Подпоследователни преобразуватели с допълнителна FIFO-памет. Регулярни правила за заместване

дипломна работа на

Стефан Владимиров Герджиков

магистър, ф.н.22028, сп. "Логика и алгоритми"
*e-mail:*st_gerdjikov@abv.bg

*дипломен ръководител:*ст.н.с Стоян Михов

Факултет по Математика и Информатика
Софийски Университет "Св. Климент Охридски"

март 2008

Съдържание

1 Увод

Замяната на определени думи в даден текст с други в зависимост от контекста, в който се намират, е от особена важност не само при машинната текст-обработка, но и при системи, които разпознават или самите те пресъздават човешка реч. Ето защо е полезно да можем да автоматизираме този процес, тоест по зададено правило да създадем машина, която ефективно да извършва замяната независимо от входния текст. За съжаление това не е възможно да бъде реализирано за произволни правила.

Един естествен клас от правила, за които такива устройства могат да се конструират са регулярните правила за заместване. При тях както левият и десният контекст, така и множеството от думи, които искаме да заменим са регулярни езици, а правилото за самата замяна се определя от рационална функция. За да се избегнат възможни конфликти, когато се оказва, че една и съща част от текста може да се замени по повече от един коректен начин, обикновено се избира стратегията на най-лявото, най-дълго срещане на правилото в текста. При нея се заменя първото отлява най-дълго срещане, а след това се търси следващото срещане, чиито фокус не се застъпва с вече заместената част от текста.

Тези ограничения се оказват в известен смисъл естествени и позволяват да бъдат описани голяма част от правила, които представляват практически интерес. От друга страна всяко регулярно правило за заместване, както следва от конструкциите, описани от Kaplan и Kay [KK94] и от Roche [Roc97], представляват рационални функции, тоест за тях може да бъде построен недетерминиран преобразувател (трансдюсер), който разпознава двойки от думи - текст и неговото коректно преобразуване.

Недетерминираните преобразуватели представляват естествено обобщение на крайните автомати и техните свойства се изучават подробно от Eilenberg [Eil74]. Въпреки че позволяват линейно траверсиране, тоест линейно по дължината на входния текст можем да намерим неговия превод, поради недетерминизма не можем локално да преценим кое ще бъде успешното изпълнение и съответно може да се наложи да задържим изхода до края, когато ще разберем кой е правилният резултат. За разлика от крайните автомати, където недетерминизмът може да бъде отстранен и всеки краен недетерминиран автомат може да се замени с еквивалентен детерминиран такъв, недетерминираните преобразуватели не притежават това хубаво свойство [Eil74]. Съществуват необходими и достатъчни условия, при които такава детерминизация е възможна, както показва Mohri [Moh94] и тогава заедно с траверсирането на входния текст, паралелно, с константна допълнителна памет, преобразувателят генерира и изходния текст. Детерминираните преобразуватели обикновено се означават като подпоследователни и за тях може да се постави и въпроса за минимизация [Moh94], който се решава ефективно от Mohri [Moh94], а след това Breslauer [Bre96] предлага по-ефективен алгоритъм, но и в двата основната идея е да се сведе проблема до минимизация на краен автомат - проблем, който решават с алоритъма на Hopcroft [Hop71]. От такава гледна точка са интересни правила, за които е възможно да се построи подпоследователен преобразувател. Пример за това е алгоритъмът на Михов и Schulz [SM06], които директно строят подпоследователен преобразувател за превод на думи в даден текст, по предварително зададен речник.

Освен с (недетерминирани) преобразуватели, регулярните правила за заместване могат да се представят и чрез бимашини. Бимашините представляват двойка автомати и функция, която по зададени състояни на двата автомата и символ връща дума от изходния текст. При подаден текст единият автомат започва да го траверсира отляво надясно, а другият - отдясно наляво. Изходът върху i-тата буква от текста зависи от това до кое състояние са стигнали двата автомата, когато им предстои да прочетат този символ. Резултатът от изпълнението на бимашината тогава е конкатенацията на тези локални изходи, конкатенирани отляво надясно. Eilenberg [Eil74] показва, че езиците, разпознавани от бимашини, съвпада с класа от рационални функции. Kaplan и Kay [KK94] също дават алгоритъм как от преобразувател за дадена функция се построява еквивалентна бимашина. Въпреки че бимашините са детерминирани и не се налага да съхраняваме произволно дълги части от изходния текст преди да ги отпечатаме, те имат недостатъка, че трябва да разполагат с целия входен текст. Нещо повече, за разлика от преобразувателите, те не позволяват да дописваме нови символи и думи към вече подадения текст, защото това ще промени поведението на "десния"автомат и съответно ще доведе до грешен резултат.

Целта на настоящата дипломна работа е да дефинира и да изучи част от свойствата на формализъм, който подобно на подпоследователните преобразуватели и бимашините е детерминиран, но за разлика от тях разполага с допълнителна памет, под формата на опашка, която може да използва за текущ запис - това, което всъщност се налага да правят недетерминираните преобразуватели, за да изведат накрая резултата. Този формализъм, който ще наричаме FIFO-трансдюсер, ще запази свойството на автоматите, да работи линейно време върху своя вход. Въпросът, който си поставяме след това, е доколко изразителни са тези устройства. Можем ли да ги използваме вместо бимашините или недетерминираните преобразуватели, за да разпознаваме правила за същото време, но потенциално по-малко памет? Ако не, то за какви правила можем да строим ефективно FIFO-трансдюсери?

От операциите, характерни за регулярни множества - конкатенация, обединение, сечение, звезда и композиция - ние ще изследваме само последната в контекста на FIFO-трансдюсери. Всъщност конкатенацията, обединението и звездата на Клини, не запазват в общия случай функциите, а както казахме - FIFO-трансдюсерите са насочени именно към разпознаването на функции. Сечението пък не запазва регулярността, когато става въпрос за релации. Накрая композицията на две функции е отново функция, но в контекста на прилагане на правила, това означава вместо последователно да прилагаме първото правило към входния текст, а след това второто върху изхода, да извършим замяната директно, като не се интересуваме от временния резултат. Това е всъщност и свойството на рационалните функции (регулярните релации), което намира най-широко практическо приложение и затова ние спираме подробно нашето внимание именно върху това свойство.

Дипломната работа е структурирана по следния начин. В част 2 ние дефинираме FIFO-трансдюсерите и формално дефинираме техния език. Показваме, че всеки FIFO-трансдюсер разпознава (частична) функция и може да бъде траверсиран детерминирано, за линейно време. Даваме пример за нерационална функция, която се разпознава от FIFO-трансдюсер и примери за рационални функции, които се разпознават от FIFO-трансдюсер. Тази част завършва с една теорема с техническо значение, която показва как можем да ограничим записите, които правим в опашката, без да загубим от изразителната способност на FIFO-трансдюсерите.

В част 3 изучаваме композицията на FIFO-трансдюсери с подпоследователни преобразуватели и доказваме, че както лявата, така и дясната композиция дава функция, разпознавана от FIFO-трансдюсер. Доказателствата са конструктивни, което ни помага да ги използваме в основата на алгоритми за конструирането на такива устройства.

Част 4 е посветена на един подклас от рационални функции, които могат да се представят чрез FIFO-трансдюсер. Първо ги дефинираме и показваме как ефективно можем да строим FIFO-трансдюсери за тях. След това показваме, че можем да построим и бимашина за всяка такава функция. В предпоследния параграф на част 4 показваме едно достатъчно условие за композиция на две такива функции, което ни позволява ефективно да конструираме функция, еквивалентна на търсената композиция. Тази част завършва с проверката, че примерите от част 2 са действително функции, разпознавани от FIFO-трансдюсер.

В част 5 конструираме рационална функция, която не е представима с FIFO-трансдюсер. Показваме също, че класът от функции, представими с FIFO-трансдюсери не е затворен относно композиция.

Част 6, мотивирана от това, че има регулярни правила, за които няма FIFO-трансдюсер, се стреми да опише клас от такива, за които съответното устройство съществува и може да се конструира ефективно. Първо дефинираме правила с маркери, за които използвайки твърденията от част 4 показваме конструкция за построяването на FIFO-трансдюсер, който да ги разпознава. След това разглеждаме и регулярни правила от общ тип. Налагаме алгоритмично проверими ограничения върху тези правила и в случай, че условията са изпълнени и като използваме теоремите от част 3 показваме как могат да се сведат към правила с маркери. Накрая разглеждаме ограничения, при които можем да композираме алгоритмично правилата. Показваме отново, че условията са алгоритмично-проверими и използваме теоремите от част 4 за да конструираме съответния FIFO-трансдюсер.

В част 7 се занимава с алгоритмичната реализация на конструкциите, описани в предходните части. Първо разработваме алгоритми, подобни на този на Михов и Schulz [SM06], за да можем да поставим необходимите ни маркери от част 6 посредством подпоследовтаен преобразувател, ефективно. След това разработваме алгоритми, с помощта на които ефективно се реализарат конструкциите за композиция с подпоследователни преобразуватели, описани в част 3. Накрая скицираме реализацията на някои от по-важните конструкции в част 4 и част 6.

В част 8 обощаваме получените резултати и поставяме въпроси, които биха могли да представляват интерес за по-нататъшно изследване.

2 Дефиниции. Примери. Каноничен FIFO-трансдюсер

2.1 Трансдюсер с опашкова памет(FIFO-трансдюсер)

Дефиниция 1 *Детерминиран трансдюсер с опашкова памет (FIFO-трансдюсер) наричаме* $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$*, където:*

1. Σ*,* Ω *и* Γ *са съответно входна, изходна и азбука на опашката.*
2. P *и* Q *са дизюнктни крайни множества от състояния.* P *наричаме основни състояния, а* Q *състояния на опашката.*
3. $i \in P$ *- начално състояние.*
4. $\Delta : P \times \Sigma \to P \cup Q$ *- функция на преходите върху основните състояния.*
5. $d : Q \times \Gamma \times \{0, 1\} \to P \cup Q$ *- функция на преходите върху състоянията на опашщата.*
6. $\lambda : P \times \Sigma \to \Gamma^*$*,* $Dom(\lambda) = Dom(\Delta)$ *- функция, записваща в опашката.*
7. $out : Q \times \Gamma \times \{0, 1\} \to \Omega^*$*,* $Dom(out) = Dom(d)$ *- изходна функция.*
8. $\phi : P \to Q$
9. $\psi : Q \to \Omega^*$.

Всички функции са частични.

Това е формалната дефиниция на FIFO-трансдюсер. Следващите няколко дефиниции ще ни позволят да въведем семантика върху този синтактичен обект.

Дефиниция 2 *Ако $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$, конфигурация $\kappa = (p, \alpha, \gamma, \omega)$ на T това е произволен елемент на $(P \cup Q) \times \Sigma^* \times \Gamma^* \times \Omega^*$.*

Интуицията ни е, че след прочитане на част от входната дума сме достигнали до състояние p, остатъкът от входната дума е α, а думата, която в момента е записана в опашката е γ. ω е текущият резултат върху изходната лента, това от което се интересуваме, всъщност.

С помощта на функциите Δ и d управляваме преходите между отделните конфигурации по следния начин:

Дефиниция 3 *Ако $\kappa' = (p', \alpha', \gamma', \omega')$ и $\kappa'' = (p'', \alpha'', \gamma'', \omega'')$ са две конфигурации на $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$, казваме, че T допуска прехода $\kappa' \vdash \kappa''$ в един от следните три случая:*

1. *$p' \in P$, $\alpha' = \sigma\alpha''$, където $\sigma \in \Sigma$ и $p'' = \Delta(p', \sigma)$, $\gamma'' = \gamma'\lambda(p', \sigma)$, $\omega'' = \omega'$.*

2. *$p' \in Q$, $\gamma' = b\gamma''$, където $b \in \Gamma$, $\gamma'' \neq \varepsilon$ и $p'' = d(p', b, 1)$, $\alpha'' = \alpha'$, $\omega'' = \omega out(p', b, 1)$.*

3. *$p' \in Q$, $\gamma' = b \in \Gamma$ и $\gamma'' = \varepsilon$, $p'' = d(p', b, 0)$, $\alpha'' = \alpha'$, $\omega'' = \omega' out(p', b, 0)$.*

Във всички останали случаи преходът $\kappa' \vdash \kappa''$ е недопустим в T.

Първият случай съответства на положението, когато трябва да прочетем следващия символ от входната дума. Тогава можем и да запишем нещо в опашката. Вторият - четем символ от опашката, който не е последният, а при третия случай - отново четем символ от опашката, след което тя остава празна. В последните два случая определяме и част от изхода.

Дефиниция 4 *Ако $\kappa' = (p', \alpha', \gamma', \omega')$ и $\kappa'' = (p'', \alpha'', \gamma'', \omega'')$ са две конфигурации на T, казваме, че T допуска извода $\kappa' \models \kappa''$ ако:*

1. *$\kappa' = \kappa''$ или*

2. *съществува конфигурация κ на T, за която $\kappa' \vdash \kappa$ и $\kappa \models \kappa''$.*

Забележка 1 *Всъщност $\models$ е рефлексивното и транзитивно затваряне на $\vdash$.*

Дефиниция 5 *Дължина на извода $\kappa' \models \kappa''$ това е естествено число l, което се определя рекурсивно по дефиницията на извод.*

1. *$\kappa' = \kappa''$, то $l = 0$ и*

2. *ако изводът е съставен от прехода*

$$\kappa' \vdash \kappa \text{ и от извода}$$
$$\kappa \models \kappa'',$$

   и дължината на извода $\kappa \models \kappa''$ е l', то:

$$l = l' + 1.$$

Забележка 2 *От дефиниция 3 за преход следва, че от една конфигурация има неповече от един допустим преход. Оттук и от дефиниция 4 следва, че между две конфигурации съществува неповече от един допустим извод.*

Дефиниция 6 *Нека $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$ е FIFO-трансдюсер. Релация или език, дефиниран от T наричаме множеството $\mathcal{L}(T)$ от всички наредени двойки от думи (α, ω), за които $\exists f \in P$, $\exists p \in Q$, $\exists \gamma \in \Gamma^*$, $\exists \omega_1, \omega_2 \in \Omega^*$ със свойствата:*

1. $(i, \alpha, \varepsilon, \varepsilon) \models (f, \varepsilon, \gamma, \omega_1)$.

2. $(\phi(f), \varepsilon, \gamma, \omega_1) \models (p, \varepsilon, \varepsilon, \omega_2)$.

3. *$p \in Q$ и $\omega = \omega_2 \psi(p)$.*

Следващото твърдение ще покаже, че всъщност езикът, дефиниран от даден FIFO-трансдюсер може да се разглежда като (частична) функция над множеството от входни думи Σ^*. Освен това, в термините на дължина на извода, FIFO-трансдюсерите позволяват линейно, последователно и детерминирано траверсиране по зададен вход - свойство, което притежават и подпоследователните преобразуватели.

Твърдение 1 *Нека $T = < \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$ е FIFO-трансдюсер. Нека още $l = \max |\lambda(p, \sigma)|$, където максимумът се взима по всички двойки $(p, \sigma) \in P \times \Sigma$, за които λ е дефинирана. Тогава са в сила следните твърдения:*

1. *За всеки $q \in Q$, $\alpha \in \Sigma^*$, $\gamma \in \Gamma^*$ и $\omega \in \Omega^*$ съществува неповече от една конфигурация $(p, \alpha, \gamma', \omega')$, за която $(q, \alpha', \gamma, \omega) \vDash (p, \alpha, \gamma', \omega')$, за която $p \in P$ и $|\alpha| = |\alpha'|$. При това ако такъв извод съществува, то дължината му е $|\gamma| - |\gamma'|$. Нещо повече γ' е наставка на γ и $\alpha = \alpha'$.*

2. *За всеки $p_1 \in P$, $\alpha_1, \alpha_2 \in \Sigma^*$, $\gamma_1 \in \Gamma^*$ и $\omega_1 \in \Omega^*$ съществува неповече от една конфигурация $(p_2, \alpha_2, \gamma_2, \omega_2)$, за която $p_2 \in P$ и $(p_1, \alpha_1, \gamma_1, \omega_1) \vDash (p_2, \alpha_2, \gamma_2, \omega_2)$. При това ако такъв извод съществува, то неговата дължина не надминава $(l+1)(|\alpha_1| - |\alpha_2|) + |\gamma_1| - |\gamma_2|$. Нещо повече α_2 е наставка на α_1.*

Доказателство:

1. Първо да забележим, че от дефиниция 3 при преход

$$(p_1, \alpha_1, \gamma_1, \omega_1) \vdash (p_2, \alpha_2, \gamma_2, \omega_2)$$

по правило 1 $|\alpha_2| = |\alpha_1| - 1$, докато при прилагане на правилата 2 и 3 $\alpha_1 = \alpha_2$. Това означава, че ако $(q, \alpha, \gamma, \omega) \vDash (p, \alpha', \gamma', \omega')$, за които $|\alpha_1| = |\alpha_2|$, то $\alpha_1 = \alpha_2$ и всеки от преходите, от които този извод е съставен отговаря на правило 2 или правило 3 на дефиниция 3. Сега всяка конфигурация от вида $(q, \alpha, \gamma, \omega)$ с $q \in Q$ еднозначно определя дали прилагаме правилото 2 или 3, и всяко от тях еднозначно определя новата конфигурация $(q', \alpha, \gamma', \omega')$. От друга страна при конфигурация от вида $(p, \alpha, \gamma', \omega')$ с $p \in P$ не е възможен преход по правило 2 или 3. Това доказва и еднозначността на всеки извод от вида $(q, \alpha, \gamma, \omega) \vDash (p, \alpha, \gamma', \omega')$ с $p \in P$. При всяко прилагане на правило 2 или 3 по дефиниция 3 при преход

$$(q_1, \alpha, \gamma_1, \omega_1) \vdash (q_2, \alpha, \gamma_2, \omega_2)$$

следва, че $|\gamma_2| = |\gamma_1| - 1$ и γ_2 е наставка на γ_1. Следователно и дължината на извода

$$(q, \alpha, \gamma, \omega) \models (p, \alpha, \gamma', \omega')$$

е $|\gamma| - |\gamma'|$ и γ' е наставка на γ.

2. Еднозначността на извода следва лесно, като се има впредвид, че правило 1 от дефиниция 3 променя дадена конфигурация

$$(p, \alpha, \gamma, \omega) \vdash (p', \alpha', \gamma', \omega')$$

еднозначно, като $|\alpha| = |\alpha'| + 1$. Това означава, че ако съществува извод $(p_1, \alpha_1, \gamma_1, \omega_1) \models (p_2, \alpha_2, \gamma_2, \omega_2)$, то той се състои точно от $|\alpha_1| - |\alpha_2|$ прехода по правило 1. Сега доказателството може да бъде извършено с индукция по броя на преходите от този тип. Ако той е 0 твърдението следва директно, освен това получаваме, че $p_1 = p_2$, $\alpha_1 = \alpha_2$, $\gamma_1 = \gamma_2$ и $\omega_1 = \omega_2$, в случай, че такъв извод съществува.

Нека твърдението е в сила за произволни $p_1 \in P$, $\gamma_1 \in \Gamma^*$, $\omega_1 \in \Omega^*$ и $\alpha_1, \alpha_2 \in \Sigma^*$ с $|\alpha_1| - |\alpha_2| = n$. Тоест за всеки $p_1 \in P$, $\gamma_1 \in \Gamma^*$, $\omega_1 \in \Omega^*$ и $|\alpha_1| - |\alpha_2| = n$ съществува неповече от един извод $(p_1, \alpha_1, \gamma_1, \omega_1) \models (p_2, \alpha_2, \gamma_2, \omega_2)$, където $p_2 \in P$ и ако такъв извод съществува, то неговата дължина ненадминава $(l+1)n + |\gamma_1| - |\gamma_2|$ и α_2 е наставка на α_1. Нека сега $\alpha_1 = \sigma\alpha_1'$, където $\sigma \in \Sigma$, $\alpha_2 \in \Sigma^*$ и $|\alpha_1| - |\alpha_2| = n + 1$ са произволни. Нека още $p_1 \in P$, $\gamma_1 \in \Gamma^*$ и $\omega_1 \in \Omega^*$. Да допуснем, че съществува поне един извод от желания вид,

$$(p_1, \alpha_1, \gamma_1, \omega_1) \models (p_2', \alpha_2, \gamma_2', \omega_2').$$

Тогава от $n + 1 > 0$ следва, че той започва с преход от вида:

$$(p_1, \alpha_1, \gamma_1, \omega_1) \vdash (p_1', \alpha_1', \gamma_1', \omega_1'),$$

защото $p_1 \in P$, следван от извод

$$(p_1', \alpha_1', \gamma_1', \omega_1') \models (p_2', \alpha_2, \gamma_2', \omega_2').$$

Освен това, тъй като прилагаме правило 1 от дефиниция 3, то $\gamma_1' = \gamma_1 \lambda(p_1, \sigma)$, в частност $|\gamma_1'| \leq |\gamma_1| + l$.

За p_1' са възможни два случая:

- **случай 1** $p_1' \in P$. В този случай можем да приложим индукционното предположение за p_1', α_1', α_2, γ_1' и $\omega_1' = \omega_1$, защото $|\alpha_1'| - |\alpha_2| = n$. Тогава ще следва, че изводът $(p_1', \alpha_1', \gamma_1', \omega_1') \models (p_2', \alpha_2, \gamma_2', \omega_2')$ е еднозначно определен от p_1', α_1', α_2, γ_1' и $\omega_1' = \omega_1$ и неговата дължина ненадминава

$$(l+1)n + |\gamma_1'| - |\gamma_2| \leq (l+1)n + l + |\gamma_1| - |\gamma_2|.$$

  Оттук директно получаваме, че изводът $(p_1, \alpha_1, \gamma_1, \omega_1) \models (p_2', \alpha_2, \gamma_2', \omega_2')$ се определя еднозначно от p_1, α_1, α_2, γ_1 и ω_1 и дължината му е ограничена отгоре с

$$1 + (l+1)n + l + |\gamma_1| - |\gamma_2| = (l+1)(n+1) + |\gamma_1| - |\gamma_2|.$$

  Освен това α_2 е наставка на α_1' и тъй като $\alpha_1 = \sigma\alpha_1'$ то α_2 е наставка и на α_1.

