Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Nedeterminizmus v konečných automatoch

Bakalárska práca

2018
Ján Rosina

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Nedeterminizmus v konečných automatoch

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	prof. RNDr. Branislav Rovan, PhD.

Bratislava, 2018
Ján Rosina

89441077

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Ján Rosina
Študijný program:	informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor:	informatika
Typ záverečnej práce:	bakalárska
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Nedeterminizmus v konečných automatoch
Nondeterminism in Finite Automata

Anotácia: V práci sú skúmané triedy regulárnych jazykov a nedeterministických konečných automatov, u ktorých ekvivalentný deterministický konečný automat má subexponenciálnu stavovu zložitosť.

Cieľ: Nájsť podtriedy regulárnych jazykov resp. nedeterministických konečných automatov, pre ktoré k n-stavovému nedeterministickému konečnému automatu existuje deterministický s menej ako exponenciálnym počtom stavov.

Vedúci:	prof. RNDr. Branislav Rovan, PhD.
Katedra:	FMFI.KI - Katedra informatiky
Vedúci katedry:	prof. RNDr. Martin Škoviera, PhD.
Dátum zadania:	29.10.2017

Dátum schválenia: 30.10.2017

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

....
študent

....
vedúci práce

Poďakovanie: Špeciálne poďakovanie patrí vedúcemu práce, profesorovi Rovanovi, za venovaný čas, ochotu, usmernenia, nápady riešení niektorých problémov a za nemalý počet prečítaní a korekcií tejto práce.

Abstrakt

Je známe, že vo všeobecnosti pre daný nedeterministický konečný automat je pre zostrojenie ekvivalentného deterministického konečného automatu potrebných a postačujúcich exponenciálne veľa stavov. V tejto práci ukážeme, že pre vybrané podtriedy regulárnych jazykov je tento nárast stavovej zložitosti menší.

Pre nedeterministické konečné automaty overujúce iba výskyt daného podslova nie je potrebný žiadny nárast počtu stavov. Pre jazyky nad unárnou abecedou ukážeme, že tento nárast súvisí s vetvením grafu silne súvislých komponentov pre daný nedeterministický konečný automat. Napokon ukážeme, že pre ohraničené jazyky je nárast stavovej zložitosti deterministického konečného automatu k nedeterministickému väčší, ako tomu bolo pri jazykoch nad unárnou abecedou napriek ich dokázateľnému súvisu.

Kľúčové slová: nedeterministický konečný automat, deterministický konečný automat, jazyky definované podslovom, jazyky nad unárnou abecedou, ohraničené jazyky

Abstract

It is well known that in general for a simulation of a given nondeterministic finite automaton by a deterministic finite automaton an exponential state complexity is sufficient and needed. In this thesis, we shall prove that for some subclasses of regular languages the growth of state complexity is smaller.

For nondeterministic finite automata which just check whether the input contains a given subword no state complexity growth is needed. For languages over unary alphabet we shall show that state complexity of a deterministic finite automaton is related to branching in a graph of strongly connected components of a given nondeterministic finite automaton. At last, we shall prove that for bounded languages the growth of state complexity of a deterministic finite automaton equivalent to a given nondeterministic finite automaton is larger than it is for languages over unary alphabet, despite of demonstrated relation between these two subclasses.

Keywords: nondeterministic finite automata, deterministic finite automata, subword languages, languages over unary alphabet, bounded languages

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Teória formálnych jazykov a automatov je jednou zo základných a zároveň najstarších oblastí teoretickej informatiky. Dodnes je predmetom aktívneho výskumu, nakoľko výsledky skúmania tejto oblasti majú rozsiahle využitie v rôznych odvetviach informatiky, bioinformatiky, ale aj lingvistiky. Konkrétnymi príkladmi sú lexikálna, či syntaktická analýza pri tvorbe kompilátorov, alebo hľadanie vzoru v texte, či v reťazcoch DNA.

Teória automatov sa zaoberá skúmaním rôznych modelov, ktoré sú schopné rozpoznávať formálne jazyky. Pre rôzne triedy formálnych jazykov sú definované rôzne modely automatov, o ktorých je dokázané, že sú schopné rozpoznať práve danú triedu jazykov. Napríklad pre triedu regulárnych jazykov sú to konečné automaty. Najznámejšími a najpoužívanejšími konečnými automatmi sú deterministické a nedeterministické konečné automaty, ktorými sa budeme v tejto práci zaoberať.

Jednou zo základných otázok v teórii automatov je meranie ich zložitosti. Túto zložitosť možno definovať viacerými spôsobmi, napríklad počtom krokov, ktoré automat vykoná pri výpočte, veľkosťou pamäte, ktorú pri výpočte použije, alebo tiež počtom stavov, ktorý je potrebný na jeho zostrojenie. Nakoľko konečné automaty pri výpočte nepoužívajú pracovnú pamäť a počet krokov výpočtu je pri nich lineárny od veľkosti vstupu, ako najlepšie kritérium merania zložitosti konečných automatov sa ukazuje ich stavová zložitosť.

Pre daný regulárny jazyk je jednoduchšie skonštruovať príslušný nedeterministický konečný automat, ako deterministický. Naproti tomu s deterministickými konečnými automatmi sa príjemnejšie pracuje (napríklad pri dokazovaní). Je známe, že tak deterministické, ako aj nedeterministické konečné automaty rozpoznávajú práve triedu regulárnych jazykov. Teda ku každému nedeterministickému konečnému automatu vieme zostrojiť deterministický konečný automat rozpoznávajúci rovnaký jazyk. Preto je pomerne zaujímavá otázka ohľadom vzťahu stavovej zložitosti medzi týmito dvoma modelmi. Ukazuje sa, že vo všeobecnosti môže byť potrebný exponenciálny počet stavov deterministického konečného automatu oproti nedeterministickému[3]. V tejto práci sa budeme venovať vzťahu stavovej zložitosti deterministického a nedeterministického konečného automatu pre vybrané podtriedy regulárnych jazykov.

Prvou podtriedou regulárnych jazykov, ktorej sa budeme v tejto práci venovať sú jazyky definované podslovom. Ako ukážeme, pre tieto jazyky je stavová zložitosť deterministických a nedeterministických konečných automatov rovnaká, teda nedochádza k žiadnemu nárastu.

Ďalej sa budeme zaoberať regulárnymi jazykmi nad unárnou abecedou. Vďaka svojej jednoduchej štruktúre už boli z pohľadu porovnania stavovej zložitosti deterministických a nedeterministických konečných automatov v minulosti predmetom výskumu. Nadviažeme na prácu Mareka Chrobáka, ktorý dokázal odhad nárastu tejto stavovej zložitosti vo všeobecnom prípade, ktorým je $e^{\sqrt{n \log(n)}}$ stavov deterministického konečného automatu k n-stavovému nedeterministickému[2]. Ukážeme, že za istých podmienok môže byť spomínaný nárast počtu stavov polynomiálny.

Napokon sa budeme venovať triede ohraničených regulárnych jazykov, ktoré sú nadtriedou spomínaných jazykov nad unárnou abecedou. Budeme preto vychádzať z dosiahnutých výsledkov pre túto podtriedu regulárnych jazykov. Ukážeme, že vo všeobecnosti môže byť nárast stavovej zložitosti deterministického konečného automatu oproti danému nedeterministickému horší, ako tomu bolo v prípade jazykov nad unárnou abecedou.

Kapitola 1

Úvod do problematiky

V tejto kapitole oboznámime čitateľa so základnými pojmami, či vetami, ktorých znalosť je nevyhnutná pre čítanie a pochopenie celej práce. Viacero tvrdení a viet uvedieme bez dôkazu, nakoľko cieľom tejto kapitoly nie je prepisovať dôkazy známych tvrdení, ale oboznámiť čitateľa s neskôr používanými tvrdeniami. Ďalšie pojmy a tvrdenia z oblasti formálnych jazykov sa dajú nájsť napríklad v [5][3].

1.1 Základné pojmy a definície

V nasledujúcej časti zadefinujeme základné pojmy formálnych jazykov, operácie na jazykoch a slovách, ktoré budeme neskôr používať, ako aj samotné definície deterministického a nedeterministického konečného automatu.

Definícia. *Abeceda je konečná neprázdna množina symbolov (písmen). Označuje sa* Σ.

Definícia. *Slovo nad abecedou* Σ *je konečná postupnosť symbolov z* Σ. *Prázdne slovo označujeme* ϵ.

Definícia. *Dĺžka slova je dĺžka postupnosti, ktorá ho vytvára. Dĺžku slova* w *budeme označovať* $|w|$.

Definícia. *Podslovo slova* $w = a_1a_2...a_n$ *je súvislá podpostupnosť* $a_ia_{i+1}...a_j$, *pričom platí* $1 \leq i \leq j \leq n$. *Špeciálne podslová sú prefix a sufix, pre prefix platí* $i = 1$ *a pre sufix platí* $j = n$.

Definícia. *Jazyk nad abecedou* Σ *je ľubovoľná množina slov nad abecedou* Σ.

Definícia. *Majme slová* $u = a_1...a_n, v = b_1...b_m$, *kde* $\forall i, a_i, b_i \in \Sigma$, *potom ich zreťazením (označujeme* $u \cdot v$, *alebo* uv*) je slovo* $a_1...a_nb_1...b_m$.

Definícia. *Nech* L_1, L_2 *sú jazyky, potom ich zreťazením (označujeme* $L_1 \cdot L_2$, *alebo* L_1L_2*) je jazyk* $L = \{uv | u \in L_1, v \in L_2\}$.

Definícia. *Nech w je slovo. Mocninu slova w (označujeme w^i pre $i \in \mathbb{N}$) definujeme induktívne takto:*

$$w^0 = \epsilon,$$

$$w^i = w \cdot w^{i-1} \textit{ pre } i > 0.$$

Definícia. *Nech L je jazyk. Mocninu jazyka L (označujeme L^i pre $i \in \mathbb{N}$) definujeme induktívne takto:*

$$L^0 = \{\epsilon\},$$

$$L^i = L \cdot L^{i-1} \textit{ pre } i > 0.$$

Definícia. *Iteráciu jazyka L definujeme ako $L^* = \bigcup_{i=0}^{\infty} L^i$.*
Kladnú iteráciu jazyka L definujeme ako $L^+ = \bigcup_{i=1}^{\infty} L^i$.

