_{i+s}.

Rozdeľme slová podľa prípadov na základe toho, či podslová w_i a w_{i+s} patria alebo nepatria do jazyka L_{p+1}:

- $w_i \notin L_{p+1}$ a $w_{i+s} \notin L_{p+1}$: V tomto prípade automat A nemá problém overovať, pretože podslovo w_i vieme overiť pomocou troch hláv a ak $w_{i+s} \notin L_{p+1}$, potom vlasnosť vieme skontrolovať pomocou dvoch hláv

nedeterministický (tipne si kde je chyba v w_{i+s}).

- $w_i \notin L_{p+1}$ a $w_{i+s} \in L_{p+1}$: Podobne ako predchádzajúci prípad, akurát nebude existovať nedeteministická vetva výpočtu, ktorý by mohol akceptovať. Čo je žiadúce, lebo pôvodné vstupné slovo nemožno akceptovať, pretože nespĺňa i-tu vlasnosť jazyka.
- $w_i \in L_{p+1}$ a $w_{i+s} \notin L_{p+1}$: Pomocou troch hláv automat A vie overiť, či $w_i \in L_{p+1}$. Tiež vie overiť v nedeterministickej vetve výpočtu, že $w_{i+s} \notin L_{p+1}$, avšak v ostatných nedeterministických vetvách výpočtu automat A nevie, či náhodou $w_{i+s} \in L_{p+1}$.
- $w_i \in L_{p+1}$ a $w_{i+s} \in L_{p+1}$: Podobne ako v predchádzajúcom prípade, vlastnosť $w_i \in L_{p+1}$ je schopný overiť. Avšak v každom nedeterministickom výpočte automat A nevie, či $w_{i+s} \in L_{p+1}$.

Z posledných dvoch možností vyplýva, že automat A nie je schopný rozlíšiť prípady, keď $w_{i+s} \in L_{p+1}$ a $w_{i+s} \notin L_{p+1}$. Preto i-tu vlasnosť musí automat A overiť neskôr, t.j. keď sa trojica hláv dostane až za @.

Existuje potenciálna možnosť, že automat A s troma hlavami môže simulovať nejaký žetón. Preto sme zadefinovali už spomínané funkcie $f(n)$ a $g(n)$, ktoré spĺňajú nasledujúcu podmienku:

$$g(n)^3 \cdot g(n)^p \cdot |K_A| \ll f(n) \tag{3.1}$$

(rôzne rozloženia hláv) $\cdot$ (rôzne rozloženia žetónov na podslove w_i) $\cdot$ (počet stavov automatu A) $\ll$ (možné pozície žetóna na podslove w_{i+s}). Ak funkcie spĺňajú túto podmienku, možno podobne ako vo vete 3.2.3 dokázať, že ani premyslená manipulácia tromi hlavami nepomôže pri overení, či $w_{i+s} \in L_{p+1}$.

Je síce pravda, že automat A môže pri overení $w_{i+s} \in L_{p+1}$, posunúť skupinu troch hláv z w_i až na w_{i+j}. Avšak j nemôže byt príliš veľké, lebo automat by po ich prejdení stratil informáciu o podslovách w_i až w_{i+j}. Problém by nastal, ak by $2^j > n^5 \cdot n^p \cdot |K_A| = n^{5+p} \cdot |K_A|$. (všetky možné príslušnosti podslov w_i až w_{i+j} do jazyka L_{p+1}) $>$ (všetky možné rozloženia piatich hláv) $\cdot$ (rozloženia žetónov) $\cdot$ (stavy automatu A). Po zjednodušení:

$$\begin{aligned} j &> \log(2^{(5+p)\cdot\log(n)} \cdot |K_A|) \\ j &> \log(2^{(5+p)\cdot\log(n)}) + \log(|K_A|) \\ j &> (5+p)\cdot\log(n) + \log(|K_A|) \end{aligned}$$

Teda ak by j bolo rádovo $\log(n)$, potom by bolo možné nájsť dve slová, ktorých niektoré podslová w_i až w_{i+j} majú rôzne príslušnosti do jazyka L_{p+1},

avšak automat A by po ich prečítaní skončil v rovnakých konfiguráciách. Čo by bol spor, pretože by mohol akceptovať aj slovo, ktoré nie je v jazyku. Z toho vyplýva, že $j < (5+p)\cdot\log(n)+\log(|K_A|) = L(n)$ ($L(n)$ - iba označenie funkcie).

