Katedra Informatiky
Fakulta Matematiky, Fyziky a Informatiky
Univerzity Komenského, Bratislava

Nepravidelné asynchrónne bunkové automaty

(Diplomová práca)

Tomáš Záthurecký

Diplomový Vedúci:
prof. RNDr. Branislav Rovan, PhD.

Bratislava, 2006

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

. . . . . .

Abstrakt

Definujeme nový výpočtový model nepravidelných asynchrónnych bunkových automatov a skúmame ich výpočtovú silu. Nepravidelný asynchrónny bunkový automat je graf, ktorý má v každom vrchole konečne stavový automat, ktorý vidí svoj stav a stavy svojich susedov. Všetky vrcholy sú riadené rovnakým „programom“, ktorý nazveme prechodová schéma. V každom kroku výpočtu urobia prechod niektoré vrcholy. Požaduje sa, aby výsledok výpočtu celého automatu (za predpokladu, že ten automat je dostatočne veľký) bol nezávislý od výberu počítajúcich vrcholov (za predpokladu, že ten výber je spravodlivý) a ich grafu prepojení (za predpokladu, že ten graf je súvislý). Dokážeme, že nepravidelné asynchrónne bunkové automaty majú rovnakú výpočtovú silu ako Turingove stroje.

Obsah

Zoznam obrázkov

Kapitola 1

Úvod

Bunkový automat [Cod68] je výpočtové zariadenie, ktoré sa skladá z konečných automatov (buniek), ktoré sú uložené v nejakej štruktúre, najčastejšie mriežke. Každá bunka mení svoj stav v závislosti od svojho pôvodného stavu a stavov susedných buniek.

V mnohých bunkových automatoch jednotlivé bunky menia svoj stav v krokoch, v každom kroku všetky naraz. S takýmto synchrónnym modelom sa síce jednoducho pracuje, ale má svoje obmedzenia. Napríklad nie je vhodný na modelovanie biologických systémov, ktorým chýba globálna synchronizácia. Synchrónne časovanie má svoje nevýhody aj v oblasti návrhu logických obvodov. Medzi výhody asynchrónnych obvodov patria najmä [Hau93], [PF93]: (1) nižšia spotreba energie a nižšie zahrievanie, pretože len aktívne časti obvodu spotrebúvajú energiu, (2) menšia réžia potrebná na distribúciu signálu na globálnu synchronizáciu, (3) vyššia priemerná rýchlosť, lebo rýchlejšie časti obvodu nemusia čakať na tie pomalšie, (4) menšia závislosť správneho fungovania obvodu od fyzikálnych podmienok a implementácie, (5) Vhodnosť pre modulárny návrh, keďže časovanie nie je už také dôležité.

Asynchrónne bunkové automaty boli študované v [Neh03]. Bolo tu ukázané, že globálna synchronizácia sa dá simulovať pomocou lokálnej synchronizácie a teda, že asynchrónne bunkové automaty sú ekvivalentné synchrónnym. V [LAPM05] bola urobená simulácia asynchrónnych obvodov pomocou asynchrónnych bunkových automatov, čím sa tiež ukázala výpočtová univerzálnosť asychrónnych bunkových automatov.

Väčšina modelov bunkových automatov má svoje bunky organizované v nejakej pravidelnej štruktúre, najčastejšie mriežke. Takáto pravidelnosť (podobne ako synchrónnosť) nezodpovedá dobre biologickým systémom, ktoré nie sú pravidelné a aj keď sú, tak sa v nich často vyskytujú chyby. Taktiež pri praktickej realizácii bunkových automatov môže dôjsť k výrobným chybám. Bolo by preto užitočné zaoberať sa aj takými bunkovými automatmi, v ktorých štruktúra prepojenia jednotlivých buniek nemusí byť pravidelná.

V tejto práci budeme skúmať, ako sa zmení výpočtová sila asynchrónnych bunkových automatov, keď odstránime predpoklad pravidelnosti štruktúry prepojení. Tým máme na mysli to, že automat dá rovnakú odpoveď nezávisle od toho, ako presne sú jeho bunky poprepájané. Ukážeme, že aj pri takýchto „zoslabených“ predpokladoch dokážu simulovať ľubovoľný Turingov stroj.

V kapitole 2 formálne definujeme tento výpočtový model. V kapitole 3 ukážeme jeho vý-

počtovú silu. V časti 3.1 definujeme zjednodušený Turingov stroj, ktorý budeme simulovať nepravidelným ACA. V časti 3.2 urobíme konštrukciu, pomocou ktorej budeme simulovať daný zjednodušený TS a v časti 3.3 dokážeme správnosť tejto konštrukcie.

Kapitola 2

Definícia výpočtového modelu

Výpočtové zariadenie, ktorým sa budeme zaoberať sa skladá z dostatočne veľkého počtu[1] buniek, ktoré sú navzájom poprepájané.[2] Každá bunka môže byť v jednom z konečne veľa stavov. Bunka vo svojom kroku výpočtu môže zmeniť stav podľa svojho doterajšieho stavu a stavov svojich susedov. Spôsob, ktorým to robí budeme volať *prechodová schéma*. V našom zariadení budú mať všetky bunky rovnakú prechodovú schému. Zariadenie bude asynchrónne. To znamená, že v kroku výpočtu celého zariadenia urobia svoj krok výpočtu len niektoré bunky. Spôsob, ktorým budú vyberané tie bunky, ktoré počítajú budeme volať *výpočtový plán*. Zariadenie bude nepravidelné. To znamená, že sa pripúšťajú rôzne spôsoby prepojenia jednotlivých buniek. Kvôli tejto vlastnosti nastávajú určité problémy s definovaním vstupu a výstupu. Riešením týchto problémov sa nebudeme hlbšie zaoberať. Budeme ich riešiť jednoducho (a možno trochu neštandartne) tak, že jedna bunka bude špeciálna – bude spojená s vonkajším svetom.[3] Táto bunka bude mať navyše aj vstupnú pásku, na ktorej bude mať (jednosmernú) čítaciu hlavu. To, ako sú bunky navzájom poprepájané a ktorá je špeciálna (budeme ju nazývať čítajúca) bude určené *schémou prepojení*.

Vo všeobecnosti výsledok práce takéhoto zariadenia bude závisieť od prechodovej schémy, vstupu, schémy prepojení a výpočtového plánu. Budeme sa však zaoberať len takými prechodovými schémami, pre ktoré výsledok nezávisí od schémy prepojení a ani od výpočtového plánu (pri splnení určitých podmienok). O takýchto prechodových schémach budeme hovoriť, že sa chovajú dobre.

Keby sme uvažovali úplne ľubovoľné spôsoby prepojenia jednotlivých buniek, mali by sme problém s konečnosťou prechodovej schémy (lebo tie prepojenia môžu mať ľubovoľne veľký stupeň). Preto sa obmedzíme len na prepojenia určitého typu. Dobré chovanie prechodovej schémy teda definujeme závisle na tom, aké prepojenia uvažujeme. Podobne aj výpočtovú silu prechodových schém budeme študovať v závislosti od typov prepojení, ktoré berieme do úvahy.

[1] Toľko, koľko bude treba aby to fungovalo.

[2] Dá sa to dobre predstaviť ako piesok v kýbli. Zrnká sú bunky a prepojené sú tie, čo sa dotýkajú.

[3] Toto sa dá predstaviť tak, že do toho kýbľa je strčený drôt, cez ktorý komunikuje zrnko, ktoré sa dotýka jeho konca.

2.1 Formálna definícia

Označenie 2.1.1 Ak G je graf, tak symbolom $V(G)$ budem značiť množinu jeho vrcholov.

Označenie 2.1.2 Ak G je graf v jeho vrchol tak symbolom $N_G(v)$ budem značiť množinu všetkých susedov vrchola v.

Definícia 2.1.3 *(prechodová schéma) Prechodová schéma je 5-tica $(K, \Sigma, \delta, q_0, F)$ taká, že K je konečná množina, Σ je abeceda, $F \subseteq K$, $q_0 \in K$, $\delta : (K \times \mathcal{M}(K) \times (\Sigma \cup \{\$, \varepsilon, \mathbf{T}\})) \to (K \cup \emptyset)$, kde $\mathcal{M}(K)$ je množina všetkých konečných multimnožín, ktorých prvky sú z K.*

Pritom požadujeme, aby množina $\{(q, Q, a) | \delta(q, Q, a) \neq \emptyset\}$ bola konečná a aby pre každé $q \in K$, $Q \in \mathcal{M}(K)$, $a \in \Sigma \cup \{\$\}$ platilo $\delta(q, Q, a) = \emptyset \vee \delta(q, Q, \varepsilon) = \emptyset$.

Poznámka 2.1.4 Prvky množiny K budeme nazývať stavy, stav q_0 budeme volať počiatočný a stavy v množine F budeme volať akceptačné. Abecedu Σ budeme volať vstupná abeceda. Funkciu δ budeme volať prechodová funkcia. Ak je tretím argumentom funkcie symbol $\mathbf{T}$, znamená to, že bunka nie je čítajúca.

Dotatočná podmienka, ktorá sa v definícii požaduje zaručuje, že prechodová schéma je konečná a deterministická.

Definícia 2.1.5 *(schéma prepojení) Ak $G = (V, E)$ je súvislý graf tak dvojicu (G, r) taká, že $r \in V$ nazveme schéma prepojení.*

Definícia 2.1.6 *(ACA) Asynchrónny bunkový automat (ACA) je trojica (G, r, H), kde (G, r) je schéma prepojení a H je prechodová schéma. Povieme, že (G, r, H) je ACA s prechodovou schémou H.*

Ďalej definujeme, ako ACA pracujú. Na to definujeme konfiguráciu, krok výpočtu, výpočet a ďalšie pojmy.

Konfigurácia je štruktúra, ktorá určuje stavy jednotlivých buniek a ešte nedočítanú časť slova na vstupnej páske.

Definícia 2.1.7 *(konfigurácia) Nech $A = (G, r, (K, \Sigma, \delta, q_0, F))$ je ACA. Konfigurácia A je dvojica (f, w), kde $f : V(G) \to K$ a $w \in (\{\varepsilon\} \cup (\Sigma^*\$))$.*

Poznámka 2.1.8 Pre $v \in V(G)$ hodnota $f(v)$ je stav vrchola (bunky) v. Slovo w je ešte nedočítaná časť vstupu.

Definícia 2.1.9 *(krok výpočtu) Nech $A = (G, r, (K, \Sigma, \delta, q_0, F))$ je ACA. Nech $M \subseteq V(G)$. Nech (f, au) a (g, u) sú konfigurácie A. Povieme že A urobil krok výpočtu z (f, au) do (g, u) pri výbere aktívnych buniek M (budeme písať $(f, au) \vdash_A^M (g, u)$) ak*

- *$g(x) = f(x)$ pre každé $x \in V - M$,*

- $g(r) = \delta(f(r), f(N_G(v)), a)$ *ak* $r \in M$,
- $g(x) = \delta(f(x), f(N_G(x)), \mathbf{T})$ *pre každé* $x \in M - \{r\}$

Poznámka 2.1.10 Krok výpočtu je teda relácia dvoch konfigurácií pri danom výbere množiny aktívnych vrcholov (buniek) a hovorí, ako sa zmení konfigurácia, keď svoj prechod urobia vybraté vrcholy.

Ďalej definujeme iteráciu kroku výpočtu. Podobne ako krok výpočtu to bude relácia dvoch konfigurácií pri zvolenej postupnosti množín buniek.

Definícia 2.1.11 *(výpočet) Nech* $A = (G, r, (K, \Sigma, \delta, q_0, F))$ *je ACA. Nech* $\mathcal{M} = \{M_i\}_{i=1}^{\infty}$ *je postupnosť podmnožín* $V(G)$. *Nech* c *a* d *sú konfigurácie* A. *Povieme že* A *urobil výpočet* z c *do* d *pri výbere* $\mathcal{M}$ *(píšeme* $c \vdash_A^{\mathcal{M}} d$*) ak existuje prirodzené číslo* n *a konfigurácie* A $c_0, c_1, \ldots, c_n$ *také, že* $c = c_0 \vdash^{M_1} c_1 \vdash^{M_2} c_2 \vdash^{M_3} \ldots \vdash^{M_n} c_n = d$.

Výpočtový plán bude postupnosť množín buniek (vrcholov), ktorá je navyše spravodlivá. Teda nestane sa to, že by nejaká bunka nepočítala. To zodpovedá intuitívnej predstave o práci nášho zariadenia, bunky počítajú rôzne rýchlo[4], ale počítajú.

Definícia 2.1.12 *(výpočtový plán) Nech* $G = (V, E)$ *je graf. Nech* $\mathcal{M} = \{M_i\}_{i=1}^{\infty}$ *je postupnosť podmnožín* $V(G)$. *Postupnosť* $\mathcal{M}$ *nazveme výpočtový plán nad* $V(G)$ *ak pre každé* $v \in V$ *a prirodzené číslo* n *existuje prirodzené číslo* $k > n$, *že* $v \in M_k$.

Teraz definujeme, čo znamená, že ACA akceptuje slovo pri určitom výpočtovom pláne. Automat akceptuje slovo vtedy, keď sa mu podarí z počiatočnej konfigurácie (takej, že všetky vrcholy sú v stave q_0) dosiahnuť konfiguráciu, v ktorej je vstupné slovo dočítané a čítajúci vrchol je v akceptačnom stave.

Definícia 2.1.13 *(akceptovanie ACA) Nech* $A = (G, r, (K, \Sigma, \delta, q_0, F))$ *je ACA. Nech* $\mathcal{M}$ *je výpočtový plán nad* $V(G)$. *Nech* $w \in \Sigma^*$. *Povieme, že* A *akceptuje* w *pri pláne* $\mathcal{M}$ *ak existuje konfigurácia* (f, ε) *taká, že* $(g_0, w\$) \vdash_A^{\mathcal{M}} (f, \varepsilon)$ *a* $f(r) \in F$, *pričom* $g_0(x) = q_0$ *pre každé* $x \in V(G)$.

Poznámka 2.1.14 Finálna konfigurácia (f, ε) je vďaka determinizmu automatom A, slovom w a plánom $\mathcal{M}$ jednoznačne určená.

Definovali sme teda, ako pracujú asynchrónne bunkové automaty. Pri takejto definícii je ale výsledok výpočtu závislý od schémy prepojení toho automatu a výpočtového plánu, podľa ktorého sa počíta. Povedali sme si však, že sa budeme zaoberať len prechodovými schémami, pre ktoré je výsledok výpočtu od týchto parametrov nezávislý (pri dodržaní istých podmienok). V nasledujúcich definíciach povieme, ktoré prechodové schémy majú túto vlastnosť.

Najprv definujeme akceptovanie slova predchodovou schémou. Povieme, že prechodová schéma akceptuje dané slovo, ak ho akceptuje skoro každý[5] ACA s prepojeniami určitého typu s tou prechodovou schémou pri každom výpočtovom pláne.

[4] Rýchlosť súvisí s tým, ako často je bunka vyberaná.

[5] Myslíme tým každý dostatočne veľký

Definícia 2.1.15 *(akceptovanie slova schémou) Nech $H = (K, \Sigma, \delta, q_0, F)$ je prechodová schéma. Nech $\mathcal{G}$ je nejaká množina schém prepojení. Nech $w \in \Sigma^*$.*

Povieme, že H akceptuje w pri prepojeniach z $\mathcal{G}$ ak existuje prirodzené číslo n také, že pre každé ACA $A = (G, r, H)$ s $(G, r) \in \mathcal{G}$ a $|V(G)| \geq n$ a každý výpočtový plán $\mathcal{M}$ nad $V(G)$: A akceptuje w pri výbere $\mathcal{M}$.

Podobne definujeme aj odmietnutie slova prechodovou schémou. Schéma slovo odmietne, ak ho odmietne každý dostatočne veľký ACA s prepojeniami určitého typu nezávisle od svojej štruktúry a výpočtového plánu.

Definícia 2.1.16 *(odmietnutie slova schémou) Nech $H = (K, \Sigma, \delta, q_0, F)$ je prechodová schéma. Nech $\mathcal{G}$ je nejaká množina schém prepojení. Nech $w \in \Sigma^*$.*

Povieme, že H odmietne w pri prepojeniach z $\mathcal{G}$ ak existuje prirodzené n také, že pre každé ACA $A = (G, r, H)$ s $(G, r) \in \mathcal{G}$ a $|V(G)| \geq n$ a každý výpočtový plán $\mathcal{M}$ nad $V(G)$: A neakceptuje w pri výbere $\mathcal{M}$.

Teda keď máme prechodovú schému a určitý typ prepojení a vezmeme si slovo z Σ^*, tak môžu nastať tri prípady. Buď ho tá schéma akceptuje, alebo odmietne, alebo výsledok výpočtu závisí od štruktúry prepojení v automate a(lebo) od výpočtového plánu. Ak nenastáva ten tretí prípad, povieme, že tá prechodová schéma sa chová dobre.

Definícia 2.1.17 *(dobré chovanie schémy) Nech H je prechodová schéma, $\mathcal{G}$ je množina schém prepojení. Povieme, že H sa chová dobre pri prepojeniach z $\mathcal{G}$ ak každé slovo buď akceptuje alebo odmietne (pri prepojeniach z $\mathcal{G}$).*

Definícia 2.1.18 *(akceptovaný jazyk) Nech $\mathcal{G}$ je množina schém prepojení. Nech H je prechodová schéma, ktorá sa chová dobre pri prepojeniach z $\mathcal{G}$. Jazyk akceptovaný H pri prepojeniach z $\mathcal{G}$ (značíme $L(H, \mathcal{G})$) je množina všetkých slov, ktoré akceptuje pri prepojeniach z $\mathcal{G}$.*

Kapitola 3

Výpočtová sila

V tejto časti dokážeme, že na prepojeniach ohraničeného stupňa[1] sú ACA ekvivalentné s Turingovymi strojmi. To znamená, že ku každému Turingovmu stroju existuje prechodová schéma, ktorá sa na prepojeniach ohraničeného stupňa chová dobre a akceptuje ten istý jazyk ako ten Turingov stroj.

Označenie 3.0.1 (stupeň grafu) Ak G je graf, tak symbolom $\Delta(G)$ budem značiť stupeň grafu G.

Definícia 3.0.2 *(prepojenia ohraničeného stupňa) Nech D je prirodzené číslo, Nech $\mathcal{G}$ je množina schém prepojení taká, že $(G, r) \in \mathcal{G}$ práve vtedy, keď $\Delta(G) \leq D$. Množine $\mathcal{G}$ budeme hovoriť trieda schém prepojení stupňa najviac D. Triedu schém prepojení stupňa najviac D budeme značiť $\mathcal{G}_D$.*

3.1 Zjednodušený TS – definícia

Aby nebola konštrukcia prechodovej schémy k danému Turingovmu stroju príliš zložitá, definujeme normálny tvar Turingovych strojov.

Turingov stroj v tomto tvare bude mať jednu pracovnú pásku nekonečnú len smerom doľava, vstupnú pásku, na ktorej sa hlava hýbe len smerom doprava (alebo stojí) a navyše zo vstupnej pásky môže čítať len keď hlava na pracovnej páske je vľavo od najľavejšieho zapísaného políčka.

Nasleduje formálna definícia:

Definícia 3.1.1 *(zjednodušený TS) Zjednodušený TS je 6-tica $M = (K, \Sigma, \Gamma, \delta, q_0, F)$ taká, že K, Σ, Γ su konečné, $\Sigma \subset \Gamma$, $q_0 \in K$, $F \subset K$, $\delta : K \times (\Gamma \cup ¢\Sigma \cup \{¢, ¢\$, \$\}) \to (K \times \Gamma \times \{-1, 0, 1\})$. Pričom δ môže byť čiastočná funkcia a navyše požadujeme, aby pre každé $q \in K$, $a \in \Sigma \cup \{\$\}$ bola aspoň jedna z hodnôt $\delta(q, ¢a)$, $\delta(q, ¢)$ nedefinovaná (teda aby to bolo deterministické).*

[1] To znamená, že počet susedov každej bunky je ohraničený nejakou konštantou.

Definícia 3.1.2 *(zjednodušený TS – konfigurácia) Nech* $M = (K, \Sigma, \Gamma, \delta, q_0, F)$ *je zjednodušený TS. Konfiguráciou* M *nazveme štvoricu* $(q, w, ¢u\$, i)$ *taký, že* $q \in K$, $w \in \Sigma^*$, $u \in \Gamma^*$, $0 \leq i \leq |u| + 1$.

Definícia 3.1.3 *(zjednodušený TS – krok výpočtu) Nech* $M = (K, \Sigma, \Gamma, \delta, q_0, F)$ *je zjednodušený TS. Krok výpočtu* M *je relácia* $\vdash_M$ *na konfiguráciach* M *taká, že*

$$(p, w, ¢u_1u_2 \ldots u_{i-1}u_iu_{i+1} \ldots u_n\$, i) \vdash_M (q, w, ¢u_1u_2 \ldots u_{i-1}yu_{i+1} \ldots u_n\$, i{+}d) \Leftrightarrow \delta_M(p, u_i) = (q, y, d)$$

$$(p, aw, ¢u_1u_2 \ldots u_n\$, 0) \vdash_M (q, w, ¢yu_1u_2 \ldots u_n\$, d) \Leftrightarrow \delta_M(p, ¢a) = (q, y, d) \text{ pre } a \in \Sigma \cup \{\varepsilon\}$$

$$(p, \varepsilon, ¢u_1u_2 \ldots u_n\$, 0) \vdash_M (q, \varepsilon, ¢yu_1u_2 \ldots u_n\$, d) \Leftrightarrow \delta_M(p, ¢\$) = (q, y, d)$$

$$(p, w, ¢u_1u_2 \ldots u_n\$, n+1) \vdash_M (q, w, ¢u_1u_2 \ldots u_n\$, n+1+d) \Leftrightarrow \delta_M(p, \$) = (q, y, d)$$

Definícia 3.1.4 *(zjednodušený TS – akceptovaný jazyk) Nech* $M = (K, \Sigma, \Gamma, \delta, q_0, F)$ *je zjednodušený TS. Jazyk akceptovaný strojom* M *je jazyk* $L(M) = \{w \in \Sigma^* \mid \exists q \in F, w' \in \Sigma^*, u \in \Gamma^*, i : (q_0, w, ¢\$, 0) \vdash_M (q, w', ¢u\$, i)\}$.

Zrejme zjednodušený TS je normálny tvar Turingovho stroja, t. j., k ľubovoľnému Turingovmu stroju existuje zjednodušený Turingov stroj akceptujúci ten istý jazyk.