Definícia. *Nech $w = a_1 a_2 ... a_n$, potom reverz slova w je slovo $w^R = a_n a_{n-1} ... a_1$. Nech L je jazyk, potom reverz jazyka L je jazyk $L^R = \{w^R | w \in L\}$.*

Automaty vo všeobecnosti pracujú tak, že na vstupe dostanú slovo, nad ktorým následne pracujú, pričom prechádzajú stavmi. Ak výpočet skončí v akceptačnom stave, vstupné slovo akceptujú. V tejto práci sa budeme zaoberať vzťahom medzi deterministickými a nedeterministickými konečnými automatmi, preto ich teraz formálne zadefinujeme. Využijeme na to štvorice definícií o automate, konfigurácii, kroku výpočtu a jazyku akceptovanom automatom:

Definícia. *Deterministický konečný automat (budeme označovať aj DKA) je 5-tica $(K, \Sigma, \delta, q_0, F)$, kde K je konečná množina stavov, Σ je abeceda, $\delta : K \times \Sigma \to K$ je prechodová funkcia, q_0 je počiatočný stav a $F \subseteq K$ je množina akceptačných stavov.*

Definícia. *Konfigurácia DKA je prvok $(q, w) \in K \times \Sigma^*$, kde q je stav automatu a w je ešte nespracovaný sufix vstupného slova.*

Definícia. *Krok výpočtu DKA A je relácia $\vdash_A$ na konfiguráciach definovaná takto:*

$$(q, aw) \vdash_A (p, w) \Leftrightarrow p = \delta(q, a).$$

Krok výpočtu budeme označovať aj $\vdash$, pokiaľ bude z kontextu zrejmé, na aký DKA sa vzťahuje.

Definícia. *Jazyk akceptovaný DKA A je množina*

$$L(A) = \{w | \exists q \in F : (q_0, w) \vdash^* (q, \epsilon)\}.$$

Poznámka. $\vdash^*$ je reflexívno-tranzitívny uzáver relácie $\vdash$, teda označenie $Q_1 \vdash_A^* Q_2$ znamená, že automat A prejde z konfigurácie Q_1 do konfigurácie Q_2 na nejaký počet krokov.

Nasleduje definícia nedeterministického konečného automatu, ktorá pozostáva podobne ako pri deterministickom zo štyroch častí. Definíciu konfigurácie a jazyka akceptovaného nedeterministickým konečným automatom si však dovoľujeme vynechať, nakoľko sú formálne rovnaké, ako pre DKA.

Definícia. *Nedeterministický konečný automat (budeme označovať aj NKA) je 5-tica $(K, \Sigma, \delta, q_0, F)$, kde K je konečná množina stavov, Σ je abeceda, $\delta : K \times (\Sigma \cup \{\epsilon\}) \to 2^K$ je prechodová funkcia, q_0 je počiatočný stav a $F \subseteq K$ je množina akceptačných stavov.*

Definícia. *Krok výpočtu NKA A je relácia $\vdash_A$ na konfiguráciach definovaná takto:*

$$(q, aw) \vdash_A (p, w) \Leftrightarrow p \in \delta(q, a)$$

.

Poznámka. Hlavný rozdiel medzi deterministickým a nedeterministickým konečným automatom spočíva v tom, že nedeterministický automat sa môže rozhodovať, do ktorého zo stavov daných prechodovou funkciou (táto množina môže byť aj prázdna) sa automat po nasledujúcom kroku výpočtu dostane. Táto vlastnosť môže viesť k výraznému rozdielu v počte stavov NKA oproti DKA akceptujúcich rovnaký jazyk. NKA sa od DKA líši aj tým, že môže urobiť tzv. prechod na ϵ, teda urobiť krok výpočtu bez prečítania ďalšieho písmena zo vstupu. Ako neskôr ukážeme, bez tejto vlastnosti by NKA nielenže nestratil na sile, ale nebol by ani potrebný nárast počtu stavov, pre zostrojenie ekvivalentného NKA bez ϵ-prechodov.

1.2 Charakterizácia konečných automatov

V predchádzajúcej časti sme zadefinovali dva podobné modely automatov, naskytá sa preto prirodzená otázka, aký je vzťah medzi ich silou, teda medzi množinami jazykov, ktoré sú schopné akceptovať. Zjavne ku každému DKA vieme zostrojiť ekvivalentný NKA. Ukazuje sa, že to platí aj opačne a teda modely sú rovnako silné. Množine jazykov, ktoré sú schopné akceptovať hovoríme trieda regulárnych jazykov.

Veta 1.1. *K ľubovoľnému NKA A existuje NKA A' bez ϵ-prechodov taký, že $L(A) = L(A')$ a počet stavov A a A' je rovnaký.*

Načrtneme iba ideu dôkazu: A' zostrojíme z A tak, že prechody na ϵ zlúčime do jedného kroku výpočtu s predošlým krokom výpočtu na písmeno.

Veta 1.2. *K ľubovoľnému NKA A existuje DKA A' taký, že $L(A) = L(A')$.*

Dôkaz. Nebudeme robiť celý dôkaz, ukážeme však štandardnú konštrukciu, ktorá sa v ňom používa, pretože zhora ohraničuje počet stavov potrebný na konštrukciu DKA pre daný NKA.
Bez újmy na všeobecnosti predpokladajme, že A neobsahuje ϵ-prechody. Zostrojme $A' = (K', \Sigma', \delta', q_0', F')$ takto:
$K' = 2^K$,
$\Sigma' = \Sigma$,
$q_0' = \{q_0\}$,
$F' = \{X | X \cap F \neq \emptyset\}$ a
$\delta'(Q, a) = \bigcup_{q \in Q} \delta(q, a)$.
A' pracuje tak, že akoby robí paralelne všetky výpočty na A' a akceptuje vtedy, keď aspoň jeden výpočet skončí v akceptačnom stave. Dôkaz, že $L(A) = L(A')$ prenechávame na čitateľa. $\square$

V tomto odseku sa budeme zaoberať vetou, ktorá nám bude neskôr veľmi užitočná, pretože okrem toho, že charakterizuje regulárne jazyky, hovorí o počte stavov minimálneho DKA akceptujúceho daný regulárny jazyk. Najskôr však zadefinujme potrebné pojmy:

Definícia. *Binárna relácia $\square$ na Σ^* sa nazýva sprava invariantná (vzhľadom na operáciu zreťazenia), ak platí:*

$$x \square y \Longrightarrow \forall w : xw \square yw.$$

Definícia. *Počet tried relácie ekvivalencie nazývame index. Ak je tento počet konečný, hovoríme, že relácia je konečného indexu.*

Veta 1.3. *(Myhill-Nerode) Majme jazyk $L \subseteq \Sigma^*$. Nasledujúce tvrdenia sú ekvivalentné:*

1. *L je regulárny jazyk*
2. *L je zjednotením niekoľkých tried ekvivalencie nejakej sprava invariantnej relácie ekvivalencie konečného indexu*
3. *Relácia R_L definovaná $uR_Lv \iff (\forall x : ux \in L \iff vx \in L)$ je reláciou ekvivalencie konečného indexu.*

Nebudeme robiť dôkaz vety, spomenieme iba úvahy vedúce k dôkazu implikácie $3 \Rightarrow 1$, pretože z nich vyplýva, že minimálny DKA pre daný regulárny jazyk musí obsahovať toľko stavov, koľko má relácia R_L tried ekvivalencie.
Snažíme sa vytvoriť na základe R_L DKA A pre jazyk L. Uvažujme slová u, v, ktoré sú v relácii R_L. Keď položíme $x = \epsilon$, dostávame $u \in L \iff v \in L$. Potom je L zjednotením niekoľkých tried ekvivalencie R_L. Uvažujme teraz u, v také, že nie sú v R_L. To znamená, že existuje x také, že bez újmy na všeobecnosti $ux \in L$ a $vx \notin$

L. Pre ľubovoľný DKA akceptujúci L potom platí, že po prečítaní slov u a v musí skončiť v rôznych stavoch, inak by skončil v rovnakých stavoch aj pre slová ux a vx. Z toho vyplýva, že DKA musí mať aspoň toľko stavov, koľko má R_L tried ekvivalencie. Automat A teda zostrojíme tak, že si bude pri výpočte v stave pamätať, do ktorej triedy ekvivalencie zatiaľ prečítaná časť slova patrí. Na konci skontroluje, či sa nachádza v triede ekvivalencie, ktorá je podmnožinou L.

1.3 Ciele práce

V dôkaze tvrdenia o ekvivalencii deterministických a nedeterministických konečných automatov sme použili štandardnú konštrukciu pre zostrojenie DKA, ktorej stavmi boli podmnožiny stavov príslušného NKA. Teda pre $|K|$ stavov NKA nám bude stačiť $2^{|K|}$ stavov pre zostrojenie ekvivalentného DKA. Naskytá sa prirodzená otázka, či je tento exponenciálny nárast stavov nutný. Ukážeme príklad jazyka, v ktorom exponenciálny nárast potrebný je: $L = \{w \in \{0,1\}^* | \textit{n-té písmeno od konca je 1} \}$[3]. Pre L poľahky zostrojíme NKA s $n+1$ stavmi. Na vytvorenie ekvivalentného DKA však bude potrebných 2^n stavov, pretože v R_L bude samostatná trieda ekvivalencie pre každý n-bitový suffix.
Ďalšou otázkou, ktorú si môžme položiť je, či existujú regulárne jazyky, pre ktoré je stavová zložitosť DKA pre NKA "dobrá", teda rádovo lepšia ako exponenciálna. Jednoduchým príkladom takého jazyka je $\{\epsilon\}$. Otázka, ktorú si teda budeme klásť a na ktorú sa budeme snažiť v tejto práci hľadať odpovede znie, pre aké regulárne jazyky je nárast stavovej zložitosti zostrojeného DKA k danému NKA rádovo lepší ako exponenciálny.