V podmienke 3.1 sme pre jednoduchosť predpokladali, že skupina troch hláv sa bude vyskytovať iba na podslove w_i. Čo sa nemusí vždy stať, preto podmienku 3.1 upravíme tak, že sa trojica hláv bude môcť vyskytovať aj na $L(n)$ susedných podslovách:

$$(g(n)\cdot L(n))^3\cdot(g(n)\cdot L(n))^p\cdot|K_A| \ll f(n) \tag{3.2}$$

Zvolíme funkcie, ktoré budú asymptoticky spĺňať predchádzajúcu podmienku:

$$f(n) = n^{1/2}$$

$$g(n) = \log(n)$$

Z toho možno určiť počet podslov w_i:

$$n = 2\cdot(f(n)+g(n))\cdot s$$

$$s = \frac{n}{2\cdot(n^{1/2}+\log(n))} \geq \frac{n^{1/2}}{2\cdot 2}$$

Z predchádzajúcich časti vyplýva, že ak aspoň $\frac{s}{4}$ podslov z w_1 až $w_{\frac{s}{2}}$ sú z jazyka L_{p+1}, potom aspoň $\frac{s}{4}$ podslov z w_{1+s} až $w_{\frac{s}{2}+s}$ je potrebné overiť či patria do jazyka L_{p+1}. Avšak v tom prípade rôznych príslušnosti podslov w_1 až $w_{\frac{s}{2}}$ do jazyka L_{p+1} je aspoň $\binom{\frac{s}{2}}{\frac{s}{4}}$. Pričom všetkých možných konfigurácii, keď skupina troch hláv je v strede slova, je len polynomiálny počet od veľkosti vstupu: $n^5\cdot n^p\cdot|K_A|$. Zjavne existuje slovo, ktorého niektoré podslovo $w_{i+s} \in L_{p+1}$ sa v následnej fáze neoverí. Tým dostávame spor, pretože slovo w_{i+s} možno zameniť za w'_{i+s} tak, že $w'_{i+s} \notin L_{p+1}$. Túto zámenu si automat A nemôže všimnúť, pretože i-tu vlastnosť neoverí a teda slovo, ktoré nemal akceptovať akceptuje. Čím sme dostali spor a dokazuje neexistenciu päť hlavového p žetónového automatu pre jazyk $\widetilde{L_{p+1}}$. □

Dôsledok 3.4.1. *Trieda jazykov (4,p+1)-NPA ⊈ trieda jazykov(5,p)-NPA (napríklad jazyk $\widetilde{L_{p+1}}$).*

Dôsledok 3.4.2. *Trieda jazykov (4,p+1)-PA ⊈ trieda jazykov(5,p)-PA (napríklad jazyk $\widetilde{L_{p+1}}$).*

Dôkaz. Dôsledok vyplýva z dvoch dokázaných tvrdení: Podľa lemy 3.4.1 jazyk $\widetilde{L_{p+1}}$ je v triede (4,p+1)-PA, pretože zostrojený automat je deterministický. Z vety 3.4.1 ten istý jazyk nie je z triedy (5,p)-NPA a teda ani z (5,p)-PA. □

3.4.3 Rozšírenie na viac-hlavové automaty

Výsledok z predchádzajúcej časti týkajúci sa štvor-hlavových žetónových automatov možno rozšíriť aj na viac ako štvor-hlavové automaty. Preto v tejto časti ukážeme, že pre viac ako štvor-hlavové žetónové automaty stratu jedného žetóna nemožno kompenzovať pridaním hlavy bez určitého oslabenia výpočtovej sily. Pre tento cieľ potrebujeme nájsť vhodný jazyk a ako neskôr ukážeme, nasledujúci jazyk bude vhodným kandidátom.