3.2 Konštrukcia prechodovej schémy

Vezmime si ľubovoľný zjednodušený TS M a prirodzené číslo D. K stroju M a číslu D ideme konštruovať prechodovú schému H, ktorá sa na prepojeniach z $\mathcal{G}_D$ bude chovať dobre a bude $L(H, \mathcal{G}_D) = L(M)$.

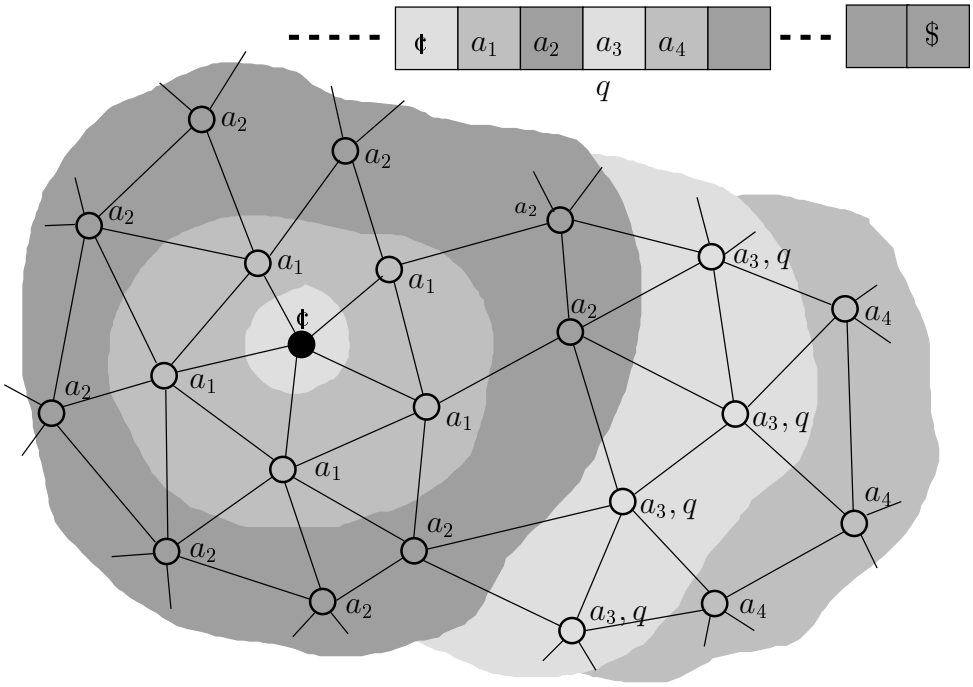
Asynschrónny bunkový automat s prechodovou schémou H bude fungovať asi takto:

Jednotlivé vrcholy sa rozdelia do vrstiev podľa vzdialenosti od čítajúceho vrchola. Každá vrstva bude simulovať jedno políčko pracovnej pásky stroja M, pričom čítajúci vrchol (ktorý tvorí nultú vrstvu) bude simulovať políčko, na ktorom je napísaný symbol ¢. Každý vrchol si vo svojom stave bude (okrem iného) pamätať svoju vzdialenosť od čítajúceho vrchola modulo 3 (aby vedel rozoznať svojich susedov na predchádzajúcej a nasledujúcej vrstve), písmenko zapísané na políčku pracovnej pásky simulovaného stroja M, ktoré simuluje a či sa na tom políčku nachádza hlava stroja M a ak áno, tak aj jeho stav. Toto rozdelenie na vrstvy je znázornené na obrázku 3.1.

3.2.1 Formálna konštrukcia

Zostrojíme $H = (K_H, \Sigma_H, \delta_H, q_H, F_H)$, pričom jednotlivé komponenty definujeme takto:

Množina stavov K_H bude obsahovať štyri špeciálne stavy (β,α,ω,φ) a stavy tvaru $[l, x, q, e, y]$. Stav β je počiatočný, v tomto stave budú vrcholy, ktoré sa zatiaľ nezúčastnili výpočtu. Pomocný stav α sa používa pri inicializácii. V stave ω budú „mŕtve“ vrcholy, ktoré sa prestali zúčastňovať výpočtu. Stav φ je akceptačný. Vrchol v stave $[l, x, q, e, y]$ je



Obr. 3.1: Rozdelenie buniek ACA na vrstvy

na vrstve l (modulo 3), pamätá si, že na políčku pracovnej pásky, ktoré tento vrchol simuluje je zapísané x a simulovaný stroj M je v stave q (ak je jeho hlava na tomto políčku, alebo niekde blízko). Komponenta e sa používa na riadenie simulácie pohybu hlavy a komponenta y sa používa pri simulovaní rozširovania pásky (posúvanie obsahu na nasledujúce vrstvy; bude objasnené neskôr).

Teda $K_H = \{\beta, \alpha, \omega, \varphi\} \cup \{[l, x, q, e, y] | l \in Z_3, x \in \Gamma_M \cup \{¢, \$\}, y \in \Gamma_M \cup \{\$, \beta, \overline{\gamma}, \gamma\}, q \in K_M \cup \{\beta\}, e \in \{\Leftarrow, \leftarrow, \Rightarrow, \Rightarrow, \rightarrow, \beta\}\}$. Ďalej bude $\Sigma_H = \Sigma_M$, $q_H = \beta$, $F_H = \{\varphi\}$.

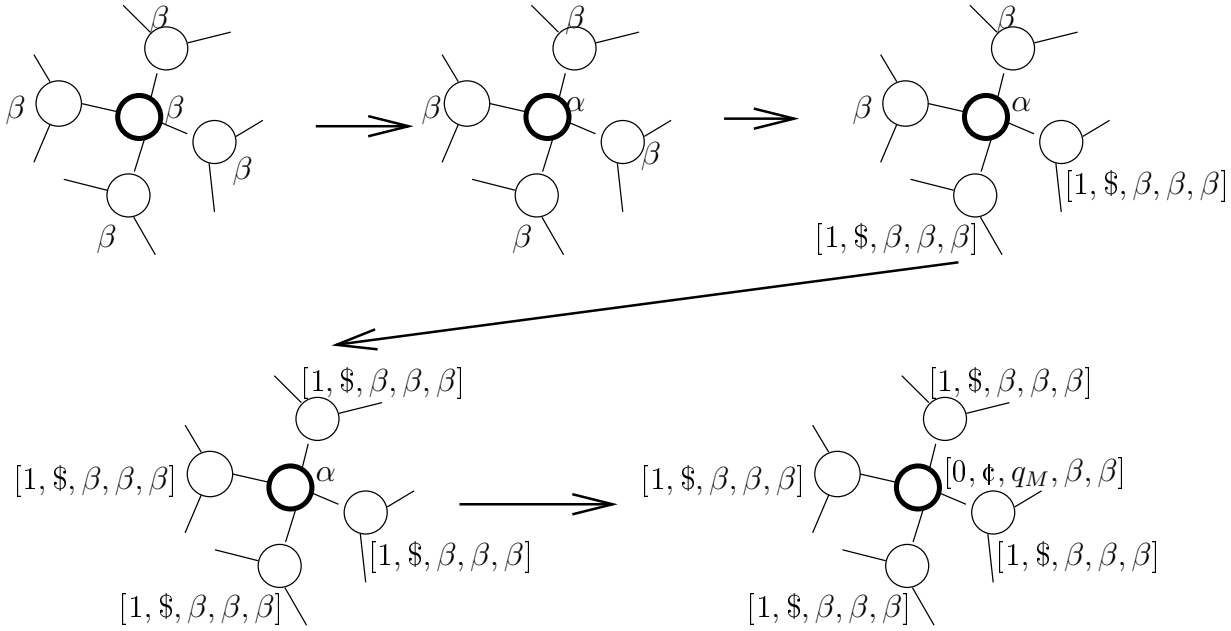
Funkcia δ_H bude definovaná nasledujúcimi rovnosťami, ktoré platia pre každú multimnožinu Q s prvkami z K_H takú, že $|Q| \leq D$. Navyše rovnosti (3.4) až (3.27) platia len pre tie Q, pre ktoré $\omega, \varphi \notin Q$.

$$\delta_H(\beta, Q, \varepsilon) = \alpha \tag{3.1}$$

$$\delta_H(\beta, Q, \mathbf{T}) = [1, \$, \beta, \beta, \beta] \text{ ak } \alpha \in Q \tag{3.2}$$

$$\delta_H(\alpha, Q, \varepsilon) = [0, ¢, q_M, \beta, \beta] \text{ ak } \forall q \in Q : q = [1, \$, \beta, \beta, \beta] \tag{3.3}$$

Rovnosti (3.1) až (3.3) zodpovedajú inicializácii zariadenia. Podľa týchto rovností ACA s prechodovou schémou H na začiatku výpočtu prejde (viď Obr. 3.2) do konfigurácie, v ktorej čítajúci vrchol je v stave $[0, ¢, q_M, \beta, \beta]$, jeho susedia sú v stave $[1, \$, \beta, \beta, \beta]$ a ostatné vrcholy sú v stave β. Táto konfigurácia teda zodpovedá počiatočnej konfigurácii simulovaného stroja M.



Obr. 3.2: Začiatok výpočtu ACA s prechodovou schémou H.

Vrchol, ktorý simuluje políčko, na ktorom je hlava simulovaného stroja M (v tretej komponente jeho stavu je zapísaný nejaký stav stroja M) má dosť informácií na to, aby začal so simuláciou kroku výpočtu stroja M, preto bude:

$$\delta_H([l,x,q,\beta,\beta],Q,\mathbf{T}) = [l,x',q',\beta,\beta] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,x) = (q',x',0) \tag{3.4}$$

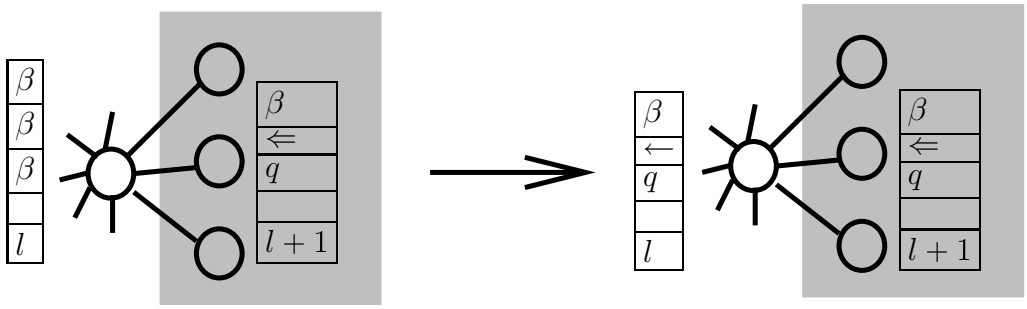
$$\delta_H([0,\text{¢},q,\beta,\beta],Q,a) = [0,\text{¢},q',\Rightarrow,x'] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,\text{¢}a) = (q',x',0) \text{ a } a \neq \mathbf{T} \tag{3.5}$$

$$\delta_H([l,x,q,\beta,\beta],Q,\mathbf{T}) = [l,x',q',\Rightarrow,\beta] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,x) = (q',x',1) \tag{3.6}$$

$$\delta_H([0,\text{¢},q,\beta,\beta],Q,a) = [0,\text{¢},q',\Rrightarrow,x'] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,\text{¢}a) = (q',x',1) \text{ a } a \neq \mathbf{T} \tag{3.7}$$

$$\delta_H([l,x,q,\beta,\beta],Q,\mathbf{T}) = [l,x',q',\Leftarrow,\beta] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,x) = (q',x',-1) \tag{3.8}$$

$$\delta_H([0,\text{¢},q,\beta,\beta],Q,a) = [0,\text{¢},q',\beta,x'] \text{ ak platí } q \neq \beta \text{ a } \delta_M(q,\text{¢}a) = (q',x',-1) \text{ a } a \neq \mathbf{T} \tag{3.9}$$



Obr. 3.3: Simulácia posunu hlavy doľava I

V prípade, že stroj M v práve simulovanom kroku výpočtu nepohol hlavou, je simulácia tohto kroku výpočtu dokončená.[2] Inak treba odsimulovať aj posun jeho hlavy (teda informáciu o stave simulovaného stroja M treba posunúť na predchádzajúcu resp. nasledujúcu vrstvu).

Simulácia posunu hlavy sa začína tým, že vrchol, ktorý simuluje políčko, z ktorého sa hlava posúva zapíše do štvrtej komponenty svojho stavu symbol $\Leftarrow$ (pri posune doľava, rovnosť 3.8) resp. $\Rightarrow$ (pri posune doprava, rovnosť 3.6). Vrchol, ktorý zistí, že všetci jeho susedia na nasledujúcej vrstve chcú simulovať posun hlavy stroja M doľava (majú v štvrtej komponente svojho stavy napísané $\Leftarrow$) skopíruje do tretej komponenty svojho stavu stav simulovaného stroja M a do štvrtej komponenty svojho stavu napíše symbol $\leftarrow$, čím „potvrdí príjem“ (viď Obr 3.3):

$$\begin{gathered}\delta_H([l,x,\beta,\beta,\beta],Q,\mathbf{T}) = [l,x,q,\leftarrow,\beta] \text{ ak platí} \\ \forall[l+1,x',q',e',y'] \in Q : q' = q \wedge e' = \Leftarrow \wedge y' = \beta\end{gathered} \tag{3.10}$$

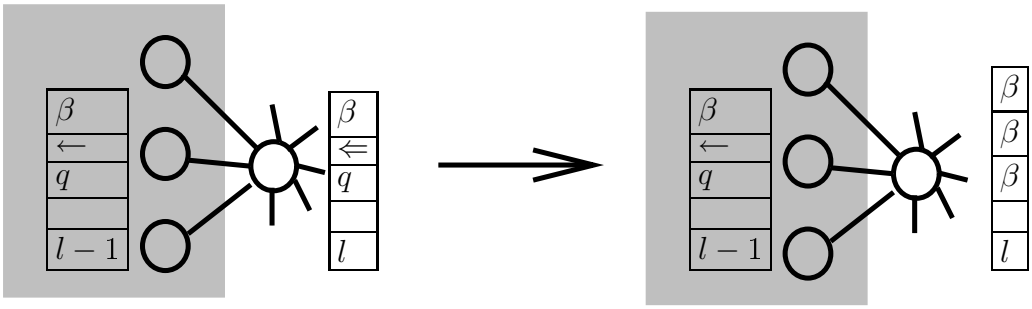
$$\begin{gathered}\delta_H([0,\not\Leftarrow,\beta,\beta,\beta],Q,\varepsilon) = [0,\not\Leftarrow,q,\leftarrow,\beta] \text{ ak platí} \\ \forall[1,x',q',e',y'] \in Q : q' = q \wedge e' = \Leftarrow \wedge y' = \beta\end{gathered} \tag{3.11}$$

Ďalej keď vrchol, ktorý chcel posunúť hlavu doľava (má v tretej komponente svojho stavu symbol $\Leftarrow$) zistí, že všetci jeho susedia už „potvrdili príjem“ (majú v tretej komponente svojho stavu symbol $\leftarrow$), tak už považuje simuláciu posúvania hlavy za dokončenú a prestane si pamätať stav stroja M (viď Obr. 3.4):

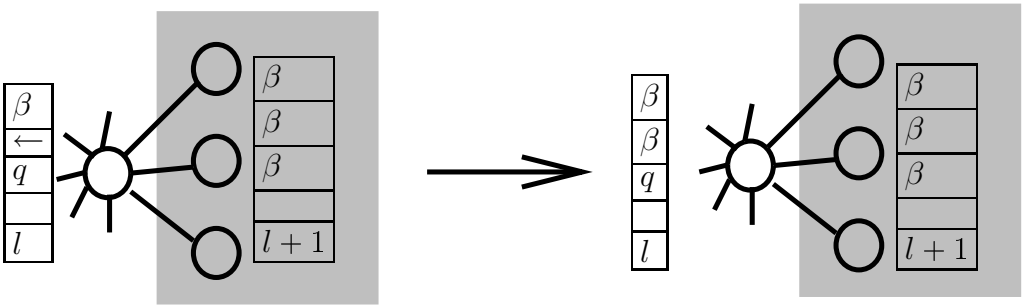
$$\begin{gathered}\delta_H([l,x,q,\Leftarrow,\beta],Q,\mathbf{T}) = [l,x,\beta,\beta,\beta] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : q' = q \wedge e' = \leftarrow \wedge y' = \beta\end{gathered} \tag{3.12}$$

No a keď vrchol, na ktorý „hlava prišla“ (tretia komponenta jeho stavu je $\leftarrow$) zistí, že všetci jeho susedia na nasledujúcej vrstve už dokončili simuláciu posunu hlavy (tretia komponenta ich stavov je β), tak aj on si do tretej komponenty svojho stavu zapíše symbol β, čím definitívne dokončí simuláciu posunu hlavy (viď Obr. 3.5):

[2] Všimnime si, že v tomto prípade sa jednotlivé vrcholy navzájom vôbec nesynchronizujú. Môže sa stať, že rôzne časti ACA s konštruovanou prechodovou schémou budú simulovať (v poradí) rôzne kroky výpočtu stroja M.



Obr. 3.4: Simulácia posunu hlavy doľava II



Obr. 3.5: Simulácia posunu hlavy doľava III

$$\begin{gathered}\delta_H([l,x,q,\leftarrow,\beta],Q,\mathbf{T}) = [l,x,q,\beta,\beta] \text{ ak platí} \\ \forall[l+1,x',q',e',y'] \in Q : q' = \beta \wedge e' = \beta \wedge y' = \beta\end{gathered} \tag{3.13}$$

$$\begin{gathered}\delta_H([0,\text{¢},q,\leftarrow,\beta],Q,\varepsilon) = [0,\text{¢},q,\beta,\beta] \text{ ak platí} \\ \forall[1,x',q',e',y'] \in Q : q' = \beta \wedge e' = \beta \wedge y' = \beta\end{gathered} \tag{3.14}$$

Simulácia posunu hlavy doprava funguje podobne. Ako sa však neskôr ukáže, budeme potrebovať simulovať aj posun hlavy o dve políčka doprava. Toto sa robí jednoducho ako dvojkrokový posun doprava. Prvý krok nám zabezpečia nasledujúce rovnosti:

$$\begin{gathered}\delta_H([l,x,\beta,\beta,\beta],Q,\mathbf{T}) = [l,x,q,\Rightarrow,\beta] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : q' = q \wedge e' = \Rightarrow \wedge y' = \beta\end{gathered} \tag{3.15}$$

$$\begin{gathered}\delta_H([0,\text{¢},q,\Rightarrow,\beta],Q,\varepsilon) = [0,\text{¢},\beta,\beta,\beta] \text{ ak platí} \\ \forall[1,x',q',e',y'] \in Q : q' = q \wedge e' = \Rightarrow \wedge y' = \beta\end{gathered} \tag{3.16}$$

A druhý krok, ako aj samotný posun hlavy o jedno políčko doprava nasledujúce:

$$\begin{gathered}\delta_H([l,x,\beta,\beta,\beta],Q,\mathbf{T}) = [l,x,q,\rightarrow,\beta] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : q' = q \wedge e' = \Rightarrow \wedge y' = \beta\end{gathered} \tag{3.17}$$

$$\begin{gathered}\delta_H([l,x,q,\Rightarrow,\beta],Q,\mathbf{T}) = [l,x,\beta,\beta,\beta] \text{ ak platí} \\ \forall[l+1,x',q',e',y'] \in Q : q' = q \wedge e' = \rightarrow \wedge\, y' = \beta \text{ a} \\ \forall[l-1,x',q',e',y'] \in Q : q' = \beta \wedge e' = \beta \wedge y' = \beta\end{gathered} \tag{3.18}$$

$$\begin{gathered}\delta_H([0,\text{¢},q,\Rightarrow,\beta],Q,\varepsilon) = [0,\text{¢},\beta,\beta,\beta] \text{ ak platí} \\ \forall[1,x',q',e',y'] \in Q : q' = q \wedge e' = \rightarrow \wedge y' = \beta\end{gathered} \tag{3.19}$$

$$\begin{gathered}\delta_H([l,x,q,\rightarrow,\beta],Q,\mathbf{T}) = [l,x,q,\beta,\beta] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : q' = \beta \wedge e' = \beta \wedge y' = \beta\end{gathered} \tag{3.20}$$

V prípade, že hlava simulovaného stroja M bola na ľavom konci pracovnej pásky (až na políčku, kde je napísaný symbol ¢), je situácia zložitejšia. V tomto prípade simulovaný stroj M prepíše symbol ¢, čím sa rozšíri jeho pracovná páska. Toto rozšírenie pracovnej pásky bude ACA s konštruovanou prechodovou schémou simulovať tak, že písmenká zapamätané na vrstvách simulujúcich jednotlivé políčka pracovnej pásky stroja M sa posunú na nasledujúce vrstvy. Toto posúvanie začne čítajúci vrchol tým, že do piatej komponenty svojho stavu zapíše písmenko, ktoré treba posunúť (teda to, ktorým by simulovaný stroj M prepísal symbol ¢) na nasledujúcu (prvú) vrstvu (viď rovnosti 3.5, 3.9 a 3.7).