Kapitola 2

Jazyky definované podslovom

Uvažujme jazyky, ktoré sú tvorené práve takými slovami w, ktoré obsahujú nejaké dané neprázdne podslovo x. Ak konštruujeme konečný automat A akceptujúci takýto jazyk, stačí, ak A pri výpočte overí, či vstupné slovo obsahuje požadované podslovo a zvyšok vstupu môže byť ľubovoľný. Ukážeme, že nedeterministický automat potrebuje na overenie podslova aspoň $|x|+1$ stavov a deterministickému na to postačí rovnako veľa stavov.

Označenie. *Nech je daná ľubovoľná konečná abeceda Σ, nech je dané ľubovoľné $n \geq 1$ a slovo $x = a_1a_2...a_n$; $a_1, a_2, ..., a_n \in \Sigma$. Jazyk $L = \{uxv | u, v \in \Sigma^*\}$ budeme označovať L_x.*

Lema 2.1. *Nech je dané ľubovoľné neprázdne slovo x, nech $A = (K, \Sigma, \delta, q_0, F)$ je nedeterministický konečný automat taký, že $L(A) = L_x$, potom počet stavov A je aspoň $|x| + 1$.*

Dôkaz. Sporom nech počet stavov A je najviac $|x|$. Označme $q_0, q_1, ..., q_m, m < |x|$ stavy automatu A. Uvažujme slovo $v = x = a_1a_2...a_n, n \in \mathbb{N}, a_1, a_2, ..., a_n \in \Sigma$. Skúmajme výpočet A pre vstupné slovo v. Platí $v = x$, teda $v \in L_x$, teda existuje taký výpočet, že A dočíta celé slovo v a skončí v nejakom akceptačnom stave $q_f \in F$. V každom kroku výpočtu pritom prečíta najviac jedno písmeno zo vstupu, teda výpočet obsahuje aspoň $|v| = n$ bez-ϵ krokov. Keďže $|v| = |x| > |K - \{q_0\}|$, z dirichletovho princípu dostávame, že musia existovať stavy $q_i, q_j, i, j \in \mathbb{N}, i, j < m$ také, že

$$(q_0, v) \vdash^* (q_i, a_ra_{r+1}...a_n) \vdash^* (q_j, a_sa_{s+1}...a_n) \vdash^* (q_f, \epsilon),$$

pričom $q_i = q_j$ a $r < s$. Nech $w = a_1a_2...a_{r-1}a_sa_{s+1}...a_n$. Potom existuje výpočet

$$(q_0, w) \vdash^* (q_i, a_sa_{s+1}...a_n) \vdash^* (q_f, \epsilon)$$

Dostávame $w \in L(A)$ a zároveň $w \notin L_x$ pretože $|w| < |x|$, čo je spor s predpokladom. $\square$

Lema 2.2. *Nech je dané ľubovoľné neprázdne slovo x, potom existuje deterministický konečný automat A taký, že $L(A) = L_x$ a A má $|x| + 1$ stavov.*

Dôkaz. V dôkaze zostrojíme DKA s požadovanými vlastnosťami. Ten si bude v stavoch pamätať najdlhší prefix daného slova x z naposledy prečítaných písmen (pre jednoduchšie čítanie označme tento prefix u). Po prečítaní ďalšieho písmena zo vstupu (označme ho c) bude hľadať aktuálne najdlhší prefix (teda nasledujúci stav) tak, že overí či je $v = uc$ prefix slova x. Ak áno, musí byť toto najdlhší aktuálny prefix, keďže ním bolo u. Ak nie, overí, či je v bez prvého písmena takým prefixom, teda či je v' prefix slova x, pričom $av' = v$ pre nejaké písmeno a. Tento proces sa opakuje až kým automat nájde prefix slova x (ten musí byť pri tomto postupe najdlhší, keďže všetky potenciálne dlhšie už automat overil). V krajnom prípade takto po jednom písmene skracuje pôvodné slovo, až kým ostane overiť prázdne slovo, ktoré je prefixom každého slova, teda aj x. Ak sa raz automat dostane do stavu, v ktorom najdlhší prefix je rovný slovu x, bude sa cykliť už len v tomto stave, keďže vstup obsahuje celé požadované podslovo.

Formálne, pre dané $x = a_1a_2...a_n, n \in \mathbb{N}, a_1, a_2, ..., a_n \in \Sigma$ zostrojme deterministický konečný automat A pre L_x takto:
$A = (K, \Sigma, \delta, q_0, F)$
$K = \{q_0, q_1, ..., q_n\}$
$F = \{q_n\}$
$\delta(q_n, c) = q_n$; pre $c \in \Sigma$
$\delta(q_i, c) = q_j$; pre $c \in \Sigma, i \in \{0, 1, ..., n-1\}$, kde j je maximálne číslo také, že $\exists k \in \{1, ..., i+1\} : a_ka_{k+1}...a_ic = a_1a_2...a_j$, ak také j neexistuje, položíme $j = 0$.

Prípad keď $j = 0$ reprezentuje, že sme nenašli dlhší prefix x ako prázdne slovo, k reprezentuje vynechávanie začiatočných písmen pri hľadaní aktuálne najdlhšieho prefixu x, pre $k = 1$ dostávame pôvodné, neskrátené slovo, pre $k = i + 1$ dostávame $c = a_1$)
Poznamenávame, že zo stavu $q_i \neq q_n$ sa po prečítaní písmena a_{i+1} dostaneme do stavu q_{i+1}, keďže maximálne j dostaneme pre $k = 1 : a_1a_2...a_ia_{i+1} = a_1...a_{i+1}$. Formálny dôkaz, že A akceptuje práve požadovaný jazyk prenechávame na čitateľa. □

Veta 2.3. *Nech je dané ľubovoľné slovo x, nech $A_1 = (K_1, \Sigma_1, \delta_1, q_{0,1}, F_1)$ je nedeterministický konečný automat taký, že $L(A_1) = L_x$, potom existuje deterministický konečný automat $A_2 = (K_2, \Sigma_2, \delta_2, q_{0,2}, F_2)$ taký, že $L(A_2) = L_x$ a súčasne $|K_2| = |K_1|$.*

Dôkaz. Dôsledok predchádzajúcich dvoch liem. □

Kapitola 3

Jazyky nad unárnou abecedou

Pri jazykoch nad unárnou abecedou je štruktúra slova zrejmá, konečnému automatu teda stačí overovať, či má slovo na vstupe správnu dĺžku. Ukazuje sa, že pri konštrukcii deterministického konečného automatu k danému n-stavovému nedeterministickému konečnému automatu na to vo všeobecnosti potrebujeme $O(e^{\sqrt{n \log n}})$ stavov [2]. Je to spôsobené potrebou simulovať viacero rôznych cyklov $c_1, ..., c_k$ v rôznych vetvách výpočtu, na čo je potrebný cyklus dĺžky $lcm(c_1, ..., c_k)$. V tejto kapitole ukážeme, že za určitých podmienok je potrebný nárast stavov DKA konštruovaného k NKA nad unárnou abecedou rádovo menší, ako vo všeobecnom prípade.

Definícia. *Postupnosť stavov konečného automatu $q_0, ..., q_n$ takú, že zo stavu $q_i, i < n$ existuje jediný prechod a to do stavu q_{i+1} a súčasne do stavu $q_j, 1 \leq j \leq n$ vedie jediný prechod a to zo stavu q_{j-1}, budeme nazývať reťaz.*

Označenie. *Označme $|K_{cyk}(A)|$ počet stavov konečného automatu A, ktoré sú súčasťou nejakého cyklu.*

Nech sú dané prirodzené čísla $x_1, ..., x_n$, také, že $gcd(x_1, ..., x_n) = 1$. Funkcia $G(x_1, ..., x_n)$ nadobúda hodnotu najväčšieho čísla y takého, že rovnica $x_1k_1 + ... + x_nk_n = y$ nemá riešenie v prirodzených číslach. Problém hľadania $G(x_1, ..., x_n)$ je známy ako Frobéniov problém. Uvedieme známy výsledok skúmania tohto problému [1][4]. Nie je najpresnejší, ale pre naše potreby bude postačujúci.

Lema 3.1. *Nech sú dané prirodzené čísla $1 < x_1 < ... < x_n$ také, že $gcd(x_1, ..., x_n) = 1$, pričom $n \geq 2$. Potom*

$$G(x_1, ..., x_n) \leq (x_1 - 1)(x_n - 1) - 1.$$

Z uvedenej lemy vyplýva dôsledok, ktorý použijeme v neskoršom dôkaze:

Dôsledok 3.1.1. *Nech sú dané prirodzené čísla $1 \leq x_1 < ... < x_n \leq m$. Nech*

$$Y = \{y | \exists k_1, ..., k_n \in \mathbb{N} : x_1k_1 + ... + x_nk_n = y\}.$$

Potom množina $Y' = \{y | y \in Y, y > m^2\}$ *tvorí aritmetickú postupnosť s periódou* $gcd(x_1, ..., x_n)$.

Veta 3.2. *Nech je daný nedeterministický konečný automat* $A = (K, \Sigma, \delta, q_0, F)$ *s* n *stavmi, pričom* $\Sigma = \{a\}$. *Zostrojme orientovaný graf* $G_A = (V, E)$ *takto:* $V = K$, $E = \{(p, q) | q \in \delta(p, a)\}$. *Ďalej zostrojme* G'_A *tak, že nahradíme každý silne súvislý komponent v* G_A *jedným vrcholom. Nech platí*

$$\text{Z každého vrchola v } G'_A \text{ vedie najviac jedna hrana,} \tag{3.1}$$

Potom existuje deterministický konečný automat $A' = (K', \Sigma, \delta', q'_0, F')$ *taký, že* $L(A') = L(A)$ *a* $|K'| \leq n^2 + 2n$.