- **случай 2** $p_1' \in Q$. Тогава от първата част на твърдението следва, че съществува единствен извод

$$(p_1', \alpha_1', \gamma_1', \omega_1') \models (p_1'', \alpha_1', \gamma_1'', \omega_1''),$$

  където $p_1'' \in P$ и неговата дължина е $|\gamma_1'| - |\gamma_1''|$. Това означава, че изводът $(p_1, \alpha_1, \gamma_1, \omega_1) \models (p_2', \alpha_2, \gamma_2', \omega_2')$ може да се представи като:

$$(p_1, \alpha_1, \gamma_1, \omega_1) \vdash (p_1', \alpha_1', \gamma_1', \omega_1') \models (p_1'', \alpha_1', \gamma_1'', \omega_1'') \models (p_2', \alpha_2', \gamma_2', \omega_2').$$

  Тъй като $p_1'' \in P$ и $|\alpha_1'| - |\alpha_2'| = n$, то можем да приложим индукционното предположение за p_1'', α_1', α_2', γ_1'' и ω_1'' . Така заключаваме, че p_1'', α_1', γ_1'' и ω_1'' еднозначно определят p_2', γ_2' и ω_2'. Тъй като те се определят, от твърдение 1 еднозначно от p_1', γ_1' и $\omega_1' = \omega_1$, следва, че и целият извод е еднозначно определен от p_1, γ_1, ω_1, α_1 и α_2. Освен това неговата дължина е равна сумата от дължините на изводите

$$(p_1', \alpha_1', \gamma_1', \omega_1') \models (p_1'', \alpha_1', \gamma_1'', \omega_1'') \text{ и}$$
$$(p_1'', \alpha_1', \gamma_1'', \omega_1'') \models (p_2', \alpha_2, \gamma_2', \omega_2')$$

плюс 1. Но според първата част на твърдението, дължината на първия извод е $|\gamma_1'| - |\gamma_1''|$, а дължината на втория ненадминава, по индукционната хипотеза, $(l+1)n + |\gamma_1''| - |\gamma_2'|$. Следователно дължината на целия извод е ограничена отгоре от:

$$\begin{aligned} 1 + |\gamma_1'| - |\gamma_1''| + (l+1)n \quad &+ \quad |\gamma_1''| - |\gamma_2'| = \\ 1 + |\gamma_1'| + (l+1)n - |\gamma_2'| \quad &\leq \quad 1 + l + |\gamma_1| + (l+1)n - |\gamma_2'| = \\ (n+1)(l+1) \quad &+ \quad |\gamma_1| - |\gamma_2'|. \end{aligned}$$

Освен това α_2 е наставка на α_1', от където следва, че е наставка и на α_1.

С това идукционната стъпка е завършена, а с това и твърдението1 е доказано.

Следствие 1 *За всеки FIFO-трансдюсер* $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$ *с* $l = \max |\lambda(p, \sigma)|$*, където максимумът е по всички двойки* $(p, \sigma) \in P \times \Sigma$ *от дефиниционната област на* λ *и завсяко* $\alpha \in \Sigma^*$ *е в сила, че:*

1. *съществува неповече от една тройка* (f, γ, ω_1)*, където* $f \in P$*,* $\gamma \in \Gamma^*$ *и* $\omega_1 \in \Omega^*$*, за която* T *допуска извода*

   $$(i, \alpha, \varepsilon, \varepsilon) \models (f, \varepsilon, \gamma, \omega_1).$$

   При това ако такъв извод съществува, неговата дължина ненадминава $(l+1)|\alpha| - |\gamma|$*.*

2. *ако такава тройка (f, γ, ω_1) съществува, то съществува не повече от една двойка (q, ω_2), за която $q \in P \cup Q$ и $\omega_2 \in \Omega^*$, за които*

$$(\phi(f), \varepsilon, \gamma, \omega_1) \vDash (q, \varepsilon, \varepsilon, \omega_2).$$

   В случай, че съществува дължината на този извод е $|\gamma|$.

3. *$\mathcal{L}(T)$ е графика на частична функция $f_T : \Sigma^* \to \Omega^*$. Ако стойностите на всяка една от функциите в дефиницията на T могат да се пресмятат за $O(1)$ при фиксирани параметри, то изчислението на $f_T(\alpha)$ може да се извърши итеративно за време $O(|\alpha|)$.*

1. Това е директно следствие от втората част на твърдение 1, приложено за $p_1 = i$, $\alpha_1 = \alpha$, $\gamma_1 = \varepsilon$, $\omega_1 = \varepsilon$ и $\alpha_2 = \varepsilon$. Оценката за дължината на извода също следва непосредствено.

2. Тъй като вторите компоненти на двете конфигурации са еднакви, то един извод между тях може да се състои само от преходи, отговарящи на правилата 2 и 3 от дефиниция 3. С разсъждения, аналогични на тези при доказателството на първата част на твърдение 1, виждаме, че съществува най-много един такъв извод и неговата дължина е $|\gamma|$.

3. Директно от 1 и 2 и дефиниция 6 за $\mathcal{L}(T)$ получаваме, че f_T е функция. Започваме с конфигурацията $(i, \alpha, \varepsilon, \varepsilon)$ и на всяка стъпка на алгоритъма извършваме по един преход, докато не стигнем в конфигурация $(f, \varepsilon, \gamma, \omega_1)$. След това преминаваме в конфигурацията $(\phi(f), \varepsilon, \gamma, \omega_1)$ и отново траверсираме, движим се по преходите, докато не попаднем в конфигурация $(q, \varepsilon, \varepsilon, \omega_2)$. Ако $q \in Q$, то $f_T(\alpha) = \omega_2\psi(q)$. Паметта, която ще ни е необходима е тази за представянето на T плюс размера на максималната възможна дължина на γ, която ще се наложи да изчислим.

2.2 Примери за FIFO-трансдюсери

В тази част ще разгледаме три функции, които могат да се зададат чрез FIFO-трансдюсер. От една страна те ще ни дадат интуиция за изразителната способност на нашия формализъм и върху тях ще демонстрираме техниката на доказателство, която ще използваме и в следващите части. От друга страна, тези примери ще ни дадат възможност да сравним езиците, разпознавани от FIFO-трансдюсерите с класа на рационалните функции, който е добре изучен.

Функциите, на които ще посветим този параграф са следните:

$$ir : (\Sigma \cup \$)^* \to (\Sigma \cup \$)^*, \text{ дефинирана като}$$
$$ir(\alpha) = \begin{cases} \alpha, \text{ ако } \alpha = \omega\$\omega \text{ за някоя дума } \omega \in \Sigma^* \\ \neg! \text{ иначе.} \end{cases}$$

Втората функция, която ще разгледаме ще бъде:

$$ins_{0,1} : \{a, b\}^+ \to \{a, b, 0, 1\}^*.$$

За дума от вида:

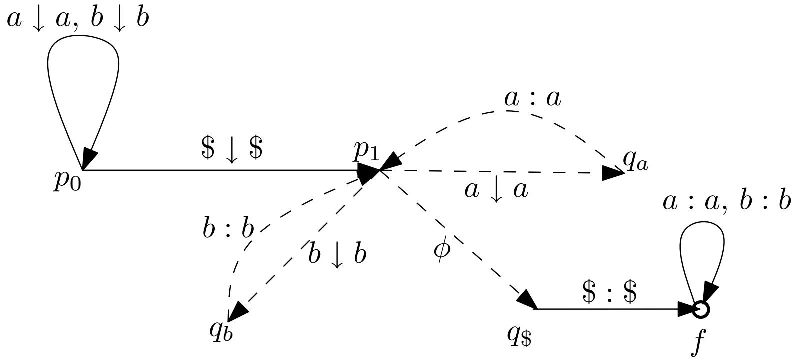
$$\prod_{i=0}^{k}(a^{m_i}b^{n_i}, \text{ където } \forall \quad i > 0\, m_i > 0 \,\&\, \forall \quad j < k\, n_j > 0$$
$$ins_{0,1}\left(\prod_{i=0}^{k}(a^{m_i}b^{n_i})\right) = \prod_{i=0}^{k}(c_i a^{m_i}b^{n_i})0,$$

където буквите $c_i \in \{0, 1\}$ удовлетворяват условието:

$$c_i \equiv \sum_{j=i}^{k} a_{n_j} (\mod 2).$$

Накрая ще построим и FIFO-трансдюсер за функцията:

$$rep_{0,1} : \{a, b, 0, 1\}^* \to \{A, a, b\}^*, \text{ зададена от}$$
$$rep_{0,1}(c_i) = \begin{cases} \varepsilon, c_i \in \{0, 1\} \\ \neg! \text{ иначе.} \end{cases}$$
$$rep_{0,1}(\alpha c \beta) = \begin{cases} \alpha_A \circ rep_{0,1}(\beta) \text{ ако } \alpha \in \{a, b\}^*, c = 1 \\ \alpha \circ rep_{0,1}(\beta) \text{ ако } \alpha \in \{a, b\}^*, c = 0 \\ \neg! \text{ иначе,} \end{cases}$$



Фигура 1: Конструкцията на T_{ir} при $\Sigma = \{a, b\}$.

където α_A е думата, получена от α след замяната на всяко срещане на a с буквата A.

В сила са следните две теореми:

Теорема 1 *Съществува FIFO-трансдюсер T_{ir}, за който:*

$$f_{T_{ir}} = ir.$$

Теорема 2 *1. Съществува FIFO-трансдюсер T_{ins}, за който:*

$$f_{T_{ins}} = ins_{0,1}.$$

2. Съществува FIFO-трансдюсер T_{rep}, за който:

$$f_{T_{rep}} = rep_{0,1}.$$

Започваме с конструкцията на FIFO-трансдюсер T_{ir}, който ще разпознава функцията ir (вж. фигура 1).

$$T_{ir} =< (\Sigma \cup \$) \times (\Sigma \cup \$)^*, \Sigma \cup \$, \{p_0, p_1\}, \{q_a \,|\, a \in \Sigma \cup \$\} \cup \{f\}, p_0, \Delta, d, \lambda, out, \phi, \psi >$$

Функцията на преходите върху основните състояния

$$\Delta : \{p_0, p_1\} \times (\Sigma \cup \$) \to \{p_0, p_1\} \cup \{q_a \,|\, a \in \Sigma \cup \$\} \cup \{f\}$$

се определя по следния начин:

$$\Delta(p_0, a) = \begin{cases} p_0 \text{ ако } a \in \Sigma \\ p_1 \text{ ако } a = \$ \end{cases}$$

$$\Delta(p_1, a) = \begin{cases} q_a \text{ ако } a \in \Sigma \\ \neg! \text{ иначе.} \end{cases}$$

Функцията

$$\lambda : \{p_0, p_1\} \times (\Sigma \cup \$) \to \Sigma \cup \$$$

се дефинира тривиално:

$$\lambda(p_i, a) = a,$$

за всяка двойка (p_0, a) и (p_1, a) с изключение на $(p_1, \$)$, за която λ е недефинирана.

Функцията

$$d : (\{q_a \,|\, a \in \Sigma \cup \$\} \cup \{f\}) \times (\Sigma \cup \$) \times \{0, 1\} \to \{p_0, p_1\} \cup \{q_a \,|\, a \in \Sigma \cup \$\} \cup \{f\}$$

се задава с равенствата:

$$d(q, b, j) = \begin{cases} p_1 \text{ ако } q = q_b, b \in \Sigma, j \in \{0, 1\} \\ f \text{ ако } q = q_\$, j \in \{0, 1\} \\ f \text{ ако } q = f,\ b \in \Sigma \cup \$,\ j \in \{0, 1\} \\ \neg! \text{ иначе.} \end{cases}$$

Функцията на изходите

$$out : (\{q_a \,|\, a \in \Sigma \cup \Omega\} \cup \{f\}) \times (\Sigma \cup \$) \times \{0, 1\} \to (\Sigma \cup \$)^*$$

се определя като:

$$out(q_a, b, j) = b$$

за всеки елемент от дефиниционната област на d.

Накрая дефинираме и функциите ϕ и ψ:

$$\phi(p_1) = q_\$$$
$$\psi(f) = \varepsilon.$$

За всички останали аргументи тези две функции са недефинирани.

Сега лесно се проверяват следните твърдения:

Твърдение 2 *1.* $< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_0, \beta, \gamma, \omega >$ *тогава и само тогава, когато* $\alpha = \gamma \in \Sigma^*$ *и* $\omega = \varepsilon$.

2. $< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_1, \beta, \gamma, \omega >$ *тогава и само тогава, когато* $\omega\gamma = \alpha$ *и* $\alpha = \omega z\$\omega$, *където* $z \in \Sigma^*\$$, *а* $\omega \in \Sigma^*$.

Доказателство: Доказателството и на двете твърдения е с индукция. В едната посока индукцията е по дължината на извода, а в другата по дължината на думата α.

1. Нека:
$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_0, \beta, \gamma, \omega > .$$
Ако дължината на извода е 0, то по дефиниция следва, че $\alpha = \varepsilon$ и няма какво да доказваме. Нека твърдението е вярно за всеки извод с дължина по-малка от n и нека дължината на извода:
$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_0, \beta, \gamma, \omega >$$
е n. Тогава тъй като всеки от преходите, определен от някое от състоянията, различни от p_0 води в състояние различно от p_0, то можем да представим дадения извод като:
$$\begin{aligned} < p_0, \alpha' a\beta, \varepsilon, \varepsilon > &\models < p_0, a\beta, \gamma', \omega' > \\ < p_0, a\beta, \gamma', \omega' > &\vdash < p_0, \beta, \gamma, \omega > . \end{aligned}$$
Сега можем да приложим индукционното предположение за извода:
$$< p_0, \alpha' a\beta, \varepsilon, \varepsilon > \models < p_0, a\beta, \gamma', \omega' >,$$
от където получаваме, че $\omega' = \varepsilon$ и $\gamma' = \alpha' \in \Sigma^*$. Сега от това, че имаме прехода:
$$< p_0, a\beta, \gamma', \omega' > \vdash < p_0, \beta, \gamma, \omega >$$

следва, че $\Delta(p_0, a) = p_0$. Това е възможно само ако $a \in \Sigma$. Освен това $\lambda(p_0, a) = a$ и следователно:

$$\gamma = \gamma' \circ \lambda(p_0, a) = \gamma' \circ a = \alpha' \circ a = \alpha.$$

Накрая $\omega = \omega' = \varepsilon$.

В обратната посока твърдението следва с индукция по дължината на α. Ако $\alpha = \varepsilon$ твърдението е очевидно. Нека $\alpha = \alpha' \circ a \in \Sigma^*$. Тогава от индукционното предположение имаме извода:

$$< p_0, \alpha' a\beta, \varepsilon, \varepsilon > \models < p_0, a\beta, \gamma', \omega' > .$$

Тъй като $\alpha \in \Sigma^*$, то и $a \in \Sigma$ и следователно $\Delta(p_0, a) = p_0$, което показва, че:

$$< p_0, a\beta, \gamma', \omega' > \vdash < p_0, \beta, \gamma, \omega > .$$

С това първото твърдение е доказано.

2. Нека е в сила, че:

$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_1, \beta, \gamma, \omega > .$$

Ще покажем, че $\alpha = \omega\gamma = \omega z \omega$, където $\omega \in \Sigma^*$, $z \in \Sigma^*\$$. Доказателството ще извършим с индукция по броя на конфигурациите от извода от вида $\kappa =< p_1, x, y, z >$. Да забележим, че съществуват два начина да достигнем до такава конфигурация. Първият е когато предходната конфигурация има вида $< p_0, x', y', z' >$, а втората е когато предходната конфигурация има вида $< q_a, x', y', z' >$ за някое $a \in \Sigma$.

Нека броят на конфигурациите $< p_1, x, y, z >$ в разглеждания извод е 1. Тогава всички предходни конфигурации в извода са от вида $< p_0, x', y', z' >$. Следователно ако $\alpha = \alpha' a$, то:

$$< p_0, \alpha' a\beta, \varepsilon, \varepsilon > \models < p_0, a\beta, \gamma', \omega' > \vdash < p_1, \beta, \gamma, \omega > .$$

От твърдение 1 имаме, че $\gamma' = \alpha'$ и $\omega' = \varepsilon$. Сега тъй като $\Delta(p_0, a) = p_1$ от дефиницията на Δ следва, че $a = \$$. Освен това от дефиницията на λ можем да заключим, че:

$$\gamma = \gamma' \circ \lambda(p_0, \$) = \gamma'\$ = \alpha.$$

Твърдението сега следва от съображението, че $\alpha = \varepsilon\alpha\varepsilon$ и $\alpha'\$ \in \Sigma^*\$$.

Нека сега в извода:

$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_1, \beta, \gamma, \omega > .$$

има $n + 1$ конфигурации от вида $< p_1, x, y, z >$, където $n \geq 1$. Нека $< p, x'\beta, y', z' >$ е конфигурацията преди $< p_1, \beta, \gamma, \omega >$, тоест:

$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p, x'\beta, y', z' > \vdash < p_1, \beta, \gamma, \omega > .$$

Сега $p = p_0$ или $p = q_a$ за някое $a \in \Sigma$. Тъй като обаче преди $< p, x'\beta, y', z' >$ съществува поне още една конфигурация $< p_1, x_0\beta, y_0, z_0 >$, то $(\alpha\beta)(x')^{-1} \notin \Sigma^*$ и следователно $p \neq p_0$. Нека $p = q_a$. Тогава единственият преход, който води до $< q_a, x'\beta, y', z' >$ е $< p_1, ax'\beta, y'', z'' >$. Следователно дадения извод има вида:

$$\begin{aligned} &< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_1, ax'\beta, y'', z'' > \vdash \\ &\qquad < q_a, x'\beta, y', z' > \vdash < p_1, \beta, \gamma, \omega > \end{aligned}$$

Сега изводът:

$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \models < p_1, ax'\beta, y'', z'' >$$

съдържа по-малко конфигурации със състоянието p_1 и за него можем да приложим индукционното предположение. Тогава тъй като:

$$< q_a, x'\beta, y', z' > \vdash < p_1, \beta, \gamma, \omega >$$

и q_a е състояние на опашката, то $x' = \varepsilon$. Нека $\alpha = \alpha' \circ a$. Тогава от индукционното предположение получаваме, че:

$$\alpha' = z''y'' = z''uz''$$

като $u \in \Sigma^*\$$, а $z'' \in \Sigma^*$. Сега лесно получаваме, че:

$$\begin{gathered} y' = y'' \circ \lambda(p_1, a) = y' \circ a \text{ и} \\ z' = z'' \end{gathered}$$

Тъй като имаме преход от $< q_a, \beta, y', z' > \vdash < p_1, \beta, \gamma, \omega >$, то първата буква на y' е a и следователно:

$$\begin{aligned} \gamma &= a^{-1}y' \text{ и} \\ \omega &= z'a. \end{aligned}$$

Тогава получаваме, че:

$$\begin{aligned} \omega\gamma = z'a(a^{-1}y''a) = z''y''a = \alpha'a = \alpha \text{ и} \\ z''uz''a = \omega(a^{-1}u)\omega. \end{aligned}$$

Тъй като $a \in \Sigma$, то $a^{-1}u \in \Sigma^*\$$, с което доказателството е завършено.

Нека сега $\alpha = \omega u \omega$, където $\omega \in \Sigma^*$, $u \in \Sigma^*\$$. Ще покажем, че:

$$< p_0, \alpha\beta, \varepsilon, \varepsilon > \vDash < p_1, \beta, u\omega, \omega >$$

с индукция по дължината на ω. При $\omega = \varepsilon$ имаме, че $v = u\$^{-1} \in \Sigma^*$ и от първото твърдение получаваме, че съществува изводът:

$$< p_0, v\$\beta, \varepsilon, \varepsilon > \vDash < p_0, \$\beta, v, \varepsilon > .$$

Тогава, тъй като:

$$\begin{aligned} \Delta(p_0, \$) &= p_1 \\ \lambda(p_0, \$) &= \$, \end{aligned}$$

получаваме и извода:

$$< p_0, v\$\beta, \varepsilon, \varepsilon > \models < p_0, \$\beta, v, \varepsilon > \vdash < p_1, \beta, v\$, \varepsilon >$$

Нека твърдението е вярно за всяка дума ω' с дължина ненадминаваща n. Нека $\omega = \omega' a$ и нека $\alpha = \omega u \omega$. Нека $\alpha' = \omega'(au)\omega'$. Тогава $\alpha = \alpha' a$. От индукционното предположение за ω' получаваме, че съществува изводът:

$$< p_0, \alpha' a\beta, \varepsilon, \varepsilon > \models < p_1, a\beta, au\omega', \omega' > .$$

Сега от p_1 трябва да направим преход с буква a и следователно, тъй като $a \in \Sigma$ получаваме:

$$< p_1, a\beta, au\omega', \omega' > \vdash < q_a, \beta, au, \omega' a > .$$

Накрая прилагаме и дефиницията за преход от q_a. Тъй като au започва с a, то преходът е дефиниран и:

$$< q_a, \beta, au\omega', \omega' > \vdash < p_1, u\omega' a, \omega' a >,$$

което означава, че:

$$< p_1, a\beta, au\omega', \omega' > \models < p_1, \beta, u\omega, \omega >,$$

откъдето следва и твърдението.