Ďalej, vrchol, ktorý zistí, že všetci jeho susedia na predchádzajúcej vrstve chcú posunúť nejaké písmenko (majú ho zapísané v piatej komponente svojho stavu; neskôr sa ukáže, že všetci budú chcieť posunúť to isté) prijme toto písmenko (zapíše ho do druhej komponenty svojho stavu) a bude chcieť posunúť to, ktoré si pamätal doteraz (teda písmenko, ktoré mal doteraz v druhej komponente svojho stavu prepíše do piatej komponenty):

$$\begin{gathered}\delta_H([l,x,q,e,\beta],Q,\mathbf{T}) = [l,y,q,e,x] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : y' = y \text{ a } y \notin \{\beta,\gamma,\overline{\gamma}\}\end{gathered} \tag{3.21}$$

Keď vrchol u, ktorý sa doteraz nezúčastňoval výpočtu (je v stave β) zistí, že nejaký jeho sused v chce posunúť písmenko na nasledujúcu vrstvu a aj všetci jeho susedia na tej istej vrstve ako v chcú posunúť písmenko na nasledujúcu vrstvu (majú ho napísané v piatej komponente svojho stavu), stane sa tento vrchol u časťou novej[3] vrstvy, ktorá si bude pamätať písmenko, ktoré posielal vrchol v. Vrchol u navyše dá vedieť, že tvorí poslednú vrstvu tým, že do piatej komponenty svojho stavu zapíše symbol $\overline{\gamma}$:

$$\begin{gathered}\delta_H(\beta,Q,\mathbf{T}) = [l+1,y,\beta,\beta,\overline{\gamma}] \text{ ak platí} \\ \exists x',q',e' : [l,x',q',e',y] \in Q \text{ a } \forall[l,x',q',e',y'] \in Q : y' = y \text{ a } y \notin \{\beta,\gamma,\overline{\gamma}\}\end{gathered} \tag{3.22}$$

Keď vrchol, ktorý chce posunúť písmenko na nasledujúcu vrstvu (má ho zapísané v piatej komponente svojho stavu) zistí, že všetci jeho susedia na nasledujúcej vrstve už toto písmenko prijali (v piatej komponente svojho stavu majú písmenko, alebo symbol $\overline{\gamma}$), považuje posúvanie vrstiev za „skoro dokončené“. Ak aj všetci jeho susedia na predchádzajúcej vrstve „skoro dokončili“ posúvanie vrstiev (v piatej komponente ich stavu je symbol γ), tak aj tento vrchol zapíše do piatej komponenty svojho stavu symbol γ:

[3]Keď simulujeme rozširovanie pracovnej pásky zvýši sa nám počet vrstiev.

$$\begin{gathered}\delta_H([l,x,q,e,y],Q,\mathbf{T}) = [l,x,q,e,\gamma] \text{ ak platí} \\ y \notin \{\beta,\gamma,\overline{\gamma}\} \text{ a } \beta \notin Q \text{ a} \\ \forall[l-1,x',q',e',y'] \in Q : y' = \gamma \text{ a } \forall[l+1,x',q',e',y'] \in Q : y' \notin \{\beta,\gamma\}\end{gathered} \tag{3.23}$$

$$\begin{gathered}\delta_H([0,\text{¢},q,e,y],Q,\varepsilon) = [0,\text{¢},q,e,\gamma] \text{ ak platí} \\ y \notin \{\beta,\gamma,\overline{\gamma}\} \text{ a } \forall[l+1,x',q',e',y'] \in Q : x' = y \text{ a } \beta \notin Q\end{gathered} \tag{3.24}$$

Keď sa takýmto spôsobom signál o „skoro dokončení“ posúvania vrstiev dostane až k poslednej vrstve, tak sa posúvanie považuje za skutočne dokončené. Teda až vrchol na poslednej (novo vzniknutej) vrstve (v piatej komponente svojho stavu má napísaný symbol $\overline{\gamma}$) zistí, že všetci jeho susedia na predchádzajúcej vrstve už „skoro dokončili“ posúvanie (v piatej komponente svojho stavu majú zapísaný symbol γ), tak tento vrchol do piatej komponenty svojho stavu zapíše symbol β, čím sa preňho posúvanie vrstiev končí:

$$\begin{gathered}\delta_H([l,x,q,e,\overline{\gamma}],Q,\mathbf{T}) = [l,x,q,e,\beta] \text{ ak platí} \\ \forall[l-1,x',q',e',y'] \in Q : y' = \gamma\end{gathered} \tag{3.25}$$

No a keď vrchol, ktorý „skoro dokončil“ posúvanie vrstiev (v piatej komponente svojho stavu má symbol γ) zistí, že všetci jeho susedia už posúvanie naozaj dokončili (v piatej komponente svojho stavu majú symbol β), tak aj on už definitívne dokončí toto posúvanie vrstiev (teda do piatej komponenty svojho stavu zapíše symbol β):

$$\begin{gathered}\delta_H([l,x,q,e,\gamma],Q,\mathbf{T}) = [l,x,q,e,\beta] \text{ ak platí} \\ \forall[l+1,x',q',e',y'] \in Q : y' = \beta\end{gathered} \tag{3.26}$$

$$\begin{gathered}\delta_H([0,\text{¢},q,e,\gamma],Q,\varepsilon) = [0,\text{¢},q,e,\beta] \text{ ak platí} \\ \forall[1,x',q',e',y'] \in Q : y' = \beta\end{gathered} \tag{3.27}$$

Až sa tento signál o dokončení posúvania vrstiev dostane k čítajúcemu vrcholu, bude sa pokračovať simuláciou posunu hlavy (bolo popísané vyššie). Keďže sme však posunuli jednotlivé vrstvy, musíme hlavu posunúť o jedno políčko doprava navyše (viď rovnosti 3.5, 3.9 a 3.7).

Ak nejaký vrchol zistí, že sa simulovaný stroj M dostal do akceptačného stavu (teda ten vrchol má v tretej komponente svojho stavu nejaký akceptačný stav stroja M), prejde tento vrchol do stavu φ:

$$\delta_H([l,x,q,e,y],Q,a) = \varphi \text{ ak } q \in F_M \tag{3.28}$$

Tento stav sa potom „ako choroba“ bude šíriť po celom ACA:

$$\delta_H(q,Q,a) = \varphi \text{ ak } \varphi \in Q \tag{3.29}$$

Potrebujeme ešte ošetriť prípad, že v grafe prepojení budú nejaké „slepé vetvy“, na ktorých by sa výpočet zasekol (najmä tá simulácia rozširovania pracovnej pásky). Teda

keď vrchol zistí, že nemá žiadnych nasledovníkov (žiadnych susedov na nasledujúcej vrstve, ani žiadnych susedov v stave β), prejde do stavu ω:

$$\begin{gathered}\delta_H([l,x,q,e,y],Q,\mathbf{T})=\omega \text{ ak platí}\\ \varphi\notin Q \text{ a } \forall[l',x',q',e',y']\in Q: l'\neq l+1 \text{ a } \beta\notin Q\end{gathered} \tag{3.30}$$

No a vrcholy, ktoré sú v stave ω sa tvária, že vôbec nie sú:

$$\delta_H(q,Q,a)=\delta_H(q,Q\cap(K_H-\{\omega\}),a) \tag{3.31}$$

Definíciu funkcie δ_H zúplníme tým, že položíme $\delta_H(q,Q,x)=q$, ak sa hodnota $\delta_H(q,Q,x)$ nedá odvodiť z rovností 3.1 až 3.31.

3.3 Dôkaz ekvivalencie stroja M a schémy H na prepojeniach z $\mathcal{G}_D$

Aby sa nám jednoduchšie dokazovalo, zavedieme najprv nejaké označenia a pojmy.

Označenie 3.3.1 Budeme značiť $K_H^B=K_H-\{\beta,\alpha,\omega,\varphi\}$. Ďalej zavedieme zobrazenia stavov z K_H^B takto:

$\pi_1([l,x,q,e,y])=l$

$\pi_2([l,x,q,e,y])=x$

$\pi_3([l,x,q,e,y])=q$

$\pi_4([l,x,q,e,y])=e$

$\pi_5([l,x,q,e,y])=y$

$\pi_{34}([l,x,q,e,y])=[q,e]$

Tieto zobrazenia budeme chápať aj ako homomorfizmy definované na abecede K_H^B.

Definícia 3.3.2 *(vetva) Nech (G,r,H) je ACA s prechodovou schémou, ktorý sme skonštruovali. Nech (f,w) je jeho konfigurácia. Nech p je nejaká najkratšia*[4] *cesta v grafe G vychádzajúca z vrchola r. Nech $f(u)\in K_H^B$ pre každý vrchol u na ceste p.*

Budeme hovoriť, že p je vetva *pri konfigurácii (f,w). Ak navyše existuje vrchol v taký, že $f(v)=\beta$ a pv je najkratšia cesta z r tak povieme, že p je* otvorená vetva.

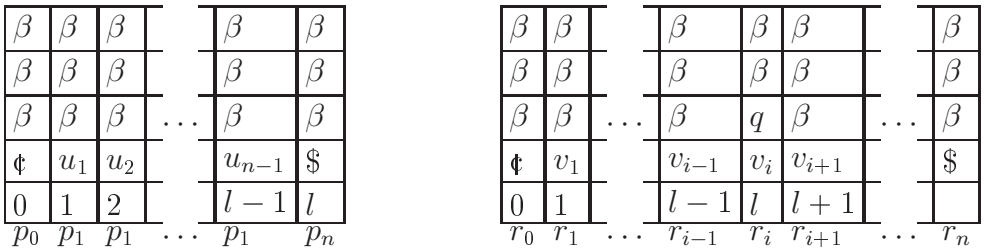
V ďalšom sa budeme zaoberať stavmi vrcholov na vetvách. Aby sa nám jednoduchšie vyjadrovalo, budeme takéto postupnosti stavov považovať za slová nad abecedou K_H.

Definícia 3.3.3 *(dobré očíslovanie) Nech $s=s_0s_1\ldots s_n\in(K_H^B)^*$.*

Ak $\pi_1(s_i)\equiv i \pmod 3$ pre každé $0\leq i\leq n$, tak povieme, že s je dobre očíslované.

V ďalšom uvidíme, že postupnosti stavov vrcholov na vetvách budú dobre očíslované.

[4] Budeme hovoriť, že $p=p_1p_2\ldots p_k$ je najkratšia cesta vychádzajúca z p_1 ak je to najkratšia cesta z p_1 do p_k.



Obr. 3.6: Postupnosť stavov $p_0p_1 \dots p_n$ typu NOSTATE a postupnosť stavov $r_0r_1 \dots r_n$ typu READY

3.3.1 Typy vetiev

V ďalšom texte sa budeme zaoberať určitými typmi postupností stavov vrcholov na vetvách. V tejto časti definujeme tieto typy a aj niektoré významné vlastnosti postupností stavov.

Definícia 3.3.4 *(typy vetiev I) Nech $s \in (K_H^B)^*$ je dobre očíslované a $|s| = n$. Povieme, že s je typu*

NOSTATE ak $\pi_3(s) = \beta^n$, $\pi_4(s) = \beta^n$ a $\pi_5(s) = \beta^n$.

Definícia 3.3.5 *(typy vetiev II) Nech $s \in (K_H^B)^*$ je dobre očíslované, $|s| = n$, $0 \leq i \leq n$, $q \in K_M$. Povieme, že s je typu*

READY ak $\pi_3(s) = \beta^i q \beta^{n-i-1}$, $\pi_4(s) = \beta^n$ a $\pi_5(s) = \beta^n$.

SHIFT-L-1 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Leftarrow][\beta,\beta]^{n-i-1}$ a $\pi_5(s) = \beta^n$.

SHIFT-L-2 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\leftarrow][q,\Leftarrow][\beta,\beta]^{n-i-2}$ a $\pi_5(s) = \beta^n$.

SHIFT-L-3 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\leftarrow][\beta,\beta]^{n-i-1}$ a $\pi_5(s) = \beta^n$.

SHIFT-R-1 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Rightarrow][\beta,\beta]^{n-i-1}$ a $\pi_5(s) = \beta^n$.

SHIFT-R-2 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Rightarrow][q,\rightarrow][\beta,\beta]^{n-i-2}$ a $\pi_5(s) = \beta^n$.

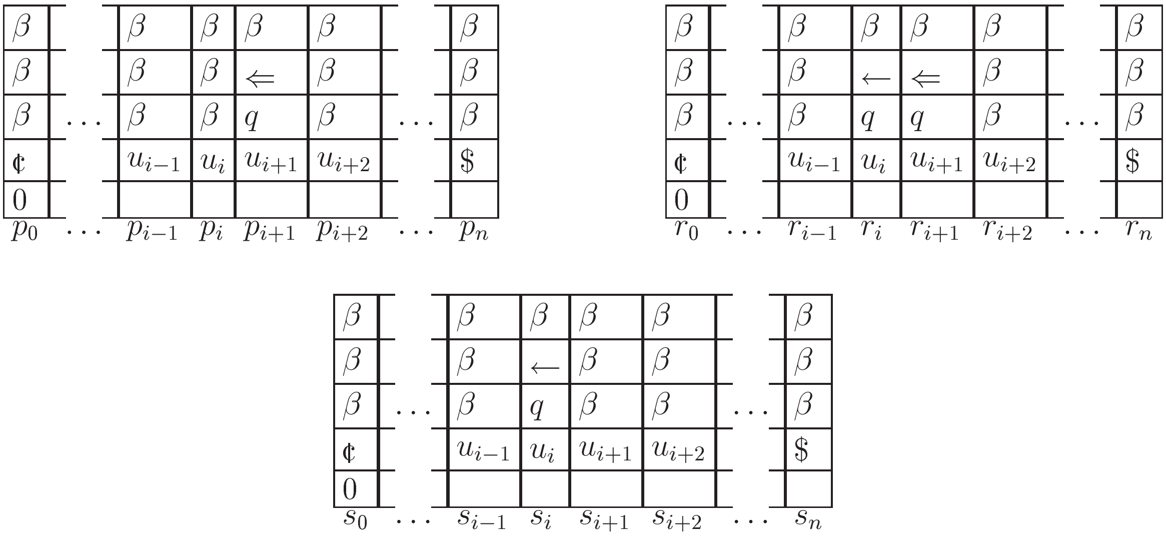
SHIFT-R-3 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\rightarrow][\beta,\beta]^{n-i-1}$ a $\pi_5(s) = \beta^n$.

SHIFT-RR-1 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Rrightarrow][\beta,\beta]^{n-i-1}$ a $\pi_5(s) = \beta^n$.

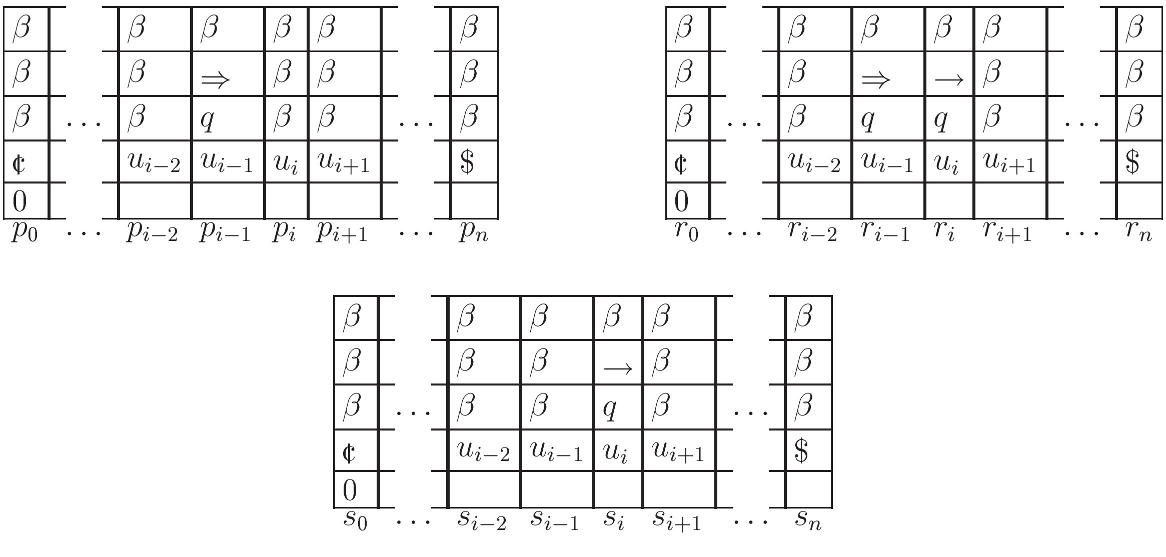
SHIFT-RR-2 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Rrightarrow][q,\Rightarrow][\beta,\beta]^{n-i-2}$ a $\pi_5(s) = \beta^n$.

SHIFT-RR-3 ak $\pi_{34}(s) = [\beta,\beta]^i[q,\Rrightarrow][q,\Rightarrow][q,\rightarrow][\beta,\beta]^{n-i-3}$ a $\pi_5(s) = \beta^n$.

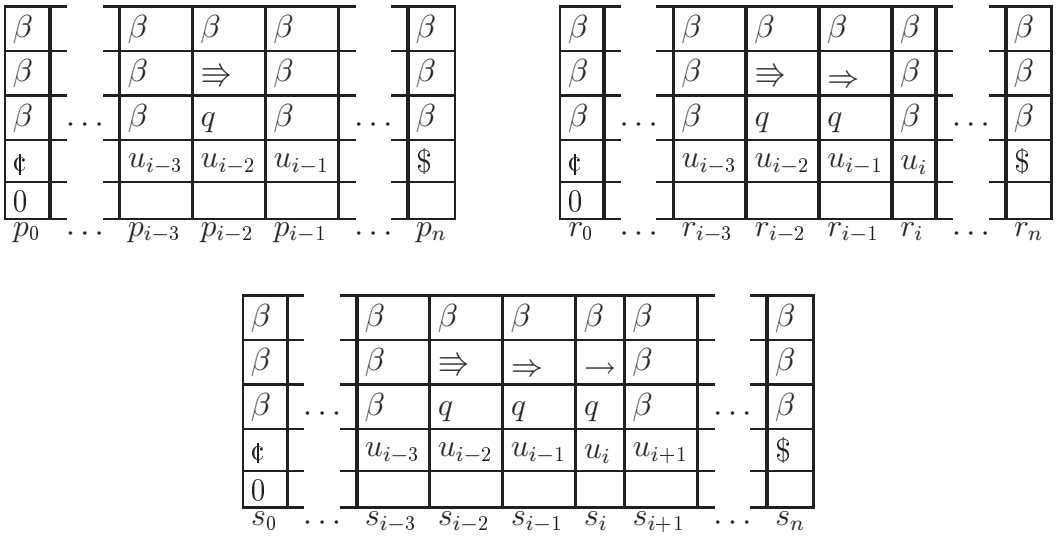
Definícia 3.3.6 *(typy vetiev III) Nech $s = uvxy \in (K_H^B)^*$ je dobre očíslované, pričom $x \in K_H^B$, $\pi_3(s) = q\beta^{n-1}$, $\pi_4(s) \in \{\beta,\Rightarrow,\Rrightarrow\}\beta^{n-1}$, $|s| = n$, $q \in K_M$. Povieme, že s je typu*



Obr. 3.7: Postupnosti stavov $p_0p_1\dots p_n$ typu SHIFT-L-1, $r_0r_1\dots r_n$ typu SHIFT-L-2 a $s_0s_1\dots s_n$ typu SHIFT-L-3



Obr. 3.8: Postupnosti stavov $p_0p_1\dots p_n$ typu SHIFT-R-1, $r_0r_1\dots r_n$ typu SHIFT-R-2 a $s_0s_1\dots s_n$ typu SHIFT-R-3



Obr. 3.9: Postupnosti stavov $p_0p_1 \dots p_n$ typu SHIFT-RR-1, $r_0r_1 \dots r_n$ typu SHIFT-RR-2 a $s_0s_1 \dots s_n$ typu SHIFT-RR-3

EXP-1 *ak* $\pi_5(u) \in \gamma^*$, $\pi_5(v) \in (\Gamma_M \cup \{\$\})^*$, $\pi_5(x) \in \Gamma_M \cup \{\$\}$, $\pi_5(y) \in \beta^*$

EXP-2 *ak* $\pi_5(u) \in \gamma^*$, $\pi_5(v) \in (\Gamma_M \cup \{\$\})^*$, $\pi_5(x) = \overline{\gamma}$, $y = \varepsilon$.

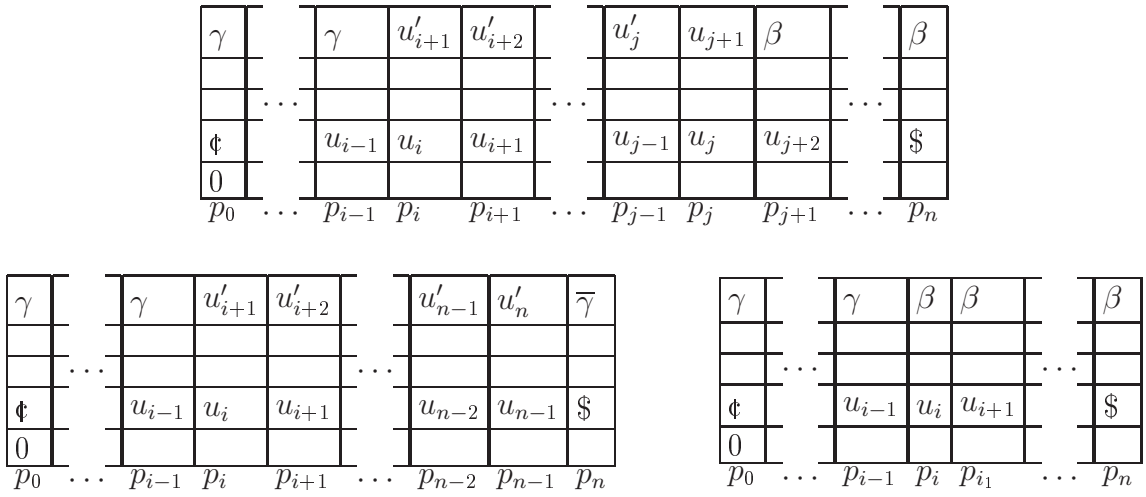
EXP-3 *ak* $\pi_5(u) \in \gamma^*$, $\pi_5(vx) \in \beta^*$, $y = \varepsilon$.

Neskôr uvidíme, že v postupnostiach stavov vrcholov na vetvách sú „zaznamenané“ konfigurácie simulovaného stroja M a tieto typy zodpovedajú rôznym fázam simulácie prechodu z jednej konfigurácie (stroja M) k nasledujúcej. V postupnostiach stavov typu NOSTATE nie je zaznamenaný stav simulovaného stroja M. Toto sa stáva na krátkych vetvách keď je hlava simulovaného stroja príliš vpravo. Ak postupnosť stavov vrcholov na vetve je typu READY, znamená to, že simulácia jedného kroku výpočtu stroja M je dokončená a môže sa začať simulácia ďalšieho kroku. Postupnosti typu SHIFT-L-1, SHIFT-L-2 a SHIFT-L-3 vznikajú pri simulácii posunu hlavy stroja M doľava. Podobne postupnosti typu SHIFT-R-1, SHIFT-R-2 a SHIFT-R-3 vznikajú pri simulácii posunu hlavy stroja M doprava a postupnosti typu SHIFT-RR-1, SHIFT-RR-2 a SHIFT-RR-3 vznikajú pri simulácii posunu hlavy stroja M doprava o dve políčka. Postupnosti typu EXP-1, EXP-2a EXP-3 vznikajú pri posúvaní zapamätaného obsahu pracovnej pásky pri simulácii pripísania písmenka tesne pred ľavý koniec pásky.

Teraz definujeme zobrazenia z týchto postupností, pomocou ktorých zistíme jednotlivé komponenty konfigurácií stroja M.