Definícia 3.4.2. *Jazyk* $\widetilde{L_p^k} = \{w_1\&w_2\&\ldots\&w_s@w_{s+1}\&w_{s+2}\&\ldots\&w_{2s} | s \geq 0 \wedge (\forall i)(w_i \in L_p^{k-2} \Leftrightarrow w_{i+s} \in L_p)\}$*, kde* L_p *je jazyk z definície 3.2.2 a* L_p^k *je jazyk z definície 3.2.3, pre* $p \in \{1, 2, \ldots\}$ *a* $k \in \{4, 5, \ldots\}$

Lema 3.4.2. *Existuje (k,p)-NPA A, ktorý akceptuje jazyk* $\widetilde{L_p^k}$*.*

Dôkaz. Automat A, ktorý skonštruujeme, bude postupne overovať všetky vlastnosti jazyka $w_i \in L_p^{k-2} \Leftrightarrow w_{i+s} \in L_p$. Na začiatku pošle prvú dvojicu hláv za oddeľovač @ a zvyšných $k-2$ hláv zostane na začiatku slova. Následne sa v cykle overí, či $w_i \in L_p^{k-2} \Leftrightarrow w_{i+s} \in L_p$ pre všetky i. Jedna iterácia cyklu pozostáva zo zistení, či $w_i \in L_p^{k-2}$ a $w_{i+s} \in L_p$. Overenie $w_i \in L_p^{k-2}$ je na základe lemy 3.2.5 možné pomocou $k-2$ hláv a p žetónov, druhé overenie $w_{i+s} \in L_p$ je z lemy 3.2.2 možné s využitím 2 hláv a p žetónov. Preto stačí ak prvá dvojica hláv prejde slovo w_{i+s} použijúc pri tom p žetónov a následne zvyšných $k-2$ hláv prejde slovo w_i s použitím p dostupných žetónov. Následne je možné vyhodnotiť, či vstupné slovo dodržuje vlastnosť jazyka $w_i \in L_p^{k-2} \Leftrightarrow w_{i+s} \in L_p$. Takto skonštruovaný žetónový automat zjavne akceptuje jazyk $\widetilde{L_p^k}$ s k hlavami a p žetónmi. □

Veta 3.4.2. *Jazyk* $\widetilde{L_{p+1}^k}$ *nepatrí do triedy* $(k+1,p)$*-NPA.*

Dôkaz. sporom. Nech teda existuje jednosmerný žetónový automat A s p žetónmi a $k+1$ hlavami, ktorý akceptuje jazyk $\widetilde{L_{p+1}^k}$.

Možno si všimnúť, že dokazovaná veta je všeobecný prípad vety 3.4.1, v ktorej volíme za $k = 4$. Preto aj myšlienka dôkazu je podobná dôkazu vety 3.4.1 rozšírená o viac hláv.

Problém automatu A je, že musí overiť každú i-tu vlastnosť jazyka $\widetilde{L^k_{p+1}}$ ($w_i \in L^{k-2}_{p+1} \Leftrightarrow w_{i+s} \in L_{p+1}$). Avšak k dispozícii má $k+1$ hláv a len p žetónov. Lenže na overenie vlastnosti $w_i \in L^{k-2}_{p+1}$ je potrebných až $p+1$ žetónov, ktoré automat A nemá k dispozícii, alebo $k-1$ hláv. Podobne na overenie vlastnosti $w_{i+s} \in L_{p+1}$ je tiež potrebných až $p+1$ žetónov alebo až 3 hlavy. Keďže automat A má len p žetónov, pre súčastné overenie podslov w_i a w_{i+s} je v súčte potrebných až $k-1+3 = k+2$ hláv. Lenže automat A má len $k+1$ hláv a preto nie je schopný overiť niektoré i-te vlastnosti. Túto myšlienku by sme mohli rozpísať do detailov ako v dôkaze vety 3.4.1, avšak dôkaz by bol približne rovnaký ako v už spomínanej vete. Jediný rozdiel by bol, že v jazyku $\widetilde{L^k_{p+1}}$ overujeme $w_i \in L^{k-2}_{p+1}$, pričom vo vete 3.4.1 sme v jazyku $\widetilde{L_{p+1}}$ overovali $w_i \in L_{p+1}$. Všetky ostatné detaily by boli rovnaké ako v už spomínanom dôkaze vety. □

Dôsledok 3.4.3. *Trieda jazykov $(k,p+1)$-NPA $\not\subseteq$ trieda jazykov $(k+1,p)$-NPA (napríklad jazyk $\widetilde{L^k_{p+1}}$).*

Dôsledok 3.4.4. *Trieda jazykov $(k,p+1)$-PA $\not\subseteq$ trieda jazykov $(k+1,p)$-PA (napríklad jazyk $\widetilde{L^k_{p+1}}$).*