Сега лесно можем да покажем, че:

$$f_{T_{ir}} = ir.$$

Нека първо $!f_{T_{ir}}(t)$. Тогава съществуват $p_i \in \{p_0, p_1\}$, γ и ω, за които:

$$< p_0, t, \varepsilon, \varepsilon > \models < p_i, \varepsilon, \gamma, \omega >$$

и е дефинирана $\phi(p_i)$. Следователно $p_i = p_1$ и $\phi(p_1) = q_\$$. Тогава от втората част на твърдение 2 заключаваме, че $t = \omega\gamma$ и $\gamma = u\omega$ за някое $u \in \Sigma^*\$$. Сега обаче знаем, че съществува изводът:

$$< q_\$, \varepsilon, \gamma, \omega > \models < q, \varepsilon, \varepsilon, \omega' > .$$

В частност има преход от състоянието $q_\$$, което означава, че $u \in \$(\Sigma \cup \$)^*$, което заедно с $u \in \Sigma^*\$$ влече, че $u = \$$. Сега от дефиницията на d и out върху $q_\$$ и f лесно се вижда, че:

$$q = f \text{ и } \omega' = \omega\$\omega.$$

Накрая $\psi(f) = \varepsilon$ и следователно $f_{T_{ir}}(t) = \omega\$\omega = \omega u \omega = t = ir(t)$.

Обратно нека $t = \omega\$\omega$ тогава от втората част на твърдение 2 за $\beta = \varepsilon$ получаваме, че е дефиниран изводът:

$$< p_0, t, \varepsilon, \varepsilon > \models < p_1, \varepsilon, \$\omega, \omega >$$

Сега $\phi(p_1) = q_\$$. И отново лесно се вижда, че:

$$< q_\$, \varepsilon, \$\omega, \omega > \vdash < f, \varepsilon, \omega, \omega\$ > .$$

Накрая тъй като за всяко $a \in \Sigma$ е в сила, че:

$$d(f, a, j) = f$$
$$out(f, a, j) = a$$

за $j = 0, 1$, то очевидно имаме и извода:

$$< f, \varepsilon, \omega, \omega\$ > \models < f, \varepsilon, \varepsilon, \omega\$\omega > .$$

Тъй като $\psi(f) = \varepsilon$ е дефинирана, то t се приема от T_{ir} и $f_{T_{ir}}(t) = t$.

Следствие 2 *Съществува функция, която не е рационална, но се разпознава от FIFO-трансдюсер.*

Доказателство: Както видяхме T_{ir} разпознава ir. От друга страна $Dom(ir) = \{\omega\$\omega \,|\, \omega \in \Sigma^*\}$, което не е регулярен език и тъй като дефиниционната област на всяка рационална функция е регулярен език, то ir не е рационална.

Сега преминаваме към конструкцията на FIFO-трансдюсер за функцията $ins_{0,1} : \{a, b\} \to \{a, b, 0, 1\}$, която дефинирахме като:

$$ins_{0,1}\left(\prod_{i=0}^{k}(a^{m_i} b^{n_i})\right) = \prod_{i=0}^{k}(c_i a^{m_i} b^{n_i})0,$$

за всеки m_i, n_i, за които $m_i > 0$ за $i > 0$ и $b_i > 0$ за $i < k$, а буквите $c_i \in \{0, 1\}$ удовлетворяват условието:

Първо ще построим подпоследователен преобразувател за функцията:

$$pref_{0,1}\left(\prod_{i=0}^{k}(a^{m_i}b^{n_i})\right) = \prod_{i=0}^{k}(c_i a^{m_i} b^{n_i})c_{k+1},$$

при същите условия както за функцията $ins_{0,1}$, но c_i се дефинират като:

$$c_i \equiv \sum_{j=0}^{i-1} n_j (\mod 2).$$

Ако построим такъв преобразувател ще можем да построим и FIFO-трансдюсер за $ins_{0,1}$. В основните състояния ще броим четността на a-тата в цялата дума, докато междувременно я записваме в опашката. След като прочетем думата, в зависимост от тази четност ще предпочетем преобразувателя за $pref_{0,1}$ или $pref_{1,0}$ - двойнствения на $pref_{0,1}$, който на мястото на 0 пише 1 и обратно. Ключовият момент е в това, че:

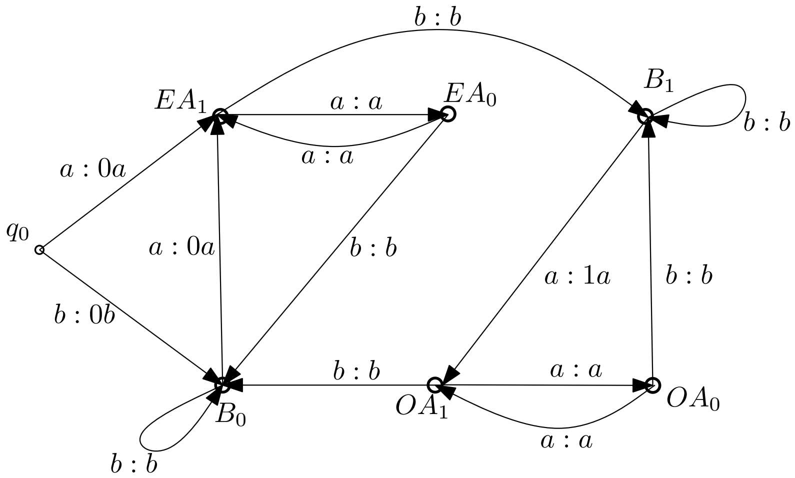
$$\sum_{j=i}^{k} m_j = \sum_{j=0}^{k} m_j - \sum_{j=0}^{i-1} m_j.$$

И така преминаваме към конструкцията на подпоследователен преобразувател, вж. фиг. 2:

$$T_{pref} =< \{a, b\} \times \{a, b, 0, 1\}^*, Q_0, q_0, \delta_0, \lambda_0, \phi_0 >,$$

който ще разпознава $pref_{0,1}$. Да разгледаме следните подмножества от думи, които завършват на a:

$$\begin{aligned}
EA_0 &= \{\alpha a^k \,|\, \alpha \in \{a,b\}^* b \cup \{\varepsilon\}, \alpha \text{ съдържа нечетен брой } a \text{ и } k > 0,\ k \equiv 0(\mod 2)\} \\
EA_1 &= \{\alpha a^k \,|\, \alpha \in \{a,b\}^* b \cup \{\varepsilon\}, \alpha \text{ съдържа четен брой } a \text{ и } k \equiv 1(\mod 2)\} \\
OA_0 &= \{\alpha a^k \,|\, \alpha \in \{a,b\}^* b \cup \{\varepsilon\}, \alpha \text{ съдържа нечетен брой } a \text{ и } k > 0, k \equiv 0(\mod 2)\} \\
OA_1 &= \{\alpha a^k \,|\, \alpha \in \{a,b\}^* b \cup \{\varepsilon\}, \alpha \text{ съдържа нечетен брой } a \text{ и } k \equiv 1(\mod 2)\}
\end{aligned}$$



Фигура 2: Подпоследователен преобпазувател за $pref_{0,1}$

Ясно е че множествата EA_0, EA_1, OA_0 и OA_1 дефинират разбиване на множеството от думи $\{a, b\}^*a$. Дефинираме и множествата:

$$B_0 = \{\alpha \in \{a, b\}^*b \,|\, \alpha \text{ съдържа четен брой } a\} \text{ и}$$
$$B_1 = \{\alpha \in \{a, b\}^*b \,|\, \alpha \text{ съдържа нечетен брой } a\}$$

Ясно е, че B_0 и B_1 дефинират разбиване на множеството от думи $\{a, b\}^*b$. По този начин EA_0, EA_1, OA_0, OA_1, B_0, B_1 и $\{\varepsilon\}$ дефинират едно разбиване на всички думи от $\{a, b\}^*$. Нещо повече лесно се вижда, че всъщност те определят класовете на еквивалентност на една дясно-инвариантна релация на еквивалентност. Именно, директно се проверява, че:

1. За всяка дума $\alpha \in EA_0$, $\alpha \circ a \in EA_1$ и $\alpha \circ b \in B_0$,

2. За всяка дума $\alpha \in EA_1$, $\alpha \circ a \in EA_0$ и $\alpha \circ b \in B_1$,

3. За всяка дума $\alpha \in OA_0$, $\alpha \circ a \in OA_1$ и $\alpha \circ b \in B_1$,

4. За всяка дума $\alpha \in OA_1$, $\alpha \circ a \in OA_0$ и $\alpha \circ b \in B_0$,

5. За всяка дума $\alpha \in B_0$, $\alpha \circ a \in EA_1$ и $\alpha \circ b \in B_0$,

6. За всяка дума $\alpha \in B_1$, $\alpha \circ a \in OA_1$ и $\alpha \circ b \in B_1$,

Нека $Q_0 = \{EA_0, EA_1, OA_0, OA_1, B_0, B_1, q_0\}$. В съответствие с горните наблюдения дефинираме δ_0 като:

$$\begin{aligned}
\delta_0(q_0, a) &= EA_1, \quad & \delta_0(q_0, b) &= B_0 \\
\delta_0(EA_0, a) &= EA_1, \quad & \delta_0(EA_0, b) &= B_0 \\
\delta_0(EA_1, a) &= EA_0, \quad & \delta_0(EA_1, b) &= B_1 \\
\delta_0(OA_0, a) &= OA_1, \quad & \delta_0(OA_0, b) &= B_1 \\
\delta_0(OA_1, a) &= OA_0, \quad & \delta_0(OA_1, b) &= B_0 \\
\delta_0(B_0, a) &= EA_1, \quad & \delta_0(B_0, b) &= B_0 \\
\delta_0(B_1, a) &= OA_1, \quad & \delta_0(B_1, b) &= B_1
\end{aligned}$$

При тази дефиниция лесно се вижда, че:

$$\delta_0^*(q_0, \alpha) = S \in Q_0,$$

за $\alpha \in \{a, b\}^+$, точно когато $\alpha \in S$. Сега дефинираме и функцията на преходите:

$$\begin{aligned}
\lambda_0(q_0, a) &= 0a, \quad & \lambda_0(q_0, b) &= 0b \\
\lambda_0(EA_0, a) &= a, \quad & \lambda_0(EA_0, b) &= b \\
\lambda_0(EA_1, a) &= a, \quad & \lambda_0(EA_1, b) &= b \\
\lambda_0(OA_0, a) &= a, \quad & \lambda_0(OA_0, b) &= b \\
\lambda_0(OA_1, a) &= a, \quad & \lambda_0(OA_1, b) &= b \\
\lambda_0(B_0, a) &= 0a, \quad & \lambda_0(B_0, b) &= b \\
\lambda_0(B_1, a) &= 1a, \quad & \lambda_0(B_1, b) &= b
\end{aligned}$$

Накрая полагаме:

$$\begin{aligned}
\phi_0(EA_0) = \phi_0(OA_1) = \phi_0(B_0) = 0 \\
\phi_0(EA_1) = \phi_0(OA_0) = \phi_0(B_1) = 1.
\end{aligned}$$

Твърдение 3 *Нека* $S \in Q_0 \backslash \{q_0\}$*. Нека* $\alpha \in \Sigma^+$*. Тогава:*

1. $$\alpha \in S \Leftrightarrow \delta_0^*(q_0, \alpha) = S.$$

2. *Ако* $\alpha \in S$*, то*

$$pref_{0,1}(\alpha) = \lambda_0^*(q_0, \alpha)\phi_0(S).$$

Доказателство: Предвид забележката по горе, трябва да докажем само втората част на твърдението. Това правим с индукция по дължината на α. При $\alpha = a$ или $\alpha = b$ твърдението се проверява непосредствено. Тъй като изходът зависи само от предходните символи, то при индукционната стъпка единствените нетривиални случаи са преходите от $\delta_0(B_0, a)$ и $\delta_0(B_1, a)$, защото само при тях започваме нов блок от букви a, пред които трябва да запишем съответното c. Но ако $\alpha \in B_0$, то броят на буквите a в α е четен и съответно $\lambda(B_0, a) = 0a$, а в другия случай $\lambda(B_1, a) = 1a$. По същия начин лесно се съобразява, че функцията ϕ_0 обективно отразява четността на буквите a във всяка дума $\alpha \in S$.

С това доказателството е завършено.

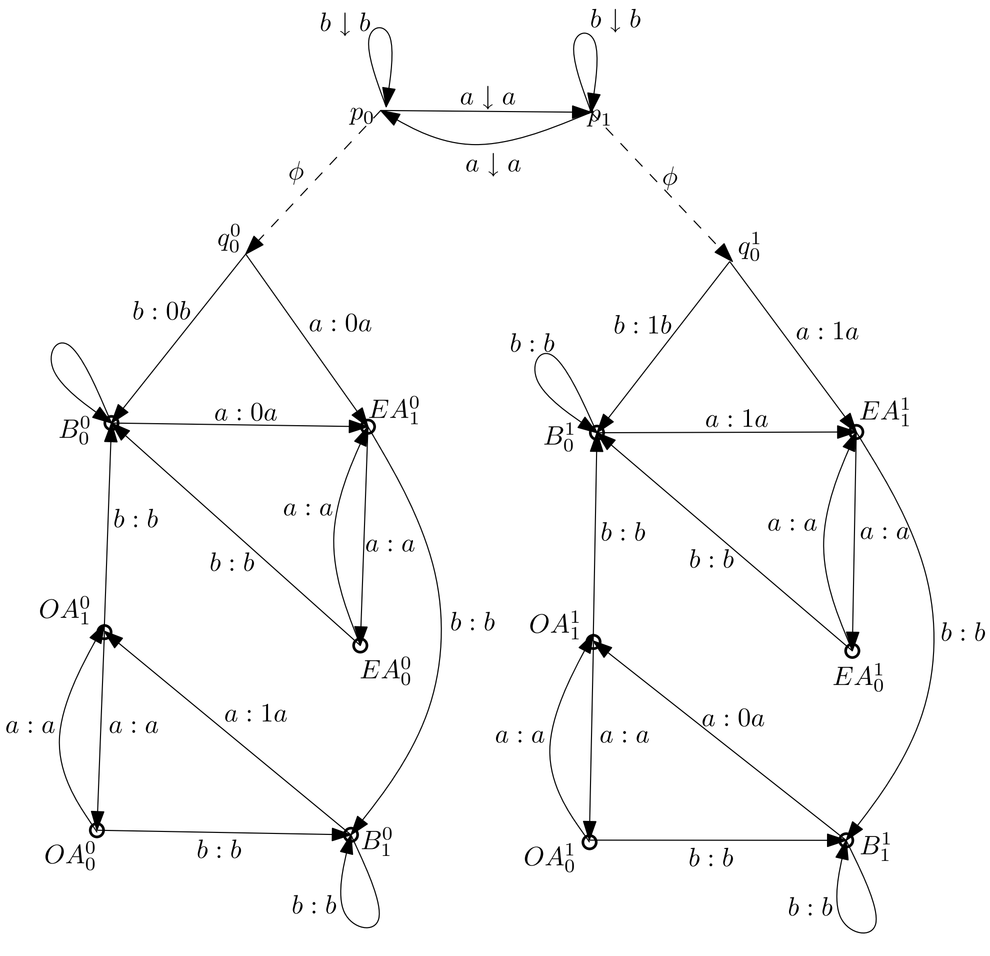
Сега лесно може да се посрои FIFO-трансдюсер за функцията $ins_{0,1}$. Идеята е, че в основните състояния трябва да преброим четността на общия брой букви a, а след като прочетем думата в зависимост от тази четност да изберем преобразувателя $pref_{0,1}$ или $pref_{1,0}$, който вместо 1 пише 0 и обратно. Така можем ще получим FIFO-трансдюсерер, показан на фиг. 3:

$$T_{ins} =< \{a, b\} \times \{a, b, 0, 1\}^*, \{a, b\}, \{p_0, p_1\}, Q_0^0 \cup Q_0^1, p_0, \Delta, d, \lambda, out, \phi, \psi >,$$

където:

1. $Q_0^0 = \{q_0^0, EA_0^0, EA_1^0, OA_0^0, OA_1^0, B_0^0, B_1^0, \}$.

2. $Q_0^1 = \{q_0^1, EA_0^1, EA_1^1, OA_0^1, OA_1^1, B_0^1, B_1^1, \}$.



Фигура 3: FIFO-трансдюсер за функцията $ins_{0,1}$

3. $\Delta : \{p_0, p_1\} \times \{a, b\} \to \{p_0, p_1\} \cup Q_0^0 \cup Q_0^1$ се дефинира като:

$$\begin{aligned}
\Delta(p_0, a) &= p_1 \\
\Delta(p_0, b) &= p_0 \\
\Delta(p_1, a) &= p_0 \\
\Delta(p_1, b) &= p_1.
\end{aligned}$$

4. $\lambda : \{p_0, p_1\} \times \{a, b\} \to \{a, b\}$, се задава като:

$$\begin{aligned}
\lambda(p_i, a) &= a \text{ за } i = 0, 1 \\
\lambda(p_i, b) &= b \text{ за } i = 0, 1.
\end{aligned}$$

5. $d : Q_0^0 \cup Q_0^1 \times \{a, b\} \times \{0, 1\} \to \{p_0, p_1\} \cup Q_0^0 \cup Q_0^1$ е определена от равенствата:

$$d(S^j, x, 0) = d(S^j, x, 1) = T^j, \text{ за всеки } j \in \{0, 1\}, x \in \{a, b\}, \text{ където } T = \delta_0(S, x).$$

6. $out : Q_0^0 \cup Q_0^1 \times \{a, b\} \times \{0, 1\} \to \{a, b, 0, 1\}^*$ се задава по подобен начен, именно:

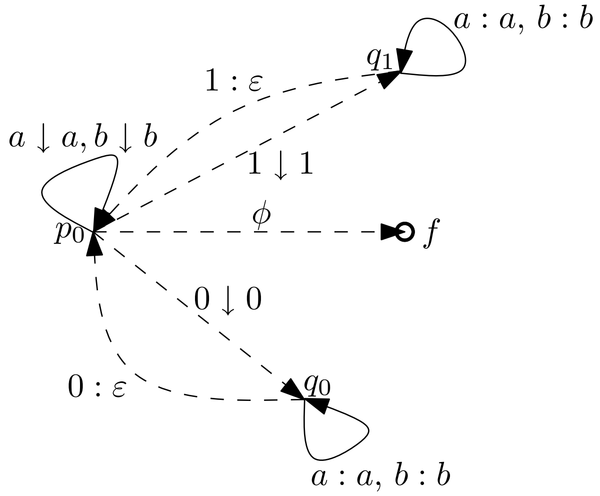
$$out(S^j, x, 0) = \begin{cases} \lambda_0(S, x) \text{ при } j = 0 \\ \lambda_0(S, x), \text{ където всяка } 0 \text{ е заменена с } 1 \text{ и обратно при } j = 1. \end{cases}$$

7. $\phi : \{p_0, p_1\} \to Q_0^0 \cup Q_0^1$ е определена като:

$$\begin{aligned}
\phi(p_0) &= q_0^0 \\
\phi(p_1) &= q_0^1.
\end{aligned}$$

8. $\psi : Q_0 \cup Q_1 \to \{a, b, 0, 1\}^*$ се дефинира като:

$$\begin{aligned}
\psi(S^0) &= \phi_0(S) \text{ за всяко } S \in Q_0 \\
\psi(S^1) &= \phi_0(S), \text{ в която всяка } 0 \text{ е заменена с } 1 \text{ и обратно.}
\end{aligned}$$



Фигура 4: FIFO-трансдюсер за функцията $rep_{0,1}$

Аналогично можем да построим FIFO-трансдюсер T_{rep} и за $rep_{0,1}$, виж фигура 4:

$T_{rep} =< \{a, b, 0, 1\} \times \{a, A, b\}, \{a, b, 0, 1\}^*, \{p_0\}, \{q_0, q_1, f\}, p_0, \Delta, d, \lambda, out, \phi, \psi >$,

където:

1. $\Delta : \{p_0\} \times \{a, b, 0, 1\} \to \{p_0\} \cup \{q_0, q_1\}$ се дефинира като:

$$\begin{aligned} \Delta(p_0, a) = \Delta(p_0, b) &= p_0 \\ \Delta(p_0, 0) &= q_0 \\ \Delta(p_0, 1) &= q_1. \end{aligned}$$

2. $\lambda : \{p_0\} \times \{a, b, 0, 1\} \to \{a, b, 0, 1\}^*$ се определя като:

$$\lambda(p_0, x) = x \text{ за всяко } x = a, b, 0, 1.$$

3. $d : \{q_0, q_1, f\} \times \{a, b, 0, 1\} \to \{p_0\} \cup \{q_0, q_1, f\}$ е дефинирана само

за q_0 и q_1 и:

$$d(q_0, x, j) = \begin{cases} q_0 \text{ ако } x \in \{a, b\}, j \in \{0, 1\} \\ p_0 \text{ ако } x = 0, j \in \{0, 1\} \\ \neg! \text{ иначе.} \end{cases}$$

$$d(q_0, x, j) = \begin{cases} q_1 \text{ ако } x \in \{a, b\}, j \in \{0, 1\} \\ p_0 \text{ ако } x = 1, j \in \{0, 1\} \\ \neg! \text{ иначе.} \end{cases}$$

4. $out : \{q_0, q_1, f\} \times \{a, b, 0, 1\} \times \{0, 1\} \to \{a, A, b\}^*$ се дефинира от равенствата:

$$out(q_0, x, j) = \begin{cases} x \text{ ако } x \in \{a, b\}, j = 0, 1 \\ \varepsilon \text{ ако } x = 0, j = 0, 1 \\ \neg! \text{ иначе.} \end{cases}$$

$$out(q_1, x, j) = \begin{cases} A \text{ ако } x = a, j = 0, 1 \\ b \text{ ако } x = b, j = 0, 1 \\ \varepsilon \text{ ако } x = 1, j = 0, 1 \\ \neg! \text{ иначе.} \end{cases}$$

5. $\phi(p_0) = f$.