Funkcia *state* priradí postupnosti stavov vrcholov stav simulovaného stroja M, ktorý je v nej zapamätaný.

Definícia 3.3.7 *(*state*)* *Definujme zobrazenie* state *z* $(K_H^B)^*$ *do* K_M *takto: Nech* $s \in (K_H^B)^*$ *je niektorého z vyššie definovaných typov, nie však typu NOSTATE. Nech existujú* $u, v \in K_M^*$ *a* $q \in K_M$ *také, že* $\pi_3(s) = uqv$. *Potom bude* $\text{state}(s) = q$.



Obr. 3.10: Postupnosti stavov $p_0p_1 \dots p_n$ typu EXP-1, $r_0r_1 \dots r_n$ typu EXP-2 a $s_0s_1 \dots s_n$ typu EXP-3

Funkcia *mem* priradí postupnosti stavov vrcholov obsah pracovnej pásky simulovaného stroja M, ktorý je v nej zapamätaný.

Definícia 3.3.8 (mem) *Definujme zobrazenie* mem $z\,(K_H^B)^*$ *do* Γ_M^* *takto: Nech* $s \in (K_H^B)^*$ *je niektorého z vyššie definovaných typov.*

Ak s je typu EXP-1 tak bude $\mathrm{mem}(s) = \pi_2(ux)\pi_5(x)\pi_2(v)$, *pričom u, v, x sú také, že* $s = uxv$ *a* $\pi_5(x) \in \Gamma_M \cup \{¢, \$\}$ *a* $\pi_5(v) \in \beta^*$.

inak bude $\mathrm{mem}(s) = \pi_2(s)$.

Funkcia *pos* priradí postupnosti stavov vrcholov polohu hlavy na pracovnej páske simulovaného stroja M.

Definícia 3.3.9 (pos) *Definujme zobrazenie* pos $z\,(K_H^B)^*$ *do množiny prirodzených čísel takto: Nech* $s = s_0s_1s_2 \dots s_n \in (K_H^B)^*$ *je niektorého z vyššie definovaných typov, ale nie typu NOSTATE. Bude:*

$\mathrm{pos}(s) = i$ *ak s je typu READY a* $\pi_3(s_i) \in K_M$.

$\mathrm{pos}(s) = i$ *ak s nie je typu READY a* $\pi_4(s_i) \in \{\leftarrow, \rightarrow\}$.

$\mathrm{pos}(s) = i - 1$ *ak s nie je typu READY a* $\pi_4(s_i) = \Leftarrow$.

$\mathrm{pos}(s) = i + 1$ *ak s nie je typu READY a* $\pi_4(s_i) = \Rightarrow$.

$\mathrm{pos}(s) = i + 2$ *ak s nie je typu READY a* $\pi_4(s_i) = \Rrightarrow$.

Z definícií vyššie vyplýva, že táto definícia funkcie pos *je korektná.*

Funkcia *len* priradí postupnosti stavov vrcholov počet zapísaných políčok pracovnej pásky simulovaného stroja M.

Definícia 3.3.10 *(*len*) Definujeme zobrazenie* len *z* $(K_H^B)^*$ *do množiny prirodzených čísel takto:*

$\text{len}(s) = |\text{mem}(s)|$.

Poznámka 3.3.11 Všimnime si, že ak s je typu EXP-1, tak $\text{len}(s) = |s| + 1$, inak $\text{len}(s) = |s|$.

Pre postupnosti stavov typu EXP-1 definujeme funkciu *lead*, ktorá vyjadruje, ako ďaleko od čítajúceho vrchola je „začiatok“ vlny, ktorá sa vyslala pri rozširovaní pracovnej pásky simulovaného stroja M.

Definícia 3.3.12 *(*lead*) Definujeme zobrazenie* lead *z* $(K_H^B)^*$ *do množiny prirodzených čísel takto: Nech* $s = s_0 s_1 s_2 \dots s_n \in (K_H^B)^*$ *je typu EXP-1. Nech* $\pi_5(s_i) \in \Gamma_M \cup \{¢, \$\}$ *a* $\pi_5(s_{i+1}) = \beta$. *Potom bude* $\text{lead}(s) = i$.

Pre postupnosti stavov typu EXP-1 a EXP-2 definujeme funkciu *trail*, ktorá vyjadruje, ako ďaleko od čítajúceho vrchola je „koniec“ vlny, ktorá sa vyslala pri rozširovaní pracovnej pásky simulovaného stroja M.

Definícia 3.3.13 *(*trail*) Definujeme zobrazenie* trail *z* $(K_H^B)^*$ *do množiny prirodzených čísel takto: Nech* $s = s_0 s_1 s_2 \dots s_n \in (K_H^B)^*$ *je typu EXP-1 alebo EXP-2. Nech* $\pi_5(s_i) \in \Gamma_M \cup \{¢, \$, \overline{\gamma}\}$ *a* $\pi_5(s_{i-1}) = \gamma$*(alebo* $i = 0$*). Potom bude* $\text{trail}(s) = i$.

Pre postupnosti stavov typu EXP-3 definujeme funkciu *dist*, ktorá vyjadruje, ako ďaleko od čítajúceho vrchola je vlna vracajúca sa po dokončení rozširovania pracovnej pásky simulovaného stroja M.

Definícia 3.3.14 *(*dist*) Definujeme zobrazenie* dist *z* $(K_H^B)^*$ *do množiny prirodzených čísel takto: Nech* $s = s_0 s_1 s_2 \dots s_n \in (K_H^B)^*$ *je typu EXP-3. Nech* $\pi_5(s_i) = \beta$ *a* $\pi_5(s_{i-1}) = \gamma$*(alebo* $i = 0$*). Potom bude* $\text{dist}(s) = i$.

3.3.2 Dobré konfigurácie

Definícia 3.3.15 *(dobrá konfigurácia) Nech* (G, r, H) *je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech* (f, w) *je jeho konfigurácia. Povieme, že* (f, w) *je dobrá ak platí:*

(i) Pre každú vetvu p *pri konfigurácii* (f, w) *je* $f(p)$ *jedného z typov zavedených v časti 3.3.1. Navyše ak* p *je otvorená vetva, tak* $f(p)$ *nie je typu NOSTATE.*

(ii) Ak p, q *sú vetvy pri konfigurácii* (f, w) *a ak* p *je otvorená, tak* $\text{len}(f(p)) \geq \text{len}(f(q))$.

(iii) Každý vrchol v *grafu* G, *pre ktorý je* $f(v) \in K_H^B$ *leží na nejakej vetve pri konfigurácii* (f, w).

Poznámka 3.3.16 Všimnime si, že ak (f, w) je dobrá konfigurácia a p, q sú otvorené vetvy pri tejto konfigurácii, tak $\text{len}(p) = \text{len}(q)$.

3.3.3 Jednokroková budúcnosť vetiev pri dobrých konfiguráciách

V tejto časti si ukážeme, ako sa v ACA s prechodovou schémou H, ktorú sme skonštruovali v časti 3.2 môžu zmeniť postupnosti stavov vrcholov na vetvách. Význam liem, ktoré v tejto časti vyslovíme je znázornený na obrázku 3.11, ktorý je na strane 39. Vo všeobecnosti môže nastať niekoľko prípadov:

(i) Vrcholy na konci vetvy začnú „odumierať“ podľa rovnosti 3.30. Tento prípad nastane, keď sa ukáže, že táto vetva je príliš krátka na to, aby sa na nej dala pamätať konfigurácia simulovaného stroja M. Ukážeme navyše, že tento prípad nastane iba, ak tá vetva v predchádzajúcej konfigurácii nebola otvorená (lebo otvorená vetva sa dá ešte rozšíriť).

(ii) Niektorý vrchol prejde do akceptačného stavu. V tomto prípade sa už v niektorej časti uvažovaného ACA zistilo, že simulovaný stroj M akceptoval svoj vstup. V takomto prípade nás už nezaujíma, čo robia vrcholy, ku ktorým sa táto informácia ešte nedostala.

(iii) Žiadny z vrcholov na tej vetve nebude vybratý, aby urobil svoj krok výpočtu, teda stavy vrcholov na vetvách sa nezmenia.

(iv,...) Ešte sa môže stať, že stavy vrcholov na vetvách sa zmenia (aspoň niektoré). V nasledujúcich lemách ukážeme, že sa zmenia len tým dobrým spôsobom.

Lema 3.3.17 *(čo sa stane s vetvou typu NOSTATE) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu NOSTATE.*

Potom nastane jeden z prípadov:

(i) p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).

(ii) existuje $v \in V(G)$ také, že $g(v) = \varphi$

(iii) $g(p) = f(p)$

(iv) $g(p)$ je typu SHIFT-L-3 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = |p| - 1$

Dôkaz Nech $p = v_0 v_1 \dots v_n$. Vrcholy v_0, v_1, v_2, ..., v_{n-1} (ak boli vybraté v S) môžu urobiť prechod jedine podľa rovnosti 3.29 (ľahko sa overí, že predpoklady pri ostatných rovnostiach definujúcich funkciu δ_H nie sú splnené). Ak ho niektorý spraví, tak nastane prípad (ii).

Vrchol v_n môže urobiť prechod podľa rovnosti 3.30, alebo 3.10 (prípadne 3.11). Vtedy nastane prípad (i) alebo (iv).

A zrejme ak žiadny vrchol nezmení svoj stav, tak nastane prípad (iii). □

Poznámka 3.3.18 Nasledujú ďalšie podobné lemy o tom, čo sa stane s vetvami jednotlivých typov. Ich dôkazy budú založené na podobnej myšlienke, ako ten predchádzajúci. Zistí sa, ktoré vrcholy na ceste p môžu urobiť netriviálne prechody (väčšinou ich bude len málo), a z toho sa usúdi, ako sa zmení typ vetvy.

Dôkazy nasledujúcich liem budú preto zjednodušené. Prípadu, že niektorý vrchol urobí prechod podľa rovnosti 3.29, alebo 3.30 sa nebudeme hlbšie venovať, lebo sa riešia tak isto ako v dôkaze lemy 3.3.17 (čo sa stane s vetvou typu NOSTATE).

Lema 3.3.19 *(čo sa stane s vetvou typu READY) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu READY. Nech* $\mathrm{pos}(f(p)) = i$, $\mathrm{state}(f(p)) = q$, $\mathrm{mem}(f(p)) = uxv$, $|u| = i$, $|x| = 1$.

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu SHIFT-L-1 a* $\mathrm{mem}(g(p)) = uyv$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p)) - 1$, $\mathrm{state}(g(p)) = r$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$, *pričom $\delta_M(q, x) = (r, y, -1)$*

(v) *$g(p)$ je typu SHIFT-R-1 a* $\mathrm{mem}(g(p)) = uyv$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p)) + 1$, $\mathrm{state}(g(p)) = r$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$, *pričom $\delta_M(q, x) = (r, y, +1)$*

(vi) *$g(p)$ je typu READY a* $\mathrm{mem}(g(p)) = uyv$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = r$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$, *pričom $\delta_M(q, x) = (r, y, 0)$*

(vii) *$g(p)$ je typu EXP-1 a* $\mathrm{mem}(g(p)) = ¢yv$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p)) + d$, $\mathrm{state}(g(p)) = r$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p)) + 1$, $\mathrm{lead}(g(p)) = \mathrm{trail}(g(p)) = 0$, *$u = \varepsilon$ pričom $\delta_M(q, ¢a) = (r, y, d)$ pre vhodné a*

Dôkaz Ľahko zistíme, že zaujímavý prechod mohol urobiť len ten (jediný) vrchol v, v ktorom je $\pi_3(f(v)) \neq \beta$, a to podľa niektorej z rovností 3.4 až 3.9. Podľa toho, ktorá to bola ľahko overíme, že nastane jeden z uvedených prípadov. □

Priebeh simulácie posunu hlavy doľava

Lema 3.3.20 *(čo sa stane s vetvou typu SHIFT-L-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-L-1.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) $g(p) = f(p)$

(iv) *$g(p)$ je typu SHIFT-L-2 a* $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$

Dôkaz Nech $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_i \dots v_n$. Opäť ľahko overíme, že jediný vrchol, ktorý môže urobiť zaujímavý prechod je v_i a to podľa rovnosti 3.10, resp. 3.11. □

Lema 3.3.21 *(čo sa stane s vetvou typu SHIFT-L-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-L-2.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) $g(p) = f(p)$

(iv) *$g(p)$ je typu SHIFT-L-3 a* $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$

Dôkaz Nech $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_i v_{i+1} \dots v_n$. Opäť ľahko overíme, že jediný vrchol, ktorý môže urobiť zaujímavý prechod je v_{i+1} a to podľa rovnosti 3.12. □

Lema 3.3.22 *(čo sa stane s vetvou typu SHIFT-L-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-L-3.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) $g(p) = f(p)$

(iv) *$g(p)$ je typu READY a* $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$

Dôkaz Nech $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_i v_{i+1} \dots v_n$. Opäť ľahko overíme, že jediný vrchol, ktorý môže urobiť zaujímavý prechod je v_i a to podľa rovnosti 3.13, resp. 3.14. □

Poznámka 3.3.23 Nasledujúce tri lemy sú veľmi podobné predchádzajúcim trom (je to to isté, len na opačnú stranu), preto sú uvedené bez dôkazu.

Priebeh simulácie posunu hlavy doprava

Lema 3.3.24 *(čo sa stane s vetvou typu SHIFT-R-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-R-1.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu SHIFT-R-2 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

Lema 3.3.25 *(čo sa stane s vetvou typu SHIFT-R-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-R-2*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu SHIFT-R-3 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

Lema 3.3.26 *(čo sa stane s vetvou typu SHIFT-R-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-R-3.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu READY a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

Priebeh simulácie posunu hlavy doprava o dve políčka

Lema 3.3.27 *(čo sa stane s vetvou typu SHIFT-RR-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-RR-1.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu SHIFT-RR-2 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

Dôkaz Ak $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_{i-2} v_{i-1} v_i \dots v_n$, tak jediný vrchol ktorý môže urobiť zaujímavý prechod je v_{i-1} a to podľa rovnosti 3.15, čím sa dosiahne, že nastane práve uvedený prípad. □

Lema 3.3.28 *(čo sa stane s vetvou typu SHIFT-RR-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-RR-2.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu SHIFT-RR-3 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

(v) *$g(p)$ je typu SHIFT-R-1 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

(vi) *$g(p)$ je typu SHIFT-R-2 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$*

Dôkaz Ak $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_{i-2} v_{i-1} v_i \dots v_n$, tak zaujímavé prechody môžu urobiť vrcholy v_i (podľa rovnosti 3.17) a v_{i-2} (podľa rovnosti 3.16). Ak ho urobí v_i, tak nastane prípad (vi), ako ho urobí v_{i-2}, tak nastane prípad (v) a ak ho urobia oba, tak nastane prípad (vi). □

Lema 3.3.29 *(čo sa stane s vetvou typu SHIFT-RR-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu SHIFT-RR-3.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) $g(p) = f(p)$

(iv) *$g(p)$ je typu SHIFT-R-2 a* $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$

Dôkaz Ak $\mathrm{pos}(f(p)) = i$ a $p = v_0 v_1 \dots v_{i-2} v_{i-1} v_i \dots v_n$, tak zaujímavý prechod môže urobiť len vrchol v_{i-2} (podľa rovnosti 3.16). □

Priebeh simulácie rozširovania pracovnej pásky

Lema 3.3.30 *(čo sa stane s vetvou typu EXP-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu EXP-1.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) $g(p) = f(p)$

(iv) *$g(p)$ je typu EXP-1 a* $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$ *a nastane jeden z týchto prípadov:*

$$\mathrm{lead}(g(p)) = \mathrm{lead}(f(p)) + 1,\ \mathrm{trail}(g(p)) = \mathrm{trail}(f(p))$$
$$\mathrm{lead}(g(p)) = \mathrm{lead}(f(p)),\ \mathrm{trail}(g(p)) = \mathrm{trail}(f(p)) + 1$$
$$\mathrm{lead}(g(p)) = \mathrm{lead}(f(p)) + 1,\ \mathrm{trail}(g(p)) = \mathrm{trail}(f(p)) + 1$$

(v) *$g(p)$ je typu EXP-3 a* $\exists a \in (\Gamma_M \cup \{\$\}) : \mathrm{mem}(g(p))a = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p)) - 1$. *Pritom však bolo* $\mathrm{lead}(f(p)) = |p| - 1$.

Dôkaz Nech $p = v_0 v_1 \dots v_i v_{i+1} \dots v_j v_{j+1} \dots v_n$. Nech $\pi_5(f(v_0 \dots v_i)) \in \gamma^*$, $\pi_5(f(v_{i+1} \dots v_j)) \in (\Gamma^M \cup \{\$\})^+$, $\pi_5(f(v_{j+1} \dots v_n)) \in \beta^*$. Jediné vrcholy, ktoré mohli urobiť zaujímavý prechod sú v_{i+1} podľa rovnosti 3.23, resp. 3.24 a v_{j+1} podľa rovnosti 3.21. □

Lema 3.3.31 *(rozšírenie vetvy typu EXP-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu EXP-1. Nech v je taký vrchol G, že $f(v) = \beta$ a pv je najkratšia cesta z r do v*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(v) = \beta$*

(iv) *$g(v) \neq \beta$ a $g(pv)$ je typu EXP-2 a $\mathrm{mem}(g(pv)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(pv)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(pv)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(pv)) = \mathrm{len}(f(p))$ a $\mathrm{trail}(g(pv)) = \mathrm{trail}(f(p))$ alebo $\mathrm{trail}(g(pv)) = \mathrm{trail}(f(p)) + 1$. Pritom však bolo $\mathrm{lead}(f(p)) = |p| - 1$.*

Dôkaz Nech $p = v_0 v_1 \dots v_i v_{i+1} \dots v_j v_{j+1} \dots v_n$. Nech $\pi_5(f(v_0 \dots v_i)) \in \gamma^*$, $\pi_5(f(v_{i+1} \dots v_j)) \in (\Gamma^M \cup \{\$\})^+$, $\pi_5(f(v_{j+1} \dots v_n)) \in \beta^*$. Jediné vrcholy, ktoré mohli urobiť zaujímavý prechod sú v_{i+1} podľa rovnosti 3.23, resp. 3.24 a v_{j+1} podľa rovnosti 3.21 a ešte v podľa rovnosti 3.22, ale to len ak bolo $j = n$. □

Lema 3.3.32 *(čo sa stane s vetvou typu EXP-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu EXP-2.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu EXP-2 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$ a $\mathrm{trail}(g(p)) = \mathrm{trail}(f(p)) + 1$*

(v) *$g(p)$ je typu EXP-3 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$ a $\mathrm{dist}(g(p)) = |p| - 1$. Pritom však bolo $\mathrm{trail}(f(p)) = |p| - 1$.*

Dôkaz Nech $p = v_0 v_1 \dots v_i v_{i+1} \dots v_{n-1} v_n$. Nech $\pi_5(f(v_0 \dots v_i)) \in \gamma^*$, $\pi_5(f(v_{i+1} \dots v_{n-1})) \in (\Gamma^M \cup \{\$\})^+$, Jediné vrcholy, ktoré mohli urobiť zaujímavý prechod sú v_{i+1} podľa rovnosti 3.23, resp. 3.24 a v_n podľa rovnosti 3.25, ale iba ak bolo $i = n$. □

Lema 3.3.33 *(čo sa stane s vetvou typu EXP-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je jeho dobrá konfigurácia. Nech $S \subset V(G)$. Nech $(f, w) \vdash_A^S (g, w')$. Nech p je vetva pri konfigurácii (f, w) a nech $f(p)$ je typu EXP-3.*

Potom nastane jeden z prípadov:

(i) *p nie je vetva pri konfigurácii (g, w') a p nebola otvorená pri konfigurácii (f, w).*

(ii) *existuje $v \in V(G)$ také, že $g(v) = \varphi$*

(iii) *$g(p) = f(p)$*

(iv) *$g(p)$ je typu EXP-3 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$ a $\mathrm{dist}(g(p)) = \mathrm{dist}(f(p)) - 1$*

(v) *$g(p)$ je typu READY, SHIFT-R-1 alebo SHIFT-RR-1 a $\mathrm{mem}(g(p)) = \mathrm{mem}(f(p))$, $\mathrm{pos}(g(p)) = \mathrm{pos}(f(p))$, $\mathrm{state}(g(p)) = \mathrm{state}(f(p))$, $\mathrm{len}(g(p)) = \mathrm{len}(f(p))$. Pritom však bolo $\mathrm{dist}(f(p)) = 1$.*

Dôkaz Nech $p = v_0 v_1 \dots v_i v_{i+1} \dots v_n$. Nech $\pi_5(f(v_0 \dots v_i)) \in \gamma^*$, $\pi_5(f(v_{i+1} \dots v_{n-1})) \in \beta^*$, Jediný vrchol, ktorý mohol urobiť zaujímavý prechod je v_i podľa rovnosti 3.26, resp. 3.27. Ak $i = 0$, tak nastane prípad (v). □

3.3.4 Štruktúra dosiahnuteľných konfigurácií

V tejto časti si ukážeme, ako vyzerajú dosiahnuteľné konfigurácie každého ACA s prechodovou schémou H, ktorú sme skonštruovali v časti 3.2.