Dôkaz. Využijeme predchádzajúce dokázané tvrdenia: Podľa lemy 3.4.2 jazyk $\widetilde{L^k_{p+1}}$ je v triede $(k,p+1)$-PA, pretože zostrojený automat je deterministický. Z vety 3.4.2 ten istý jazyk nie je z triedy $(k+1,p)$-NPA a teda ani z $(k+1,p)$-PA. □

Lema 3.4.3. *Triedy jazykov $(k,p+1)$-NPA a $(k+1,p)$-NPA sú neporovnateľné (pre $k \geq 4$).*

Dôkaz. Stačí dokázať, že:

1. $(k,p+1)$-NPA $\not\subseteq$ $(k+1,p)$-NPA: je zrejmé z dôsledku 3.4.3.
2. $(k,p+1)$-NPA $\not\supseteq$ $(k+1,p)$-NPA: z dôsledku 3.2.2 na strane 23.

□

Poznámka 3.4.1. Vetu 3.4.1 s jej dôsledkami a tiež vetu 3.4.2 s jej dôsledkami, nie je problém zovšeobecniť aj na žetónové automaty so „sensing heads“. Preto tiež platí:

$$(k,p+1)\text{-NPA-Sens} \not\subseteq (k+1,p)\text{-NPA-Sens (pre } k \geq 4)$$

V dôkazoch spomínaných viet by sa argumentácia pre žetónové automaty so „sensing heads“ vôbec nezmenila a dôkazy by boli korektné aj pre ne.

3.4.4 Horné ohraničenie počtu hláv kompenzujúcich žetóny

V poslednej časti nás bude zaujímať, akým minimálnym počtom „sensing“ hláv sa dá kompenzovať strata jedného žetóna, poprípade viacero žetónov. Konkrétne budeme hľadať minimálny počet hláv Δ taký, že platí:

$$(k, p+1)\text{-NPA-Sens} \subseteq (k+\Delta, p)\text{-NPA-Sens}$$

Triviálny horný odhad na Δ je rovný počtu hláv žetónového automatu (t.j. k), pretože ku každej pôvodnej hlave môžme pridať pomocnú hlavu, ktorá bude s pôvodnou hlavou asociovaná. Následne ak bude potrebné položiť chýbajúci žetón, tak sa len pomocná hlava zafixuje na vstupe. Je zjavné, že dostatočným počtom pomocných hláv sme vyriešili spätný pohyb chýbajúceho žetóna.

Existuje aj tesnejší horný odhad pre Δ, ktorý ukážeme v nasledujúcej vete.

Veta 3.4.3. *Ak* $\Delta \geq \left\lfloor \frac{k}{2} \right\rfloor$, *potom platí* $(k, p+1)$*-NPA-Sens* $\subseteq (k+\Delta, p)$*-NPA-Sens.*

Dôkaz. Bez ujmy na všeobecnosti nech $\Delta = \left\lfloor \frac{k}{2} \right\rfloor$. Ukážeme ako simulovať jeden chýbajúci žetón pomocou Δ pomocných hláv navyše.

Na začiatok si je potrebné všimnúť, že pri simulovaní chýbajúceho žetóna nie je problém, keď sa žetón presúva len smerom ku koncu slova. Problém vzniká, keď je potrebné presunúť žetón k začiatku slova, t.j. spätne. Takýto posun nemožno simulovať len jednou hlavou a aj preto Δ závisí od počtu hláv.

V každej konfigurácii k-hlavového automatu môžeme zoradiť hlavy postupne od konca vstupného slova až po začiatok slova. Preto ďalej pod pojmom „i-ta“ hlava budeme rozumieť v poradí i-tou hlavou od konca slova. Δ pomocných hláv môžme na začiatku výpočtu ľubovolne zoradiť, preto na základe tohto fixného poradia budeme pomocné hlavy označovať pojmom „i-ta pomocná“ hlava.

Ak chceme simulovať jeden chýbajúci žetón, tak položenie žetónu možno nahradiť umiestnením niektorej hlavy(spomedzi Δ pomocných hláv) na dané miesto a zdvihnutie žetónu je ešte jednoduchšie, pretože stačí, aby pomocná

hlava stratila reprezentáciu žetóna. Ešte treba ukázať, že Δ pomocných hláv bude dostatočný počet pre simuláciu žetóna.