6. $\psi : \{q_0, q_1, f\} \to \{a, A, b\}^*$ е дефинирана само в състоянието f и:

$$\psi(f) = \varepsilon.$$

Формалното доказателство, че T_{ins} и T_{rep} разпознават съответно $ins_{0,1}$ и $rep_{0,1}$ може да бъде направено по начин, подобен на този, по който доказахме коректността на FIFO-трансдюсера T_{ir}. Затова ние няма да излагаме тези доказателства, а фактът, че $ins_{0,1}$ и $rep_{0,1}$ могат да се представят с FIFO-трансдюсер ще отложим за част 4, където ще разгледаме клас от функции, за които такъв трансдюсер може да се построи и лесно ще проверим, че функциите $ins_{0,1}$ и $rep_{0,1}$ попадат в този клас.

2.3 Каноничен FIFO-трансдюсер. Еквивалентност

В тази част ще покажем, че можем да ограничим записите, които правим в опашката до еднобуквени думи, за всяко състояние и всеки входен символ. Започваме със следната:

Дефиниция 7 *FIFO-трансдюсер,* $T =< \Sigma\times\Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$, *за който* $\Gamma = P \times \Sigma$, *а* $\lambda(p, \sigma) = (p, \sigma)$ *за всяка двойка* $(p, \sigma) \in P \times \Sigma$ *ще наричаме каноничен. Когато се подразбира, че става въпрос за канонични FIFO-трансдюсери ще изпускаме,* Γ *и* λ *сигнатурата на* T*, тоест ще пишем* $T =< \Sigma \times \Omega^*, P, Q, i, \Delta, d, out, \phi, \psi >$.

В сила е следната теорема.

Теорема 3 *За всеки FIFO-трансдюсер* $T =< \Sigma\times\Omega^*, \Gamma^*, P, Q, i, \Delta, d, \lambda, out, \phi, \psi >$, *съществува каноничен FIFO-трансдюсер* $\widetilde{T} =< \Sigma\times\Omega^*, \widetilde{P}, \widetilde{Q}, \widetilde{i}, \widetilde{\Delta}, \widetilde{d}, \widetilde{out}, \widetilde{\phi}, \widetilde{\psi} >$, *който е еквивалентене на* T*, тоест* $\mathcal{L}(T) = \mathcal{L}(\widetilde{T})$.

Идеята на доказателството е да симулираме изпълнението, тоест изводите, на T. За целта винаги, когато четем символ $(p, a) \in P \times \Sigma$ от опашката на $\widetilde{T}$ ние ще се опитваме да имитираме поведението на T върху прочетена дума $\lambda(p, a)$. Това обаче няма да ни се одава винаги.

Тъй като $\widetilde{T}$ задължително трябва да пише в опашката си точно по един символ на всеки прочетен символ, това ще предизвика разминаване ежду $\widetilde{T}$ и T - от време на време $\widetilde{T}$ ще трябва да изчаква, защото той не е записвал нищо, а от време на време, ще трябва да го настига, като обработва по-големи последователности от букви наведнъж.

За да се справим с този проблем, в състоянията на трансдюсера, който ще строим ще пазим допълнително и информация, за необработената част от началото на опашката на T, която обаче $\widetilde{T}$ е прочел наведнъж, но T е обработил реално само част от нея. От друга страна може да се окаже, че в определен момент T е обработил част от началото на опашката си и е извел нещо на изходната лента. Тъй като обаче цялата тази информация ще се съдържа само в едно състояние на $\widetilde{p} \in \widetilde{P}$, то ще симулира преходите в T и ще премине в друго състояние $\widetilde{p}' \in \widetilde{P}$ без да има възможност да извърши запис върху изходната лента. Този изход също трябва да бъде запазен и това може да стане само посредством ново състояние на $\widetilde{P}$.

Проблем предизвикват и състоянията $\widetilde{Q}$ - за да направим преход от тях ние трябва да знаем дали работим с последния символ в опашката (но не тази в $\widetilde{T}$, а тази в T) или не. Опасността тук е, че е възможно оригиналният трансдюсер да не е записал нищо, докато каноничният е бил принуден да поставя записи (p, a), за които $\lambda(p, a) = \varepsilon$. Решението тук е отново да изчакаме - да запазим последната буква от думата, която четем, симулирайки оригиналния автомат, докато не се убедим, че след нея следват още значещи. Ако това не е така, то ние ще прочетем всички символи от опашката на $\widetilde{T}$ - те ще бъдат ε и тогава ще знаем, отново детерминирано, че трябва да симулираме работата на T с последния символ в опашката, който ние предвидливо сме запазили.

Възниква естественият въпрос дали ще успеем да организираме съхранението на толкова много информация само с краен брой състояния. Оказва се, че това е възможно, благодарение на факта, че наставките на думите от $\lambda(p, a)$, които T може да запише в опашката са краен брой. Всъщност те, заедно с всевъзможните изходи по представките на тези думи, е онази допълнителна информация, която ще осигури желаната от нас симулация.

Сега пристъпваме към формалното излагане на конструкцията. Започваме с дефиницията на три функции, чиято цел е да опишат възможно най-пълно поведението на T на базата на част от началото на опашката на T и състояние $q \in Q$.

Дефиниция 8 *$walk : Q \times \Gamma^* \to P \cup Q$, $rest : Q \times \Gamma^* \to \Gamma^*$ и $print : Q \times \Gamma^* \to \Omega^*$ се дефинират за всяко $q \in Q$ и всяка дума $\gamma \in \Gamma^*$, индуктивно по дължината на γ.*

1. *$|\gamma| \leq 1$, тогава $walk(q, \gamma) = q$, $rest(q, \gamma) = \gamma$ и $print(q, \gamma) = \varepsilon$.*

2. *$\gamma = b\gamma'$, $|\gamma'| \geq 1$.*

(а) ако $d(q, b, 1) \in P$*, то*

$$\begin{aligned} walk(q, \gamma) &= d(q, b, 1), \\ rest(q, \gamma) &= \gamma', \\ print(q, \gamma) &= out(q, b, 1). \end{aligned}$$

(б) ако $d(q, b, 1) \in Q$*, то*

$$\begin{aligned} walk(q, \gamma) &= walk(d(q, b, 1), \gamma'), \\ rest(q, \gamma) &= rest(d(q, b, 1), \gamma'), \\ print(q, \gamma) &= out(q, b, 1) \circ print(d(q, b, 1), \gamma'). \end{aligned}$$

(в) ако $d(q, b, 1)$ *не е дефинирана, то и функциите* $walk$*,* $rest$ *и* $print$ *са недефинирани за* q *и* γ*.*

Забележка 3 *От дефиницията на функцията на* $rest$ *директно следва, че* $rest(q, \gamma)$ *е наставка на* γ *за всеки* q *и* γ*, за които е дефинирана. Освен това ако* $|\gamma| > 1$*, то* $|rest(q, \gamma)| < |\gamma|$*.*

Следващата лема отразява нашата интуиция за тези функции:

Лема 1 *Нека* $q \in Q$*,* $\gamma_1 \in \Gamma^*$*,* $\gamma_2 \in \Gamma^*$*,* $\alpha \in \Sigma^*$ *и* $\omega \in \Omega^*$ *са произволни. Тогава:*

1. $(q, \alpha, \gamma_1\gamma_2, \omega) \vDash (walk(q, \gamma_1), \alpha, rest(q, \gamma_1) \circ \gamma_2, \omega \circ print(q, \gamma_1))$.
2. *При това ако* $|rest(q, \gamma_1)| \geq 2$*, то* $walk(q, \gamma_1) \in P$*.*
3. *Ако функциите* $walk(q, \gamma_1)$*, съответно* $rest(q, \gamma_1)$ *и* $print(q, \gamma_1)$ *не са дефинирани, то не съществува извод с начало* $(q, \alpha, \gamma_1\gamma_2, \omega)$ *с дължина по-голяма от* $|\gamma_1| - 2$*. При това всеки такъв извод има вида:*
$$(q, \alpha, \gamma_1\gamma_2, \omega) \vDash (q', \alpha, \gamma_1'\gamma_2, \omega\omega'),$$
където $q' \in Q$ *и* γ_1' *е наставка на* γ_1*.*

Доказателство: Доказателство е с индукция по дължината на γ_1. И трите твърдения са очевидни при $|\gamma_1| \leq 1$. Първото е тавтология, а 2 и 3 са с грешна предпоставка.

Да допуснем, че твърденията са верни за произволни $q \in Q$ и $\gamma_1 \in \Gamma^*$, за които $|\gamma_1| = n$. Нека $q \in Q$, $\gamma_1 \in \Gamma^*$, $\gamma_2 \in \Gamma^*$, $\alpha \in \Sigma^*$ и $\omega \in \Omega^*$ са произволни като $|\gamma_1| = n + 1$. Тогава $\gamma_1 = b\gamma_1'$ като $|\gamma_1'| = n \geq 1$. Тогава тъй като $q \in Q$ то единственият възможен преход в T от конфигурацията $(q, \alpha, \gamma_1\gamma_2, \omega)$ е:

$$(q, \alpha, \gamma_1\gamma_2, \omega) \vdash (d(q, b, 1), \alpha, \gamma_1' \circ \gamma_2, \omega \circ out(q, b, 1)).$$

От дефиниция 8 трябва да разгледаме три случая:

- $d(q, b, 1) \in P$. Тогава $walk(q, \gamma_1) = d(q, b, 1)$, $rest(q, \gamma_1) = \gamma_1'$ и $print(q, \gamma_1) = out(q, b, 1)$ и 1 е изпълнено тривиално, 2 и 3 също са в сила.

- $d(q, b, 1) \in Q$. Тогава $walk(q, \gamma) = walk(d(q, b, 1), \gamma')$, $rest(q, \gamma) = rest(d(q, b, 1), \gamma')$ и $print(q, \gamma) = out(q, b, 1)print(d(q, b, 1), \gamma')$. Тогава от индукционното предположение за $d(q, b, 1)$ и γ_1' имаме, че

$$(d(q, b, 1), \alpha, \gamma_1'\gamma_2, \omega \circ out(q, b, 1)) \models$$
$$(walk(d(q, b, 1), \gamma_1'), \alpha, rest(d(q, b, 1), \gamma_1') \circ \gamma_2, \omega \circ out(q, b, 1) \circ print(d(q, b, 1), \gamma_1')),$$

и следователно:

$$(q, \alpha, \gamma_1\gamma_2, \omega) \models (walk(q, \gamma_1), \alpha, rest(q, \gamma_1) \circ \gamma_2, \omega \circ print(q, \gamma_1).$$

Освен това ако $|rest(q, \gamma_1)| = |rest(d(q, b, 1), \gamma_1')| \geq 2$, то от индукционното предполежение получаваме, че $walk(d(q, b, 1), \gamma_1') \in P$, откъдето непосредствено следва, че и $walk(q, \gamma_1) \in P$.

Накрая, ако $walk(q, \gamma_1)$ не е дефинирана, то и $walk(d(q, b, 1), \gamma_1')$ не е дефинирана и от индукционната хипотеза следва, че няма извод от $(d(q, b, 1), \alpha, \gamma_1'\gamma_2, \omega \circ out(q, b, 1))$ с дължина по-голяма от $|\gamma_1'| - 2$. Тогава и от $(q, \alpha, \gamma_1\gamma_2, \omega) = (q, \alpha, b\gamma_1'\gamma_2, \omega)$ не съществува извод с дължина по-голяма от $1 + |\gamma_1'| - 2 = |\gamma_1| - 2$.

- $d(q,b,1)$ не е дефинирана. Тогава трябва да проверим само третото условие и то е налице, защото $|\gamma_1| \geq 2$.

С това твърденията на лемата следват от принципа за математическа индукция.

Сега можем да пристъпим към конструкцията на $\widetilde{T}$. Както вече отбелязахме важна роля при тази конструкция ще играят наставките на думите $\lambda(p,a)$ и възможните изходи по представките на тези думи. Затова нека положим:

$$\begin{aligned} Suf(\lambda) &= \{\gamma \in \Gamma^* \quad | \quad \exists p \in P, \sigma \in \Sigma, \beta \in \Gamma^*, \text{ за които} \\ &\quad\ \lambda(p,\sigma) = \beta \circ \gamma\} \text{ и} \\ Pref(out) &= \{print(q,\gamma) \,|\, q \in Q, \gamma \in Suf(\lambda)\}. \end{aligned}$$

При тези означения дефинираме компонентите на $\widetilde{T} =< \Sigma \times \Omega^*, \widetilde{P}, \widetilde{Q}, \widetilde{i}, \widetilde{\Delta}, \widetilde{d}, \widetilde{out}, \widetilde{\phi}, \widetilde{\psi} >$.

1. $\widetilde{P} = P \times Suf(\lambda) \times Pref(out)$.

2. $\widetilde{Q} = Q \times (\Gamma \cup \{\varepsilon\}) \times Pref(out)$.

3. $\widetilde{i} =< i, \varepsilon, \varepsilon >$.

4. $\widetilde{\Delta} : \widetilde{P} \times \Sigma \to \widetilde{P} \cup \widetilde{Q}$, която дефинираме така:

$$\widetilde{\Delta}(< p,\gamma,\omega >, \sigma) = \begin{cases} < \Delta(p,\sigma), \gamma, \omega > \ \text{ако } \Delta(p,\sigma) \in P \\ < walk(q,\gamma), rest(q,\gamma), \omega \circ print(q,\gamma) >, \\ \text{където } q = \Delta(p,\sigma) \in Q. \\ \\ \neg! \text{ иначе.} \end{cases}$$

5. $\widetilde{d} : \widetilde{Q} \times (\widetilde{P} \times \Sigma) \times \{0, 1\} \to \widetilde{P} \cup \widetilde{Q}$ дефинираме чрез:

$\widetilde{d}(< q, b, \omega >, (< p, \beta, \omega' >, \sigma), 1) =< walk(q, b\gamma), rest(q, b\gamma), \varepsilon >$, където $\gamma = \lambda(p, \sigma)$.

$$\widetilde{d}(< q, b, \omega >, (< p, \beta, \omega' >, \sigma), 0) = \begin{cases} < walk(q, b\gamma), rest(q, b\gamma), \varepsilon >, \\ \text{където } \gamma = \lambda(p, \sigma) \text{ и } walk(q, b\gamma) \in P, \\ \\ < d(walk(q, b\gamma), rest(q, b\gamma), 0), \varepsilon, \varepsilon >, \\ \text{където } \gamma = \lambda(p, \sigma) \text{ и } walk(q, b\gamma) \in Q \text{ и} \\ rest(q, b\gamma) \neq \varepsilon \\ \\ < q, \varepsilon, \varepsilon > \\ \text{където } \gamma = \lambda(p, \sigma) \text{ и } walk(q, b\gamma) \in Q \text{ и} \\ rest(q, b\gamma) = \varepsilon \\ \\ \neg! \text{ иначе.} \end{cases}$$

Трябва да отбележим, че d е коректно дефинирана. Това следва директно от лема 1 3 забележка 3.

6. $\widetilde{out} : \widetilde{Q} \times (\widetilde{P} \times \Sigma) \times \{0,1\} \to \Omega^*$ дефинираме чрез равенствата:

$$\widetilde{out}(< q, b, \omega >, (< p, \beta, \omega' >, \sigma), 1) = \omega \circ print(q, b\gamma), \text{ където } \gamma = \lambda(p, \sigma)$$

$$\widetilde{out}(< q, b, \omega >, (< p, \beta, \omega' >, \sigma), 0) = \begin{cases} \omega \circ print(q, b\gamma), \\ \text{където } \gamma = \lambda(p, \sigma) \text{ и } walk(q, b\gamma) \in P, \\ \\ \omega \circ print(q, b\gamma) \circ out(walk(q, b\gamma), rest(q, b\gamma), 0), \\ \text{където } \gamma = \lambda(q, \sigma) \text{ и } walk(q, b\gamma) \in Q \\ \ \text{и } rest(q, b\gamma) \neq \varepsilon \\ \\ \omega \\ \text{където } \gamma = \lambda(q, \sigma) \text{ и } walk(q, b\gamma) \in Q \text{ и} \\ rest(q, b\gamma) = \varepsilon \\ \\ \neg! \text{ иначе.} \end{cases}$$

7. $\widetilde{\phi} : \widetilde{P} \to \widetilde{Q}$ дефинираме с равенството:

$$\widetilde{\phi}(< p, \gamma, \omega >) =< walk(\phi(p), \gamma), rest(\phi(p)), \omega \circ print(\phi(p), \gamma) > .$$

8. $\widetilde{\psi} : \widetilde{Q} \to \Omega^*$ дефинираме с равенството:

$$\widetilde{\psi}(< q, b, \omega >) = \begin{cases} \omega \circ out(q, b, 0) \circ \psi(d(q, b, 0)), \text{ ако } b \in \Gamma, \\ \omega \circ \psi(q), \text{ ако } b = \varepsilon. \end{cases}$$

Преди да пристъпим към доказателството на основното твърдение, от което ще следва, че $\mathcal{L}(\widetilde{T}) = \mathcal{L}(T)$ ще ни бъде нужно едно последно означение. То ще съответства на естествения хомоморфизъм $\widetilde{\lambda}^* : \widetilde{P} \times \Sigma \to \Gamma^*$, породен от функцията $\lambda : P \times \Sigma \to \Gamma^*$. Формално това може да бъде направено така:

Дефиниция 9 $\widetilde{\lambda}^* : (\widetilde{P} \times \Sigma)^* \to \Gamma^*$ *се дефинира индуктивно по дължината на думата* $\widetilde{\gamma}$.

1. ако $\gamma = \varepsilon$, то $\widetilde{\lambda}^*(\gamma) = \varepsilon$.

2. ако $\gamma = (< p, \beta, \omega >, \sigma) \circ \gamma'$, то $\widetilde{\lambda}^*(\gamma) = \lambda(p, \sigma)\widetilde{\lambda}^*(\gamma')$.

Сега сме в състояние да формулираме основната лема в доказателството на теорема 3:

Лема 2 *Нека $\widetilde{p_1} = (p_1, \gamma_1, \omega_1) \in \widetilde{P} \cup \widetilde{Q}$ и $\widetilde{p_2} = (p_2, \gamma_2, \omega_2) \in \widetilde{P} \cup \widetilde{Q}$ са две произволни състояния на $\widetilde{T}$. Нека още $\alpha_1, \alpha_2 \in \Sigma^*$, $\beta_1, \beta_2 \in (\widetilde{P} \times \Sigma)^*$ и $\pi_1, \pi_2 \in \Omega^*$ са произволни думи. Тогава*

1. ако $(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \models (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2)$, то $(p_1, \alpha_1, \gamma_1 \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \models (p_2, \alpha_2, \gamma_2 \circ \widetilde{\lambda}^*(\beta_2), \pi_2 \circ \omega_2)$.

2. ако $p_2 \in P$ или $\widetilde{\beta_2} = \varepsilon$ и $(p_1, \alpha_1, \gamma_1 \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \models (p_2, \alpha_2, \gamma_2 \circ \widetilde{\lambda}^*(\beta_2), \pi_2 \circ \omega_2)$, то съществува и извод $(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \models (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2)$ за някои $\beta_2 \in (\widetilde{P} \times \Sigma)^*$ и $\pi_1, \pi_2 \in \Omega^*$ и някое $\widetilde{p_2} = (p_2, \gamma_2, \omega_2) \in \widetilde{P} \cup \widetilde{Q}$.

Доказателство:

1. Доказателството ще извършим с индукция по дължината на извода
$$(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \models (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2).$$

   (a) Ако $(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) = (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2)$, то получаваме, че $p_1 = p_2$, $\gamma_1 = \gamma_2$, $\omega_1 = \omega_2$, $\alpha_1 = \alpha_2$, $\beta_1 = \beta_2$ и $\pi_1 = \pi_2$. Това показва, че $\gamma_1 \circ \widetilde{\lambda}^*(\beta_1) = \gamma_2 \circ \widetilde{\lambda}^*(\beta_2)$, $\pi_1 \circ \omega_1 = \pi_2 \circ \omega_2$ и следователно
   $(p_1, \alpha_1, \gamma_1 \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \models (p_2, \alpha_2, \gamma_2 \circ \widetilde{\lambda}^*(\beta_2), \pi_2 \circ \omega_2)$.

   (б) Нека изводът
$$(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \models (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2)$$
   се състои от прехода
$$(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \vdash (\widetilde{p}, \alpha, \beta, \pi)$$

и извода

$$(\widetilde{p}, \alpha, \beta, \pi) \vDash (\widetilde{p_2}, \alpha_2, \beta_2, \pi_2)$$

В зависимост от типа на прехода

$$(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \vdash (\widetilde{p}, \alpha, \beta, \pi)$$

ще разгледме трите възможни случая:

- $\widetilde{p_1} \in \widetilde{P}$. Тогава $\alpha_1 = \sigma \circ \alpha$. Нека $p = \Delta(p_1, \sigma)$. Ако $p \in P$ то $\widetilde{p} = \widetilde{\Delta}(\widetilde{p_1}, \sigma) =< p, \gamma_1, \omega_1 >$. От друга страна тъй като $\widetilde{p_1} \in \widetilde{P}$, то и $p_1 \in P$ тогава очевидно T допуска прехода:

  $$(p_1, \alpha_1, \gamma_1 \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \vDash (p, \alpha, \gamma_1 \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1).$$

  Тогава твърдението следва от индукционната хипотеза приложена за $\widetilde{\Delta}(p, \sigma)$ и $\widetilde{p_2}$. Ако пък $q = \Delta(p_1, \sigma) \in Q$ то $\widetilde{\Delta}(\widetilde{p_1}, \sigma) =< walk(q, \gamma_1), rest(q, \gamma_1), \omega_1 \circ print(q, \gamma_1) >$. Сега от лема 1, 1 следва, че за всеки $\alpha \in \Sigma^*$, $\beta \in \Gamma^*$ и $\pi \in \Omega^*$ съществува преход:

  $$(q, \alpha, \gamma_1\beta, \pi) \vDash (walk(q, \gamma_1), rest(q, \gamma_1) \circ \beta, \pi \circ print(q, \gamma).$$

  Сега като положим $\beta = \widetilde{\lambda}^*(\beta_1)$, $\pi = \pi_1 \circ \omega_1$ получаваме, че T допуска извода:

  $$\begin{gathered}(q, \alpha, \gamma_1\widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \vDash \\ (walk(q, \gamma_1), rest(q, \gamma_1) \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1 \circ print(q, \gamma)).\end{gathered}$$

  Следователно

  $$\begin{gathered}(p, \alpha_1, \gamma_1\widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \vdash \\ (q, \alpha, \gamma_1\widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1) \vDash \\ (walk(q, \gamma_1), rest(q, \gamma_1) \circ \widetilde{\lambda}^*(\beta_1), \pi_1 \circ \omega_1 \circ print(q, \gamma))\end{gathered}$$

или, което означава, че:

$$(p,\alpha_1,\gamma_1\widetilde{\lambda}^*(\beta_1),\pi_1\circ\omega_1)\models$$
$$(walk(q,\gamma_1),rest(q,\gamma_1)\circ\widetilde{\lambda}^*(\beta_1),\pi_1\circ\omega_1\circ print(q,\gamma)).$$

Сега твърдението следва от индукционнота хипотеза, приложена за $\widetilde{\Delta}(\widetilde{p_1},\sigma)$ и $\widetilde{p_2}$.