Lema 3.3.34 *(o rozširovaní vetvy) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (g, u') je jeho dobrá konfigurácia. Nech $(g, u') \vdash_A^S (f, u)$ pre vhodné $S \subseteq V(G)$. Nech $p = v_0 v_1 \dots v_n$ je vetva pri konfigurácii (f, u), ale nie je vetva pri konfigurácii (g, u').*

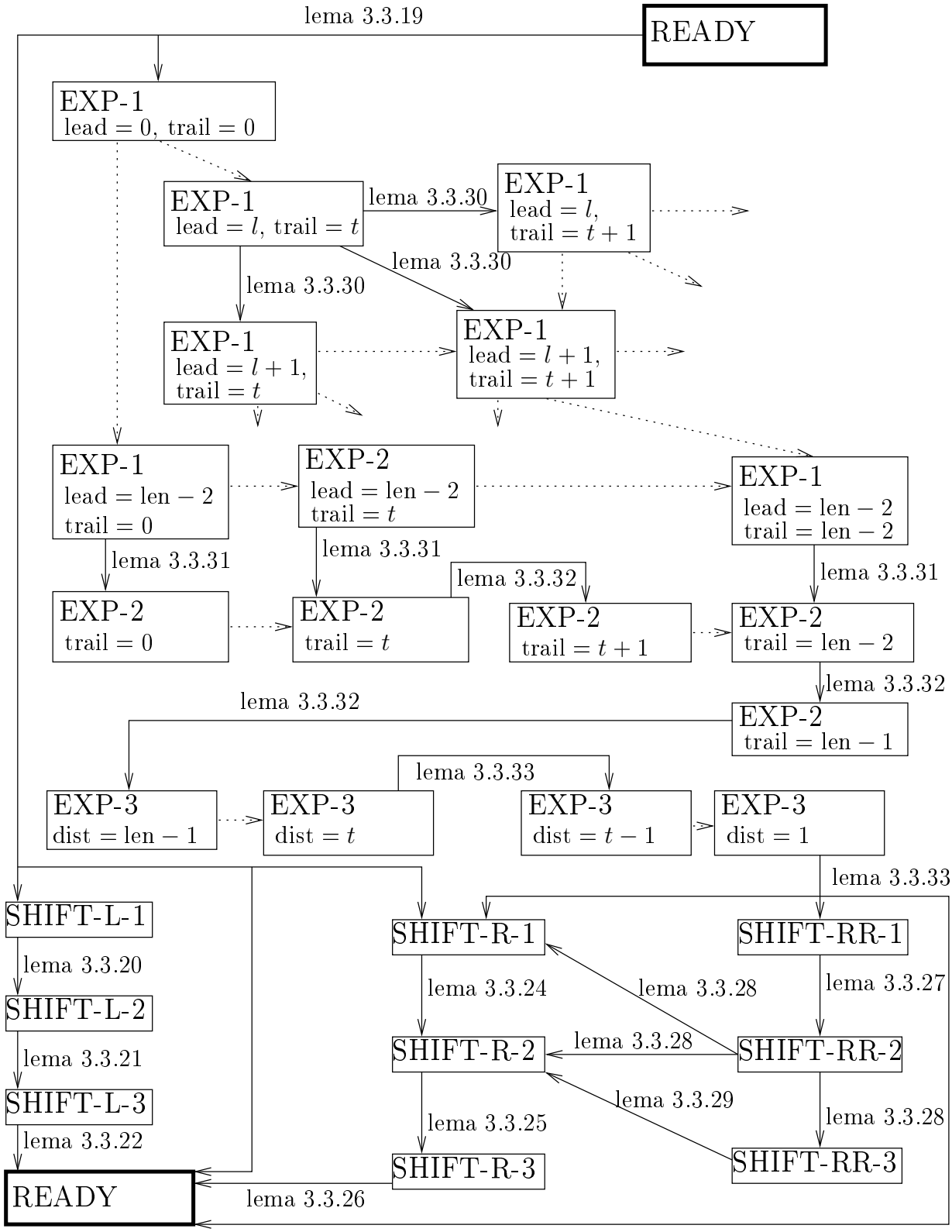
Potom $q = v_0 v_1 \dots v_{n-1}$ je otvorená vetva pri konfigurácii (g, u') a $g(q)$ je typu EXP-1 s $\mathrm{lead}(g(q)) = n - 2$ a $f(p)$ je typu EXP-2.

Dôkaz Keďže (g, u') je dobrá konfigurácia, tak zrejme $\forall i \in \{0, 1, \dots, n\} : g(v_i) \in K_H^B \cup \{\beta\}$. Nech $s = v_0 v_1 \dots v_k$ je najdlhší prefix p taký, že s je ešte vetva pri konfigurácii (g, u') (viď obr. 3.12a). Zrejme s je otvorená vetva.

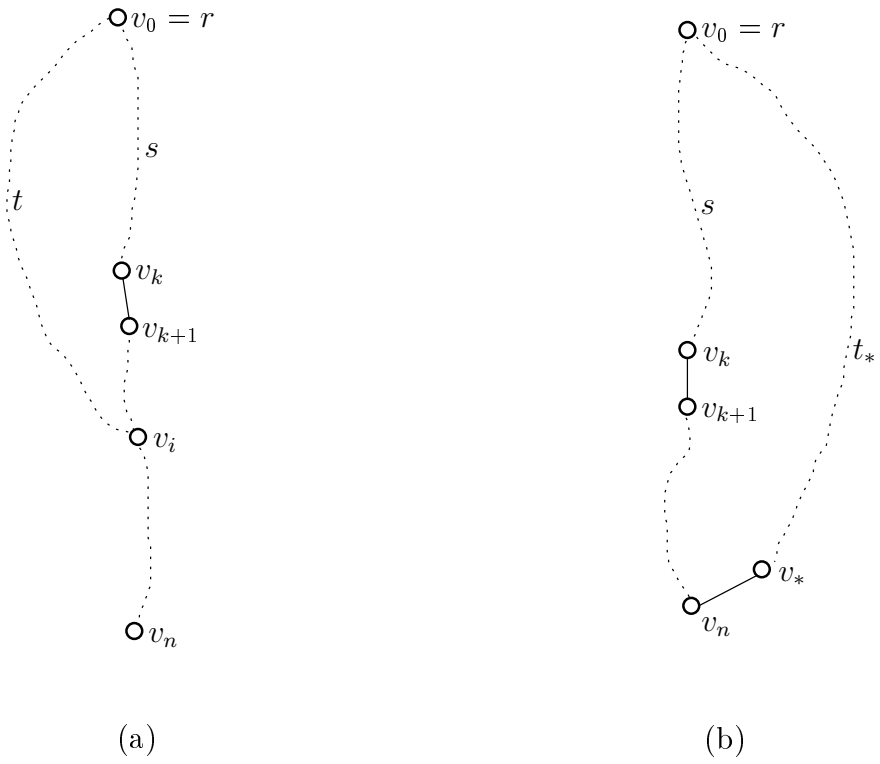
Uvažujme i také, že $k + 1 < i \leq n$. Keby bolo $g(v_i) \neq \beta$, tak (lebo (g, u') je dobrá konfigurácia) by v_i ležalo na nejakej vetve t pri konfigurácii (g, u'). Keďže ale aj $v_0 \dots v_i$ je najkratšia cesta z r do v_i, tak $i \leq \mathrm{len}(g(t)) \leq \mathrm{len}(g(s)) \leq k + 1$, čo je spor. Teda $\forall i \in \{k + 1, k + 2, \dots, n\} : g(v_i) = \beta$.

To, že $g(v_{k+1}) = \beta$ vyplýva z maximality s.

Vezmime si teraz vrchol v_n. Podľa predchádzajúceho bolo $g(v_n) = \beta$, teda zrejme v_n urobil prechod podľa rovnosti 3.22 (lebo p už je vetva pri konfigurácii (f, u)). Teda v_n



Obr. 3.11: Význam liem o jednokrokovej budúcnosti vetiev vyslovených v časti 3.3.3



Obr. 3.12: k dôkazu lemy 3.3.34 (o rozširovaní vetvy)

mal nejakého suseda v_*, ktorý zrejme ležal na nejakej vetve t_* pri konfigurácii (g, u') (viď obr. 3.12b). Navyše z predpokladu pri rovnosti 3.22 a toho, že (g, u') je dobrá vyplýva, že $g(t_*)$ je typu EXP-1. Nech l_* je dĺžka cesty t_*. Zrejme $\text{len}(g(t_*)) = l_* + 1$ Potom platí $k + 1 \leq n \leq l_* + 1 = \text{len}(g(t_*)) \leq \text{len}(g(s)) = k + 1$

Dostali sme teda, že q je vetva pri konfigurácii (g, u') a že $g(q)$ je typu EXP-1. Že $\text{lead}(g(q)) = n - 2$ a že $f(p)$ je typu EXP-2 už vyplýva z rovnosti 3.22 a toho, že vrchol v_n urobil prechod práve podľa nej. □

Lema 3.3.35 *(o štruktúre dosiahnuteľných konfigurácií) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, u) je jeho dosiahnuteľná konfigurácia na vstupe w. Nech $f(r) \in K_H^B$. Nech $f(v) \neq \varphi$ pre každý vrchol v grafu G.*

Potom:

1. *(f, w) je dobrá.*
2. *ak p je vetva pri konfigurácii (f, u) a $f(p)$ nie je typu NOSTATE, tak existuje z také, že $(\text{state}(f(p)), u, \text{mem}(f(p))z, \text{pos}(f(p)))$ je dosiahnuteľná konfigurácia stroja M na vstupe w. Navyše ak p je otvorená, tak $x = \varepsilon$.*

Dôkaz Dokážeme indukciou vzhľadom na dĺžku výpočtu, ktorým sa dosiahla konfigurácia (f, u).

1° Ak (f, u) je prvá konfigurácia, ktorá spĺňa predpoklad, tak zrejme

$f(r) = [0, ¢, q_M, \beta, \beta]$
$f(v) = [1, \$, \beta, \beta, \beta]$ pre každé $v \in V(G)$, ktoré je susedom r
$f(v) = \beta$ pre ostatné vrcholy v grafu G

V tomto prípade tvrdenie zrejme platí.

2° Nech (g, u') je konfigurácia, z ktorej sa na jeden krok dosiahla konfigurácia (f, u). Dokážeme postupne platnosť tvrdení (1) a (2).

(1) Dokážeme platnosť jednotlivých podmienok v definícii 3.3.15 (dobrá konfigurácia)

(i) Nech p je vetva pri konfigurácii (f, u). Mohli nastať tieto prípady:

p je vetva aj pri konfigurácii (g, u'):
V tomto prípade tvrdenie vyplýva priamo z indukčného predpokladu a liem vyslovených v časti 3.3.3.

p nie je vetva pri konfigurácii (g, u'):
V tomto prípade tvrdenie vyplýva z lemy 3.3.34 (o rozširovaní vetvy), ktorej predpoklady vyplývajú z indukčného predpokladu.

(ii) Pre q mohli nastať tieto prípady:

(a) q je vetva pri konfigurácii (g, u') a $g(q)$ je typu READY a $f(q)$ je typu EXP-1:
V tomto prípade podľa lemy 3.3.19 (čo sa stane s vetvou typu READY) je $\text{len}(g(q)) + 1 = \text{len}(f(q))$.

(b) q je vetva pri konfigurácii (g, u') a nenastáva predchádzajúci prípad:
Označme si $q_* = q$. Potom podľa liem vyslovených v časti 3.3.3 platí $\text{len}(g(q_*)) \geq \text{len}(f(q))$.

(c) q nie je vetva pri konfigurácii (g, u'):
V tom prípade podľa lemy 3.3.34 (o rozširovaní vetvy) a 3.3.31 (rozšírenie vetvy typu EXP-1) existuje vetva q_* pri konfigurácii (g, u') taká, že $\text{len}(g(q_*)) = \text{len}(f(q))$.

Pre p mohli nastať podobné prípady:

(a) p je vetva pri konfigurácii (g, u') a $g(p)$ je typu READY a $f(p)$ je typu EXP-1:
Všimnime si, že tento prípad nastane práve vtedy, keď nastane prípad (a) pre vetvu q. V tomto prípade podľa lemy 3.3.19 (čo sa stane s vetvou typu READY) a indukčného predpokladu je $\text{len}(f(p)) = \text{len}(g(p)) + 1 \geq \text{len}(g(q)) + 1 = \text{len}(f(q))$.

(b) p je vetva pri konfigurácii (g, u') a nenastáva predchádzajúci prípad:
Podľa liem vyslovených v časti 3.3.3 platí $\text{len}(g(p)) = \text{len}(f(q))$ a p je otvorená aj pri konfigurácii (g, u'). Teda podľa indukčného predpokadu platí $\text{len}(f(p)) = \text{len}(g(p)) \geq \text{len}(g(q_*)) \geq \text{len}(f(q))$.

(c) p nie je vetva pri konfigurácii (g, u'):
V tom prípade podľa lemy 3.3.34 (o rozširovaní vetvy) a 3.3.31 (rozšírenie vetvy typu EXP-1) existuje vetva p_* pri konfigurácii (g, u') taká, že p_* je pri konfigurácii (g, u') otvorená a $\text{len}(g(p_*)) = \text{len}(f(p))$. Teda podľa indukčného predpokladu platí $\text{len}(f(p)) = \text{len}(g(p_*)) \geq \text{len}(g(q_*)) \geq \text{len}(f(q))$.

(iii) Vezmime si vrchol v taký, že $f(v) \in K_H^B$. Mohli nastať dva prípady:

(*) $g(v) \in K_H^B$: Podľa indukčného predpokladu existuje vetva p pri konfigurácii (g, u'), na ktorej leží v. Podľa liem v časti 3.3.3 nastane jeden z týchto prípadov:

- $f(v') = \varphi$ pre niektorý vrchol v': Ale to je spor s predpokladom dokazovanej lemy.
- p je vetva aj pri konfigurácii (f, u).
- p nie je vetva pri konfigurácii (f, u):
  Označme si $p = v_0 v_1 \dots v_n$. Aby nastal tento prípad, musel niektorý vrchol na ceste p urobiť prechod podľa rovnosti 3.30. Podľa indukčného predpokladu u vrcholov $v_0, v_1, \dots v_{n-1}$ nie sú splnené predpoklady pri tej rovnosti. Ak ten prechod urobil vrchol $v_n = v$, tak dostávame spor s voľbou vrcholu v.

(**) $g(v) = \beta$: Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r do v. Nech $q = v_0 v_1 \dots v_k$ $(k < n)$ je najdlhší prefix p taký, že q je vetva pri konfigurácii (g, u'). Zrejme q je otvorená vetva pri konfigurácii (g, u'). Zrejme vrchol $v = v_n$ urobil prechod podľa rovnosti 3.22. Teda mal nejakého suseda v' takého, že $g(v') \in K_H^B$. Podľa indukčného predpokladu v' ležal na nejakej vetve p' pri konfigurácii (g, u') (viď obr. 3.13). Ďalej zrejme $g(p')$ je typu EXP-1. Nech l' je dĺžka cesty p'. Zrejme platí $n \leq l' + 1$ lebo p je najkratšia cesta z r do v a p' je najkratšia cesta z r do v' a v a v' sú susedia. Teda platí $n - 1 \geq k \geq \text{len}(g(q)) - 1 \geq \text{len}(g(p')) - 1 = (l' + 1) - 1 = l' \geq n - 1$. Teda $k = n - 1$. Z toho a lemy 3.3.31 (rozšírenie vetvy typu EXP-1) vyplýva, že p je vetva pri konfigurácii (f, u).

(2) Nech p je vetva pri konfigurácii (f, u), taká, že $f(p)$ nie je typu NOSTATE. Mohli nastať dva prípady:

(*) p nie vetva pri konfigurácii (g, u'):
Potom podľa liem 3.3.34 (o rozširovaní vetvy) a 3.3.31 (rozšírenie vetvy typu EXP-1) a podľa indukčného predpokladu platí: $(\text{state}(f(p)), u, \text{mem}(f(p)), \text{pos}(f(p))) = (\text{state}(g(p)), u', \text{mem}(g(p)), \text{pos}(g(p)))$ je dosiahnuteľná konfigurácia stroja M na vstupe w.

(**) p je vetva pri konfigurácii (g, u'):
Podľa liem vyslovených v časti 3.3.3 mohli nastať tieto prípady:

- $g(p) = f(p)$:
  Niet čo dokazovať.

- $g(p) \neq f(p)$, ale $\text{state}(g(p)) = \text{state}(f(p))$, $\text{mem}(g(p)) = \text{mem}(f(p))$ a $\text{pos}(g(p)) = \text{pos}(f(p))$:
  Opäť niet čo dokazovať.
- $g(p)$ bolo typu READY:
  Označme si $\text{pos}(g(p)) = i$, $\text{mem}(g(p)) = xay$ (kde $uv \in \Gamma^*$, $a \in \Gamma$, $|u| = i$), $\text{state}(g(p)) = q$. Nech $z \in \Gamma^*$ je také, že $(q, u', xayz, i)$ je dosiahnuteľná konfigurácia stroja M na vstupe w (také podľa indukčného predpokladu existuje). Nech $(r, b, d) = \delta_M(q, a)$ ak $i > 0$ a $(r, d, a) = \delta(q, ¢c)$ ak $i = 0$ a $u' = cu$.
  Podľa lemy 3.3.19 (čo sa stane s vetvou typu READY) bude $\text{state}(f(p)) = r$, $\text{mem}(f(p)) = xby$ a $\text{pos}(f(p)) = i + d$, teda
  $(\text{state}(f(p)), u, \text{mem}(f(p)), \text{pos}(f(p))) = (r, u, xbyz, i + d)$ je dosiahnuteľná (nasledujúca) konfigurácia stroja M na vstupe w. Navyše ak p je pri konfigurácii (f, u) otvorená, tak zrejme bola otvorená aj pri konfigurácii (g, u'), teda podľa indukčného predpokladu je $z = \varepsilon$.
- $g(p)$ bolo typu EXP-1 a nastal piaty prípad lemy 3.3.30 (čo sa stane s vetvou typu EXP-1):
  Nech z je také, že $(\text{state}(g(p)), u', \text{mem}(g(p))z, \text{pos}(g(p)))$ je dosiahnuteľná konfigurácia stroja M na vstupe w (také z podľa indukčného predpokladu existuje). Potom podľa tvrdenia piateho prípadu v leme 3.3.30 (čo sa stane s vetvou typu EXP-1) je $(\text{state}(f(p)), u, \text{mem}(f(p))az, \text{pos}(f(p))) = (\text{state}(g(p)), u', \text{mem}(g(p))z, \text{pos}(g(p)))$. V tomto prípade zrejme vetva p nebola otvorená pri konfigurácii (g, u'), teda nie je otvorená ani pri konfigurácii (f, u).

□

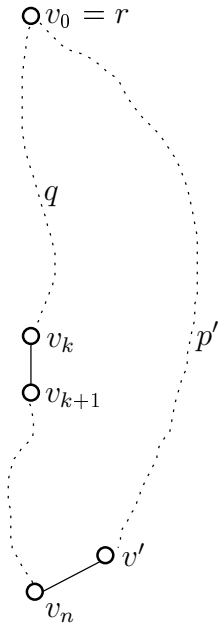
Z lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) vyplýva nasledujúca lema 3.3.36 (o korektnosti) o tom, že naša schéma H nikdy neakceptuje iné slová, než stroj M.

Lema 3.3.36 *(o korektnosti) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, u) je jeho dosiahnuteľná konfigurácia na vstupe w. Nech existuje vrchol v taký, že $f(v) = \varphi$*

Potom $w \in L(M)$.

Dôkaz Vezmime si výpočet, ktorým sa dosiahla konfigurácia (f, u). V ňom nech (g, x) je posledná konfigurácia, v ktorej ešte $\forall v \in V(G) : g(v) \neq \varphi$. A nech (h, y) je prvá konfigurácia v tomto výpočte taká, že $\exists v \in V(G) : h(v) = \varphi$. Teda $(g, x) \vdash_A^S (h, y)$ pre vhodné $S \subseteq V(G)$. Nech v_* je ten vrchol, pre ktorý $h(v_*) = \varphi$.

Pri prechode z (g, x) na (h, y) zrejme vrchol v_* urobil prechod podľa rovnosti 3.28. Teda bolo $g(v_*) \in K_H^B$. Z predpokladu pri rovnosti 3.28 vyplýva, že $\pi_3(g(v_*)) \in F_M$. Podľa lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) vrchol v_* leží na nejakej vetve pri konfigurácii (g, x). Táto vetva nech je p. Z predchádzajúceho vyplýva, že $g(p)$ nie je



Obr. 3.13: k dôkazu lemy 3.3.35, prípad 2°(1)(iii)(**)

typu NOSTATE. Teda podľa lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) je $(\text{state}(g(p)), x, \text{mem}(g(p))z, \text{pos}(g(p)))$ pre vhodné $z \in (\Gamma \cup \{¢, \$)^*\}$ dosiahnuteľná konfigurácia stroja M na vstupe w. Ale $\text{state}(g(p)) = \pi_3(g(v_*)) \in F_M$, teda $w \in L(M)$. □

Lema 3.3.37 *(o očíslovaní) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, u) je jeho dosiahnuteľná konfigurácia. Nech $x, y \in V(G)$ sú susedné a $\pi_1(f(x)) + 1 = \pi_1(f(y))$. Nech p je vetva pri konfigurácii (f, u) končiaca vo vrchole x.*

Potom py je vetva pri konfigurácii (f, u).

Dôkaz Podľa lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) je (f, u) dobrá. Teda aj vrchol y je na konci nejakej vetvy q pri konfigurácii (f, u). Zároveň z dobrého očíslovania vetiev p a q vyplýva, že $|py| \equiv |q| \pmod 3$. Ďalej, keďže p je najkratšia cesta z r do x platí $|p| \leq |q| + 1$, teda $|py| \leq |q| + 2$. A ešte, keďže q je najkratšia cesta z r do y platí $|py| \geq |q|$.

Teda dostávame, že $|py| = |q|$, teda aj py je najkratšia cesta z r do y. Cesta py zrejme spĺňa aj ostatné podmienky definujúce vetvu. □

3.3.5 Nutne dosiahnuté konfigurácie

V predchádzajúcich častiach sme dokázali, že žiadny ACA s prechodovou schémou H, ktorú sme skonštruovali v časti 3.2 nikdy neakceptuje slovo, ktoré neakceptuje stroj M. V tejto

a nasledujúcich častiach budeme dokazovať, že každý (dosť veľký) ACA s prechodovou schémou H skonštruovanou v časti 3.2 vždy akceptuje každé slovo, ktoré akceptuje stroj M. Dokážeme to tak, že ukážeme, že každý (dosť veľký) ACA s prechodovou schémou H pri simulácii stroja M „dosiahne“ každú dosiahnuteľnú konfiguráciu stroja M.

O vrcholoch bude užitočné vedieť, aká až dlhá vetva by cez ne potenciálne mohla prechádzať. Túto maximálnu dĺžku budeme volať dosah vrchola.

Definícia 3.3.38 *(dosah vrchola) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech $v \in V(G)$.*

Dosah vrchola v je najväčšie číslo l také, že existuje vrchol $u \in V(G)$ taký, že nejaká najkratšia cesta z r do u prechádza cez v a má dĺžku l.

Dosah vrchola v budeme značiť $\mathrm{range}(v)$ *(predpokladáme, že ACA A, pre ktorý to uvažujeme bude známy z kontextu).*

Cez vrcholy s príliš malým dosahom teda nikdy nebude môcť prechádzať dlhá vetva. To by mohlo spôsobiť problém ak by si ACA potreboval zapamätať dlhé konfigurácie simulovaného stroja M. Aby tieto problémy nevznikali, je prechodová schéma H (ako neskôr ukážeme) definovaná tak, že vrcholy, o ktorých sa zistí, že majú príliš malý dosah niekedy prejdú do stavu ω. Túto vlastnosť vrcholov voláme *odsúdenosť na zánik*.