Dôkaz. V dôkaze vety o normálnom tvare NKA bez prechodov na ϵ sme použili konštrukciu, ktorá zachovávala počet stavov. Môžeme teda bez újmy na všeobecnosti predpokladať, že A neobsahuje prechody na ϵ. Deterministický automat môže obsahovať maximálne jeden cyklus, naproti tomu nedeterministický automat ich môže obsahovať viacero. Kľúčovým problémom dôkazu preto bude simulácia týchto cyklov v jednom deterministickom cykle za použitia dokázateľne málo stavov. Pre jednoduchšie pochopenie štruktúry automatu A, ho rozdelíme na silne súvislé komponenty a ukážeme, že predpoklad (3.1) zaručuje, že nebude potrebné simulovať viacero nesúdeliteľných cyklov v rôznych vetvách výpočtu. Na základe Dôsledku 3.1.1 nájdeme cyklus postačujúci na simuláciu všetkých cyklov v A aj odhadneme nárast stavov, ktoré nie sú súčasťou žiadneho cyklu v A'.

Graf G'_A je acyklický, lebo v opačnom prípade by sme mohli nejaký silno súvislý komponent zväčšiť a dostali by sme spor s definíciou silne súvislého komponentu. Podmienka (3.1) zaručuje, že vetvenie je možné len vrámci jedného silne súvislého komponentu. Preto je postupnosť silne súvislých komponentov, ktorými pri výpočte automat A prechádza jednoznačne daná až na dĺžku v závislosti od toho, v ktorom stave výpočet skončí.

Každý silne súvislý komponent buď obsahuje aspoň jeden cyklus, alebo je to jediný vrchol (stav). Ľubovoľný akceptačný výpočet, ktorý prechádza silno súvislými komponentami $K_1, ..., K_l$ môžeme zapísať ako postupnosť stavov[1] $M_1 ... M_l$, kde M_i je podpostupnosť stavov patriacich komponentu K_i. Z acyklickosti grafu G'_A vyplýva, že ak automat A vo výpočte opustil nejaký silno súvislý komponent, už sa doň v tomto výpočte nevrátil. Vrámci jedného silne súvislého komponentu môže výpočet prejsť niekoľkými cyklami a tiež prejsť niekoľkými stavmi bez ukončenia cyklu. Pre silne súvislý komponent K_i obsahujúci cykly $C_{i,1}, ..., C_{i,r_i}$ teda môžme počet stavov podpostupnosti

[1] Výpočet sme definovali ako postupnosť konfigurácií. V tomto prípade však uvažujeme automat bez prechodov na ϵ nad unárnou abecedou, preto je výpočet jednoznačne daný postupnosťou stavov, ktorými prechádza.

M_i rozpísať nasledovne $|M_i| = x_{i,0}+x_{i,1}|C_{i,1}|+...+x_{i,r_i}|C_{i,r_i}|$, kde $x_{i,j}$ je počet výskytov ukončeného cyklu $C_{i,j}$ a $x_{i,0}$ je počet stavov ktoré ostanú v M_i po vynechaní ukončených cyklov. Pre rôzne výpočty automatu A sa môže $x_{i,0}$ meniť, ale nikdy nepresiahne počet stavov daného silne súvislého komponentu. Dĺžku celého výpočtu teda môžme vyjadriť ako $|M_1| + ... + |M_l| = x_0 + x_{1,1}|C_{1,1}| + ... + x_{l,r_l}|C_{l,r_l}|$, kde $x_0 = x_{1,0} + ... + x_{l,0}$. Hodnota x_0, ako ani hodnoty $|C_{1,1}|, ..., |C_{l,r_l}|$ pritom nepresiahne n. Podľa Dôsledku 3.1.1 dostávame $|M_1| + ... + |M_l| = p + y \cdot gcd(|C_{1,1}|...|C_{l,r_l}|)$ pre nejaké y a pre

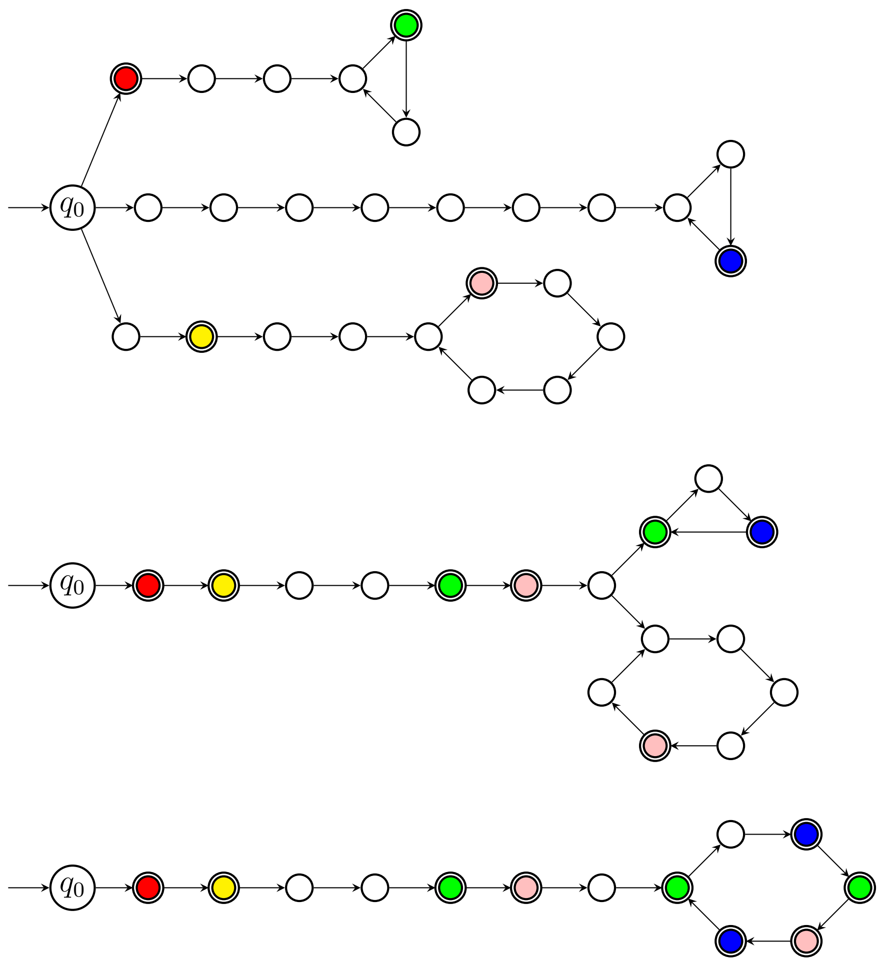
$$p \leq n^2 + n. \tag{3.2}$$

Takýmto spôsobom môžme (pre rôzne veľké x_0 aj rôzne silne súvislé komponenty) vyjadriť dĺžku každého akceptačného výpočtu a teda aj každého $w \in L(A)$. Počet rôznych postupností silne súvislých komponentov, ktorými pri výpočte A prechádza je konečný, lebo graf G'_A je acyklický, podobne aj počet hodnôt, ktoré môže nadobudnúť x_0 je nanajvýš n, teda konečný. Z toho vyplýva, že vieme zostrojiť automat A''' ekvivalentný k A, ktorý sa rozvetví do niekoľkých reťazí dĺžky najviac p a na konci každej reťaze je prechod do cyklu. Ak zarátame rovnako dlhé cykly z A''' ako jeden cyklus, potom platí, že $|K_{cyk}(A''')| \leq |K_{cyk}(A)|$. Vyplýva to zo spôsobu tvorby cyklov v A'''.

Ku automatu A''' môžeme zostrojiť ekvivalentný automat A'', ktorý bude mať len jednu reťaz. Postačí nám najdlhšia reťaz a patričná úprava akceptačných stavov. Pomôžeme si technikou využitou pri štandardnej konštrukcii DKA k NKA, teda prehľadávaním do šírky. Na konci reťaze bude prechod do každého z cyklov. Samozrejme treba upraviť aj akceptačné stavy v cykloch, to si môžeme predstaviť ako cyklický posun akceptačného stavu (stavov).

Deterministický automat môže obsahovať najviac jeden cyklus, teda v A' potrebujeme simulovať niekoľko stavovo disjunktných cyklov z A'' v jednom cykle. Analyzujme dĺžky cyklov v A''. Dĺžka každého cyklu je daná na základe spomenutých úvah Dôsledkom 3.1.1 ako najväčší spoločný deliteľ všetkých cyklov obsiahnutých v silne súvislých komponentoch, ktorými A pri výpočte prešiel. Pre rôzne postupnosti silne súvislých komponentov teda môžeme dostať rôzne dĺžky cyklov. Označme $K_{F_1}, ..., K_{F_k}$ rôzne silne súvislé komponenty obsahujúce akceptačné stavy. Potom existuje k rôznych postupností silne súvislých komponentov, pre ktoré existuje akceptačný výpočet v A. Bez újmy na všeobecnosti predpokladajme, že najdlhšia takáto postupnosť prechádza komponentami v poradí $K_{1,0}, ..., K_{F_1}, K_{2,0}, ..., K_{F_2}, ..., K_{F_k}$. Teda je potrebné simulovať súčasne k rôznych cyklov $C_1, ..., C_k$, pričom platí $|C_i| = gcd(K_{1,0}, ..., K_{F_i})$, kde $gcd(K_x)$ znamená najväčší spoločný deliteľ všetkých cyklov v komponente K_x. Teda platí, že $|C_i|$ delí $|C_j|$ pre $i > j$. Každý cyklus možno simulovať v násobne dlhom cykle. Všetky cykly z A'' teda môžme simulovať v cykle dĺžky

$$lcm(|C_1|, ..., |C_k|) = lcm(gcd(K_{1,0}, ..., K_{F_1}), ..., gcd(K_{1,0}, ..., K_{F_k})) = |C_1|.$$



Obr. 3.1: Automat s viacerými reťazami A''', automat s jednou reťazou A'' a deterministický automat A'

Na obrázku 3.1 môžme vidieť príklad opísaného postupu konštrukcie automatu s jednou reťazou a následnej konštrukcie deterministického konečného automatu. Ku každému akceptačnému stavu z A''' sú rovnakou farbou zvýraznené stavy z A'' a A', ktoré sú na základe neho v množine akceptačných stavov A'' a A'. Poznamenávame, že pri tomto postupe počet stavov A'' ani A' nepresiahne počet stavov A'''.