Vlastne je potrebné navrhnúť také správanie (pohyb) pomocných hláv, že keď automat bude chcieť položiť chýbajúci žetón na miesto niektorej hlavy, potom bude existovať nejaká pomocná hlava, ktorej pozícia je pred alebo na danom mieste. Ak by taká pomocná hlava neexistovala, nemohli by sme na danom mieste zanechať chýbajúci žetón v podobe pomocnej hlavy.

Ďalej budeme ignorovať triviálny prípad, keď k je nepárne a posledná hlava by chcela položiť žetón, ktorý už nie je potrebné simulovať. Podobne budeme ignorovať ukladanie a v ďalšom kroku následné zdvihnutie žetónu z toho istého miesta, pretože takáto manipulácia sa dá simulovať v stave a preto je nezaujímavá.

Správanie pomocných hláv bude jednoduché, pretože väčšinu času budú pomocné hlavy stacionárne. Jediný pohyb niektorej pomocnej hlavy nastane až vtedy, keď niektorá bežná hlava bude chcieť položiť chýbajúci žetón. V tejto situácii ak $2i$-ta hlava alebo $(2i-1)$-ta hlava (v poradí od konca vstupného slova) bude chcieť položiť chýbajúci žetón, potom sa iba i-ta pomocná hlava (vo fixnom poradí) presunie na miesto $2i$-tej alebo $(2i-1)$-tej hlavy.

Ešte je potrebné ukázať, že i-ta pomocná hlava nie je za $2i$-tou hlavou, čo by bol problém. Na začiatku výpočtu je to splnené, pretože všetky pomocné hlavy sú na začiatku vstupného slova. Ešte je potrebné rozanalyzovať pozíciu pomocnej hlavy pri ukladaní a zdvíhaní chýbajúceho žetóna. Ak $2i$-ta hlava bude chcieť položiť chýbajúci žetón, tak i-ta pomocná hlava bude stále pred $2i$-tou hlavou ako aj pred $(2i-1)$-tou hlavou. Avšak ak $(2i-1)$-ta hlava bude chcieť položiť chýbajúci žetón, tak jediná hlava ktorá by mohla zdvihnúť chýbajúci žetón je iba $2i$-ta hlava. Preto aj po „zdvihnutí“ chýbajúceho žetóna bude platiť, že i-ta pomocná hlava bude stále pred $2i$-tou hlavou ako aj pred $(2i-1)$-tou hlavou.

Z toho všetkého vyplýva, že $\Delta \geq \left\lfloor \frac{k}{2} \right\rfloor$ pomocných hláv je postačujúca, t.j. aspoň pred každou párnou hlavou je nejaká pomocná hlava. □

Dôsledok 3.4.5. (k,p)*-NPA-Sens* $\subseteq \left(k + p\left\lfloor \frac{k}{2} \right\rfloor, 0\right)$*-NPA-Sens*

Dôkaz. Stačí spraviť konštrukciu z vety 3.4.3 pre každý žetón, pričom pre jeden žetón je potrebné pridať $\left\lfloor \frac{k}{2} \right\rfloor$ pomocných hláv. Preto pre p žetónov stačí pridať $\left(p\left\lfloor \frac{k}{2} \right\rfloor\right)$ pomocných hláv. □

Záver

V práci sme poskytli základný prehľad žetónových automatov a prinášame rad nových výsledkov pre jednosmerné žetónové automaty. Pracovali sme s dvomi výpočtovými modelmi, s dvojsmernými konečnými automatmi so žetónmi a jednosmernými konečnými automatmi so žetónmi.

Pre dvojsmerné žetónové automaty sme uviedli základné tvrdenia, ktoré sú v súčastnosti známe o dvojsmerných žetónových automatoch. Pre niektoré jednoduchšie tvrdenia sme uviedli aj myšlienky ich dôkazov ako napríklad pri ekvivalencií s turingovym strojom (s obmedzením na logaritmický priestor) alebo ekvivalenciou so „sensing“ viac-hlavovými dvojsmernými konečnými automatmi. Tiež sme spomenuli otvorený problém ekvivalencie nedeterministických a deterministických dvojsmerných žetónových automatov.