Definícia 3.3.39 *(odsúdenosť na zánik) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech $v \in V(G)$.*

Povieme, že vrchol v je odsúdený na zánik na vstupe w pri výpočtom pláne $\mathcal{M}$, ak existuje konfigurácia (g, u) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(v) = \omega$.

O niektorých postupnostiach stavov (reprezentujúcich konfigurácie simulovaného stroja M) ukážeme, že budú *nutne dosiahnuté*. To znamená, že na každej dosť dlhej najkratšej ceste vychádzajúcej z čítajúceho vrchola bude niekedy počas výpočtu táto postupnosť dosiahnutá (teda, že vrcholy na tej ceste budú v tých stavoch).

Definícia 3.3.40 *(nutné dosiahnutie) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech $u \in (K_H^B)^*$. Nech $s \in \Sigma_H^*$.*

Povieme, že $[u, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$, ak sú splnené tieto dve podmienky:

- *Pre každú najkratšiu cestu p z vrchola r takú, že $|p| > |u|$ existuje konfigurácia (g, s) ACA A taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, s)$ a $g(p) = u\beta^{|p|-|u|}$.*
- *Každý vrchol, ktorého dosah je najviac $|u|$ je odsúdený na zánik na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Poznámka 3.3.41 Prvá podmienka v definícii 3.3.40 (nutné dosiahnutie) hovorí o tom, že tá postupnosť stavov s sa dosiahne na každej dosť dlhej ceste vychádzajúcej z r (i keď nie nutne na všetkých naraz). Druhá podmienka hovorí o tom, že vrcholy s príliš malým dosahom niekedy „umrú“.

Niekedy sa výpočet ACA simulujúceho stroj M môže uberať viacerými cestami (v závislosti od výpočtového plánu). V takýchto prípadoch sa nedá hovoriť o nutnom dosiahnutí určitej postupnosti stavov, ale o nutnom dosiahnutí nejakej postupnosti stavov s určitými vlastnosťami. Takéto prípady riešia nasledujúce definície.

Nasledujúca definícia je pre prípad, keď sa posúva hlava simulovaného stroja M o dve políčka doprava (tesne po dokončení simulácie rozširovania pracovnej pásky stroja M). Hovorí o nutnom dosiahnutí postupnosti stavov, v ktorej už hlava odišla z políčka, na ktorom bola, ale ešte nemusela (ale mohla) prijsť na políčko, na ktoré ide.

Definícia 3.3.42 *(nutné dosiahnutie SHIFT-R-1.5[q,u,i],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech $q \in K_M$ a $u \in (\Gamma \cup \{¢, \$\})^*$ a i je prirodzené číslo. Nech $s \in \Sigma_H^*$.*

Povieme, že SHIFT-R-1.5[q,u,i],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$, ak sú splnené tieto dve podmienky:

- *Pre každú najkratšiu cestu p z vrchola r takú, že $|p| > |u|$ existuje konfigurácia (g, s) ACA A taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, s)$ a $g(q)$ je typu SHIFT-R-1 alebo SHIFT-R-2,* $\mathrm{state}(g(q)) = q$, $\mathrm{mem}(g(q)) = u$ *a* $\mathrm{pos}(g(q)) = i$ *a q je otvorená vetva pri konfigurácii (g, s), kde q je prefix p taký, že $|q| = |u|$.*

- *Každý vrchol, ktorého dosah je najviac $|u|$ je odsúdený na zánik na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Nasledujúca definícia hovorí o nutnom dosiahnutí postupnosti stavov, ktorá vyjadruje, že začiatok vlny, ktorá sa šíri pri posúvaní zapamätaného obsahu pásky je aspoň k políčok od čítajúceho vrchola.

Definícia 3.3.43 *(nutné dosiahnutie EXP-1[q,u,i,k],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech $q \in K_M$, $u \in (\Gamma \cup \{¢, \$\})^*$ a i, k sú prirodzené čísla. Nech $s \in \Sigma_H^*$.*

Povieme, že EXP-1[q,u,i,k],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$ ak platí:

- *Pre každú najkratšiu cestu z vrchola r takú, že $|p| > |u| - 1$ existuje konfigurácia (g, s) ACA A taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, s)$ a $g(q)$ je typu EXP-1 a* $\mathrm{state}(g(q)) = q$, $\mathrm{mem}(g(q)) = u$, $\mathrm{pos}(g(q)) = i$ *a* $\mathrm{lead}(g(q)) \geq k$ *a q je otvorená vetva pri konfigurácii (g, s), kde q je prefix p taký, že $|q| = |u| - 1$.*

- *Každý vrchol, ktorého dosah je najviac* $|u|-1$ *je odsúdený na zánik na vstupe* w *pri výpočtovom pláne* $\mathcal{M}$.

Nasledujúca definícia hovorí o nutnom dosiahnutí postupnosti stavov, ktorá vyjadruje, že začiatok vlny, ktorá sa šíri pri posúvaní zapamätaného obsahu pásky je už na konci pásky a koniec tej vlny je aspoň k políčok od čítajúceho vrchola.

Definícia 3.3.44 *(nutné dosiahnutie EXP-2[q,u,i,k],s) Nech* $A = (G, r, H)$ *je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech* (f, w) *je počiatočná konfigurácia ACA A. Nech* $\mathcal{M}$ *je výpočtový plán. Nech* $q \in K_M$, $u \in (\Gamma \cup \{¢, \$\})^*$ *a* i, k *sú prirodzené čísla. Nech* $s \in \Sigma_H^*$.

Povieme, že EXP-2[q,u,i,k],s bude nutne dosiahnuté na vstupe w *pri výpočtovom pláne* $\mathcal{M}$ *ak platí:*

- *Pre každú najkratšiu cestu z vrchola* r *takú, že* $|p| > |u|$ *existuje konfigurácia* (g, s) *ACA A taká, že* $(f, w) \vdash_A^{\mathcal{M}} (g, s)$ *a* $g(q)$ *je typu EXP-2 a* $\mathrm{state}(g(q)) = q$, $\mathrm{mem}(g(q)) = u$, $\mathrm{pos}(g(q)) = i$ *a* $\mathrm{trail}(g(q)) \geq k$ *a* q *je otvorená vetva pri konfigurácii* (g, s), *kde* q *je prefix* p *taký, že* $|q| = |u|$.

- *Každý vrchol, ktorého dosah je najviac* $|u|$ *je odsúdený na zánik na vstupe* w *pri výpočtovom pláne* $\mathcal{M}$.

Nasledujúca definícia hovorí o nutnom dosiahnutí postupnosti stavov, ktorá vyjadruje, že vlna, ktorá sa šíri späť po posunutí zapamätaného obsahu pásky je už najviac k políčok od čítajúceho vrchola.

Definícia 3.3.45 *(nutné dosiahnutie EXP-3[q,u,i,k],s) Nech* $A = (G, r, H)$ *je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech* (f, w) *je počiatočná konfigurácia ACA A. Nech* $\mathcal{M}$ *je výpočtový plán. Nech* $q \in K_M$, $u \in (\Gamma \cup \{¢, \$\})^*$ *a* i, k *sú prirodzené čísla. Nech* $s \in \Sigma_H^*$.

Povieme, že EXP-3[q,u,i,k],s bude nutne dosiahnuté na vstupe w *pri výpočtovom pláne* $\mathcal{M}$ *ak platí:*

- *Pre každú najkratšiu cestu z vrchola* r *takú, že* $|p| > |u|$ *existuje konfigurácia* (g, s) *ACA A taká, že* $(f, w) \vdash_A^{\mathcal{M}} (g, s)$ *a* $g(q)$ *je typu EXP-3 a* $\mathrm{state}(g(q)) = q$, $\mathrm{mem}(g(q)) = u$, $\mathrm{pos}(g(q)) = i$ *a* $\mathrm{dist}(g(q)) \leq k$ *a* q *je otvorená vetva pri konfigurácii* (g, s), *kde* q *je prefix* p *taký, že* $|q| = |u|$.

- *Každý vrchol, ktorého dosah je najviac* $|u|$ *je odsúdený na zánik na vstupe* w *pri výpočtovom pláne* $\mathcal{M}$.

V ďalšom texte vyslovíme a dokážeme lemy o nutnom dosiahnutí niektorých významných postupností stavov. V dôkazoch týchto liem bude treba vyjadriť zložitejšie vzťahy medzi konfiguráciami, ktoré sa vyskytnú počas výpočtu. Na vyjadrenie týchto vzťahov zavedieme nasledujúce označenie, ktoré vysvetlíme v poznámke, ktorá za ním nasleduje.

Označenie 3.3.46 (postupné dosiahnutie konfigurácií počas výpočtu) Nech (G, r, H) je ACA. Nech $\mathcal{M} = \{M_i\}_{i=0}^{\infty}$ je postupnosť podmnožín $V(G)$. Nech $c_0, c_1, c_2, \ldots c_n$ sú konfigurácie A.

Budeme značiť $c_0 \vdash_A^{\mathcal{M}} c_1 \vdash^{t_1} c_2 \vdash^{t_2} \ldots \vdash^{t_{n-1}} c_n$, kde $t_i \in \{1, *, S, \ni v, \not\ni v, [v=], [v\neq] | S \subseteq V(G), v \in V(G)\}$ ak existuje potupnosť konfigurácií $k_0, k_1, \ldots k_m$ a indexov $0 = j_0 \leq j_1 \leq \ldots \leq j_n$ takých, že:

- pre každé $i \in \{0, 1, \ldots, m-1\}$ platí $k_i \vdash_A^{M_i} k_{i+1}$.
- pre každé $i \in \{0, 1, \ldots, n\}$ platí $k_{j_i} = c_i$.
- pre každé $i \in \{0, 1, \ldots, n-1\}$ platí:
  - ak $t_i = 1$, tak $j_{i+1} = j_i + 1$
  - ak $t_i = S$, tak $j_{i+1} = j_i + 1$ a $M_{j_i} = S$
  - ak $t_i = \ni v$, tak $j_{i+1} = j_i + 1$ a $v \in M_{j_i}$
  - ak $t_i = \not\ni v$, tak $v \notin M_j$ pre každé $j \in \{j_i, j_i + 1, j_i + 2, \ldots, j_{i+1} - 1\}$
  - ak $t_i = [v=]$, tak pre každé $j \in \{j_i, j_i + 1, \ldots j_{i+1}\}$ platí $f_j(v) = f_{j_i}(v)$ pre $k_j = (f_j, u_j)$
  - ak $t_i = [v\neq]$, tak $j_{i+1} = j_i + 1$ a $f_{j_i}(v) \neq f_{j_i+1}(v)$ pre $k_j = (f_j, u_j)$.

Poznámka 3.3.47 (k označeniu 3.3.46) Teda $c_0 \vdash_A^{\mathcal{M}} c_1 \vdash^{t_1} c_2 \vdash^{t_2} \ldots \vdash^{t_{n-1}} c_n$ znamená, že A pri výpočtovom pláne $\mathcal{M}$ začínajúc v konfigurácii c_0 postupne (ale nie nutne súvislo po sebe) dosiahol konfigurácie $c_1, \ldots, c_n$. Pričom ak sa použil symbol $\vdash^1$, tak medzi tými konfiguráciami spravil práve jeden krok, ak sa použil symbol $\vdash^*$, tak spravil žiadny, alebo viac krokov, ak sa použil symbol $\vdash^S$, tak spravil jeden krok a počítali práve vrcholy z S, $\vdash^{\ni v}$ znamená, že spravil jeden krok a počítal aj vrchol v, no a $\vdash^{\not\ni v}$ znamená, že spravil žiadny, alebo viac krokov a medzi tým vrchol v nepočítal. $\vdash^{[v\neq]}$ znamená, že spravil jeden krok a vrchol v zmenil stav a $\vdash^{[v=]}$ znamená, že spravil žiadny, alebo viac krokov a medzi tým vrchol v nezmenil svoj stav.

Namiesto $\vdash^1$ budeme písať $\vdash$.

Nasledujúca lema hovorí o tom, že ak nejaký vrchol niekedy dosiahne stav φ, tak tento stav určite niekedy dosiahne aj čítajúci vrchol.

Lema 3.3.48 *(o akceptovaní) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech existuje konfigurácia (g, u) ACA A taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a pre niektorý vrchol $v \in V(G)$ platí $g(v) = \varphi$.*

Potom A akceptuje w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Dokážeme indukciou vzhľadom na vzdialenosť vrchola v od vrchola r. Bez újmy na všeobecnosti nech v je najbližšie k r spomedzi vrcholov v stave φ.

1° Ak $v = r$ tak niet čo dokazovať.

2° Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r to v. (teda $v_0 = r$ a $v_n = v$) Z toho, že $\mathcal{M}$ je spravodlivý vyplýva, že existujú konfigurácie (h_1, u_1) a (h_2, u_2) také, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^{\nexists v_{n-1}} (h_1, u_1) \vdash^{\ni v_{n-1}} (h_2, u_2)$. Z definície H vyplýva, že $h_1(v) = \varphi$ a $h_2(v_{n-1}) = \varphi$ (pri prechode z (h_1, u_1) do (h_2, u_2) vrchol v_{n-1} zrejme urobil prechod podľa rovnosti 3.29).

Teda $(f, w) \vdash_A^{\mathcal{M}} (h_2, u_2)$ a $h_2(v_{n-1}) = \varphi$ a v_{n-1} je k r bližšie, než v. Z toho a indukčného predpokladu vyplýva tvrdenie lemy.

□

Teraz už konečne vyslovíme a dokážeme sľúbené lemy o nutnom dosiahnutí niektorých významných postupností stavov. Význam týchto liem je znázornený na obrázku 3.14, ktorý je na strane 3.14.

Priebeh simulácie posunu hlavy doľava

Nasledujúca lema hovorí, že ak sa nutne dosiahne postupnosť stavov typu SHIFT-L-1, tak sa nutne dosiahne aj postupnosť stavov typu SHIFT-L-2, pričom zapamätaná konfigurácia simulovaného stroja M sa nezmení.

Lema 3.3.49 *(o nutnom dosiahnutí SHIFT-L-2 z SHIFT-L-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in K_H^{B^*}$ je typu SHIFT-L-1. Nech $y \in K_H^{B^*}$ je typu SHIFT-L-2 a* $\mathrm{state}(y) = \mathrm{state}(x)$, $\mathrm{pos}(y) = \mathrm{pos}(x)$, $\mathrm{mem}(y) = \mathrm{mem}(x)$, $\mathrm{len}(y) = \mathrm{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| > |x|$. Nech $q = v_0 v_1 \dots v_{|x|}$. Označme si $i = \mathrm{pos}(x)$.

Z predpokladu vyplýva, že existuje konfigurácia (g, u) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(q) = x\beta$.

Z lemy 3.3.20 (čo sa stane s vetvou typu SHIFT-L-1) vyplýva, že pre každé konfigurácie (g', u') a (g'', u'') také, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^{[v_i =]} (g', u') \vdash^{[v_i \neq]} (g'', u'')$ platí $g''(q) = y\beta$. Teda stačí nám dokázať, že existuje taká konfigurácia (g_*, u_*), že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g_*, u_*)$ a $g(v_i) \neq g_*(v_i)$.

Toto dokážeme sporom: Predpokladajme, že pre každú konfiguráciu (g', u') takú, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g', u')$ je $g'(v_i) = g(v_i)$.

Vezmime si množinu $W \subseteq V(G)$ vrcholov $t \in V(G)$ takých, že $\pi_1(g(t)) = \pi_1(g(v_i)) + 1$. A nech $W_* \subseteq W$ obsahuje práve tie vrcholy $t \in W$, ktoré majú dosah väčší než $|x|$. Pre každé $t \in W_*$ si vezmime nejakú najkratšiu cestu p_t z vrchola r takú, že $v_0 v_1 \dots v_i t$ je prefixom p_t a $|p_t| = |x| + 1$. Podľa lemy 3.3.37 (o očíslovaní) také p_t existujú.

Z predpokladu vyplýva, že pre každé $t \in W_*$ existuje konfigurácia (g_t, u_t) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g_t, u_t)$ a $g_t(p_t) = x\beta$. A pre každé $t \in W - W_*$ existuje konfigurácia (g_t, u_t) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g_t, u_t)$ a $g_t(t) = \omega$.

Vezmime si také $t_* \in W$, pre ktoré tá konfigurácia (g_{t_*}, u_{t_*}) príde najneskôr. Ďalej z toho, že $\mathcal{M}$ je spravodlivý vyplýva, že existujú konfigurácie (g_+, u_+) a (g_*, u_*) také, že $(f, w) \vdash_A^{\mathcal{M}} (g_{t_*}, u_{t_*}) \vdash^{\not\ni v_i} (g_+, u_+) \vdash^{\ni v_i} (g_*, u_*)$.

Z predpokladu dôkazu sporom a lemy 3.3.20 (čo sa stane s vetvou typu SHIFT-L-1) vyplýva, že pre každé $t \in W_*$ je $g_+(p_t) = x\beta$. To teda znamená, že v konfigurácii (g_+, u_+) sú pre vrchol v_i splnené predpoklady pri rovnosti 3.10, resp. 3.11 a navyše vtedy bude vrchol v_i vybraný pre výpočet. Teda $g_*(v_i) \neq g_+(v_i)$, čo je spor.

Tým sme dokázali prvú podmienku nutnej dosiahnuteľnosti $[y, s]$. Tá druhá triviálne vyplýva z predpokladu. □

Nasledujúca lema hovorí, že ak sa nutne dosiahne postupnosť stavov typu SHIFT-L-2, tak sa nutne dosiahne aj postupnosť stavov typu SHIFT-L-3, pričom zapamätaná konfigurácia simulovaného stroja M sa nezmení.

Lema 3.3.50 *(o nutnom dosiahnutí SHIFT-L-3 z SHIFT-L-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-L-2. Nech $y \in (K_H^B)^*$ je typu SHIFT-L-3 a* $\text{state}(y) = \text{state}(x)$, $\text{pos}(y) = \text{pos}(x)$, $\text{mem}(y) = \text{mem}(x)$, $\text{len}(y) = \text{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne m.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne m.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| > |x|$. Nech $q = v_0 v_1 \dots v_{|x|}$. Označme si $i = \text{pos}(x) - 1$.

Z predpokladu vyplýva, že existuje konfigurácia (g, u) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(q) = x\beta$.

Z lemy 3.3.21 (čo sa stane s vetvou typu SHIFT-L-2) vyplýva, že pre každé konfigurácie (g', u') a (g'', u'') také, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^{[v_i =]} (g', u') \vdash^{[v_i \neq]} (g'', u'')$ platí $g''(q) = y\beta$. Teda stačí nám dokázať, že existuje taká konfigurácia (g_*, u_*), že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g_*, u_*)$ a $g(v_i) \neq g_*(v_i)$.

Toto dokážeme sporom: Predpokladajme, že pre každú konfiguráciu (g', u') takú, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g', u')$ je $g'(v_i) = g(v_i)$.

Vezmime si množinu $U \subseteq V(G)$ vrcholov $t \in V(G)$ takých, že $\pi_1(g(t)) = \pi_1(g(v_i)) - 1$. Pre každé $t \in U$ nech q_t je nejaká najkratšia cesta z r to t. Nech $p_t = q_t v_i v_{i+1} \dots v_{|x|}$. Zrejme p_t je najkratšia cesta r t do $v_{|x|}$, teda aj $|p_t| = |x| + 1$.

Z predpokladu vyplýva, že pre každé $t \in U$ existuje konfigurácia (g_t, u_t) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u) \vdash^* (g_t, u_t)$ a $g_t(p_t) = x\beta$.

Vezmime si také $t_* \in U$, pre ktoré tá konfigurácia (g_{t_*}, u_{t_*}) príde najneskôr. Ďalej z toho, že $\mathcal{M}$ je spravodlivý vyplýva, že existujú konfigurácie (g_+, u_+) a (g_*, u_*) také, že $(f, w) \vdash_A^{\mathcal{M}} (g_{t_*}, u_{t_*}) \vdash^{\not\ni v_i} (g_+, u_+) \vdash^{\ni v_i} (g_*, u_*)$.

Z predpokladu dôkazu sporom a lemy 3.3.21 (čo sa stane s vetvou typu SHIFT-L-2) vyplýva, že pre každé $t \in U$ je $g_+(p_t) = x\beta$. To teda znamená, že v konfigurácii (g_+, u_+) sú pre vrchol v_i splnené predpoklady pri rovnosti 3.12 a navyše vtedy bude vrchol v_i vybraný pre výpočet. Teda $g_*(v_i) \neq g_+(v_i)$, čo je spor.

Tým sme dokázali prvú podmienku nutnej dosiahnuteľnosti $[y, s]$. Tá druhá triviálne vyplýva z predpokladu. □

Poznámka 3.3.51 (k lemám 3.3.49 a 3.3.50 a nasledujúcim) Nesledujú ďalšie lemy podobné predchádzajúcim. Budú hovoriť o tom, čo všetko sa nutne podarí dosiahnuť počas výpočtu na každej dosť dlhej najkratšej ceste z r.

Ich dôkaz bude podobný dôkazom predchádzajúcich dvoch liem.

Pre ľubovoľnú dosť dlhú cestu p použitím predpokladu nájdeme konfiguráciu (g, u), pri ktorej sa na p dosiahne stav, akého dosiahnutie sa predpokladá. Potom použitím niektorej z liem vyslovených v časti 3.3.3 sa ukáže, že na to aby sa dosiahol aj stav, ktorý chceme, stačí aby svoj stav zmenil len jeden vrchol v na ceste p. Potom použitím predpokladu na vhodne zvolené cesty prechádzajúce vrcholom v, spravodlivosti výpočtového plánu $\mathcal{M}$ a niektorej z liem vyslovených v časti 3.3.3 sa ukáže, že niekedy vrchol v svoj stav zmeniť musel.

Konštrukciu tých pomocných ciest v dôkaze lemy 3.3.49 (o nutnom dosiahnutí SHIFT-L-2 z SHIFT-L-1) nazveme vetvenie doprava a ich konštrukciu v dôkaze lemy 3.3.50 (o nutnom dosiahnutí SHIFT-L-3 z SHIFT-L-2) nazveme vetvenie doľava.

Vzhľadom na túto podobnosť budú dôkazy nasledujúcich liem zjednodušené.

Nasledujúca lema hovorí, že ak sa nutne dosiahne postupnosť stavov typu SHIFT-L-3, tak sa nutne dosiahne aj postupnosť stavov typu READY, pričom zapamätaná konfigurácia simulovaného stroja M sa nezmení. Hovorí teda, že raz začatá simulácia posunu hlavy doľava sa niekedy určite dokončí.