Analyzujme počet stavov A' vzhľadom k n. Najskôr sme vytvorili automat A''', ktorý obsahoval niekoľko reťazí a cyklov, pričom platilo, že najdlhšia reťaz nemala viac ako p stavov. Potom sme zostrojili automat A'', ktorý obsahoval jednu reťaz dĺžky p a rôzne dlhé cykly z A''', pričom platilo, že $|K_{cyk}(A'')| \leq |K_{cyk}(A)|$. Napokon sme zostrojili deterministický konečný automat A' obsahujúci rovnakú raťaz ako A'' a najdlhší cyklus z A''. Z nerovnosti 3.2 potom dostávame, že počet stavov A' je nanajvýš $n^2 + n + |K_{cyk}(A)| \leq n^2 + 2n$. □

V predošlom dôkaze sme zostrojili automat A'' s pomerne jednoduchou a prehľadnou štruktúrou. Zavedieme pre túto štruktúru normálny tvar, ktorý využijeme v neskoršom dôkaze.

Definícia. *Nedeterministický konečný automat* $A = (K, \{a\}, \delta, q_0, F)$ *je v normálnom tvare, ak obsahuje iba reťaz* $q_0, ..., q_k$ *a stavovo disjunktné cykly* $C_1, ..., C_l$*, kde* $C_i = \{q_{i,0}, ..., q_{i,x_i}\}$*, pričom* $\delta(q_k, a) = \{q_{1,0}, ..., q_{l,0}\}$.

Lema 3.3. *[2] K ľubovoľnému nedeterministickému konečnému automatu* $A = (K, \{a\}, \delta, q_0, F)$ *nad unárnou abecedou existuje nedeterministický konečný automat* $A' = (K', \{a\}, \delta', q_0', F')$ *v normálnom tvare taký, že* $|K'| \leq |K|^2 + 2|K|$.

Dôkaz tejto lemy nebudeme uvádzať, nakoľko je veľmi podobný prvej časti dôkazu Vety 3.2 a je obsiahnutý aj v dokumente [2].

V nasledujúcej vete ukážeme, že nárast stavovej zložitosti DKA oproti ekvivalentnému NKA je pre jazyky nad unárnou abecedou závislý od "rozvetvenosti" daného NKA.

Veta 3.4. *Nech je daný nedeterministický konečný automat* $A = (K, \{a\}, \delta, q_0, F)$ *s* n *stavmi. Zostrojme orientovaný graf* $G_A = (V, E)$ *takto:* $V = K$, $E = \{(p, q) | q \in \delta(p, a)\}$. *Ďalej zostrojme* G'_A *tak, že nahradíme každý silne súvislý komponent v* G_A *jedným vrcholom. Nech* k *je počet rôznych nepredĺžiteľných ciest v* G'_A*. Potom existuje deterministický konečný automat* $A' = (K', \{a\}, \delta', q_0', F')$ *taký, že* $L(A') = L(A)$ *a zároveň platí, že* $|K'| \leq n^k + n^2 + 2n$.

Dôkaz. Z Lemy 3.3 vieme, že k A existuje ekvivalentný nedeterministický konečný automat A'' v normálnom tvare s najviac $n^2 + n$ stavmi. Označme v_0 vrchol v G'_A, ktorému zodpovedá v G_A silne súvislý komponent obsahujúci q_0. Odhadnime počet cyklov v A''. Tieto cykly sú dané na základe vyššie spomenutých úvah silne súvislými komponentami, ktorými A pri výpočte prechádza. Teda budú nás zaujímať rôzne cesty v G'_A, ktoré začínajú vo v_0. Z Vety 3.2 a jej dôkazu vyplýva, že pre dve cesty P_1, P_2 v grafe G'_A, kde P_2 je predĺžením P_1 stačí v A'' simulovať cyklus daný cestou P_1. Teda ak graf G'_A obsahuje k rôznych nepredĺžiteľných ciest[2], potom A'' neobsahuje viac ako k rôznych cyklov, označme ich $C_1, ..., C_l$, kde $l \leq k$. Pritom platí, že $|C_1| + ... + |C_l| \leq n$. Cykly $C_1, ..., C_l$ môžme simulovať v násobne dlhšom cykle, teda pre dĺžku cyklu $C_{A'}$ v deterministickom A' dostávame

$$|C_{A'}| = lcm(|C_1|, ..., |C_l|) \leq |C_1| \cdot ... \cdot |C_l| \leq n^l \leq n^k.$$

Počet stavov A', ktoré nie sú súčasťou cyklu podobne ako v A'' nepresiahne $n^2 + 2n$, teda $|K'| \leq n^k + n^2 + 2n$. □

Poznámka. Ľahko nahliadneme, že číslo k z uvedenej vety môže presiahnuť n, keďže môže byť exponenciálne veľké vzhľadom k počtu silne súvislých komponentov v grafe

[2]Podotýkame, že bez újmy na všeobecnosti môžme predpokladať, že všetky stavy v A sú dosiahnuteľné z q_0, teda všetky nepredĺžiteľné cesty v G'_A začínajú vo v_0

G_A. Teda cieľom tejto vety zjavne nebolo zlepšiť odhad stavovej zložitosti DKA k NKA pre jazyky nad unárnou abecedou[3], ale ukázať, že pre dostatočne malé k je tento nárast počtu stavov v zásade polynomiálny.

[3]Je dokázané, že pre jazyky nad unárnou abecedou je $e^{\sqrt{n \log(n)}}$ tesným odhadom stavovej zložitosti DKA k danému n-stavovému NKA[2] .

Kapitola 4

Ohraničené jazyky

V tejto kapitole budeme skúmať ohraničené jazyky, kotré sú nadtriedou jazykov nad unárnou abecedou skúmaných v predošlej kapitole. Tieto dve triedy jazykov majú významný súvis, pretože na ohraničený jazyk môžme nahliadať ako na zreťazenie niekoľkých jazykov nad unárnymi abecedami. Ako ukážeme, tejto predstave zodpovedajú aj konečné automaty akceptujúce ohraničené jazyky. Preto sa budeme opierať o výsledky z predošlej kapitoly, ako aj o už spomínané výsledky skúmania tejto oblasti [2]. Na úvod zavedieme niekoľko pojmov a predpokladov dôležitých pre túto kapitolu.

Definícia. *Jazyk L nazývame ohraničený, ak $L \subseteq w_1^* ... w_n^*$ pre nejaké slová $w_1, ..., w_n$.*

Poznámka. Z pohľadu porovnania stavovej zložitosti deterministických a nedeterministických konečných automatov nemá dĺžka slov $w_1, ..., w_n$ žiadny význam (ako sme dokázali v kapitole 2), preto budeme pre jednoduchosť v tejto kapitole predpokladať $|w_1| = ... = |w_n| = 1$, teda sú to písmená.

V tejto kapitole budeme uvažovať automaty bez nedosiahnuteľných stavov, teda pre každý stav q platí, že existuje akceptačný výpočet prechádzajúci stavom q. Ľahko môžme nahliadnuť, že odstránením nedosiahnuteľných stavov z ľubovoľného nedeterministického konečného automatu sa nezmení jazyk akceptovaný týmto automatom.

Uvažujme NKA $A = (K, \{a_1, ..., a_n\}, \delta, q_0, F)$ a $A' = (K', \{a'_1, ..., a'_n\}, \delta', q'_0, F')$ také, že $L(A) \subseteq a_1^* ... a_n^*$ a $L(A') \subseteq a_1'^* ... a_n'^*$. Zároveň platí $K = K', F = F', q_0 = q'_0$ a $\delta(q, a_i) = \delta'(q, a'_i)$. Teda jediné, čím sa tieto automaty môžu líšiť, je vstupná abeceda. Predpokladajme, že $a_1, ..., a_n$ sú navzájom rôzne písmená. Ľahko nahliadneme, že ak zostrojíme deterministický konečný automat B ekvivalentný s A, potom vieme zostrojiť aj deterministický konečný automat B' ekvivalentný s A', ktorý sa bude s B líšiť len vstupnou abecedou, podobne ako A a A'. Môžme teda bez újmy na všeobecnosti predpokladať, že pre ľubovoľný ohraničený jazyk $L(A) \subseteq a_1^* ... a_n^*$ daný nedeterministickým konečným automatom A sú písmená $a_1, ..., a_n$ navzájom rôzne. Vďaka tomuto predpokladu sa budeme môcť vyhnúť niektorým technikalitám v dôkazoch.

Ako sme už v úvode načrtli, k ohraničeným jazykom budeme pristupovať ako k zreťazeniu niekoľkých jazykov nad unárnou abecedou. V NKA akceptujúcom ohraničený jazyk nás teda budú zaujímať stavy, do ktorých sa dá dostať v jednom kroku na daný prvok abecedy. Na základe tejto vlastnosti rozdelíme stavy do niekoľkých partícií. Do jedného stavu môže viesť aj niekoľko prechodov na rôzne písmená. Aby bolo rozdelenie do partícií jednoznačné, definujeme ju nasledovne:

Definícia. *Nech je daný nedeterministický konečný automat $A(K, \Sigma, \delta, q_0, F)$ taký, že $L(A) \subseteq a_1^*...a_n^*$. Množinu*

$$P_i = \{q | q \in K, i \textit{ je maximálne číslo také, že } q \in \delta(p, a_i) \textit{ pre nejaké } p \in K\}$$

budeme nazývať partícia, alebo aj partícia písmena a_i.