- Нека $\widetilde{p_1}\in\widetilde{Q}$ и $\beta_1=(\widetilde{q},\sigma)\circ\beta$, където $\widetilde{q}=<q,s,t>$ и $\beta\neq\varepsilon$. Тогава $\alpha=\alpha_1$, $(p,\gamma,\varepsilon)=\widetilde{p}=\widetilde{d}(\widetilde{p_1},(\widetilde{q},\sigma),1)$ и $\pi=\pi_1\circ\omega_1\circ\widetilde{out}(\widetilde{p_1},(\widetilde{q},\sigma),1)$. Тогава от дефинициите на $\widetilde{d}$ и $\widetilde{out}$ $p=walk(p_1,\gamma_1\circ\lambda(q,\sigma))$ и $\gamma=rest(p,\gamma_1\circ\lambda(p,\sigma))$. Сега, прилагайки отново лема 1, за p_1 и $\gamma_1\circ\lambda(q,\sigma)$ получаваме, че T допуска извода:

$$(p_1,\alpha,\gamma_1\circ\lambda(q,\sigma)\circ\widetilde{\lambda}^*(\beta),\pi_1\circ\omega_1)\models$$
$$(p,\alpha,\gamma\circ\widetilde{\lambda}^*(\beta),\pi_1\circ\omega_1\circ print(p_1,\gamma_1\circ\lambda(q,\sigma)))=$$
$$(p,\alpha,\gamma\circ\widetilde{\lambda}^*(\beta),\pi).$$

Сега твърдението следва от индукционната хипотеза, приложена за $\widetilde{p}=<p,\gamma,\varepsilon>$ и $\widetilde{p_2}$.

- Остана да разгледаме случая, когато $\widetilde{p_1}\in\widetilde{Q}$ и $\beta_1=(\widetilde{q},\sigma)$, където $\widetilde{q}=<q,s,t>$. От тук следва, че $\beta=\varepsilon$. Ако $walk(p_1,\gamma_1\circ\lambda(q,\sigma))\in P$, то прилагайки разсъждения, аналогични на тези от предишния случай, получаваме, че $\widetilde{p}=(walk(p_1,\gamma_1\circ\lambda(q,\sigma)),rest(p_1,\gamma_1\circ\lambda(q,\sigma)),\varepsilon)$ и $\pi=\pi_1\circ\omega_1\circ print(p_1,\gamma_1\circ\lambda(q,\sigma)$. Тогава отново от лема 1 следва, че

$$(p_1,\alpha_1,\gamma_1\circ\lambda(q,\sigma),\pi_1\circ\omega_1)\models$$
$$(p,\alpha_1,rest(p_1,\gamma_1\circ\lambda(q,\sigma)),\pi)$$

и можем да приложим индукционната хипотеза за $\widetilde{p}=<p,rest(p_1,\gamma_1\circ\lambda(q,\sigma)),\varepsilon>$ и p_2, защото $\beta=\varepsilon$.
По-интересен е случаят, когато $walk(p_1,\gamma_1\circ\lambda(q,\sigma))\in Q$. Тогава от втората част на лема 1 следва, че $|rest(p_1,\gamma_1\circ$

$\lambda(q,\sigma)| \leq 1$. Нека $p' = walk(p_1, \gamma_1 \circ \lambda(q,\sigma)$ и $b = rest(p_1, \gamma_1 \circ \lambda(q,\sigma) \in \Gamma \cup \{\varepsilon\}$. Тогава отново от лема 1 получаваме, че

$$(p_1, \alpha_1, \gamma_1 \circ \lambda(q,\sigma), \pi_1 \circ \omega_1) \vDash$$
$$(p', \alpha_1, b, \pi_1 \circ \omega_1 \circ print(p_1, \gamma_1 \circ \lambda(q,\sigma))).$$

Сега тъй като $\widetilde{d}(\widetilde{p_1}, (\widetilde{q}, \sigma)), 0)$ е дефинирана и $p' \in Q$, то е дефинирана и функцията $d(p', b, 0)$, при $b \neq \varepsilon$. В този случай съществува преход:

$$(p', \alpha_1, b, \pi_1 \circ \omega_1 \circ print(p_1, \gamma_1 \circ \lambda(q,\sigma))) \vdash$$
$$(d(p', b, 0), \alpha_1, \varepsilon, \pi_1 \circ \omega_1 \circ print(p_1, \gamma_1 \circ \lambda(q,\sigma)) \circ out(p', b, 0)) =$$
$$(p, \alpha, \varepsilon, \pi_1 \widetilde{out}(\widetilde{p_1}, (\widetilde{q}, \sigma), 0)) =$$
$$(p, \alpha, \varepsilon, \pi).$$

Сега тъй като $\widetilde{p} =< p, \varepsilon, \varepsilon >$, $\beta = \varepsilon$, то прилагайки индукционната хипотеза за $\widetilde{p}$ и $\widetilde{p_2}$ получаваме желания резултат.
Ако пък $b = \varepsilon$, то непосредствено се вижда, че и $\widetilde{\lambda}^*(\beta') = \varepsilon$ и отново твърдението следва.

2. Сега преминаваме към втората част на твърдението.

   - Да допуснем, че твърдението не е вярно и да предположим, че съществува извод в T:

     $$(p_1, \alpha_1, \gamma_1 \widetilde{\beta_1}, \pi_1 \circ \omega_1) \vDash (p_2, \alpha_2, \beta_2, \pi_2),$$

     където сме положили $\widetilde{\beta_1} = \widetilde{\lambda}^*(\beta_1)$. Нека с

     $$(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1) \vDash (\widetilde{p}, \alpha, \beta, \pi)$$

     означим най-дългият възможен извод с начало $(\widetilde{p_1}, \alpha_1, \beta_1, \pi_1)$. Да допуснем, че $|\alpha| < |\alpha_2|$. Тогава тъй като на всеки преход дължината на думата α_1 намалява най-много с 1 и това се случва точно при преходи от състоянията в $\widetilde{P}$, то ще съществува конфигурация, в този извод $(\widetilde{p'}, \alpha', \beta', \pi')$, при която $|\alpha'| = |\alpha_2|$ и $\widetilde{p'} \in \widetilde{P}$. От твърдение 1 следва, че $\alpha' = \alpha_2$. Нека $\widetilde{p'} =< p', \gamma', \omega' >$. От първата част на лемата, ще следва, че от конфигурацията $(p_1, \alpha_1, \gamma_1\widetilde{\beta_1}, \pi_1\omega_1) \vDash (p', \alpha', \gamma', \pi')$. Сега отново тъй като дължината на четената дума не расте, а намалява точно в преходи от състоянията в P ще следва, че $\alpha_2 = \alpha'$ и $p' = p_2$. Но тогава ако приложим първата част на лемата за $\widetilde{p_1}$, и $\widetilde{p'}$ ще получим точно твърдението, което твърди втората част на лемата, а ние допуснахме, че то е грешно.

И така $|\alpha| \geq |\alpha_2|$, като ако имаме равенство, то $p' \notin P$. Сега от първата част на лемата получаваме, че изводът

$$(p_1, \alpha_1, \gamma_1\widetilde{\beta_1}, \pi_1 \circ \omega_1) \vDash (p_2, \alpha_2, \beta_2, \pi_2)$$

е съставен от двата извода:

$$(p_1, \alpha_1, \gamma_1\widetilde{\beta_1}, \pi_1 \circ \omega_1) \vDash (p, \alpha, \gamma\widetilde{\lambda}^*(\beta), \pi \circ \omega)$$

и

$$(p, \alpha, \gamma\widetilde{\lambda}^*(\beta), \pi \circ \omega) \vDash (p_2, \alpha_2, \beta_2, \pi_2)$$

Тъй като това е най-дългият възможен извод от $(\widetilde{p_1}, \alpha_1, \gamma_1\beta_1, \pi_1)$, то $\widetilde{T}$ не допуска преход от $(\widetilde{p}, \alpha, \gamma, \pi)$. Да допуснем, че $\widetilde{p} \in \widetilde{P}$. Тъй като в T има преход от $(p, \alpha, \gamma\widetilde{\lambda}^*(\beta), \pi\omega)$, то $\Delta(p, \sigma)$ е дефинирана (тук σ е първата буква на α). Сега от дефиницията на $\widetilde{\Delta}$ единствената възможност $\widetilde{\Delta}(\widetilde{p}, \sigma)$ да не е дефинирана е $\Delta(p, \sigma) \in Q$ и $walk(\Delta(p, \sigma), \gamma)$ не е дефинирана. Тогава от лема 1 не съществува извод от $< \Delta(p, \sigma), \alpha', \gamma\widetilde{\lambda}^*(\beta), \pi\omega >$ с дължина по-голяма от $|\gamma| - 2$ и всеки такъв преход запазва α' и съответното състояние е от множеството Q. И в двата случая стигаме до противоречие с условието, че или $p_2 \in P$ или $\beta_2 = \varepsilon$. В частност вторият извод е невъзможен. Така остава възможността $\widetilde{p} \in \widetilde{Q}$, откъдето $p \in Q$. Ако $\beta = \varepsilon$, то и $\widetilde{\lambda}^*(\beta) = \varepsilon$ и отново

$$(p, \alpha, \gamma\widetilde{\lambda}^*(\beta), \pi \circ \omega) \vDash (p_2, \alpha_2, \beta_2, \pi_2)$$

е невъзможен за $p \neq p_2$. Нека $\beta = b\beta''$. Ако $\beta'' \neq \varepsilon$, то тъй като $\widetilde{d}(\widetilde{p'}, b)$ не е дефиниран, то от лема 1, 3 не съществува извод с дължина по-голяма от $|\gamma'\widetilde{\lambda}^*(b)| - 2$, започващ от $(p, \alpha, \gamma\widetilde{\lambda}^*(\beta), \pi\omega')$ и всеки такъв извод запазва непроменено α. Тоест отново стигаме до противоречие.

Накрая остава да разгледаме случая, когато $\beta' = b$. Ако $walk(q, \gamma\lambda(b)$ не е дефинирана, то разсъждавайки както по горе стигаме до противоречие. Нека $walk(q, \gamma\lambda(b)) = q''$, $rest(q, \gamma\lambda(b)) = b'$. Единствената възможност тогава да нямаме преход е $b' \in \Gamma$ и стойността $d(q, b', 0)$ да не е дефинирана. Това обаче отново означава, че

$$(q', \alpha', \gamma'\lambda(b), \pi'\omega') \vDash$$
$$(walk(q', \alpha', \gamma'\lambda(b)), \alpha', b', \pi'\omega' \circ print(q', \gamma'\lambda(b))),$$

където T не допуска преход от последната конфигурация като или $b' \in \Gamma$ и $walk(q', \gamma'\lambda(b)) \in Q$.

С това стигаме до противоречие с допускането, че твърдението на лемата не е в сила, с което и доказателството на лемата е завършено.

Сега можем да завършим доказателството на теорема 3. Нека първо $\alpha \in Dom(f_{\widetilde{T}})$. Тогава съществуват $\widetilde{f} \in \widetilde{P}$ и $\widetilde{q} \in \widetilde{Q}$, и думи $\beta \in (\widetilde{P} \times \Sigma)^*$ и $\pi \in \Omega^*$, за които:

$$(\widetilde{i}, \alpha, \varepsilon, \varepsilon) \vDash (\widetilde{f}, \varepsilon, \beta, \pi) \text{ и}$$
$$(\widetilde{\phi}(\widetilde{f}), \varepsilon, \beta, \pi) \vDash (\widetilde{q}, \varepsilon, \varepsilon, \pi'),$$

Нека $\widetilde{f} = (f, \gamma_1, \omega_1)$ и от първата част на лема 2 следва, че

$$(i, \alpha, \varepsilon, \varepsilon) \vDash (f, \varepsilon, \gamma_1\widetilde{\lambda}^*(\beta), \pi\omega_1).$$

По дефиниция имаме, че $\widetilde{\phi}(\widetilde{f}) = (walk(\phi(f), \gamma_1), rest(\phi(f), \gamma_1), \omega_1 \circ print(\phi(f), \gamma_1))$. Лема 1 сега ни дава, че имаме извод:

$$(\phi(f), \varepsilon, \gamma_1\widetilde{\lambda}^*(\beta), \pi\omega_1) \vDash$$
$$(walk(\phi(f), \gamma_1), \varepsilon, rest(\phi(f), \gamma_1) \circ \widetilde{\lambda}^*(\beta), \pi\omega_1 \circ print(\phi(f), \gamma_1).$$

Като комбинираме този резултат с извода

$$(\widetilde{\phi}(\widetilde{f}), \varepsilon, \beta, \pi) \models (\widetilde{q}, \varepsilon, \varepsilon, \pi')$$

и приложим лема 2 за $\widetilde{\phi}(\widetilde{f})$ и $\widetilde{q}$ получаваме, че:

$$< \phi(f), \varepsilon, \gamma_1 \widetilde{\lambda}^*(\beta), \pi\omega_1 > \models < q, \varepsilon, \gamma_2, \pi'\omega_2 >$$

където $\widetilde{q} = (q, \gamma_2, \omega_2)$. Сега тъй като $\widetilde{q} \in \widetilde{Q}$ получаваме първо, че $q \in Q$ и второ $|\gamma_2| \leq 1$. Ако $|\gamma_2| = 1$ то

$$(\phi(f), \varepsilon, \gamma_1 \widetilde{\lambda}^*(\beta), \pi\omega_1) \models$$
$$(q, \varepsilon, \gamma_2, \pi'\omega_2) \vdash (d(q, \gamma_2, 0), \varepsilon, \varepsilon, \pi'\omega_2 \circ out(q, \gamma_2, 0)).$$

Сега е ясно, че

$$f_{\widetilde{T}}(\alpha) = \pi' \circ \widetilde{\psi}(\widetilde{q}) =$$
$$\pi' \circ \omega_2 \circ out(q, b, 0) \circ \quad \psi(d(q, b, 0)) = f_T(\alpha).$$

Ако пък $\gamma_2 = \varepsilon$, то директно получаваме, че

$$f_{\widetilde{T}}(\alpha) = \pi'\omega' \circ \psi(q) = f_T(\alpha).$$

Това показва, че $f_T(\alpha)$ също е дефинирана и приема стойността на $f_{\widetilde{T}}(\alpha)$.

Обратно, нека $\alpha \in Dom(T)$. Тогава съществуват $f \in P$, $q \in Q$ и думи $\beta \in \Gamma^*$, $\pi, \pi' \in \Omega^*$, за които:

$$(i, \alpha, \varepsilon, \varepsilon) \models (f, \varepsilon, \beta, \pi) \text{ и} \tag{1}$$
$$(\phi(f), \varepsilon, \beta, \pi) \models (q, \varepsilon, \varepsilon, \pi'). \tag{2}$$

Сега от втората част на лема 2, приложена за $\widetilde{i} =< i, \varepsilon, \varepsilon >$, f, γ и π, получаваме, че съществуват $\widetilde{f} = (f, \gamma_1, \omega_1)$, $\widetilde{\beta}$ и $\widetilde{\pi}$ със свойството:

$$(\widetilde{i}, \alpha, \varepsilon, \varepsilon) \models (\widetilde{f}, \varepsilon, \widetilde{\beta}, \widetilde{\pi}).$$

Сега за последния извод можем да приложим първата част на лема 2 и получаваме, че всъщност $\beta = \gamma_1 \widetilde{\lambda}^*(\widetilde{\beta})$ и $\pi = \widetilde{\pi}\omega_1$. Сега от съществуването на извода:

$$(\phi(f), \varepsilon, \gamma_1\beta, \widetilde{\pi}\omega_1) \models (q, \varepsilon, \varepsilon, \pi')$$

заключаваме, че съществува извод от $(\phi(f), \varepsilon, \gamma_1\beta, \pi'\omega_1)$ с дължина по-голяма от $|\gamma_1| - 2$. Тогава от третия пункт на лема 1 следва, че $walk(\phi(f), \gamma_1)$ е дефинирана. Тогава от точка 1 на лема 1 следва, че

$$(\phi(f), \varepsilon, \gamma_1\beta, \widetilde{\pi}\omega_1) \models$$
$$(walk(\phi(f), \gamma_1), \varepsilon, rest(\phi(f), \gamma_1)\beta, \widetilde{\pi}\omega_1 \circ print(\phi(f), \gamma_1)).$$

Тогава всъщност имаме, че:

$$< walk(\phi(f), \gamma_1), \varepsilon, rest(\phi(f), \gamma_1)\beta, \widetilde{\pi}\omega_1 \circ print(\phi(f), \gamma_1) > \models$$
$$< q, \varepsilon, \varepsilon, \pi' > .$$

От тук следва първо, че

$$\widetilde{\phi}(\widetilde{f}) = (walk(\phi(f), \gamma_1), rest(\phi(f), \gamma_1), \omega_1 print(\phi(f), \gamma_1))$$

и второ, прилагайки втория пункт на лема 2, че съществува извод:

$$(\widetilde{\phi}(\widetilde{f}), \varepsilon, \widetilde{\beta}, \widetilde{\pi}) \models (\widetilde{q}, \varepsilon, \varepsilon, \widetilde{\pi}').$$

Сега прилагаме първия пункт на лема 2, според който $\widetilde{q} = (q', \gamma_1', \omega_1')$, то

$$(\phi(f), \varepsilon, \beta, \pi) \models (q', \varepsilon, \gamma_1', \pi''\omega_1') \models (q, \varepsilon, \varepsilon, \pi').$$

В частност $q' \in Q$ и тогава $\widetilde{q} \in \widetilde{Q}$. Следователно $|\gamma_1'| \leq 1$ и сега е ясно, че или $\gamma_1' = \varepsilon$ и тогава $q = q'$ или $|\gamma_1'| = 1$ и тогава $q = d(q', \gamma_1', 0)$. И в двата случая $\widetilde{\psi}(\widetilde{q})$ е дефинирано точно, когато $\psi(q)$ е дефинирано. Тъй като $\alpha \in Dom(f_T)$, то това показва, че $\psi(q)$ и $\widetilde{\psi}(\widetilde{q})$ са дефинирани и следователно $\alpha \in Dom(f_{\widetilde{T}}$. С това показахме, че $Dom(f_T) \subset Dom(f_{\widetilde{T}})$ и тъй като по горе видяхме, че за всяко $\alpha \in Dom(f_{\widetilde{T}})$ е в сила, че $\alpha \in Dom(f_T)$ и $f_T(\alpha) = f_{\widetilde{T}}(\alpha)$, получаваме, че $f_T = f_{\widetilde{T}}$. С това доказахме и теорема 3.

3 Композиция на FIFO-трансдюсер с подпоследователен преобразувател

Както отбелязахме в увода основен интерес, от практическа гледна точка, представлява композицията на функции, зададени от трансдюсери, в нашия случай FIFO-трансдюсери. Това, от части, може би се дължи и на факта, че останалите операции над езици - конкатенация, обединение, звезда на Клини и отрицание, приложени над функции, в общия случай не са функции, което прави въпросът им за представяне несъстоятелен. Случаят за сечение на две функции излиза извън пределите на рационалните релации [Eil74] и затова и тази естествена операция не се разглежда.

В тази част ще разгледаме композицията на FIFO-трансдюсери с подпоследователни преобразуватели. Двата основни резултата са, че както при лява, така и при дясна композиция с подпоследователен преобразувател, получаваме функция, разпознавана от FIFO-трансдюсер. Идеята и за двете доказателства е да симулираме съвместното изпълнение на двете машини.

При композицията отляво не възникват почти никакви усложнения и всъщност композицията се отразява само върху състоянията на опашката - където прилагаме, по същество, конструкцията за композиция на подпоследоватлни преобразуватели.

При композицията отдясно, обаче, се налага да бъдем по-внимателни и да приложим повече усилия, за да получим желания резултат.

Започваме с по-простия случай.

3.1 Композиция отляво на FIFO-трансдюсер и подпоследователен преобразувател

Теорема 4 *Нека* $T =< \Sigma \times \Omega^*, \Gamma^*, P, Q, s, \Delta, d, \lambda, out, \phi, \psi >$ *е FIFO-трансдюсер, а* $T' =< \Omega \times \Xi^*, Q', s', \delta', \lambda', \phi' >$ *е подпоследователен трансдюсер. Тогава съществува FIFO-трансдюсер* $\widetilde{T}$ *със свойството:*

$$f_{T'} \circ f_T = f_{\widetilde{T}}.$$

Доказателство: Нека $\widetilde{T} =< \Sigma \times \Xi^*, \Gamma^*, \widetilde{P}, \widetilde{Q}, \widetilde{s}, \widetilde{\Delta}, \widetilde{d}, \widetilde{\lambda}, \widetilde{out}, \widetilde{\phi}, \widetilde{\psi} >$ е FIFO-трансдюсерът, определен по следния начин:

1. $\widetilde{P} = P \times Q'$.