Lema 3.3.52 *(o nutnom dosiahnutí READY z SHIFT-L-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne m. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-L-3. Nech $y \in (K_H^B)^*$ je typu READY a* $\mathrm{state}(y) = \mathrm{state}(x)$, $\mathrm{pos}(y) = \mathrm{pos}(x)$, $\mathrm{mem}(y) = \mathrm{mem}(x)$, $\mathrm{len}(y) = \mathrm{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $i = \mathrm{pos}(x)$.

Podľa lemy 3.3.22 (čo sa stane s vetvou typu SHIFT-L-3) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-L-3, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.13, resp. 3.14.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.22 (čo sa stane s vetvou typu SHIFT-L-3) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doprava dostaneme dokazované tvrdenie. □

Nasledujú lemy, ktoré podobne ako tie predchádzajúce hovoria o tom, ako prebieha (a že prebieha) simulácia posunu hlavy doprava.

Priebeh simulácie posunu hlavy doprava o dve políčka

Lema 3.3.53 *(o nutnom dosiahnutí SHIFT-RR-2 z SHIFT-RR-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-RR-1. Nech $y \in (K_H^B)^*$ je typu SHIFT-RR-2 a* $\mathrm{state}(y) = \mathrm{state}(x)$, $\mathrm{pos}(y) = \mathrm{pos}(x)$, $\mathrm{mem}(y) = \mathrm{mem}(x)$, $\mathrm{len}(y) = \mathrm{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $i = \mathrm{pos}(x) - 1$.

Podľa lemy 3.3.27 (čo sa stane s vetvou typu SHIFT-RR-1) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-RR-1, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.15.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.27 (čo sa stane s vetvou typu SHIFT-RR-1) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doprava dostaneme dokazované tvrdenie. □

Lema 3.3.54 *(o nutnom dosiahnutí SHIFT-R-1.5[q,u,i],s z SHIFT-RR-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-RR-2. Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj SHIFT-R-1.5[$\mathrm{state}(x)$,$\mathrm{mem}(x)$,$\mathrm{pos}(x)$*],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $i = \mathrm{pos}(x) - 2$.

Podľa lemy 3.3.28 (čo sa stane s vetvou typu SHIFT-RR-2) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-RR-2, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.16.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.28 (čo sa stane s vetvou typu SHIFT-RR-2) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doprava dostaneme dokazované tvrdenie. □

Priebeh simulácie posunu hlavy doprava

Lema 3.3.55 *(o nutnom dosiahnutí SHIFT-R-1.5[q,u,i],s z SHIFT-R-1) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-R-1. Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj SHIFT-R-1.5[$\mathrm{state}(x)$,$\mathrm{mem}(x)$,$\mathrm{pos}(x)$*],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Dôkaz Triviálne vyplýva priamo z definície. □

Lema 3.3.56 *(o nutnom dosiahnutí SHIFT-R-2 z SHIFT-R-1.5[q,u,i],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma \cup \{¢, \$\})^*$ a i je prirodzené číslo. Nech $s \in \Sigma_H$. Nech SHIFT-R-1.5[q,u,i],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$. Nech $y \in (K_H^B)^*$ je typu SHIFT-R-2a* $\mathrm{state}(y) = q$, $\mathrm{mem}(y) = u$ *a* $\mathrm{pos}(y) = i$.

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $p' = v_0 v_1 \dots v_{|x|-1}$ Podľa predpokladu existuje konfigurácia (g, u) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(v_{|x|}) = \beta$ a $g(p')$ je typu SHIFT-R-1 alebo SHIFT-R-2 a $\mathrm{state}(p') = q$, $\mathrm{mem}(p') = u$ a $\mathrm{pos}(p') = i$. Ak $g(p')$ je typu SHIFT-R-2, tak niet čo dokazovať.

Inak podľa lemy 3.3.24 (čo sa stane s vetvou typu SHIFT-R-1) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-R-1, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.17.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.24 (čo sa stane s vetvou typu SHIFT-R-1) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doľava dostaneme dokazované tvrdenie. □

Lema 3.3.57 *(o nutnom dosiahnutí SHIFT-R-3 z SHIFT-R-2) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-R-2. Nech $y \in (K_H^B)^*$ je typu SHIFT-R-3 a* $\mathrm{state}(y) = \mathrm{state}(x)$, $\mathrm{pos}(y) = \mathrm{pos}(x)$, $\mathrm{mem}(y) = \mathrm{mem}(x)$, $\mathrm{len}(y) = \mathrm{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $i = \mathrm{pos}(x) - 1$.

Podľa lemy 3.3.25 (čo sa stane s vetvou typu SHIFT-R-2) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-R-2, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.18, resp. 3.19.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.25 (čo sa stane s vetvou typu SHIFT-R-2) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doprava aj doľava dostaneme dokazované tvrdenie. □

Lema 3.3.58 *(o nutnom dosiahnutí READY z SHIFT-R-3) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $s \in \Sigma_H$. Nech $x \in (K_H^B)^*$ je typu SHIFT-R-3. Nech $y \in (K_H^B)^*$ je typu READY a* $\mathrm{state}(y) = \mathrm{state}(x)$, $\mathrm{pos}(y) = \mathrm{pos}(x)$, $\mathrm{mem}(y) = \mathrm{mem}(x)$, $\mathrm{len}(y) = \mathrm{len}(x)$ *(týmito podmienkami je y zrejme jednoznačne určené). Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $i = \mathrm{pos}(x)$.

Podľa lemy 3.3.26 (čo sa stane s vetvou typu SHIFT-R-3) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav SHIFT-R-3, vrchol v_i zmení svoj stav, a to podľa rovnosti 3.20.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.26 (čo sa stane s vetvou typu SHIFT-R-3) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doľava dostaneme dokazované tvrdenie. □

Priebeh simulácie rozširovania pracovnej pásky

Nasledujúca lema hovorí o tom, že pri posune obsahu pracovnej pásky simulovaného stroja M sa začiatok tej „posúvacej“ vlny šíri (teda, že sa niekde nezasekne).

Lema 3.3.59 *(o nutnom dosiahnutí EXP-1[q,u,i,k + 1],s z EXP-1[q,u,i,k],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$, i a k sú prirodzené čísla, $k < |u| - 2$. Nech $s \in \Sigma_H$. Nech EXP-1[q,u,i,k],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj EXP-1[q,u,i,k+1],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| > |u| - 1$. Nech $p' = v_0 v_1 \dots v_{|u|-2}$. Podľa predpokladu existuje konfigurácia (g, x) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, x)$ a p' je pri (g, x) otvorená vetva a $g(p')$ je typu EXP-1, $\mathrm{state}(g(p')) = q$, $\mathrm{mem}(g(p')) = u$, $\mathrm{pos}(g(p')) = i$ a $\mathrm{lead}(g(p')) \geq k$. Ak dokonca je $\mathrm{lead}(g(p')) > k$, tak niet čo dokazovať.

Predpokladajme teda, že $\mathrm{lead}(g(p')) = k$. Podľa lemy 3.3.30 (čo sa stane s vetvou typu EXP-1) nám treba dokázať, že ešte aj niekedy po dosiahnutí konfigurácie (g, x) vrchol v_{k+1} zmení svoj stav, a to podľa rovnosti 3.21.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.30 (čo sa stane s vetvou typu EXP-1) na cesty prechádzajúce vrcholom v_{k+1} zostrojené vetvením doľava dostaneme dokazované tvrdenie. □

Nasledujúca lema hovorí o tom, že pri posune obsahu pracovnej pásky simulovaného stroja M sa koniec tej „posúvacej“ vlny šíri (teda, že sa niekde nezasekne).

Lema 3.3.60 *(o nutnom dosiahnutí EXP-2[q,u,i,k+1],s z EXP-2[q,u,i,k],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$, i a k sú prirodzené čísla, $k < |u| - 2$. Nech $s \in \Sigma_H$. Nech EXP-2[q,u,i,k],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj EXP-2[q,u,i,k+1],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| > |u|$. Nech $p' = v_0 v_1 \dots v_{|u|-1}$. Podľa predpokladu existuje konfigurácia (g, x) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, x)$ a p' je pri (g, x) otvorená vetva a $g(p')$ je typu EXP-2, $\mathrm{state}(g(p')) = q$, $\mathrm{mem}(g(p')) = u$, $\mathrm{pos}(g(p')) = i$ a $\mathrm{trail}(g(p')) \geq k$. Ak dokonca je $\mathrm{trail}(g(p')) > k$, tak niet čo dokazovať.

Predpokladajme teda, že $\mathrm{trail}(g(p')) = k$. Podľa lemy 3.3.32 (čo sa stane s vetvou typu EXP-2) nám treba dokázať, že ešte aj niekedy po dosiahnutí konfigurácie (g, x) vrchol v_k zmení svoj stav, a to podľa rovnosti 3.23, resp. 3.24.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.32 (čo sa stane s vetvou typu EXP-2) na cesty prechádzajúce vrcholom v_k zostrojené vetvením doprava i doľava dostaneme dokazované tvrdenie. □

Nasledujúca lema hovorí o tom, že po posune obsahu pracovnej pásky simulovaného stroja M sa tá „spätná“ vlna šíri (teda, že sa niekde nezasekne).

Lema 3.3.61 *(o nutnom dosiahnutí EXP-3[q,u,i,k−1],s z EXP-3[q,u,i,k],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$, i a k sú prirodzené čísla, $k > 1$. Nech $s \in \Sigma_H$. Nech EXP-2[q,u,i,k],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj EXP-2[q,u,i,k−1],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| > |u|$. Nech $p' = v_0 v_1 \dots v_{|u|-1}$. Podľa predpokladu existuje konfigurácia (g, x) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, x)$ a p' je pri (g, x) otvorená vetva a $g(p')$ je typu EXP-3, $\mathrm{state}(g(p')) = q$, $\mathrm{mem}(g(p')) = u$, $\mathrm{pos}(g(p')) = i$ a $\mathrm{dist}(g(p')) \leq k$. Ak dokonca je $\mathrm{dist}(g(p')) < k$, tak niet čo dokazovať.

Predpokladajme teda, že $\mathrm{dist}(g(p')) = k$. Podľa lemy 3.3.33 (čo sa stane s vetvou typu EXP-3) nám treba dokázať, že ešte aj niekedy po dosiahnutí konfigurácie (g, x) vrchol v_{k-1} zmení svoj stav, a to podľa rovnosti 3.26, resp. 3.27.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.33 (čo sa stane s vetvou typu EXP-3) na cesty prechádzajúce vrcholom v_{k-1} zostrojené vetvením doprava i doľava dostaneme dokazované tvrdenie. □

Nasledujúca lema hovorí o dokončení posúvania obsahu pracovnej pásky simulovaného stroja M po tom, čo tá „posúvacia“ vlna dorazí na koniec.

Lema 3.3.62 *(o nutnom dosiahnutí EXP-2[q,u,i,0],s z EXP-1[q,u,i,|u|−2],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$, i je prirodzené číslo. Nech $s \in \Sigma_H$. Nech EXP-1[q,u,i,|u|−2],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj EXP-2[q,u,i,0],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Tentoraz potrebujeme dokázať obe časti v definícii nutného dosiahnutia.

- Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r taká, že $|p| \geq |u|$. Nech $p' = v_0 v_1 \dots v_{|u|-2}$ a nech $j = |u| - 1$. Podľa predpokladu existuje konfigurácia (g, x) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, x)$ a p' je pri (g, x) otvorená vetva, $\mathrm{state}(g(p')) = q$, $\mathrm{pos}(g(p')) = i$, $\mathrm{mem}(g(p')) = u$ a $\mathrm{lead}(g(p')) = |u| - 2$.

  Nech $W \subseteq V(G)$ obashuje práve tie vrcholy $t \in V(G)$, ktoré sú k r bližšie než v_j. Podľa predpokladu pre každé $t \in W$ existuje konfigurácia (g_t, x_t) taká, že p_t je pri konfigurácii (g_t, x_t) otvorená vetva a $\mathrm{state}(g(p_t)) = q$, $\mathrm{pos}(g(p_t)) = i$, $\mathrm{mem}(g(p_t)) = u$ a $\mathrm{lead}(g(p_t)) = |u| - 2$.

  Nech $t_* \in W$ je také, že konfigurácia (g_{t_*}, x_{t_*}) bude dosiahnutá najneskôr. Zrejme existujú konfigurácie (g_+, x_+) a (g_*, x_*) také, že $(f, w) \vdash_A^{\mathcal{M}} (g_{t_*}, x_{t_*}) \vdash^{\not\ni v_j} (g_+, x_+) \vdash^{\ni v_j} (g_*, x_*)$.

  Ľahko sa overí, že podľa lemy 3.3.30 (čo sa stane s vetvou typu EXP-1) a lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) sú v konfigurácii (g_+, x_+) splnené predpoklady pri rovnosti 3.22. Teda bude $g_*(p'v_j)$ typu EXP-2, $\mathrm{state}(g_*(p'v_j)) = q$, $\mathrm{pos}(g_*(p'v_j)) = i$, $\mathrm{mem}(g_*(p'v_j)) = u$ a $\mathrm{trail}(g_*(p'v_j)) \geq 0$.

  Zrejme ak bolo dokonca $|p| > |u|$, tak bude aj $p'v_j$ otvorená vetva pri konfigurácii (g_*, x_*)

- Nech v je vrchol, ktorého dosah je najviac $|u|$. Ak $\mathrm{range}(v) < |u|$, tak tvrdenie vyplýva priamo z predpokladu. Nech teda $\mathrm{range}(v) = |u|$. Tvrdenie dokážeme indukciou vzhľadom na hodnotu $|u|-$(vzdialenosť v od r).

  1° vzdialenosť v od r je $|u|$.
  Z predchádzajúcich úvah vyplýva, že existuje konfigurácia (g, x) taká, že $(f, w) \vdash_A^{\mathcal{M}} (g, x)$ a $g(v') \in K_H^B$ pre každé $v' \in V(G)$ také, že v' je sused v. Z lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) ale vyplýva, že v tejto konfigurácii budú splnené predpoklady pri rovnosti 3.30.

Teda z toho, že $\mathcal{M}$ je spravodlivý vyplýva, že vrchol v niekedy prejde do stavu ω.

2° Nech $w \subseteq V(G)$ je množina tých susedov v, ktorí sú od r ďalej než v. podľa indukčného predpokladu zrejme každé $t \in W$ niekedy prejde do stavu ω. Podľa predchádzajúcich úvah vrchol v niekedy prejde do stavu z K_H^B.

Z lemy 3.3.35 (o štruktúre dosiahnuteľných konfigurácií) ľahko vyplýva, že potom budú pre vrchol v splnené predpoklady pri rovnosti 3.30.

□

Nasledujúca lema hovorí o odrazení tej „posúvacej“ vlny späť.

Lema 3.3.63 *(o nutnom dosiahnutí EXP-3[q,u,i,|u|−1],s z EXP-2[q,u,i,|u|−1],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$, i je prirodzené číslo. Nech $s \in \Sigma_H$. Nech EXP-2[q,u,i,|u| − 1],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.*

Potom aj EXP-3[q,u,i,|u| − 1],s bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |u|$. Nech $j = |u| - 1$.

Podľa lemy 3.3.32 (čo sa stane s vetvou typu EXP-2) nám treba dokázať, že potom, čo vrcholy na ceste p dosiahnú[5] také stavy, že $g(p)$ bude typu EXP-2 s $\mathrm{trail}(g(p)) = |u| - 1$, vrchol v_j zmení svoj stav, a to podľa rovnosti 3.25.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.32 (čo sa stane s vetvou typu EXP-2) na cesty prechádzajúce vrcholom v_i zostrojené vetvením doľava dostaneme dokazované tvrdenie. □

Nasledujúca lema hovorí o návrate „spätnej“ vlny a dokončení posúvania obsahu pracovnej pásky.

Lema 3.3.64 *(o nutnom dosiahnutí READY, SHIFT-R-1 alebo SHIFT-RR-1 z EXP-3[q,u,i,1],s) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in (\Gamma_M \cup \{¢, \$\})^*$ a i je prirodzené číslo. Nech $s \in \Sigma_H$. Nech EXP-3[q,u,i,1],s je nutne dosiahnuteľné pri výpočtovom pláne $\mathcal{M}$.*

Nech pre $y \in K_H^{B^*}$ platí $\mathrm{state}(y) = q$, $\mathrm{pos}(y) = i$, $\mathrm{mem}(y) = u$ *a y je typu READY ak $i = 0$, typu SHIFT-R-1 ak $i = 1$ a typu SHIFT-RR-1 ak $i = 2$. (týmito podmienkami je y zrejme jednoznačne určené).*

Potom aj $[y, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

[5] pričom (g, x) je tá konfigurácia, v ktorej sa to stane

Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$.

Podľa lemy 3.3.33 (čo sa stane s vetvou typu EXP-3) nám treba dokázať, že potom, čo sa na ceste p dosiahne stav EXP-3 s $\mathrm{dist}(g(p)) = 1$, vrchol v_0 zmení svoj stav, a to podľa rovnosti 3.27.

Použitím predpokladu, spravodlivosti výpočtového plánu $\mathcal{M}$ a lemy 3.3.33 (čo sa stane s vetvou typu EXP-3) na cesty prechádzajúce vrcholom v_0 zostrojené vetvením doprava dostaneme dokazované tvrdenie. □

Priebeh simulácie kroku výpočtu

Nasledujúca lema hovorí o tom, že ak bude nutne dosiahnutá postupnosť stavov typu READY, v ktorej je zaznamenaná nejaká konfigurácia simulovaného stroja M, tak bude nutne dosiahnutá aj postupnosť stavov, v ktorej je zaznamenaná nasledujúca konfigurácia simulovaného stroja M.

Lema 3.3.65 *(o nutnom dosiahnutí niečoho z READY) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$.*

Nech $q \in K_M$, $u_1, u_2 \in (\Gamma_M \cup \{¢, \$\})^*$, $a \in \Gamma_M \cup \{¢, \$\}$ a $i = |u_1|$, pričom $u_1 a u_2 \in ¢\Gamma_M^*\$$. Nech $s = s_0 s_* \in \Sigma_H^*$, $s_0 \in \Sigma_H \cup \{\varepsilon\}$.

Nech $x \in K_H^{B^*}$ je typu READY a $\mathrm{state}(x) = q$, $\mathrm{pos}(x) = i$, $\mathrm{mem}(x) = u_1 a u_2$.

Nech $(q', b, d) = \delta_M(q, a)$ ak $a \neq ¢$ a $(q', b, d) = \delta(q, ¢s_0)$ ak $a = ¢$. Nech $y \in K_H^{B^*}$ a $\mathrm{state}(x) = q'$, $\mathrm{pos}(x) = i + d$, $\mathrm{mem}(x) = u_1 b u_2$. Nech $s' = s$ ak $a \neq ¢$ a $s' = s_*$ ak $a = ¢$.

Nech y je typu READY ak $i > 0$, $d = 0$, typu SHIFT-R-1 ak $i > 0$, $d = 1$, typu SHIFT-L-1 ak $i > 0$, $d = -1$ a typu EXP-1 ak $i = 0$. Ak $i = 0$ tak nech ešte $\mathrm{lead}(y) = \mathrm{trail}(y) = 0$.

Nech $[x, s]$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Potom aj $[y, s']$ bude nutne dosiahnuté na vstupe w pri výpočtovom pláne $\mathcal{M}$.

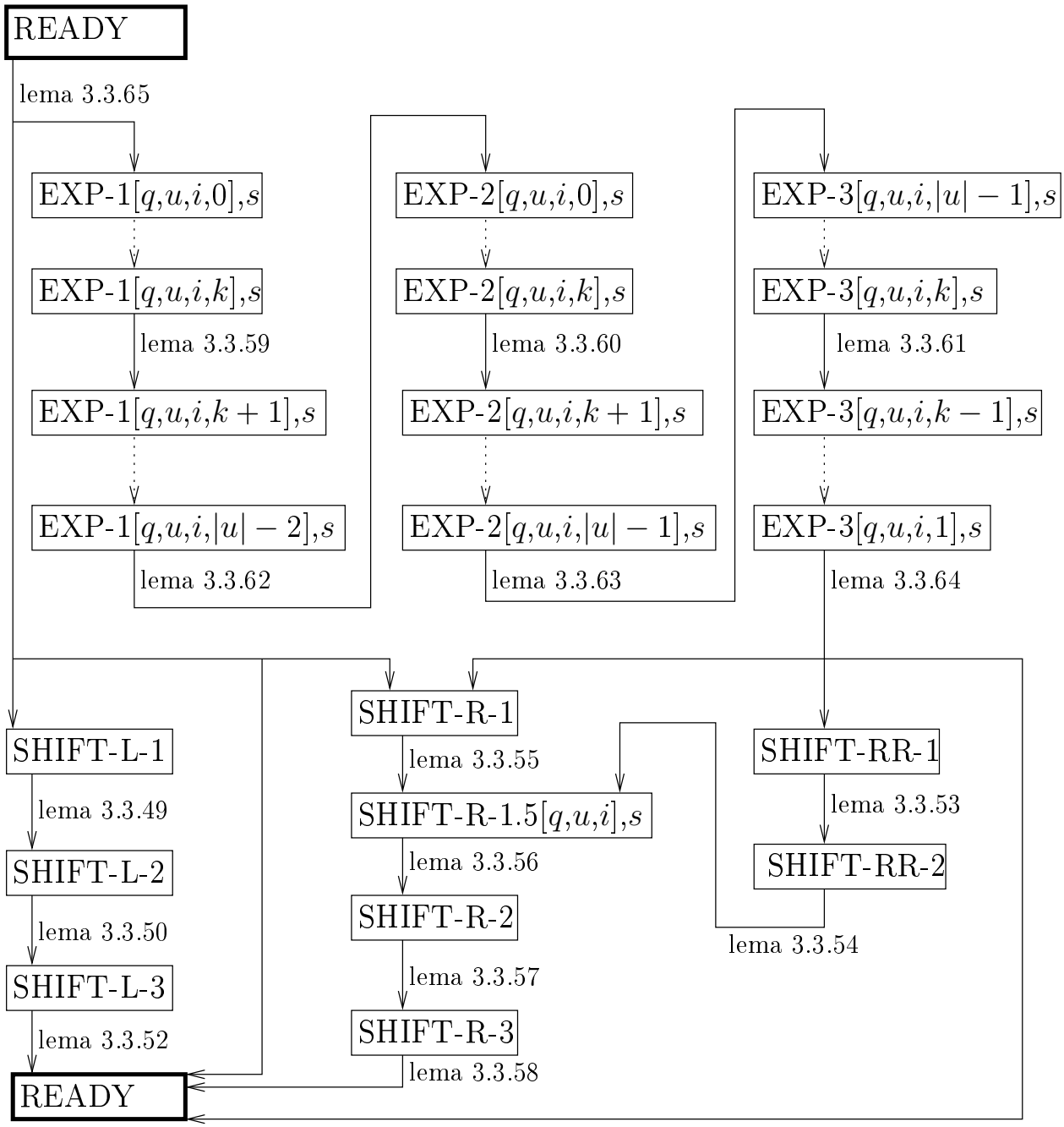
Dôkaz Nech $p = v_0 v_1 \dots v_n$ je najkratšia cesta z r taká, že $|p| > |x|$. Nech $p' = v_0 v_1 \dots v_{|x|-1}$

Podľa predpokladu existuje konfigurácia (g, z) taká, že $g(p') = x$. Zrejme existujú konfigurácie (g_+, z_+) a (g_*, z_*) také, že $(f, w) \vdash_A^{\mathcal{M}} (g, z) \vdash^{\not\ni v_i} (g_+, z_+) \vdash^{\ni v_i} (g_*, z_*)$.