Pre takto definovanú partíciu sa všetky stavy[1] nedeterministického konečného automatu akceptujúceho ohraničený jazyk dajú jednoznačne rozdeliť do niekoľkých po dvoch disjunktných partícií. V nasledujúcej leme ukážeme, že automat pri výpočte prechádza partíciami akoby jedným smerom, teda nevracia sa späť do partícií, ktoré prešiel. Podobným spôsobom pracujú aj ohraničené zásobníkové automaty, o ktorých je dokázané, že akceptujú práve triedu ohraničených jazykov [7]

Lema 4.1. *Uvažujme regulárny ohraničený jazyk $L \subseteq a_1^*...a_n^*$ a NKA $A = (K, \Sigma, \delta, q_0, F)$ taký, že $L(A) = L$. Potom pre ľubovoľnú jeho partíciu P_i a stav $q \in P_i$ platí, že $\delta(q, a_j) = \emptyset$ pre $j < i$.*

Dôkaz. Predpokladajme sporom, že by pre nejaké $q \in P_i$ existoval[2] stav p taký, že $p \in \delta(q, a_j)$, kde $j < i$. Potom by existoval aj výpočet na nejakom vstupe $w = xa_ia_jy$, kde $x, y \in \{a_1, ..., a_n\}^*$ taký, že $(q_0, w) \vdash^* (q', a_ia_jy) \vdash^{a_i} (q, a_jy) \vdash^{a_j} (p, y) \vdash^* (p', \epsilon)$, pre počiatočný stav q_0, akceptačný stav p' a nejaký stav q'. Teda pre slovo w platí, že $w \in L(A)$, potom ale dostávame spor, pretože $L(A) \subseteq a_1^*...a_j^*...a_i^*...a_n^*$. □

Ako sme ukázali, na ohraničené jazyky môžme nahliadať ako na zreťazenie jazykov nad unárnymi abecedami. Mohlo by sa teda zdať, že pre NKA akceptujúci ohraničený jazyk bude pri konštrukcii ekvivalentného DKA stačiť simulovať každú partíciu zvlášť ako samostatný NKA nad unárnou abecedou. Problém je však v tom, že medzi jednotlivými partíciami môže viesť viacero prechodov z rôznych stavov, čo môže spôsobiť, že túto partíciu bude potrebné simulovať akoby viac krát. Nasledujúcim kontrapríkladom ukážeme, že táto vlastnosť má za následok že pre ohraničené jazyky je potrebná väčšia stavová zložitosť DKA k NKA ako tomu bolo pri jazykoch nad unárnou abecedou.

[1] Jedinou výnimkou môže byť stav q_0, do ktorého nevedie žiadny prechod. Táto výnimka nám však pri ďalších úvahách nebude prekážať.

[2] Existencia takéhoto výpočtu je zaručená predpokladom dosiahnuteľnosti všetkých stavov

Veta 4.2. *Existuje NKA* $A = (K, \Sigma, \delta, q_0, F)$ *taký, že* $L(A) \subseteq a^*b^*$*, pre ktorý neexistuje ekvivalentný DKA* $A' = (K', \Sigma', \delta', q_0', F')$*, pre ktorý by platilo* $|K|' \leq ce^{\sqrt{|K|\log(|K|)}}$ *pre nejakú konštantu c.*

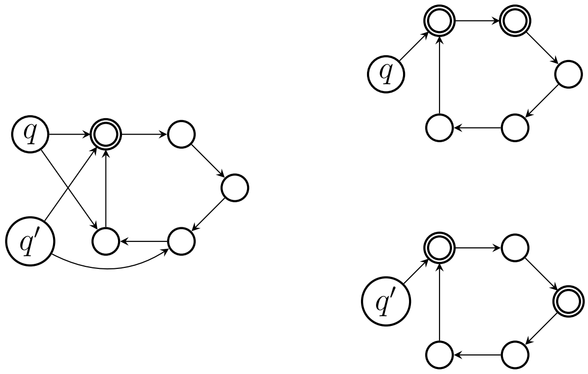
Dôkaz. Ohraničené jazyky si vďaka Leme 4.1 môžme predstaviť ako na seba nadväzujúce automaty nad unárnou abecedou. Napriek tomu je nárast stavov DKA k NKA väčší, ako tomu bolo pri unárnych jazykoch. Je to spôsobené tým, že z jednej partície do druhej môže nedeterministický automat prejsť z viacerých stavov, pričom sa môže ešte nedeterministicky rozhodovať, do ktorého stavu z ďalšej partície prejde. Teda ďalšiu partíciu môžme vnímať ako NKA nad unárnou abecedou s viacerými alternatívnymi vstupnými stavmi. Môže sa stať, že bude potrebné vytvoriť samostatnú kópiu tohto "automatu" (partície) pre každý z alternatívnych "počiatočných stavov". Najskôr spomenieme dve jednoduché úvahy, ktoré ukážu spôsob, ako vytvoriť potrebu simulovať jeden cyklus viac krát v osobitnom cykle, čo je hlavná myšlienka konštrukcie automatu A.

Nech p je nejaké prvočíslo. Analyzujme, koľko rôznych[3] cyklov dĺžky p vieme vytvoriť zmenami množiny akceptačných stavov. Ak nie je v cykle žiadny akceptačný stav, alebo sú akceptačné všetky stavy, potom ho môžeme nahradiť cyklom dĺžky jeden, teda tieto možnosti vylúčime. Ľahko vidno, že pre rôzny počet akceptačných stavov dostávame rôzne cykly. Tých môže byť $p-1$. Počínajúc číslom 5 možno pre každé prvočíslo vytvoriť dva rôzne cykly obsahujúce dva akceptačné stavy. Teda pre prvočíslo $p \geq 5$ existuje aspoň p rôznych cyklov dĺžky p.

Uvažujme cyklus C prvočíselnej dĺžky s jedným akceptačným stavom. Ďalej uvažujme stav $q \notin C$, z ktorého vedie niekoľko prechodov do rôznych, no nie všetkých stavov cyklu C. Deterministicky vieme túto situáciu simulovať pomocou štandardnej konštrukcie viacerými akceptačnými stavmi, ako je to znázornené na Obrázku 4.1. Keďže existuje $p' \in C$ taký, že $p' \notin \delta(q, a)$ pre ľubovoľné písmeno a a keďže C má prvočíselnú dĺžku, akceptačné stavy v takto zostrojenej deterministickej konštrukcii sa neopakujú s periódou menšou ako je dĺžka C. Teda nevieme túto situáciu deterministicky simulovať lepšie ako spomenutým spôsobom. Uvažujme ďalej stav q' pre ktorý sú definované prechody do niekoľkých stavov C tak, že nie sú len cyklickým posunom oproti prechodom z q. Keď chceme simulovať túto novo vzniknutú situáciu deterministicky, nemáme lepšiu možnosť ako pre každý zo stavov q, q' vytvoriť cyklus dĺžky C s príslušnými akceptačnými stavmi, podobne ako na Obrázku 4.1.

Uvažujme prvočísla $p_1, ..., p_k \geq 5$ také, že $p_1 + ... + p_k = n$. Zostrojme nedeterministický konečný automat $A = (K, \Sigma, \delta, q_0, F)$ takto:
$K = \{q_0\} \cup \{q_{1,0}, ..., q_{1,p_1}, ..., q_{k,0}, ..., q_{k,p_k}\} \cup \{q'_{1,0}, ..., q'_{1,p_1}, ..., q'_{k,0}, ..., q'_{k,p_k}\}$

[3]Pod rôznymi cyklami máme na mysli cykly s rôznym rozložením akceptačných stavov. Teda také rozloženie akceptačných stavov, ktoré nebude iba cyklickým posunutím.



Obr. 4.1: Pre deterministickú simuláciu cyklu s niekoľkými vstupnými stavmi môže byť potrebný samostatný cyklus pre každý zo vstupných stavov

$\Sigma = \{a, b\}$
$\delta(q_0, \epsilon) = \{q_{1,0}, ..., q_{k,0}\}$,
$\delta(q_{i,j}, a) = \{q_{i,(j+1)mod(p_i)}\}$ pre všetky $q_{i,j}$
$\delta(q_{i,j}, \epsilon) = \{q'_{i,0}, ..., q'_{i,j}\}$ pre $j < p_i - 1$
$\delta(q_{i,j}, \epsilon) = \{q'_{i,0}, q'_{i,2}\}$ pre $j = p_i - 1$
$\delta(q'_{i,j}, b) = \{q'_{i,(j+1)mod(p_i)}\}$
$F = \{q'_{1,p_1-1}, ..., q'_{k,p_k-1}\}$

V tomto prípade $a_1 = a, a_2 = b$. Platí, že do P_1 patria všetky $q_{i,j}$ a do P_2 patria všetky $q'_{i,j}$, pričom $|P_1| = |P_2| = n$. Tiež platí, že $L(A) \subseteq a^*b^*$. Automat pracuje tak, že v počiatočnom stave sa nedeterministicky rozhodne, do ktorého z k cyklov prejde, pričom i-ty cyklus má dĺžku p_i. V tomto cykle sa cyklí, kým sú na vstupe písmená a. Keď sa na vstupe objaví písmeno b, prejde do druhej partície opäť do cyklu rovnakej dĺžky. Pre každý stav cyklu z P_1 má pritom na výber inú množinu stavov z cyklu z P_2, do ktorých môže prejsť. Pre každý stav z P_1 bude teda potrebné vytvoriť nový cyklus s príslušnými akceptačnými stavmi, podobne ako na Obrázku 4.1. Príklad takéhoto automatu pre cykly dĺžky 5 a 7 je znázornený na obrázku 4.2. Neoznačené prechody na tomto obrázku sú prechodmi na ϵ, čo sme v obrázku neznačili kvôli prehľadnosti. Z toho, ako automat A pracuje vidno, že

$$L(A) = \{a^x b^y | \exists p \in \{p_1, ..., p_k\} :$$
$$y \geq p - 1 - x(mod(p)), \text{ pre } x < p - 1(mod(p));$$
$$y \equiv p - l(mod(p)), \text{ kde } l \in \{1, 3\}, \text{ pre } x \equiv p - 1(mod(p))\}.$$

Analyzujme index relácie ekvivalencie R_L, definovanej vo Vete 1.3. Ukážeme, že pre ľubovoľné rôzne $u = a^{x_1}b^{y_1}, v = a^{x_2}b^{y_2}$, pričom $x_1, x_2, y_1, y_2 < lcm(p_1, ..., p_k)$ neplatí uR_Lv. Teda budeme hľadať slovo b^z také, že práve jedno zo slov ub^z, vb^z patrí do jazyka $L(A)$. Dôkaz rozdelíme na tri prípady.