Vlastné výsledky sme dokázali v kapitole 3, ktorej obsah je iba o jednosmerných žetónových automatoch. Prvý zaujímavý výsledok, ktorý sme ukázali je vplyv počtu hláv na jednosmerný žetónový automat. Konkrétne sme výsledok pre model jednosmerných konečných automatov rozšírili na žetónové automaty. Tento výsledok vypovedá o tom, že ak pridáme čo i len jednu hlavu jednosmernému žetónovému automatu, dostaneme výpočtovo silnejší model. Dokonca ďalším netriviálny dôsledkom je, že stratu jednej hlavy nemožno kompenzovať ľubovolným konečným počtom žetónov. Ďalší dokázaní výsledok sa týkal vplyvu počtu žetónov na výpočtovú silu. Ukázali sme zväčšovanie akceptačnej sily modelu jednoduchým pridávaním žetónov. Tiež sme uvažovali problém ekvivalencie deterministických a nedeterministických jednosmerných žetónových automatov, pričom sme ukázali rozdielnosť takýchto modelov a teda nedeterminizmus má vplyv na výpočtovú silu modelu. Následne sme si položili otázku, či je možné kompenzovať stratu jedného žetóna za jednu „sensing“ hlavu. Ako sme ukázali takáto zámena žetóna za hlavu nie je možná. Nakoniec sme sa pokúsili zhora ohraničiť počet „sensing“ hláv, ktoré už sú postačujúce pre kompenzovanie jedného žetóna.

Väčšinu spomenutých výsledkov sme dokázali pre nedeterministické ako aj deterministické jednosmerné žetónové automaty. Všetky základné spomenuté výsledky platia aj pre jednosmerné žetónové „sensing“ automaty.

Výsledky z tejto práce poukázali na zopár otázok, ktoré by bolo zaujímavé ďalej preskúmať. Najzaujímavejšia sa nám zdá otázka, či spomenuté horné ohraničenie počtu hláv (kompenzujúce stratu žetónov) je možné zmenšiť. Naša hypotéza je, že odhadnuté horné ohraničenie z vety 3.4.3 je najtesnejšie možné.

Literatúra

[BH67] Manuel Blum and Carl Hewitt. Automata on a 2-dimensional tape. In *FOCS* [DBL67], pages 155–160.

[CC97] Bogdan S. Chlebus and Ludwik Czaja, editors. *Fundamentals of Computation Theory, 11th International Symposium, FCT '97, Kraków, Poland, September 1-3, 1997, Proceedings*, volume 1279 of *Lecture Notes in Computer Science*. Springer, 1997.

[DBL67] *Conference Record of 1967 Eighth Annual Symposium on Switching and Automata Theory, 18-20 October 1967, Austin, Texas, USA*. IEEE, 1967.

[GH96] Noa Globerman and David Harel. Complexity results for two-way and multi-pebble automata and their logics. *Theor. Comput. Sci.*, 169(2):161–184, 1996.

[GI09] Viliam Geffert and Lubomíra Istonová. Translation from classical two-way automata to pebble two-way automata. *CoRR*, abs/0907.5127, 2009.

[HKM11] Markus Holzer, Martin Kutrib, and Andreas Malcher. Complexity of multi-head finite automata: Origins and directions. *Theor. Comput. Sci.*, 412(1-2):83–96, 2011.

[IIIO05] Atsuyuki Inoue, Akira Ito, Katsushi Inoue, and Tokio Okazaki. Some properties of one-pebble turing machines with sublogarithmic space. *Theor. Comput. Sci.*, 341(1):138–149, 2005.

[Pet95] H. Petersen. Automata with sensing heads. In *ISTCS*, pages 150–157, 1995.

[Pet97] H. Petersen. The equivalence of pebbles and sensing heads for finite automata. In Chlebus and Czaja [CC97], pages 400–410.

[Rav07] Bala Ravikumar. On some variations of two-way probabilistic finite automata models. *Theor. Comput. Sci.*, 376(1-2):127–136, 2007.

[Sze06] Andrzej Szepietowski. A note on alternating one-pebble turing machines with sublogarithmic space. *Inf. Process. Lett.*, 98(5):174–176, 2006.

[YR78] Andrew Chi-Chih Yao and Ronald L. Rivest. k+1 heads are better than k. *J. ACM*, 25(2):337–340, 1978.