2. $\widetilde{Q} = Q \times Q'$.

3. $\widetilde{s} =< s, s' >$.

4. $\widetilde{\Delta} : \widetilde{P} \times \Sigma \to \widetilde{P} \cup \widetilde{Q}$ се дефинира като:

$$\widetilde{\Delta}(< p, q' >, \sigma) =< \Delta(p, \sigma), q' > .$$

5. $\widetilde{\lambda} : \widetilde{P} \times \Sigma \to \Gamma^*$ е определена от равенството:

$$\widetilde{\lambda}(< p, q' >, \sigma) = \lambda(p, \sigma).$$

6. $\widetilde{d} : \widetilde{Q} \times \Gamma \times \{0, 1\} \to \widetilde{P} \cup \widetilde{Q}$ се дефинира като:

$$\widetilde{d}(< p, q' >, b, j) =< d(p, b, j), (\delta')^*(q', out(p, b, j)) > .$$

7. $\widetilde{out} : \widetilde{Q} \times \Gamma \times \{0, 1\} \to \Xi^*$ се задава от равенството:

$$\widetilde{out}(< p, q' >, b, j) = (\lambda')^*(q', out(p, b, j)).$$

8. $\widetilde{\phi} : \widetilde{P} \to \widetilde{Q}$ е:

$$\widetilde{\phi}(< p, q' >) =< \phi(p), q' > .$$

9. $\widetilde{\psi} : \widetilde{Q} \to \Xi^*$ е:

$$\widetilde{\psi}(< p, q' >) = (\lambda')^*(q', \psi(p)) \circ \phi'((\delta')^*(q', \psi(p))).$$

Ще покажем, че $\widetilde{T}$ удовлетворява заключението на теоремата. За целта ще ни послужи следната:

Теорема 5 *Нека $p_i \in P \cup Q$, $q'_i \in Q'$, $\alpha_i \in \Sigma^*$, $\gamma_i \in \Gamma^*$, $\widetilde{\omega_1} \in \Xi^*$ и $\omega_i \in \Omega^*$ за $i \in \{1, 2\}$ са произволни. Тогава ако означим с $\widetilde{p_i} =< p_i, q'_i >$, то изводът:*

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} >\models \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} > \qquad (3)$$

съществува, точно когато съществува $\omega_2 \in \Omega^*$, за което са изпълнени следните три условия:

1. $< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega_2 >$.
2. $(\delta')^*(q'_1, \omega_2) = q'_2$ *и*
3. $(\lambda')^*(q'_1, \omega_2) = \widetilde{\omega_2}$.

Доказателство: Нека първо условията 1–3 са изпълнени. Ще покажем, че съществува изводът 3 с индукция по дължината на извода

$$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega_2 > .$$

1. Ако дължината му е 0, то получаваме, че $p_1 = p_2$, $\alpha_1 = \varepsilon$, $\gamma_1 = \gamma_2$ и $\omega_2 = \varepsilon$. Тогава от условие 2 получаваме, че $q'_2 = q'_1$, а от 3, че $\widetilde{\omega_2} = \varepsilon$. Тогава лесно се вижда, че $\widetilde{p_1} = \widetilde{p_2}$ и че двете конфигурации в извода 3 съвпадат, което показва, че между тях съществува извод.

2. Нека изводът

   $$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega_2 >$$

   е съставен от прехода:

   $$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\vdash< p, \alpha\alpha_2, \gamma, \omega_1\omega >$$

   и от извода:

   $$< p, \alpha\alpha_2, \gamma, \omega_1\omega >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega_2 > .$$

   В зависимост от това дали $p_1 \in P$ или $p_1 \in Q$ имаме два случая:

- $p_1 \in P$. Тогава от дефиницията за преход получаваме, че $\alpha_1 = \sigma\alpha$, $p = \Delta(p_1, \sigma)$, $\gamma = \gamma_1 \circ \lambda(p_1, \sigma)$ и $\omega = \varepsilon$. Сега получаваме, че $< p_1, q_1' > \in \widetilde{P}$ и съответно $\widetilde{\Delta} =< \Delta(p_1, \sigma), q_1' > =< p, q_1' >$ и $\widetilde{\lambda}(\widetilde{p_1}, \sigma) = \lambda(p_1, \sigma)$. Това показва, че съществува преходът:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \vdash << p, q_1' >, \alpha\alpha_2, \gamma, \widetilde{\omega_1} > .$$

  Да означим с $\widetilde{p} =< p, q_1' >$. Тогава от индукционното предположение, приложено за $\widetilde{p}$ и $\widetilde{p_2}$, тъй като равенствата 2 и 3, касаещи q_1' е се променят, заключаваме, че:

$$< \widetilde{p}, \alpha\alpha_2, \gamma, \widetilde{\omega_1} > \vDash < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

  Сега като вземе в предвид и прехода:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \vdash << p, q_1' >, \alpha\alpha_2, \gamma, \widetilde{\omega_1} >,$$

  получаваме и извода:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \vDash < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

- Нека сега $p_1 \in Q$. Тогава от дефиницията за преход получаваме, че $\gamma_1 = b\gamma$, $\alpha = \alpha_1$, $p = d(p_1, b, j)$ и $\omega = out(p_1, b, j)$, където $j \in \{0, 1\}$ в зависимост от това дали $\gamma = \varepsilon$ или не. Нека положим $q' = (\delta')^*(q_1', \omega)$ и $\widetilde{\omega} = (\lambda')^*(q_1', \omega)$. От условия 2 и 3 получаваме, че q' и $\widetilde{\omega}$ съществуват и са коректно дефинирани, а от дефиницията на $\widetilde{d}$ и $\widetilde{out}$ заключаваме, че:

$$\widetilde{d}(\widetilde{p_1}, b, j) =< p, q' >$$
$$\widetilde{out}(\widetilde{p_1}, b, j) = \widetilde{\omega}.$$

  Тогава обаче е дефиниран и преходът:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \vdash < \widetilde{p}, \alpha\alpha_2, \gamma, \widetilde{\omega_1}\widetilde{\omega} > .$$

Остана да забележим, че условия 2 и 3 за q_1' сега ни дават, че:

$$q_2' = (\delta')^*(q_1', \omega_2) =$$
$$(\delta')^*((\delta')^*(q_1', \omega), (\omega^{-1}\omega_2)) = (\delta')^*(q', \omega^{-1}\omega_2)$$

и аналогично от условие 3:

$$\widetilde{\omega_2} = (\lambda')^(q_1', \omega_2) =$$
$$(\lambda')^*(q_1', \omega) \circ (\lambda')^*((\delta')^*(q_1', \omega), \omega^{-1}\omega_2) = \widetilde{\omega}(\lambda')^*(q', \omega^{-1}\omega_2).$$

Това показва, че:

$$(\delta')^*(q', \omega^{-1}\omega_2) = q_2' \text{ и}$$
$$(\lambda')^*(q', \omega^{-1}\omega_2) = \widetilde{\omega}^{-1}\widetilde{\omega_2}.$$

Сега можем да приложим индукционното предположение за $\widetilde{p}$, $\widetilde{p_2}$ и извода:

$$< p, \alpha\alpha_2, \gamma, \omega_1\omega > \models < p_2, \alpha_2, \gamma_2, \omega_1\omega_2 > =$$
$$< p_2, \alpha_2, \gamma_2, \omega_1\omega(\omega^{-1}\omega_2) > .$$

От него получаваме, че съществува и изводът:

$$< \widetilde{p}, \alpha\alpha_2, \gamma, \widetilde{\omega_1}\widetilde{\omega} > \models < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

С това показахме верността на твърдението и в този случай, което завършва и индукционната стъпка.

Доказателството в обратната посока ще извършим с индукция по дължината на извода:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \models < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

1. При дължина на извода 0, получаваме, че двете конфигурации съвпадат. Оттук следва, че $p_1 = p_2$, $q_1' = q_2'$, $\alpha_1 = \varepsilon$, $\gamma_1 = \gamma_2$, $\widetilde{\omega_2} = \varepsilon$. Сега непосредствено се проверява, че двете конфигурации, от двете страни на извода от условие 1 съвпадат, а условията 2 и 3 също следват непосредствено.

2. Нека изводът

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \models < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} >$$

е съставен от прехода:

$$< \widetilde{p_1}, \alpha_1\alpha_2, \gamma_1, \widetilde{\omega_1} > \vdash < \widetilde{p_2}, \alpha\alpha_2, \gamma, \widetilde{\omega_1}\widetilde{\omega} >$$

и от извода:

$$< \widetilde{p}, \alpha\alpha_2, \gamma, \widetilde{\omega_1}\widetilde{\omega} > \models < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} >$$

Имаме два случая, в зависимост от това дали $\widetilde{p_1} \in \widetilde{P}$ или $\widetilde{p_1} \in \widetilde{Q}$.

- $\widetilde{p_1} \in \widetilde{P}$. Тогава от дефиницията на $\widetilde{P}$ заключаваме, че $p_1 \in P$. Сега от дефиницията за преход следва, че $\alpha_1 = \sigma\alpha$, $\widetilde{\omega} = \varepsilon$, $\widetilde{p} = \widetilde{\Delta}(\widetilde{p_1}, \sigma)$ и $\gamma = \gamma_1 \circ \widetilde{\lambda}(\widetilde{p_1}, \sigma)$. Но от дефинициите на $\widetilde{\Delta}$ и $\widetilde{\lambda}$ следва, че $\widetilde{p} =< \Delta(p_1, \sigma), q_1' >$ и съответно $\widetilde{\lambda}(\widetilde{p_1}, \sigma) = \lambda(p, \sigma)$. Така получаваме, че е дефиниран и преходът:

  $$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 > \vdash < p, \alpha\alpha_2, \gamma, \omega_1 >$$

  Сега, прилагайки индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$ получаваме, че

  $$\begin{gathered} < p, \alpha\alpha_2, \gamma, \omega_1 > \models < p_2, \alpha_2, \gamma_2, \omega_1\omega_2 > \\ (\delta')^*(q_1', \omega_2) = q_2' \\ (\lambda')^*(q_1', \omega_2) = \widetilde{\omega_2}. \end{gathered}$$

  Остава да се възползваме и от прехода:

  $$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 > \vdash < p, \alpha\alpha_2, \gamma, \omega_1 >,$$

  от където получаваме и извода:

  $$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 > \models < p_2, \alpha_2, \gamma_2, \omega_1\omega_2 >$$

.

Сега и условията 2 и 3 следват непосредствено.

- $\widetilde{p_1} \in \widetilde{Q}$. Тогава и $p \in Q$ и от дефиницията за преход получаваме, че $\gamma_1 = b\gamma$, $\alpha = \alpha_1$, $\widetilde{p} = \widetilde{d}(\widetilde{p_1}, b, j)$ $\widetilde{\omega} = \widetilde{out}(\widetilde{p_1}, b, j)$, където $j \in \{0, 1\}$ в зависимост от това дали $\gamma = \varepsilon$ или не. Нека $< p, q' >= \widetilde{p}$. Тогава от дефиницията на $\widetilde{d}$ заключаваме, че $q' = (\delta')^*(q'_1, out(p_1, b, j))$ и съответно $\widetilde{\omega} = (\lambda')^*(q'_1, out(p_1, b, j))$. Нека $\omega = out(p, b, j)$. Тогава, непосредствено от дефиницията за преход и разсъжденията, направени по горе, получаваме, че е дефиниран преходът:

$$< p, \alpha_1\alpha_2, \gamma_1, \omega_1 >\vdash< p_1, \alpha\alpha_2, \gamma, \omega_1\omega > .$$

Освен това

$$(\delta')^*(q'_1, \omega) = q' \text{ и}$$
$$(\lambda')^*(q'_1, \omega) = \widetilde{\omega}.$$

Сега прилагаме индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$. Тъй като:

$$< \widetilde{p}, \alpha\alpha_2, \gamma, \widetilde{\omega_1}\widetilde{\omega} >\models< \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega_2} >= \\ < \widetilde{p_2}, \alpha_2, \gamma_2, \widetilde{\omega_1}\widetilde{\omega}\widetilde{\omega}^{-1}\widetilde{\omega_2}) >,$$

то заключаваме, че съществува $\omega_2 \in \Omega^*$, за което:

$$< p, \alpha\alpha_2, \gamma_2, \omega_1\omega >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega(\omega^{-1}\omega_2) >$$
$$(\delta')^*(q', \omega^{-1}\omega_2) = q'_2$$
$$(\lambda')^*(q', \omega^{-1}\omega_2) = \widetilde{\omega}^{-1}\widetilde{\omega_2}.$$

Оттук прибавяйки прехода:

$$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\vdash< p, \alpha\alpha_2, \gamma, \omega_1\omega >$$

получаваме, че съществува и изводът:

$$< p_1, \alpha_1\alpha_2, \gamma_1, \omega_1 >\models< p_2, \alpha_2, \gamma_2, \omega_1\omega_2 > .$$

Накрая като вземем под внимание, че

$$(\delta')^*(q_1', \omega) = q'$$
$$(\lambda')^*(q_1', \omega) = \widetilde{\omega},$$

получаваме:

$$(\delta')^*(q_1', \omega_2) = (\delta')^*(q', (\omega^{-1}\omega_2)) = q_2'$$
$$(\lambda')^*(q_1', \omega_2) = (\lambda')^*(q_1', \omega) \circ (\lambda')^*((\delta')^*(q_1, \omega), (\omega^{-1}\omega_2)) =$$
$$\widetilde{\omega}(\lambda')^*(q', \omega^{-1}\omega_2) = \widetilde{\omega}(\widetilde{\omega}^{-1}\widetilde{\omega_2}) = \widetilde{\omega_2}.$$

С това доказахме твърдението и в този случай, с което индукционната стъпка е направена.

Сега фактът, че:

$$f_{\widetilde{T}} = f_{T'} \circ f_T$$

следва почти непосредствено.

Наистина, нека $\alpha \in Dom(f_{\widetilde{T'}})$. Тогава съществуват изводите:

$$< \widetilde{s}, \alpha, \varepsilon, \varepsilon > \models < \widetilde{p_1}, \varepsilon, \gamma, \widetilde{\omega} >$$
$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma, \widetilde{\omega_1} > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \widetilde{\omega_1}\widetilde{\omega_2} >,$$

където $\widetilde{p_1} =< p_1, q_1' > \in \widetilde{P}$ и $\widetilde{p_2} =< p_2, q_2' > \in \widetilde{Q}$, като $\widetilde{\psi}(\widetilde{p_2})$ е дефинирана. Сега прилагаме твърдение 5 за $\widetilde{s}$ и $\widetilde{p_1}$. Тогава получаваме, че съществува ω_1 със свойствата:

$$< s, \alpha, \varepsilon, \varepsilon > \models < p_1, \varepsilon, \gamma, \omega_1 >$$
$$(\delta')^*(s', \omega_1) = q_1'$$
$$(\lambda')^*(s', \omega_1) = \widetilde{\omega_1}.$$

Сега от дефиницията за $\widetilde{\phi}$ имаме, че $\widetilde{\phi}(\widetilde{p_1}) =< \phi(p_1), q_1' >$. Тъй като е дефиниран изводът:

$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma, \widetilde{\omega_1} > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \widetilde{\omega_1}\widetilde{\omega_2} >$$

можем да приложим твърдение 5 за него и тогава намираме ω_2 със свойствата:

$$< \phi(p_1), \varepsilon, \gamma, \omega_1 > \models < p_2, \varepsilon, \varepsilon, \omega_1\omega_2 >$$
$$(\delta')^*(q_1', \omega_2) = q_2'$$
$$(\lambda')^*(q_1', \omega_2) = \widetilde{\omega_2}.$$

Сега като обединим тези два резултата получаваме, че:

$$< s, \alpha, \varepsilon, \varepsilon > \models < p_1, \varepsilon, \gamma, \omega_1 >$$
$$< \phi(p_1), \varepsilon, \gamma, \omega_1 > \models < p_2, \varepsilon, \varepsilon, \omega_1\omega_2 >$$
$$(\delta')^*(s', \omega_1\omega_2) = q_2'$$
$$(\lambda')^*(s', \omega_1\omega_2) = \widetilde{\omega_1}\widetilde{\omega_2}.$$

Накрая тъй като, по дефиниция:

$$\widetilde{\psi}(< p_2, q_2' >) = (\lambda')(q_2', \psi(p_2)) \circ \phi'((\delta')^*(q_2', \psi(p_2))$$

можем да заключим, че $\psi(p_2)$ е дефинирана. Тогава получаваме, че:

$$f_T(\alpha) = \omega_1\omega_2\psi(p_2).$$

От друга страна имаме, че:

$$f_{\widetilde{T}}(\alpha) = \widetilde{\omega_1}\widetilde{\omega_2}\widetilde{\psi}(< p_2, q_2' >) =$$
$$(\lambda')^*(s', \omega_1\omega_2)(\lambda')^*(q_2', \psi(p_2))\phi'((\delta')^*(q_2', \psi(p_2))) =$$
$$(\lambda')^*(s', \omega_1\omega_2 \circ \psi(p_2)) \circ \phi'((\delta')^*(s', \omega_1\omega_2 \circ \psi(p_2))),$$

където използвахме, че:

$$(\delta')^*(s', \omega_1\omega_2) = q_2'.$$

Така получаваме, че всъщност:

$$f_{\widetilde{T}}(\alpha) = (\lambda')^*(\omega_1\omega_2 \circ \psi(p_2)) \circ (\phi'((\delta')^*(s', \omega_1\omega_2 \circ \psi(p_2))) = f_{T'}(\omega_1\omega_2 \circ \psi(p_2)) = f_{T'}(f_T(\alpha)).$$

Обратно нека $\alpha \in Dom(f_T)$ и $f_T(\alpha) \in Dom(f_{T'})$. Тогава от първото получаваме, че съществуват изводите:

$$< s, \alpha, \varepsilon, \varepsilon > \models < p_1, \varepsilon, \gamma, \omega_1 >$$
$$< \phi(p_1), \varepsilon, \varepsilon, \omega_1 > \models < p_2, \varepsilon, \varepsilon, \omega_1\omega_2 >,$$

където $p_1 \in P$ и $p_2 \in Q$ като $\psi(p_2)$ е дефинирано. Сега имаме, че:

$$f_T(\alpha) = \omega_1\omega_2 \circ \psi(p_2).$$

Тъй като $f_T(\alpha) \in Dom(f_{T'})$, то са дефинирани и:

$$(\delta')^*(s', \omega_1) = q'_1$$
$$(\delta')^*(q'_1, \omega_2) = q'_2$$
$$(\delta')^*(q'_2, \psi(p_2)) = q'_3.$$

Сега ако означим:

$$\widetilde{\omega_1} = (\lambda')^*(s', \omega_1)$$
$$\widetilde{\omega_2} = (\lambda')^*(q'_1, \omega_2),$$

то от твърдение 5 получаваме, че

$$<< s, s' >, \alpha, \varepsilon, \varepsilon > \models << p_1, q'_1 >, \varepsilon, \gamma, \widetilde{\omega_1} > \text{ и}$$
$$<< \phi(p_1), q'_1 >, \varepsilon, \gamma, \widetilde{\omega_1} > \models << p_2, q'_2 >, \varepsilon, \varepsilon, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

Сега означаваме $\widetilde{p_1} =< p_1, q'_1 >$ и $\widetilde{p_2} =< p_2, q'_2 >$. От дефиницията на $\widetilde{\phi}$ получаваме, че

$$< \widetilde{s}, \alpha, \varepsilon, \varepsilon > \models < \widetilde{p_1}, \varepsilon, \gamma, \widetilde{\omega_1} >$$
$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma, \widetilde{\omega_1} > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \widetilde{\omega_1}\widetilde{\omega_2} > .$$

Това означава, че:

$$f_{\widetilde{T}} = \widetilde{\omega_1}\widetilde{\omega_2} \circ \widetilde{\psi}(\widetilde{p_2}) =$$
$$(\lambda')^*(s', \omega_1\omega_2) \circ (\lambda')^*(q'_2, \psi(p_2)) \circ (\phi')(q'_3) =$$
$$(\lambda')^*(s', \omega_1\omega_2 \circ \phi(p_2)) \circ \phi'(q'_3) = f_{T'}(f_T(\alpha)).$$

Тук използвахме, че $q_2' = (\delta')^*(s', \omega_1\omega_2)$ и $q_3' = (\delta')^*(q_2', \phi(p_2))$.

С това показахме и че:

$$f_{T'}(f_T(\alpha)) = f_{\widetilde{T}}(\alpha).$$

Тъй като α беше произволно, то получаваме, че:

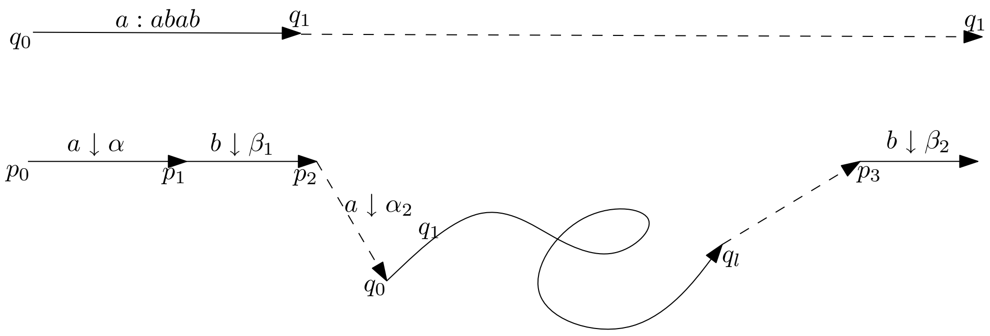
$$f_{\widetilde{T}} = f_{T'} \circ f_T.$$

С това доказахме и теорема 4.