Prípad 1: $x_1 = x_2, y_1 \neq y_2$. Keďže y_1 a y_2 sú rôzne, musí existovať $p_j \in \{p_1, ..., p_k\}$

také, že $y_1 \not\equiv y_2(mod(p_j))$. Položme

$$z_i = p_i - y_1(mod(p_i)), \text{ pre } i \neq j.$$

Ďalej hľadáme z_j také, aby platilo:

$$z_j < p_j - 1 - x_1 - y_1(mod(p_j))$$

a súčasne $$z_j \geq p_j - 1 - x_1 - y_2(mod(p_j))$$

pre $x_1 = x_2 < p_j - 1(mod(p_j))$, v prípade $x_1 = p_j - 1$ položíme z_j, aby platilo:

$$z_j \not\equiv p_j - l - y_1(mod(p_j))$$

a súčasne $$z_j \equiv p_j - l - y_2(mod(p_j))$$

pre $l \in \{1, 3\}$. Existencia takého z_j vyplýva z toho, že y_1, y_2 sú rôzne a z toho, že $p_1, ..., p_k \geq 5$ sú navzájom nesúdeliteľné. Keďže $p_1, ..., p_k$ sú navzájom nesúdeliteľné, podľa Čínskej vety o zvyškoch existuje práve jedno $z < p_1 \cdot ... \cdot p_k$ také, že pre $i = 1, ..., k$ platí, že $z \equiv z_i(mod(p_i))$. Z toho, ako sme definovali z_i pre všetky i potom dostávame, že $ub^z \notin L(A)$, ale z definície z_j vyplýva, že $vb^z \in L(A)$.

Prípad 2: $x_1 \neq x_2, y_1 = y_2$. Opäť budeme hľadať slovo b^z tak, aby platilo $ub^z \notin L(A)$ a súčasne $vb^z \in L(A)$. Podobne ako v predošlom prípade existuje $p_j \in \{p_1, ..., p_k\}$ také, že $x_1 \not\equiv x_2(mod(p_j))$. Položme

$$z_i \equiv p_i - y_1(mod(p_i)), \text{ pre } i \neq j.$$

Najskôr analyzujme prípad $x_1, x_2 < p_j - 1(mod(p_j))$. Bez újmy na všeobecnosti predpokladajme, že $x_1 < x_2(mod(p_j))$. Položme

$$z_j \equiv p_j - 1 - x_2 - y_2(mod(p_j)).$$

Opäť podľa Čínskej vety o zvyškoch existuje práve jedno z také, že $z \equiv z_i$ pre všetky i. Analyzujme príslušnosť slov ub^z, vb^z do jazyka $L(A)$. Pre $i \neq j$ dostávame $y_1 + z_i \equiv 0(mod(p_i))$ a pre $i = j$ dostávame $y_1 + z_i \equiv p_i - 1 - x_2 < p_i - 1 - x_1(mod(p_i))$. Teda $ub^z \notin L(A)$. Naproti tomu vďaka $y_2 + z_j \equiv p_j - 1 - x_2(mod(p_j))$ platí $vb^z \in L(A)$. Prípad keď jedno z čísel x_1, x_2 je rovné $p_j - 1$ modulo p_j rozdelíme na dva prípady, v ktorých budeme bez újmy na všeobecnosti predpokladať[4], či $x_1 \equiv p_j - 1(mod(p_j))$, alebo $x_2 \equiv p_j - 1(mod(p_j))$. Prvý prípad je $x_1 \equiv p_j - 1(mod(p_j)), x_2 \geq 2(mod(p_j))$. Položíme

$$z_j \equiv p_j - 2 - y_2(mod(p_j)).$$

[4]Ak by nastal opačný prípad, stačí dokazovať rovnakým spôsobom $ub^z \in L(A) \wedge vb^z \notin L(A)$

Znovu nájdeme z podľa Čínskej vety o zvyškoch. Dostávame $y_1 + z \equiv y_2 + z \equiv p_j - 2(mod(p_j))$, teda $ub^z \notin L(A)$, naopak $vb^z \in L(A)$. Druhým prípadom je $x_1 \equiv 1(mod(p_j)), x_2 \equiv p_j - 1(mod(p_j))$. Položíme

$$z_j \equiv p_j - 3 - y_2(mod(p_j)).$$

Potom pre podobne nájdené z platí $y_1 + z \equiv y_2 + z \equiv p_j - 3(mod(p_j))$, teda opäť $ub^z \notin L(A) \wedge vb^z \in L(A)$.

Prípad 3: $x_1 \neq x_2, y_1 \neq y_2$. Ak pre nejaké $p_j \in \{p_1, ..., p_k\}$ platí $y_1 \equiv y_2(mod(p_j))$ resp. $x_1 \equiv x_2(mod(p_j))$, potom môžme postupovať podobne, ako v prípade 1, resp. 2. V opačnom prípade budeme opäť najskôr uvažovať prípad $x_1, x_2 < p_j - 1(mod(p_j))$ pre nejaké $p_j \in \{p_1, ..., p_k\}$, pričom BÚNV $x_1 < x_2(mod(p_j))$. Z toho ako je definovaný jazyk $L(A)$ vyplýva, že ak na A neexistuje akceptačný výpočet prechádzajúci cyklami dĺžky p_j pre vstup $w = a^x b^y$, kde $x < p_j - 1(mod(p_j))$, potom takýto výpočet neexistuje ani pre vstup $w' = a^{x_1} b^y$, kde $x_1 < x(mod(p_j))$. Teda v tomto prípade môžme opäť úspešne použiť rovnaký postup ako v prípade 1. Pre zvyšné prípady budeme podobne ako v prípade 2 BÚNV predpodkladať, či $x_1 \equiv p_j - 1(mod(p_j))$ alebo $x_2 \equiv p_j - 1(mod(p_j))$. Najskôr nech $x_1 \equiv p_j - 1(mod(p_j)), x_2 \geq 2(mod(p_j))$. Potom pre

$$z_j \equiv p_j - 1 - y_2(mod(p_j))$$

alebo

$$z_j \equiv p_j - 2 - y_2(mod(p_j))$$

platí, že $y_1 + z_j \not\equiv p_j - 1(mod(p_j)), y_1 + z_j \not\equiv p_j - 3(mod(p_j))$ a súčasne $y_2 + z_j \geq p_j - 2(mod(p_j))$. Teda pre takto zvolené z_j a pre $z_i, i \neq j$ zvolené rovnako ako v prípade 1 dostávame pre $z \equiv z_i(mod(p_i)), i = 1, ..., k$ znova $ub^z \notin L(A)$ a zároveň $vb^z \in L(A)$. Napokon nech $x_1 \equiv 1(mod(p_j)), x_2 \equiv p_j - 1(mod(p_j))$. Pre

$$z_j \equiv p_j - 1 - y_2(mod(p_j))$$

alebo

$$z_j \equiv p_j - 3 - y_2(mod(p_j))$$

platí, že $y_1 + z_j \not\equiv p_j - 1(mod(p_j))$ a zároveň $y_2 + z_j \equiv p_j - l(mod(p_j))$, kde $l \in \{1, 3\}$. Teda existuje z pre ktoré $ub^z \notin L(A)$ ale $vb^z \in L(A)$.

Ukázali sme, že pre všetky rôzne $x, y < lcm(p_1, ..., p_k)$ existuje pre vstupné slovo $a^x b^y$ v R_L samostatná trieda ekvivalencie. Pre funkciu

$$f(x) = max\{lcm(c_1, ..., c_m) | c_1 + ... + c_m = x\}$$

platí, že

$$\lim_{x \to \infty} \frac{f(x)}{e^{\sqrt{x \log(x)}}} = 1 \text{ [6]}.$$

Teda index relácie R_L je najmenej rádovo

$$\Omega(lcm(p_1, ..., p_k) \cdot lcm(p_1, ..., p_k)) = \Omega(e^{2\sqrt{n \log(n)}}).$$

Počet stavov automatu A je $2n+1$. Platí

$$\lim_{n\to\infty} \frac{e^{2\sqrt{n\log(n)}}}{e^{\sqrt{(2n+1)\log(2n+1)}}} = \infty.$$

□

Veta 4.3. *Pre ľubovoľný n-stavový nedeterministický konečný automat $A = (K, \Sigma, \delta, q_0, F)$ taký, že $L(A) \subseteq a_1^+ ... a_k^+$ existuje ekvivalentný deterministický konečný automat $A' = (K', \Sigma, \delta', q_0', F'$ taký, že $|K'| \leq ke^{k\sqrt{n\log(n)}}$.*

Dôkaz. Z Lemy 4.1 vyplýva, že na jednotlivé partície môžme nahliadať ako na automaty nad unárnou abecedou s viacerými alternatívnymi vstupnými stavmy (stavy z predošlých partícií). V najhoršom prípade je potrebné simulovať túto partíciu pre každý vstupný stav osobitne. Keďže $L(A) \subseteq a_1^+ ... a_k^+$ partície nemožno vo výpočte "preskočiť", ale výpočet musí prejsť všetkými partíciami postupne. Tiež vieme, že m-stavový unárny nedeterministický automat (partíciu) možno simulovať deterministickým automatom s $O(e^{\sqrt{m\log(m)}})$ stavmi. Pre partície $P_1, ...P_k$, teda v najhoršom prípade dostávame

$$\sum_{i=1}^{k} \prod_{j=1}^{i} e^{\sqrt{|P_j|\log(|P_j|)}} \leq k \prod_{i=1}^{k} e^{\sqrt{|P_i|\log(|P_i|)}} \leq ke^{k\sqrt{n\log(n)}}$$

stavov potrebných na konštrukciu ekvivalentného deterministického konečného automatu. □

Poznámka. Podarilo sa nám ukázať, že vo všeobecnosti stačí na konštrukciu DKA k NKA akceptujúcemu ohraničený jazyk exponenciálne veľa stavov vzhľadom k počtu partícií NKA. Môže sa zdať, že tento nárast je priveľký. Treba si však uvedomiť, že na to, aby bolo n-stavovú partíciu potrebné simulovať $e^{\sqrt{n\log(n)}}$ stavmi, je potrebné dosiahnuť dostatočne veľké n. Počet partícií je v takomto prípade výrazne nižší, ako celkový počet stavov nedeterministického automatu.