Ľahko sa nahliadne, že podľa lemy 3.3.19 (čo sa stane s vetvou typu READY) bude $g_*(p') = y$. □

Nasledujúca lema hovorí o tom, že ak bude nutne dosiahnutá postupnosť stavov typu READY, v ktorej je zaznamenaná nejaká konfigurácia simulovaného stroja M, tak bude nutne dosiahnutá aj postupnosť stavov typu READY, v ktorej je zaznamenaná nasledujúca konfigurácia simulovaného stroja M.

Lema 3.3.66 *(o nutnom dosiahnutí nasledujúcej konfigurácie) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$.*



Obr. 3.14: Význam liem o nutnom dosiahnutí vyslovených v časti 3.3.5.

Nech $s, t \in ¢\Gamma_M^*\$$ *a* $x, y \in K_H^{B^*}$ *sú typu READY. Nech* $(\text{state}(x), s, \text{mem}(x), \text{pos}(x)) \vdash_M (\text{state}(y), t, \text{mem}(y), \text{pos}(y))$. *Nech* $[x, s]$ *bude nutne dosiahnuté na vstupe* w *pri výpočtovom pláne* $\mathcal{M}$.

Potom aj $[y, t]$ *bude nutne dosiahnuté na vstupe* w *pri výpočtovom pláne* $\mathcal{M}$.

Dôkaz Môžu nastať tieto prípady:

- $\text{pos}(x) = 0$:
  Podľa lemy 3.3.65 (o nutnom dosiahnutí niečoho z READY) bude nutne dosiahnuté EXP-1[state(y),mem(y),pos(y),0],t.

  Potom podľa lemy 3.3.59 (o nutnom dosiahnutí EXP-1[q,u,i,k+1],s z EXP-1[q,u,i,k],s) bude nutne doisahnuté EXP-1[state(y),mem(y),pos(y),$|\text{mem}(y)| - 2$],t. [6]

  Potom podľa lemy 3.3.62 (o nutnom dosiahnutí EXP-2[q,u,i,0],s z EXP-1[q,u,i,$|u| - 2$],s) bude nutne dosiahnuté EXP-2[state(y),mem(y),pos(y),0],t.

  Potom podľa lemy 3.3.60 (o nutnom dosiahnutí EXP-2[q,u,i,k+1],s z EXP-2[q,u,i,k],s) bude nutne doisahnuté EXP-2[state(y),mem(y),pos(y),$|\text{mem}(y)| - 1$],t.

  Potom podľa lemy 3.3.63 (o nutnom dosiahnutí EXP-3[q,u,i,$|u|-1$],s z EXP-2[q,u,i,$|u|-1$],s) bude nutne doisahnuté EXP-3[state(y),mem(y),pos(y),$|\text{mem}(y)| - 1$],t.

  Potom podľa lemy 3.3.61 (o nutnom dosiahnutí EXP-3[q,u,i,$k-1$],s z EXP-3[q,u,i,k],s) bude nutne doisahnuté EXP-3[state(y),mem(y),pos(y),1],t.

  Ďalej môžu nastať tieto prípady:

  - $\text{pos}(y) = 0$:
    Potom podľa lemy 3.3.64 (o nutnom dosiahnutí READY, SHIFT-R-1 alebo SHIFT-RR-1 z EXP-3[q,u,i,1],s) bude nutne dosiahnuté $[y, t]$.
  - $\text{pos}(y) = 1$:
    Potom podľa lemy 3.3.64 (o nutnom dosiahnutí READY, SHIFT-R-1 alebo SHIFT-RR-1 z EXP-3[q,u,i,1],s) bude nutne dosiahnuté $[x_1, t]$, kde x_1 je typu SHIFT-R-1 a $\text{state}(x_1) = \text{state}(y)$, $\text{mem}(x_1) = \text{mem}(y)$ a $\text{pos}(x_1) = \text{pos}(y)$.

    Potom podľa lemy 3.3.55 (o nutnom dosiahnutí SHIFT-R-1.5[q,u,i],s z SHIFT-R-1) bude nutne dosiahnuté SHIFT-R-1.5[state(y),mem(y),pos(y)],t.

    Potom podľa lemy 3.3.56 (o nutnom dosiahnutí SHIFT-R-2 z SHIFT-R-1.5[q,u,i],s) bude nutne dosiahnuté $[x_2, t]$, kde x_2 je typu SHIFT-R-2 a $\text{state}(x_2) = \text{state}(y)$, $\text{mem}(x_2) = \text{mem}(y)$ a $\text{pos}(x_2) = \text{pos}(y)$.

    Potom podľa lemy 3.3.57 (o nutnom dosiahnutí SHIFT-R-3 z SHIFT-R-2) bude nutne dosiahnuté $[x_3, t]$, kde x_3 je typu SHIFT-R-3 a $\text{state}(x_3) = \text{state}(y)$, $\text{mem}(x_3) = \text{mem}(y)$ a $\text{pos}(x_3) = \text{pos}(y)$.

[6] Toto sa dá jednoducho dokázať indukciou, ktorej báza je nutné dosiahnutie EXP-1[state(y),mem(y),pos(y),0],t a indukčný krok je lema 3.3.59. Podobne je tomu aj v ostatných prípadoch použitia liem 3.3.59, 3.3.60 a 3.3.61 v tomto dôkaze.

Potom podľa lemy 3.3.58 (o nutnom dosiahnutí READY z SHIFT-R-3) bude nutne dosiahnuté $[y, t]$.

– $\mathrm{pos}(y) = 2$:
Potom podľa lemy 3.3.64 (o nutnom dosiahnutí READY, SHIFT-R-1 alebo SHIFT-RR-1 z EXP-3[q,u,i,1],s) bude nutne dosiahnuté $[x_1, t]$, kde x_1 je typu SHIFT-RR-1 a $\mathrm{state}(x_1) = \mathrm{state}(y)$, $\mathrm{mem}(x_1) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_1) = \mathrm{pos}(y)$.

Potom podľa lemy 3.3.53 (o nutnom dosiahnutí SHIFT-RR-2 z SHIFT-RR-1) bude nutne dosiahnuté $[x_2, t]$, kde x_1 je typu SHIFT-RR-2 a $\mathrm{state}(x_1) = \mathrm{state}(y)$, $\mathrm{mem}(x_1) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_1) = \mathrm{pos}(y)$.

Potom podľa lemy 3.3.54 (o nutnom dosiahnutí SHIFT-R-1.5[q,u,i],s z SHIFT-RR-2) bude nutne dosiahnuté SHIFT-R-1.5[$\mathrm{state}(y)$,$\mathrm{mem}(y)$,$\mathrm{pos}(y)$],t.

Potom podľa lemy 3.3.56 (o nutnom dosiahnutí SHIFT-R-2 z SHIFT-R-1.5[q,u,i],s) bude nutne dosiahnuté $[x_3, t]$, kde x_2 je typu SHIFT-R-2 a $\mathrm{state}(x_3) = \mathrm{state}(y)$, $\mathrm{mem}(x_3) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_3) = \mathrm{pos}(y)$.

Potom podľa lemy 3.3.57 (o nutnom dosiahnutí SHIFT-R-3 z SHIFT-R-2) bude nutne dosiahnuté $[x_4, t]$, kde x_3 je typu SHIFT-R-3 a $\mathrm{state}(x_4) = \mathrm{state}(y)$, $\mathrm{mem}(x_4) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_4) = \mathrm{pos}(y)$.

Potom podľa lemy 3.3.58 (o nutnom dosiahnutí READY z SHIFT-R-3) bude nutne dosiahnuté $[y, t]$.

- $\mathrm{pos}(x) > 0$, $\mathrm{pos}(y) = \mathrm{pos}(x)$:
Podľa lemy 3.3.65 (o nutnom dosiahnutí niečoho z READY) bude nutne dosiahnuté $[y, t]$.

- $\mathrm{pos}(x) > 0$, $\mathrm{pos}(y) = \mathrm{pos}(x) - 1$:
Podľa lemy 3.3.65 (o nutnom dosiahnutí niečoho z READY) bude nutne dosiahnuté $[x_1, t]$, kde x_1 je typu SHIFT-L-1 a $\mathrm{state}(x_1) = \mathrm{state}(y)$, $\mathrm{mem}(x_1) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_1) = \mathrm{pos}(y)$.

  Potom podľa lemy 3.3.49 (o nutnom dosiahnutí SHIFT-L-2 z SHIFT-L-1) bude nutne dosiahnuté $[x_2, t]$, kde x_2 je typu SHIFT-L-2 a $\mathrm{state}(x_2) = \mathrm{state}(y)$, $\mathrm{mem}(x_2) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_2) = \mathrm{pos}(y)$.

  Potom podľa lemy 3.3.50 (o nutnom dosiahnutí SHIFT-L-3 z SHIFT-L-2) bude nutne dosiahnuté $[x_3, t]$, kde x_3 je typu SHIFT-L-3 a $\mathrm{state}(x_3) = \mathrm{state}(y)$, $\mathrm{mem}(x_3) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_3) = \mathrm{pos}(y)$.

  Potom podľa lemy 3.3.52 (o nutnom dosiahnutí READY z SHIFT-L-3) bude nutne dosiahnuté $[y, t]$.

- $\mathrm{pos}(x) > 0$, $\mathrm{pos}(y) = \mathrm{pos}(x) + 1$:
Podľa lemy 3.3.65 (o nutnom dosiahnutí niečoho z READY) bude nutne dosiahnuté $[x_1, t]$, kde x_1 je typu SHIFT-R-1 a $\mathrm{state}(x_1) = \mathrm{state}(y)$, $\mathrm{mem}(x_1) = \mathrm{mem}(y)$ a $\mathrm{pos}(x_1) = \mathrm{pos}(y)$.

Potom podľa lemy 3.3.55 (o nutnom dosiahnutí SHIFT-R-1.5[q,u,i],s z SHIFT-R-1) bude nutne dosiahnuté SHIFT-R-1.5[state(y),mem(y),pos(y)],t.

Potom podľa lemy 3.3.56 (o nutnom dosiahnutí SHIFT-R-2 z SHIFT-R-1.5[q,u,i],s) bude nutne dosiahnuté $[x_2, t]$, kde x_2 je typu SHIFT-R-2 a state(x_2) = state(y), mem(x_2) = mem(y) a pos(x_2) = pos(y).

Potom podľa lemy 3.3.57 (o nutnom dosiahnutí SHIFT-R-3 z SHIFT-R-2) bude nutne dosiahnuté $[x_3, t]$, kde x_3 je typu SHIFT-R-3 a state(x_3) = state(y), mem(x_3) = mem(y) a pos(x_3) = pos(y).

Potom podľa lemy 3.3.58 (o nutnom dosiahnutí READY z SHIFT-R-3) bude nutne dosiahnuté $[y, t]$.

□

Nasledujúca lema je dôsledkom predchádzajúcej a hovorí, že každý ACA s prechodovou schémou H „objaví“ každú dosiahnuteľnú konfiguráciu simulovaného stroja M (ak skôr neakceptuje).

Lema 3.3.67 *(o nutnom dosiahnutí každej konfigurácie) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech (f, w) je počiatočná konfigurácia ACA A. Nech $\mathcal{M}$ je výpočtový plán. Nech A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech $q \in K_M$, $u \in \Sigma_M$, $v \in \text{¢}\Gamma_M^*\$$, i je prirodzené číslo. Nech $x \in (K_H^B)^*$ je také, že x je typu READY a* state(x) = q, mem(x) = v *a* pos(x) = i. *Nech (q, u, v, i) je dosiahnuteľná konfigurácia stroja M na vstupe w.*

Potom $[x, u]$ bude nutne dosiahnuté automatom A na vstupe w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Dokážeme indukciou vzhľadom na dĺžku výpočtu, ktorým sa dosiahla konfigurácia (q, u, v, i).

1° Ak (q, u, v, i) je počiatočná konfigurácia, tak $q = q_M$, $u = w$, $v = \text{¢}\$$ a $i = 0$. Teda $x = [0, \text{¢}, q_M, \beta, \beta][1, \$, \beta, \beta, \beta]$. Ľahko sa však nahliadne, že $[x, u]$ bude dosiahnuté tesne po inicializácii zariadenia (teda v prvej dobrej konfigurácii).

2° Nech $(q'u'v'i') \vdash_M (q, u, v, i)$. Nech $x' \in (K_H^B)^*$ je typu READY a state(x') = q, mem(x') = v a pos(x') = i. Podľa indukčného predpokladu bude nutne dosiahnuté $[x', u']$ a podľa lemy 3.3.66 (o nutnom dosiahnutí nasledujúcej konfigurácie) bude nutne dosiahnuté aj $[x, u]$.

□

Z predchádzajúcej lemy zase vyplýva nasledujúca, ktorá hovorí, že každý ACA s prechodovou schémou H, v ktorom sú dostatočne dlhé najkratšie cesty vychádzajúce z čítajúceho vrchola akceptuje každé slovo z jazyka $L(M)$.

Lema 3.3.68 *(o úplnosti) Nech $A = (G, r, H)$ je ACA s prechodovou schémou, ktorú sme skonštruovali v časti 3.2. Nech $\mathcal{M}$ je výpočtový plán. Nech M akceptuje slovo w a pri tom použije najviac k políčok pracovnej pásky. Nech v G existuje najkratšia cesta z vrchola r, ktorej dĺžka je aspoň $k + 3$.*

Potom A akceptuje w pri výpočtovom pláne $\mathcal{M}$.

Dôkaz Dokážeme sporom. Predpokladajme, že A neakceptuje w pri výpočtovom pláne $\mathcal{M}$. Nech (f, w) je počiatočná konfigurácia ACA A.

Z toho, že $w \in L(M)$ vyplýva, že existuje konfigurácia (q, u, v, i) stroja M taká, že $q \in F_M$ a (q, u, v, i) je dosiahnuteľná konfigurácia stroja M na vstupe w. Teda podľa lemy 3.3.67 (o nutnom dosiahnutí každej konfigurácie) bude nutne dosiahnuté $[x, u]$, kde $x \in (K_H^B)^*$ je typu READY a $\text{state}(x) = q$, $\text{mem}(x) = v$ a $\text{pos}(x) = i$.

Nech $p = v_0 v_1 \dots v_n$ je nejaká najkratšia cesta z r, ktorej dĺžka je $|v| + 1$ (jej existencia je zaručená predpokladom). Z definície nutného dosiahnutia vyplýva, že existuje taká konfigurácia (g, u) ACA A, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(p) = x\beta$.

Teda (g, u) je taká konfigurácia ACA A, že $(f, w) \vdash_A^{\mathcal{M}} (g, u)$ a $g(v_i) = \varphi$. Z toho ale podľa lemy 3.3.48 (o akceptovaní) vyplýva, že A akceptuje w pri výpočtovom pláne $\mathcal{M}$. □

3.4 Výsledok

Z toho, čo sme dokázali v predchádzajúcich častiach a toho, že v dostatočne veľkých grafoch existujú dostatočne dlhé cesty vyplýva nasledujúce tvrdenie.

Lema 3.4.1 *(o tom, že to funguje) Prechodová schéma H sa na triede prepojení[7] $\mathcal{G}_D$ chová dobre a $L(H, \mathcal{G}_D) = L(M)$.*

Dôkaz Nech $w \in L(M)$. Nech M pri (akceptačnom) výpočte na vstupe w použije najviac k políčok pracovnej pásky. Zrejme ak $G \in \mathcal{G}_D$ a $|V(G)| > D^{k+7} = B$, tak ku každému $r \in V(G)$ existuje najkratšia cesta z r, ktorej dĺžka je aspoň $k + 3$. Teda každý ACA $A = (G, r, H)$ s $G \in \mathcal{G}_D$ a $|V(G)| > B$ spĺňa predpoklady lemy 3.3.68 (o úplnosti), teda akcetptuje w pri ľubovoľnom výpočtovom pláne.

To ale znamená, že ak $w \in L(M)$, tak H akceptuje w.

Ak $w \notin L(M)$, tak podľa lemy 3.3.36 (o korektnosti) H neakceptuje w. □

Veta 3.4.2 *(o ekvivalencii TS a prechodových schém pre ACA na grafoch ohraničeného stupňa) Nech $D > 1$ je prirodzené číslo.*

Ku každému Turingovmu stroju M existuje prechodová schéma H taká, že H sa na $\mathcal{G}_D$ chová dobre a $L(H, \mathcal{G}_D) = L(M)$

Dôkaz Vyplýva z lemy 3.4.1 (o tom, že to funguje) a toho, že M a D bolo na začiatku ľubovoľne zvolené a toho, že zjednodušený TS je normálny tvar Turingových strojov. □

[7] Trieda prepojení $\mathcal{G}_D$ bola definovaná v definícii 3.0.2. Číslo D bolo zavedené v časti 3.2.

Kapitola 4

Záver

Dokázali sme, že výpočtová sila asynchrónnych bunkových automatov s prepojeniami z triedy $\mathcal{G}_k$ (zavedenej v definícii 3.0.2 (prepojenia ohraničeného stupňa)) je pre každú konštantu $k > 1$ rovnaká ako výpočtová sila Turingovych strojov. Z definície uvažovaného výpočtového modelu vyplýva, že podobné tvrdenie platí aj v prípade, že by sme zobrali prepojenia z ľubovoľnej nekonečnej podmnožiny $\mathcal{G}_k$. Ukazuje to teda, že bunkovým automatom asynchrónnosť a nepravidelnosť nezhorší výpočtovú silu.

Konštrukcia, ktorú sme použili na simuláciu Turingovho stroja je ale dosť neefektívna. Na to, aby asynchrónny bunkový automat s prechodovou schémou, ktorú sme skonštruovali fungoval treba, aby bol exponenciálne veľký v závislosti od pamäťovej zložitosti simulovaného Turingovho stroja. Ak vezmeme do úvahy, že sa pripúšťajú aj veľmi symetrické schémy prepojení a výpočtové plány dalo by sa dokázať, že na prepojeniach z $\mathcal{G}_k$ sa efektívnejšie počítať nedá.

Na dosiahnutie nižšej "priestorovej" zložitosti[1] by sme potrebovali nájsť nejaký spôsob na rozbitie symetrie. Jednou z možností, ako to dosiahnuť by mohlo byť nájdenie inej triedy prepojení. A to takej, na ktorej by sa dala zaviesť lepšia štruktúra než rozdelenie na vrstvy. Samozrejme chceme, aby tá trieda prepojení nebola až príliš obmedzujúca. Inou možnosťou by bolo upraviť definíciu výpočtového plánu tak, aby neumožňoval príliš symetrický priebeh výpočtu. Ešte by sa dalo rozbiť symetriu tak, že jednotlivé vrcholy by boli od seba odlíšené stavom už v počiatočnej konfigurácii.

Spôsob, akým je riešený vstup a výstup nepravidelných asynchrónnych bunkových automatov sa podstatne líši od toho, ktorý je použitý u ostatných modelov bunkových automatov. Ostatné bunkové automaty majú jednoduchú a často pevnú prechodovú schému a vstup ako aj to, čo sa má počítať je zakódované v počiatočnej konfigurácii. V našom prípade priame použitie tohto prístupu nie je možné. Kvôli nepravidelnosti totiž nie je možné (jednoducho) všeobecne popísať, ako by vyzerala počiatočná konfigurácia pre daný vstup. Distribúcia vstupu medzi viac vrcholov však má svoj význam aj v tom, že by umožnila zostrojiť automat, ktorý by rýchlejšie počítal. Bolo by teda vhodné nájsť jednoduchšiu triedu schém prepojení, takú pri ktorej by sa to zakódovanie vstupu do počiatočnej konfigurácie

[1] Myslíme tým potrebnú veľkosť automatu, pri ktorej už automat funguje tak, ako má.

dalo urobiť.

Ďalej by bolo treba definovať a analyzovať časovú zložitosť nepravidelných asynchrónnych bunkových automatov (teda presnejšie ich prechodových schém). Problém, ktorý tu vzniká spočíva v tom, že výpočtové plány síce zaručujú, že každý vrchol počíta, ale nijak neohraničujú rýchlosť tohto počítania. To teda znamená, že výpočet môže trvať ľubovoľne dlho. Jednou z možností ako tento problém vyriešiť by bolo pri definícii časovej zložitosti uvažovať len niektoré výpočtové plány (napríklad len synchrónne).

Literatúra

[Cod68] E. F. Codd. *Cellular Automata*. Academic Press, Inc., Orlando, FL, USA, 1968.

[Hau93] S. Hauck. Asynchronous design methodologies: An overview. Technical Report TR-93-05-07, Department of Computer Science and Engineering, University of Washington, Seattle, 1993.

[LAPM05] Jia Lee, Susumu Adachi, Ferdinand Peper, and Shinro Mashiko. Delay-insensitive computation in asynchronous cellular automata. *J. Comput. Syst. Sci.*, 70(2):201–220, 2005.

[Neh03] Chrystopher L. Nehaniv. Evolution in asynchronous cellular automata. In *ICAL 2003: Proceedings of the eighth international conference on Artificial life*, pages 65–73, Cambridge, MA, USA, 2003. MIT Press.

[PF93] Priyadarsan Patra and Donald S. Fussell. Building-blocks for designing DI circuits. Technical Report CS-TR-93-23, Dept. of Computer Sciences, Univ of Texas at Austin, November 1993.