3.2 Композиция отдясно на FIFO-трансдюсер с подпоследователен преобразувател

Както отбелязахме в началото на тази част, композицията с подпоследователен преобразувател е технически по-трудна. Това се дължи на следния проблем. Да си представим, че четем дума α според дефиницията на подпоследователния преобразувател T'. Докато траверсира думата α T' конструира и дума $f_{T'}(\alpha)$ и нашата цел е да симулираме изпълнението на FIFO-трансдюсера T върху тази дума. Ние получаваме $f_{T'}(\alpha)$ на порции, в зависимост от функцията на изхода на преобразувателя T', но тя може да е по-дълга от един символ и тогава това ще наложи да направим повече от един преход по състоянията P на FIFO-трансдюсера. Ако успеем да прочетем всички букви от текущата порция всичко е наред. Трудностите обаче възникват при преходите от състоянията P към състоянията на опашката. Тогава преди да продължим да четем буквите на $f_{T'}(\alpha)$ трябва да изчакаме опашката преди да се върнем в състояние на P(виж фиг. 5).

Сега можем вече да симулираме преходите по останалата, непрочетена част от порцията, която ни е предоставил T', но не трябва да забравяме за записите, които се генерират и трябва да бъдат добавени в опашката (виж фиг. 6). Този процес може да се повтори няколко пъти преди да изчерпим цялата порция, която ни е предоставил T' и натрупаната за това информация трябва да пазим в състоянията



Фигура 5: Можем да влезем в опашката до като траверсираме подпоследователния преобразувател, затова трябва той да изчака, а не обработените символи да запомним.

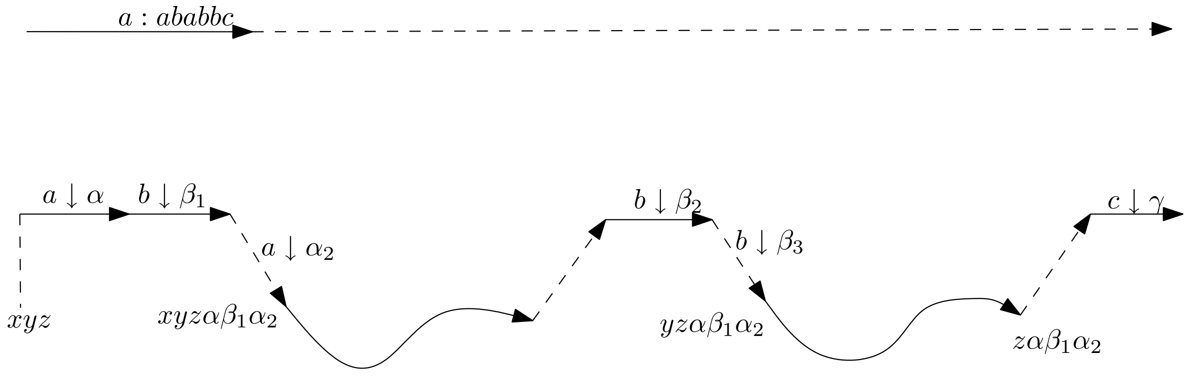
на FIFO-трансдюсера. Последният проблем, който възниква е изпразването на опашката, въпреки че опашката на оригиналния FIFO-трансдюсер не би се изпразнила - тогава трябва да използваме точно отложените записи, които не са били направени.(виж фиг. 7)

При доказателството на следващата теорема, ще видим как можем да преодолеем всички тези трудности и като резултат да получим FIFO-трансдюсер за желаната композиция.

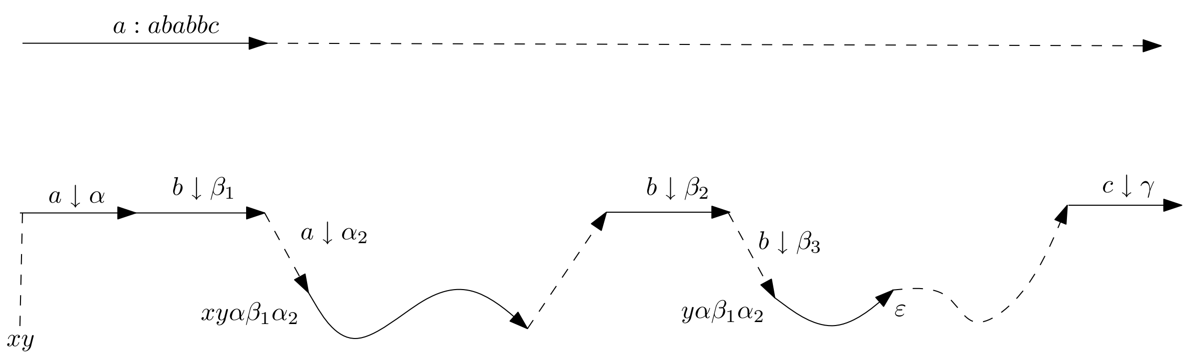
Теорема 6 *Нека* $T =< \Sigma \times \Xi^*, \Gamma^*, P, Q, s, \Delta, d, \lambda, out, \phi, \psi >$ *е FIFO-трансдюсер, а* $T' =< \Omega \times \Sigma^*, Q', s', \delta', \lambda', \phi' >$ *е подпоследователен трансдюсер. Тогава съществува FIFO-трансдюсер* $\widetilde{T}$*, за който* $f_{\widetilde{T}} = f_T \circ f_{T'}$.

Идеята на доказателството се състои в две основни стъпки. Първата е да симулираме по-дълги поредици от преходи, които съдържат само състояния от P. Това ще ни позволи да симулираме успешно поведението на T върху всевъзможните изходи на T', генерирани от функцията λ' на една стъпка. При това обаче възниква проблем с недокрай обработените поредици. За това във втората стъпка ще симулираме тези допълнителни преходи като ги отразим в по-дълги преходи от състоянията на опашката.

Пристъпваме към реализацията на първата стъпка. За целта ще ни бъде необходима следната:



Фигура 6: Може да се наложи да симулираме част от преходите по основните състояния вътре в самата опашка. Това ще наложи да запомним и техните записи, които те биха добавили в опашката.



Фигура 7: Може да се наложи да използваме тези записи, ако опашката се изпразни.

Дефиниция 10 *$walk_P : P \times \Sigma^* \to P \cup Q$, $rest_P : P \times \Sigma^* \to \Sigma^*$, $print_P : P \times \Sigma^* \to \Gamma^*$ са частични функции, които се дефинират рекурентно по дължината на думата $\alpha \in \Sigma^*$.*

1. *Ако $\alpha = \varepsilon$, то:*

$$walk_P(p, \varepsilon) = p$$
$$rest_P(p, \varepsilon) = \varepsilon$$
$$print_P(p, \varepsilon) = \varepsilon.$$

2. *Ако $\alpha = \sigma\alpha'$, $\sigma \in \Sigma$, то:*

$$walk_P(p, \sigma\alpha') = \begin{cases} \Delta(p,\sigma), \text{ ако } \Delta(p,\sigma) \in Q \\ walk_P(\Delta(p,\sigma), \alpha'), \text{ ако } \Delta(p,\sigma) \in P \\ \neg! \text{ иначе.} \end{cases}$$

$$rest_P(p, \sigma\alpha') = \begin{cases} \alpha', \text{ ако } \Delta(p,\sigma) \in Q \\ rest_P(\Delta(p,\sigma), \alpha'), \text{ ако } \Delta(p,\sigma) \in P \\ \neg! \text{ иначе.} \end{cases}$$

$$print_P(p, \sigma\alpha') = \begin{cases} \lambda(p,\sigma), \text{ ако } \Delta(p,\sigma) \in Q \\ \lambda(p,\sigma) \circ print_P(\Delta(p,\sigma), \alpha'), \text{ ако } \Delta(p,\sigma) \in P \\ \neg! \text{ иначе.} \end{cases}$$

Следващата лема описва основните свойства на тези функции:

Лема 3 *За всеки $p \in P$, $\alpha, \beta \in \Sigma^*$, $\gamma \in \Gamma^*$, $\omega \in \Xi^*$ са в сила следните твърдения:*

1. *ако $walk_P(p, \alpha)$ е дефинирана, то $rest_P(p, \alpha)$ е наставка на α и*

$$< p, \alpha\beta, \gamma, \omega > \models < walk_P(p,\alpha), rest_P(p,\alpha) \circ \beta, \gamma \circ print_P(p,\alpha), \omega > . \tag{4}$$

2. *ако $walk_P(p, \alpha) \in P$, то $rest_P(p, \alpha) = \varepsilon$.*

3. *ако $walk_P(p, \alpha) \in Q$, то $|rest_P(p, \alpha)| < |\alpha|$.*

4. ако $walk_P(p, \alpha)$ не е дефинирана, то не съществува извод от $< p, \alpha\beta, \gamma, \omega >$ с дължина по-голяма от $|\alpha| - 1$.

Доказателство: Индукция по дължината на α.

1. Ако $\alpha = \varepsilon$, то $walk_P(p, \varepsilon) = p$, $rest_P(p, \varepsilon) = \varepsilon$, $print_P(p, \varepsilon) = \varepsilon$. Тогава двете конфигурации в 4 са равни и твърдението следва непосредствено.

2. Нека $\alpha = \sigma\alpha'$ и нека $p \in P$ е произволно. В зависимост от $\Delta(p, \sigma)$ имаме три случая:

   (а) $\Delta(p, \sigma) \in Q$. Тогава $walk_P(p, \sigma) = \Delta(p, \sigma)$, $rest_P(p, \sigma\alpha') = \alpha'$, $print_P(p, \sigma\alpha') = \lambda(p, \sigma)$. От друга страна по дефиницията за преход, съществува преходът:

   $$< p, \sigma\alpha'\beta, \gamma, \omega > \vdash < \Delta(p, \sigma), \alpha'\beta, \gamma \circ \lambda(p, \sigma), \omega >$$

   Сега лесно се вижда, че дясната страна на последния преход съвпада с дясната страна на 4. Нещо повече $|rest_P(p, \sigma\alpha')| = |\alpha'| < |\alpha|$. Сега твърденията на лемата следват директно.

   (б) $\Delta(p, \sigma) \in P$. Тогава

   $$walk_P(p, \alpha) = walk_P(\Delta(p, \sigma), \alpha'),$$
   $$rest_P(p, \alpha) = rest_P(\Delta(p, \sigma), \alpha') \text{ и}$$
   $$print_P(p, \alpha) = \lambda(p, \sigma) \circ print_P(\Delta(p, \sigma), \alpha').$$

   От друга страна отново от дефиницията за преход е в сила, че:

   $$< p, \sigma\alpha'\beta, \gamma, \omega > \vdash < \Delta(p, \sigma), \alpha'\beta, \gamma \circ \lambda(p, \sigma), \omega > .$$

   Сега ако $walk_P(p, \alpha)$ не е дефинирана, то и $walk_P(\Delta(p, \sigma), \alpha')$ не е дефинирана. Сега по индукциотто предположение, от 4, не съществува извод от $< \Delta(p, \sigma), \alpha'\beta, \gamma \circ \lambda(p, \sigma), \omega >$ с дължина по-голяма от $|\alpha'| - 1$. Тогава от $< p, \sigma\alpha'\beta, \gamma, \omega >$ не съществува извод с дължина по-голяма от $1 + (|\alpha'| - 1) = |\alpha| - 1$.

Да разгледаме сега случая, когато $walk_P(\Delta(p,\sigma),\alpha')$ е дефинирана. Тогава от индукционното предположение, приложено за $\Delta(p,\sigma)$ и α' получаваме, че съществува изводът:

$$< \Delta(p,\sigma), \alpha'\beta, \gamma \circ \lambda(p,\sigma), \omega > \models$$
$$< walk_P(\Delta(p,\sigma),\alpha'), rest_P(\Delta(p,\sigma),\alpha') \circ \beta,$$
$$\gamma \circ \lambda(p,\sigma) \circ print_P(\Delta(p,\sigma),\alpha'), \omega > =$$
$$< walk_P(p,\alpha), rest_P(p,\alpha) \circ \beta, \gamma \circ print_P(p,\alpha), \omega >$$

по дефиниция. Това означава, че 4 е в сила и за двойката (p,α). Нещо повече отново от индукционното предполежение за $rest_P(\Delta(p,\sigma),\alpha')$ получаваме, че е наставка на α' и следователно

$$|rest_P(p,\alpha)| = |rest_P(\Delta(p,\sigma),\alpha')| \leq |\alpha'| < |\alpha|.$$

Оттук получаваме верността и на 3. Ако пък $walk_P(p,\alpha) \in P$, то от дефиницията на $walk_P$ следва, че и $walk_P(\Delta(p,\sigma),\alpha') \in P$. Сега от индукционното предполжение за $\Delta(p,\sigma)$ и α' получаваме, че $rest_P(\Delta(p,\sigma),\alpha') = \varepsilon$. Тогава обаче и $rest_P(p,\alpha) = rest_P(\Delta(p,\sigma),\alpha') = \varepsilon$. С това проверихме верността на твърденията на лемата и в този случай.

Остана да разгледаме и случая:

(в) $\Delta(p,\sigma)$ не е дефинирано. Тогава тъй като $|\alpha| \geq 1$ твърдение 4 следва непосредствено.

С това по метода на математическата индукция твърденията на лемата са в сила за произволни p и α.

Сега преминаваме към реализацията на втората част от нашата идея. За целта ще ни бъде необходима следната дефиниция, която ще ни позволи да извършваме по-дълги изводи от състояние в Q като познаваме съдържанието на опашката.

Дефиниция 11 *Дефинираме функциите* $walk_Q : Q \times \Sigma^* \times \Gamma^* \to P \cup Q$, $rest_Q : Q \times \Sigma^* \times \Gamma^* \to \Sigma^*$, $print_Q : Q \times \Sigma^* \times \Gamma^* \to \Gamma^*$ *и* $output_Q : Q \times \Sigma^* \times \Gamma^* \to \Xi^*$ *за всяко* $q \in Q$, $\alpha \in \Sigma^*$ *и* $\gamma \in \Gamma^*$ *рекурсивно по следния начин:*

1. $\gamma = \varepsilon$,

$$\begin{aligned} walk_Q(q, \alpha, \varepsilon) &= q \\ rest_Q(q, \alpha, \varepsilon) &= \alpha \\ print_Q(q, \alpha, \varepsilon) &= \varepsilon \\ output_Q(q, \alpha, \varepsilon) &= \varepsilon. \end{aligned}$$

2. $\gamma = b \in \Gamma$

$$walk_Q(q, \alpha, b) = \begin{cases} d(q, b, 0) \text{ ако } d(q, b, 0) \in Q \\ walk_P(d(q, b, 0), \alpha), \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in P \\ \\ walk_Q(walk_P(d(q, b, 0), \alpha), rest_P(d(q, b, 0), \alpha), \\ print_P(d(q, b, 0), \alpha)) \\ \quad \text{ако } d(q, b, 0) \in P \text{ и } walk_P(d(q, b, 0), \alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

$$rest_Q(q, \alpha, b) = \begin{cases} \alpha, \text{ АКО } d(q, b, 0) \in Q \\ \\ \varepsilon, \text{ ако } d(q, b, 0) \in P \text{ и } walk_P(d(q, b, 0), \alpha) \in P \\ \\ rest_Q(walk_P(d(q, b, 0), \alpha), rest_P(d(q, b, 0), \alpha), \\ print_P(d(q, b, 0), \alpha)) \\ \text{ако } d(q, b, 0) \in P \text{ и } walk_P(d(q, b, 0), \alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

$$print_Q(q,\alpha,b) = \begin{cases} \varepsilon,\ \ \textit{ако}\ d(q,b,0) \in Q \\ \\ print_P(d(q,b,0),\alpha),\ \ \textit{ако}\ d(q,b,0) \in P\ \ \textit{и} \\ walk_P(d(q,b,0),\alpha) \in P \\ \\ print_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), \\ print_P(d(q,b,0),\alpha)), \\ \textit{ако}\ d(q,b,0) \in P\ \ \textit{и}\ walk_P(d(q,b,0),\alpha) \in Q \\ \\ \neg!\ \textit{иначе}. \end{cases}$$

$$output_Q(q,\alpha,b) = \begin{cases} out(q,b,0),\ \textit{ако}\ d(q,b,0) \in Q \\ \\ out(q,b,0),\ \textit{ако}\ d(q,b,0) \in P\ \ \textit{и}\ walk_P(d(q,b,0),\alpha) \in P \\ \\ out(q,b,0) \circ output_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), \\ print_P(d(q,b,0),\alpha)), \\ \textit{ако}\ d(q,b,0) \in P\ \ \textit{и}\ walk_P(d(q,b,0),\alpha) \in Q \\ \\ \neg!\ \textit{иначе}. \end{cases}$$

3. $\gamma = b\gamma'$, $\gamma' \neq \varepsilon$.

$$walk_Q(q, \alpha, b\gamma') = \begin{cases} walk_Q(d(q,b,1), \alpha, \gamma') \text{ ако } d(q,b,1) \in Q \\ walk_P(d(q,b,1), \alpha), \text{ ако } d(q,b,1) \in P \text{ и} \\ walk_P(d(q,b,1), \alpha) \in P \\ \\ walk_Q(walk_P(d(q,b,1), \alpha), rest_P(d(q,b,1), \alpha), \\ \gamma' \circ print_P(d(q,b,1), \alpha)), \\ \text{ако } d(q,b,1) \in P \text{ и } walk_P(d(q,b,1), \alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

$$rest_Q(q, \alpha, b\gamma') = \begin{cases} rest_Q(d(q,b,1), \alpha, \gamma'), \text{ ако } d(q,b,1) \in Q \\ \\ \varepsilon, \text{ ако } d(q,b,1) \in P \text{ и } walk_P(d(q,b,1), \alpha) \in P \\ \\ rest_Q(walk_P(d(q,b,1), \alpha), rest_P(d(q,b,1), \alpha), \\ \gamma' \circ print_P(d(q,b,1), \alpha)), \\ \text{ако } d(q,b,1) \in P \text{ и } walk_P(d(q,b,1), \alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

$$print_Q(q,\alpha,b\gamma') = \begin{cases} walk_Q(d(q,b,1),\alpha,\gamma'), \text{ ako } d(q,b,1) \in Q \\ \\ \gamma' \circ print_P(d(q,b,1),\alpha), \\ \text{ako } d(q,b,1) \in Q \text{ и } walk_P(d(q,b,1),\alpha) \in P \\ \\ print_Q(walk_P(d(q,b,1),\alpha), rest_P(d(q,b,1),\alpha), \\ \gamma' \circ print_P(d(q,b,1),\alpha)), \\ \text{ako } d(q,b,1) \in P \text{ и } walk_P(d(q,b,1),\alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

$$output_Q(q,\alpha,b\gamma') = \begin{cases} out(q,b,1) \circ output_Q(q,\alpha,\gamma'), \text{ ako } d(q,b,1) \in Q \\ \\ out(q,b,1), \text{ ako } d(q,b,1) \in Q \text{ и} \\ walk_P(d(q,b,1),\alpha) \in P \\ \\ out(q,b,1) \circ output_Q(walk_P(d(q,b,1),\alpha), rest_P(d(q,b,1),\alpha), \\ \gamma' \circ print_P(d(q,b,1),\alpha)), \\ \text{ako } d(q,b,1) \in P \text{ и } walk_P(d(q,b,1),\alpha) \in Q \\ \\ \neg! \text{ иначе.} \end{cases}$$

Следващата лема характериза така дефинираните функции, като освен това показва, че те са коректни.

Лема 4 *За всяко $q \in Q$, $\alpha, \beta \in \Sigma^*$, $\gamma \in \Gamma^*$ и $\omega \in \Omega^*$ са в сила следните твърдения:*

1. *ако $walk_Q(q,\alpha,\gamma)$ е дефинирана, то*

$$< q, \alpha\beta, \gamma, \omega > \models \tag{5}$$
$$< walk_Q(q,\alpha,\gamma), rest_Q(q,\alpha,\gamma)\beta, print_Q(q,\alpha,\gamma), \omega \circ output_Q(q,\alpha,\gamma) > .$$

При това ако $rest_Q(q,\alpha,\gamma) \neq \varepsilon$, *то това е най-дългият възможен извод с начало* $< q, \alpha\beta, \gamma, \omega >$.

2. *Ако* $walk_Q(q,\alpha,\gamma)$ *не е дефинирана, то не съществува извод*

$$< q, \alpha\beta, \gamma, \omega > \models < p, \beta', \gamma', \omega >,$$

за който $|\beta'| < |\beta|$ *или* $\beta' = \beta$ *и* $\gamma' = \varepsilon$.

Доказателство: Да разгледаме множеството $\Sigma^* \times \Omega^*$ с наредба $\prec$, дефинирана чрез:

$$(\alpha',\gamma') \prec (\alpha'',\gamma'') \Leftrightarrow$$
$$|\alpha'| < |\alpha''| \text{ или } |\alpha'| = |\alpha''| \text{ и } |\gamma'| < |\gamma''|.$$

Лесно се вижда, че $(\Sigma^* \times \Omega^*, \prec)$ е фундирано множество с най-малък елемент $(\varepsilon,\varepsilon)$.

Твърденията на лемата ще направим посредством структурна индукция над $(\Sigma^* \times \Omega^*, \prec)$.

- Ако $(\alpha,\gamma) = (\varepsilon,\varepsilon)$, то $rest_Q(q,\alpha,\gamma) = \alpha = \varepsilon$, $walk_Q(q,\alpha,\gamma) = q$, $print_Q(q,\alpha,\gamma) = \varepsilon$ и $output_Q(q,\alpha,\beta,\gamma) = \varepsilon$. Тогава се вижда, че двете страни на извода 5 съвпадат и твърдението е на лице.