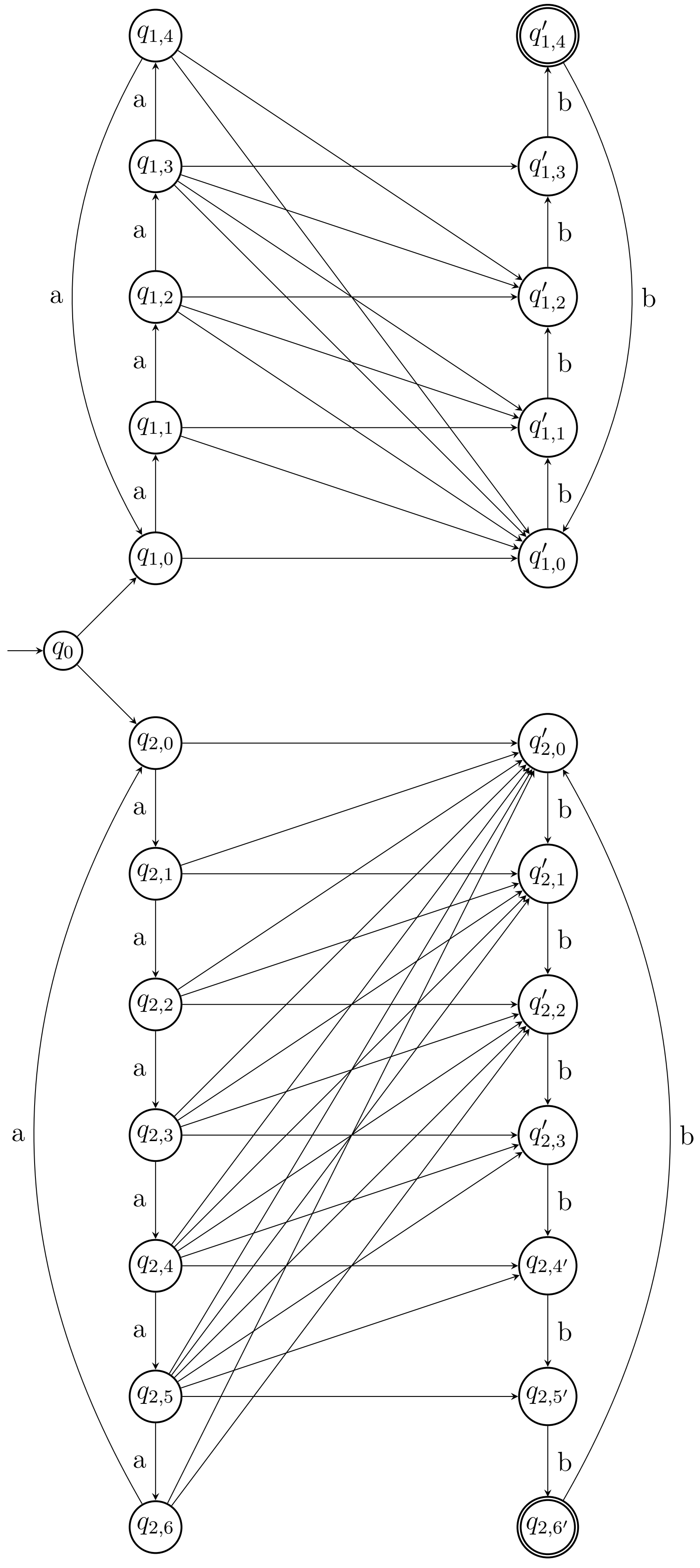
Ukázali sme, že vo všeobecnosti môže byť nárast stavovej zložitosti potrebný na zostrojenie DKA k NKA nad regulárnymi ohraničenými jazykmi rádovo väčší, ako tomu bolo pri jazykoch nad unárnou abecedou. Teraz ukážeme, že ak dostatočne obmedzíme prechody medzi jednotlivými partíciami, bude tento nárast pre obe podtriedy regulárnych jazykov podobný.

Veta 4.4. *Nech je daný n-stavový nedeterministický konečný automat $A = (K, \Sigma, \delta, q_0, F)$ taký, že $L(A) \subseteq a_1^* ... a_k^*$. Nech $P_1, ..., P_k$ sú partície automatu A, pričom P_i je partíciou písmena a_i. Nech pre každé P_i platí, že pre každé $P_j, j > i$ existuje najviac jeden stav $q \in P_i$ taký, že $p \in \delta(q, a)$, kde $p \in P_j$, $a \in \Sigma$. Potom existuje deterministický konečný automat $A' = (K', \Sigma, \delta', q_0, F)$ taký, že $L(A) = L(A')$ a platí $|K'| \leq \frac{k(k+1)e^{\sqrt{n\log(n)}}}{2}$.*

Dôkaz. Na základe vyššie spomenutých úvah môžeme automat A simulovať po partíciach. Počet stavov postačujúci na simulovanie jednej partície závisí od počtu stavov, z ktorých sme sa do partície mohli dostať. Pre jeden takýto stav dostávame najviac $e^{\sqrt{|P_i|\log(|P_i|)}}$ stavov potrebných na simuláciu partície $|P_i|$ deterministicky. Z prepokladu vety vyplýva, že pre partíciu P_i existuje najviac jeden stav pre každú partíciu $P_j, j < i$, z ktorého existuje prechod do partície P_i. Do každej partície ešte môže existovať prechod z q_0. Teda ľubovoľnú partíciu P_i vieme deterministicky simulovať najviac $ie^{\sqrt{|P_i|\log(|P_i|)}}$ stavmi. Pre ľubovoľné $m > 0$ platí $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$. Teda počet stavov potrebný na zostrojenie deterministického A' ekvivalentného k A je najviac

$$\sum_{i=1}^{k} ie^{\sqrt{|P_i|\log(|P_i|)}} \leq \frac{k(k+1)e^{\sqrt{n\log(n)}}}{2}$$

$\square$



Obr. 4.2: Automat A z Vety 4.2 pre $p_1 = 5$, $p_2 = 7$

Záver

V tejto práci sme skúmali stavovú zložitosť deterministických konečných automatov konštruovaných k nedeterministickým konečným automatom pre vybrané podtriedy regulárnych jazykov. Nadviazali sme pritom na už známe výsledky skúmania tejto oblasti. Vo všeobecnom prípade je potrebných 2^n stavov DKA oproti ekvivalentnému n-stavovému NKA. Pre všetky zo skúmaných podtried však postačuje rádovo menší nárast.

Pre jazyky definované podslovom sme dokázali, že stavová zložitosť ekvivalentného DKA a NKA nad týmito jazykmi je rovnaká, pretože oba musia overiť, či vstup obsahuje dané podslovo. Nedeterminizmus neumožňuje toto overovanie "skrátiť" bez toho, aby spôsobil že NKA akceptuje aj vstup, ktorý neobsahuje dané podslovo.

V kapitole o jazykoch nad unárnou abecedou sme prezentovali známy výsledok vzťahu stavovej zložitosti DKA a NKA akceptujúcich tieto jazyky. Ukázali sme, že nárast stavovej zložitosti pre DKA k NKA súvisí s rozvetvenosťou grafu silne súvislých komponentov daného nedeterministickým konečným automatom. Pre graf silne súvislých komponentov G daný n-stavovým nedeterministickým konečným automatom platí, že počet stavov ekvivalentného DKA nepresiahne $n^k + n^2 + 2n$, kde k je počet rôznych nepredĺžiteľných ciest v G. To pre malé k znamená prakticky polynomiálny nárast.

V predošlej kapitole sme skúmali vzťah stavovej zložitosti DKA k NKA akceptujúce ohraničené jazyky. Inšpirujúc sa ideou ohraničeného zásobníkového automatu sme rozdelili stavy NKA na partície, v ktorých automat spracúva zreťazenie niekoľkých rovnakých písmen. Dokázali sme, že týmito partíciami prechádza NKA pri výpočte postupne a do žiadnej z už prejdených partícií sa už nevracia. Napriek tomuto zjavnému súvisu s NKA akceptujúcimi jazyky nad unárnou abecedou je však stavová zložitosť DKA konštruovaného k NKA akceptujúcemu ohraničený jazyk rádovo väčšia, ako tomu bolo pri jazykoch nad unárnou abecedou. To sme demonštrovali kontrapríkladom, ktorému stačila dvojpísmenová abeceda a teda aj dve partície na to, aby bol nárast počtu stavov potrebný na konštrukciu DKA k NKA väčší, ako v najhoršom prípade pre jazyky nad unárnou abecedou.

Ďalej sme ukázali, že pre ohraničené jazyky môže byť stavová zložitosť DKA k ekvivalentnému NKA exponenciálna vzhľadom na počet partícií, ktoré daný NKA obsahuje. Tiež sme ukázali, že ak povolíme z každej partície daného NKA len jeden prechod do každej z následujúcich partícií, nárast stavovej zložitosti ekvivalentného DKA k NKA s k partíciami nepresiahne $\frac{k(k+1)}{2}$ násobok nárastu počtu stavov DKA konštruovanému k NKA akceptujúcemu jazyk nad unárnou abecedou.

Ak by mal autor tejto práce viac času a lepšie matematické základy, venoval by sa v závere kapitoly o ohraničených jazykoch vzťahu počtu partícii k celkovému počtu stavov v NKA. Videli sme totiž, že pre dosiahnutie maximálneho nárastu stavovej zložitosti pre DKA k NKA nad unárnymi jazykmi bol potrebný dostatočne veľký počet stavov NKA. Podobná závislosť potom platí aj pre jednotlivé partície v NKA akceptujúcom ohraničený jazyk. Teda hľadali by sme pomer počtu stavov v partíciach k celkovému počtu partícií, pre ktorý je nárast stavovej zložitosti DKA k NKA akceptujúcemu ohraničený jazyk maximálny.

Literatúra

[1] Alfred Brauer and James E Shockley. On a problem of frobenius. *J. reine angew. Math*, 211(19):215–220, 1962.

[2] Marek Chrobak. Finite automata and unary languages. *Theoretical Computer Science*, 47:149–158, 1986.

[3] M. Chytil. *Automaty a gramatiky.* SNTL, Praha, 1984.

[4] Paul Erdos and Ronald L Graham. On a linear diophantine problem of frobenius. *Acta Arith*, 21(1):399–408, 1972.

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

[6] Edmund Landau. Über die maximalordnung der permutationen gegebenen grades. *Archiv der Math. und Phys*, 3:92–103, 1903.

[7] Branislav Rovan. Bounded push down automata. *Kybernetika*, 5(4):261–265, 1969.