- Нека твърденията 1 и 2 са изпълнени за всички (α',γ'), за които $(\alpha',\gamma') \prec (\alpha,\gamma)$. Нека $q \in Q$, $\beta \in \Sigma^*$ и $\omega \in \Xi^*$ са произволни. За γ са възможни три случая:

  1. $\gamma = \varepsilon$. Тогава не съществува преход от конфигурацията $< q, \alpha\beta, \gamma, \omega >$, тъй като $q \in Q$. Сега 5 ледва директно от факта, че $rest_Q(\alpha,\varepsilon) = \alpha$ и $|\alpha| > 0$ и освен това $print_Q(\alpha,\varepsilon) = \varepsilon$ $output_Q(q,\alpha,\varepsilon) = \varepsilon$.
  2. $\gamma = b \in \Gamma$. Тогава ако съществува извод от $< q, \alpha\beta, \gamma, \omega >$ с дължина поне едно, то той започва с прехода:

$$< q, \alpha\beta, \gamma, \omega > \vdash < d(q,b,0), \alpha\beta, \varepsilon, \omega \circ out(q,b,0) > .$$

Това показва, че ако $d(q, b, 0)$ не е дефинирана, то заключението в 2 е изпълнено. Затова без ограничение на общността нека $d(q, b, 0)$ е дефинирано. Ако $d(q, b, 0) \in Q$, то можем да приложим индукционното предположение за α и ε. В този случай се убеждаваме, че 1 е изпълнено, защото от $d(q, b, 0)$ не са възможен преход, а $rest_Q(q, \alpha, b) = \alpha$ $print_Q(q, b, 0) = \varepsilon$ и $output_Q(q, \alpha, b)out(q, b, 0)$.
Да допуснем, че $d(q, b, 0) \in P$. Тогава ако $walk_P(d(q, b, 0), \alpha)$ не е дефинирано от лема 3 получаваме, че 2 е изпълнено, защото от $< d(q, b, 0), \alpha\beta, \varepsilon, \omega \circ out(q, b, 0) >$ не съществува извод с дължина по-голяма от $|\alpha| - 1$, а на всеки преход $\alpha\beta$ намалява с неповече от 1. И така нека $walk_P(d(q, b, 0), \alpha)$ е дефинирано. Тогава отново от лема 3 получаваме, че съществува извод:

$$\begin{gathered} < d(q, b, 0), \alpha\beta, \varepsilon, \omega \circ out(q, b, 0) > \models \\ < walk_P(d(q, b, 0), \alpha), rest_P(d(q, b, 0), \alpha) \circ \beta, \\ print_P(d(q, b, 0), \alpha), \omega \circ out(q, b, 0) > \end{gathered}$$

Сега ако $walk_P(d(q, b, 0), \alpha) \in P$ отново от лема 3 заключаваме, че $rest_Q(q, \alpha, b) = rest_P(d(q, b, 0), \alpha) = \varepsilon$. Освен това от дефиницията на $walk_Q$, $rest_Q$, $print_Q$ и $output_Q$ получаваме, че

$$\begin{aligned} walk_Q(q, \alpha, b) &= walk_P(d(q, b, 0), \alpha) \\ rest_Q(q, \alpha, b) &= rest_P(d(q, b, 0), \alpha) \\ print_Q(q, \alpha, b) &= print_P(d(q, b, 0), \alpha) \\ output_Q(q, \alpha, b) &= out(q, b, 0), \end{aligned}$$

откъдето следва, че:

$$\begin{gathered} < d(q, b, 0), \alpha\beta, \varepsilon, \omega \circ out(q, b, 0) > \models \\ < walk_Q(q, \alpha, b), rest_Q(q, \alpha, b) \circ \beta, print_Q(q, \alpha, b), \omega \circ output_Q(q, \alpha, b). \end{gathered}$$

Като вземем в предвид прехода

$$< q, \alpha\beta, b, \omega > \vdash < d(q, b, 0), \alpha\beta, \varepsilon, \omega \circ out(q, b, 0) >$$

получаваме желания резултат.

Ако пък $walk_P(d(q,b,0),\alpha) \in Q$, то отново от лема 3 получаваме, че $|rest_P(d(q,b,0),\alpha)| < |\alpha|$. Тогава от индукционното предположение за двойката:

$$(rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha))$$

получаваме, че ако

$$walk_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha))$$

е дефиниран, то

$$\begin{gathered} < walk_P(d(q,b,0), rest_P(d(q,b,0),\alpha) \circ \\ \beta, print_P(d(q,b,0),\alpha), \omega \circ out(q,b,0) > \models \\ < walk_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha)), \\ \\ rest_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha)) \circ \beta, \\ \\ print_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha)), \\ \\ \omega \circ out(q,b,0) \circ \\ output_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha)) > . \end{gathered}$$

последното обаче, по дефиниция е точно:

$$< walk_Q(q,\alpha,b), rest_Q(q,\alpha,b) \circ \beta, print_Q(q,\alpha,b), \omega \circ output_Q(q,\alpha,b) > .$$

Сега, обединявайки двата извода, получаваме точно резултата в 4. Ако пък

$$walk_Q(walk_P(d(q,b,0),\alpha), rest_P(d(q,b,0),\alpha), print_P(d(q,b,0),\alpha))$$

не е дефиниран, то лесно се вижда, че твърдение 2 за него влече верността на твърдение 2 за q, α и γ.

3. Нека сега $\gamma = b\gamma'$, където $\gamma' \neq \varepsilon$. Ако $d(q, b, 1)$ не е дефинирана, то по дефиниция не съществува преход от конфигурацията:
$$< q, \alpha\beta, \gamma, \omega > .$$
Това показва, че условието 2 на лемата е изпълнено. Остава да разгледаме случая, когато $q' = d(q, b, 1)$ е добре дефинирано. Това определя прехода:
$$< q, \alpha\beta, \gamma, \omega > \vdash < q', \alpha\beta, \gamma', \omega \circ out(q, b, 1) > .$$
Сега ако $q' \in Q$, то тъй като $|\gamma'| < |\gamma|$, то можем да приложим индукционното предположение за q', γ' и α. Тогава ако $walk_Q(q', \alpha, \gamma')$ е дефинирана, получаваме, че съществува изводът:
$$\begin{gathered} < q', \alpha\beta, \gamma', \omega \circ out(q, b, 1) > \models \\ < walk_Q(q', \alpha, \gamma'), rest_Q(q', \alpha, \gamma')\beta, \\ print_Q(q', \alpha, \gamma'), \omega \circ out(q, b, 1) \circ output_Q(q', \alpha, \gamma') > . \end{gathered}$$
Това показва, че е допустим и изводът:
$$\begin{gathered} < q, \alpha\beta, \gamma, \omega > \models \\ < walk_Q(q', \alpha, \gamma'), rest_Q(q', \alpha, \gamma')\beta, \\ print_Q(q', \alpha, \gamma'), \omega \circ out(q, b, 1) \circ output_Q(q', \alpha, \gamma') > . \end{gathered}$$
Сега остана да забележим, че по дефиниция, в този случай:
$$\begin{aligned} walk_Q(q, \alpha, \gamma) &= walk_Q(q', \alpha, \gamma') \\ rest_Q(q, \alpha, \gamma) &= rest_Q(q', \alpha, \gamma') \\ print_Q(q, \alpha, \gamma) &= print_Q(q', \alpha, \gamma') \\ output_Q(q, \alpha, \gamma) &= out(q, b, 1) \circ output_Q(q', \alpha, \gamma'). \end{aligned}$$
Следователно твърдение 1 на лемата е изпълнено и за (q, α, γ). Ако $walk_Q(q', \alpha, \gamma')$ не е дефинирана, то и $walk_Q(q, \alpha, \gamma)$ не е дефинирана, което влече верността на твърдение 2 за (q, α, γ).

Накрая, трябва да разгледаме и случая, $q' = d(q, b, 1) \in P$. Тогава ако $walk_P(q', \alpha)$ не е дефинирана, то от лема 3 следва, че не съществува извод с дължина по-голяма от $|\alpha| - 1$, започващ от конфигурацията

$$< q', \alpha\beta, \gamma', \omega \circ out(q, b, 1) > .$$

Тъй като при всеки преход дължината на $\alpha\beta$ намалява най-много с едно - това показва верността на 2 в този случай. И така нека $walk_P(q', \alpha)$ е дефинирана. Тогава отново от лема 3 получаваме, че съществува изводът:

$$< q', \alpha\beta, \gamma', \omega \circ out(q, b, 1) > \models$$
$$< walk_P(q', \alpha), rest_P(q', \alpha) \circ \beta >, \gamma' \circ print_P(q', \alpha), \omega \circ out(q, b, 1) > .$$

Нека:

$$q'' = walk_P(q', \alpha)$$
$$\alpha'' = rest_P(q', \alpha)$$
$$\gamma'' = \gamma' \circ print_P(q', \alpha)$$
$$\omega'' = \omega \circ out(q, b, 1).$$

Сега ако $q'' \in P$, то отново от лема 3 получаваме, че $rest_P(q', \alpha) = \varepsilon$ и твърдение 1 на лемата, следва директно като вземем в предвид, че

$$walk_Q(q, \alpha, \gamma) = walk_P(q', \alpha)$$
$$rest_Q(q, \alpha, \gamma) = \varepsilon$$
$$print_Q(q, \alpha, \gamma) = \gamma' \circ print_P(q', \alpha)$$
$$output_Q(q, \alpha, \gamma) = out(q, b, 1).$$

Остана да разгледаме случая, когато $q' \in Q$. Тогава от лема 3 знаем, че $|\alpha''| = |rest_P(q', \alpha)| < |\alpha|$. Това означава, че $(\alpha'', \gamma'') \prec (\alpha, \gamma)$ и следователно за тях можем да приложим индукционното предположение. Да отбележим, че в този случай, по дефиниция, имаме, че:

$$\begin{aligned} walk_Q(q,\alpha,\gamma) &= walk_Q(q'',\alpha'',\gamma'') \\ rest_Q(q,\alpha,\gamma) &= rest_Q(q'',\alpha'',\gamma'') \\ print_Q(q,\alpha,\gamma) &= print_Q(q'',\alpha'',\gamma'') \\ output_Q(q,\alpha,\gamma) &= out(q,b,1)\circ output_Q(q'',\alpha'',\gamma''). \end{aligned}$$

Сега тъй като:

$$< q,\alpha\beta,\gamma,\omega > \vdash < q',\alpha\beta,\gamma',\omega\circ out(q,b,1) > \models \\ < q'',\alpha''\beta,\gamma'',\omega'' >$$

твърденията 1 и 2 за (q,α,γ) следват непосредствено от съответните твърдения за (q'',α'',γ'').

С това от принципа за структурна индукция, следва, че за всеки $(\alpha,\gamma)\in\Sigma^*\times\Gamma^*$ и $q\in Q$ твърденията на лемата са изпълнени.

За да опишем компонентите на FIFO-трансдюсера $\widetilde{T}$, който ще представлява композицията на FIFO-трансдюсера T с подпоследователния трансдюсер T' ще ни бъдат необходими още следните означения:

Дефиниция 12 *1. Множеството от всевъзможните едностъпкови изходи в T':*

$$Out(\lambda') = \{\lambda'(q',a)\,|\,q'\in Q', a\in\Omega\}\cup\{\phi'(q')\,|\,q'\in Q'\}\cup \\ \{\lambda'(q',a)\phi'(\delta'(q',a))\,|\,q'\in Q,\, a\in\Omega\}.$$

2.

$$Rest(\lambda') = \Big\{(x,z)\,|\,z=\textstyle\prod_{i=1}^k print_P(p_i,t_i),\; x=rest_P(p_k,t_k), \\ \text{където } t_0\in Out(\lambda'), p_i\in P \text{ удовлетворяват условието} \\ t_{i+1}=rest_P(p_i,t_i) \text{ за всяко } 0\le i<k\Big\}$$

3.

$$Print(\lambda') = \{print_Q(q,x,bz)\,|\,q\in Q,(x,z)\in Rest(\lambda'), b\in\Gamma\}$$

Забележка 4 *$Out(\lambda')$, $Rest(\lambda')$ и $Print(\lambda')$ са крайни.*

За $Out(\lambda')$ това е очевидно.

$Rest(\lambda')$ е крайно, защото от лема 3 $|rest(p,t)| < |t|$ за всяко p и всяко $t \neq \varepsilon$. Това означава, че в конкатенцията могат да участват ограничен брой думи различни от ε.

Сега е ясно, че $Print(\lambda')$, като образ на функция с крайна дефиниционна област, също е крайно множество.

Сега пристъпваме към описанието на конструкцията:

1. $\widetilde{P} = Q' \times P \times Print(\lambda')$

2. $\widetilde{Q} = Q' \times Q \times (Rest(\lambda') \cup (\{\varepsilon\} \times Print(\lambda'))) \times \{0,1\}$.

3. $\widetilde{s} =< s', s, \varepsilon >$.

4. $\widetilde{\Delta} : \widetilde{P} \times \Omega \to \widetilde{P} \cup \widetilde{Q}$ дефинираме така:

$$\widetilde{\Delta}(< q', p, \gamma >, a) = \begin{cases} < \delta'(q',a), walk_P(p, \lambda'(q',a)), \varepsilon > \text{, ако } walk_P(p, \lambda'(q',a)) \in P \\ < \delta'(q',a), walk_P(p, \lambda(q',a)), rest_P(p, \lambda'(q',a)), \varepsilon > \text{ иначе.} \end{cases}$$

5. $\widetilde{d} : \widetilde{Q} \times \Gamma \times \{0,1\} \to \widetilde{P} \cup \widetilde{Q}$ дефинираме чрез равенствата:

$$\widetilde{d}(< q', q, \alpha, \gamma, 0 >, b, 1) = \begin{cases} < q', d(q,b,1), \alpha, \gamma, 0 > \text{ ако } d(q,b,1) \in Q \\ \\ < q', walk_P(q_1, \alpha), rest_P(q_1, \alpha), \gamma \circ print_P(q_1, \alpha), 0 >, \\ \text{ако } q_1 = d(q,b,1) \in P \text{ и } walk_P(q_1, \alpha) \in Q \\ \\ < q', walk_P(q_1, \alpha), \gamma \circ print_P(q_1, \alpha) >, \\ \text{ако } q_1 = d(q,b,1) \text{ и } walk_P(q_1, \alpha) \in P. \end{cases}$$

$$\widetilde{d}(< q', q, \alpha, \gamma, 0 >, b, 0) = \begin{cases} < q', walk_Q(q, \alpha, b\gamma), print_Q(q, \alpha, b\gamma) >, \\ \text{ако } walk_Q(q, \alpha, b\gamma) \in P \\ \\ < q', walk_Q(q, \alpha, b\gamma), rest_Q(q, \alpha, b\gamma), print_Q(q, \alpha, b\gamma), 0 >, \\ \text{ако } walk_Q(q, \alpha, b\gamma) \in Q. \end{cases}$$

$$\widetilde{d}(< q', q, \alpha, \gamma, 1 >, b, 1) = \begin{cases} < q', d(q, b, 1), \alpha, \gamma, 1 >, \text{ ако } d(q, b, 1) \in Q \\ \\ < q', walk_P(q_1, \alpha), rest_P(q_1, \alpha), \gamma \circ print_P(q_1, \alpha), 1 >, \\ \text{ако } q_1 = d(q, b, 1) \in P \text{ и } walk_P(q_1, \alpha) \in Q \\ \\ < q', \phi(walk_P(q_1, \alpha)), \varepsilon, \gamma \circ print_P(q_1, \alpha), 0 >, \\ \text{ако } q_1 = d(q, b, 1) \text{ и } walk_P(q_1, \alpha) \in P. \end{cases}$$

$$\widetilde{d}(< q', q, \gamma, \alpha, 1 >, b, 0) = \begin{cases} < q', \phi(walk_Q(q, \alpha, b\gamma)), \varepsilon, print_Q(q, \alpha, b\gamma) >, \\ \text{ако } walk_Q(q, \alpha, b\gamma) \in P \\ \\ < q', walk_Q(q, \alpha, b\gamma), rest_Q(q, \alpha, b\gamma), print_Q(q, \alpha, b\gamma), 1 >, \\ \text{ако } walk_Q(q, \alpha, b\gamma) \in Q. \end{cases}$$

6. $\widetilde{\lambda} : \widetilde{P} \times \Omega \to \Gamma^*$ се задава от равенството:

$$\widetilde{\lambda}(< q', p, \gamma >, a) = \gamma \circ print_P(p, \lambda'(q', a)).$$

7. $\widetilde{out} : \widetilde{Q} \times \Gamma \times \{0, 1\} \to \Xi^*$ се определя като:

$$\widetilde{out}(< q', q, \alpha, \gamma, j >, b, 1) = out(q, b, 1)$$
$$\widetilde{out}(< q', q, \alpha, \gamma, j >, b, 0) = output_Q(q, \alpha, b\gamma).$$

за $j \in \{0, 1\}$.

8. $\widetilde{\phi} : \widetilde{P} \to \widetilde{Q}$ се задава чрез равенството:

$$\widetilde{\phi}(< q', p, \gamma >) = \begin{cases} < q', walk_P(p, \phi'(q')), rest_P(p, \phi'(q')), \gamma \circ print_P(p, \phi'(q')), 1 >, \\ \text{ако } walk_P(p, \phi(q')) \in Q \\ \\ < q', \phi(p_1), \varepsilon, \gamma \circ print_P(p, \phi'(q')), 0 >, \\ \text{ако } p_1 = walk_P(p, \phi(q')) \in P \end{cases}$$

9. $\widetilde{\psi} : \widetilde{Q} : \Xi^*$ дефинираме като:

$$\widetilde{\psi}(< q', q, \alpha, \gamma, j >) = \begin{cases} output_Q(q, \alpha, \gamma) \circ \psi(q_1) \\ \text{където } q_1 = walk_Q(q, \alpha, \gamma) \in Q, \quad j = 0, \\ \alpha = \varepsilon \text{ и } print_Q(q, \alpha, \gamma) = \varepsilon \\ \\ \neg! \text{ иначе.} \end{cases}$$

За краткост в следващите твърдения ще бележим:

$$\widetilde{Q_0} = Q' \times Q \times (Rest(\lambda') \cup Print(\lambda')) \times \{0\}$$

и аналогично:

$$\widetilde{Q_1} = Q' \times Q \times (Rest(\lambda') \cup Print(\lambda')) \times \{1\}$$

Следващото твърдение ще ни даде възможност да докажем, че $\widetilde{T}$ действително разпознава $f_T \circ f_{T'}$.

Твърдение 4 *Нека $\widetilde{p_1}, \widetilde{p_2} \in \widetilde{P} \cup \widetilde{Q_0}$. Нека q'_i, p_i, x_i и z_i за $i = 1, 2$ са такива, че $\widetilde{p_i} =< q'_i, p_i, x_i, z_i, 0 >$ ако $\widetilde{p_i} \in \widetilde{Q_0}$ и $\widetilde{p_i} =< q'_i, p_i, z_i >$ и $x_i = \varepsilon$ в противен случай. Тогава за всеки $\beta_1, \beta_2 \in \Omega^*$, $\gamma_1, \gamma_2 \in \Gamma^*$, $\omega_1, \omega_2 \in \Xi^*$, за които съществува изводът:*

$$< \widetilde{p_1}, \beta_1\beta_2, \gamma_1, \omega_1 >\models< \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >,$$

е изпълнено, че:

$$\begin{gathered} (\delta')^*(q_1, \beta_1) = q_2 \text{ и} \\ < p_1, x_1(\lambda')^*(q_1, \beta_1)\alpha, \gamma_1 z_1, \omega_1 >\models< p_2, x_2\alpha, \gamma_2 z_2, \omega_2 >, \end{gathered}$$

където $\alpha \in \Sigma^*$ е произволно.

Доказателство: Индукция по дължината на извода.

1. Ако

$$< \widetilde{p_1}, \beta_1\beta_2, \gamma_1, \omega_1 > = < \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >,$$

то твърдението е очевидно, защото тогава $\beta_1 = \varepsilon$, $x_1 = x_2$, $z_1 = z_2$, $q'_1 = q'_2$, $p_1 = p_2$, $\gamma_1 = \gamma_2$, $\omega_1 = \omega_2$. При това лесно се вижда, че $(\lambda')^*(q_1, \beta_1) = \varepsilon$, $(\delta')^*(q'_1, \beta_1) = q'_2$ и

$$< p_1, x_1\alpha, \gamma_1 z_1, \omega_1 > = < p_2, x_2\alpha, \gamma_2 z_1, \omega_2 >,$$

което показва верността на твърдението.

2. Нека изводът е съставен от:

$$< \widetilde{p_1}, \beta_1\beta_2, \gamma_1, \omega_1 > \vdash < \widetilde{p}, \beta\beta_2, \gamma, \omega > \models < \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >$$

Тъй като функциите $\widetilde{d}$ и $\widetilde{\Delta}$ върху аргументи от $\widetilde{P} \cup \widetilde{Q_0}$ приемат стойности от $\widetilde{P} \cup \widetilde{Q_0}$, то и $\widetilde{p} \in \widetilde{P} \cup \widetilde{Q_0}$

Нека q, p, x и z са такива, че $\widetilde{p} = < q, p, x, z, 0 >$, ако $\widetilde{p} \in \widetilde{Q_0}$ и $\widetilde{p} = < q, p, z >$, $x = \varepsilon$ в противен случай.

В зависимост от първия преход имаме три случая:

(a) $\widetilde{p_1} \in \widetilde{P}$. Тогава $\beta_1 = b\beta$ за някое $b \in \Omega$. От дефиницията на $\widetilde{\Delta}$ получаваме, че $q' = \delta'(q'_1, b)$, $p = walk_P(p, \lambda'(q'_1, b))$, $z = \varepsilon$, $\gamma = \gamma_1 \circ z_1 \circ print_P(p_1, \lambda'(q'_1, b))$ при това ако $p \in Q$, то $x = rest_P(p_1, \lambda'(q'_1, b))$. Тогава от лема 3 получаваме, че съществува изводът:

$$\begin{gathered} < p_1, \lambda'(q'_1, b)\alpha, \gamma_1 z_1, \omega_1 > \models \\ < p, rest_P(p_1, \lambda'(q'_1, b)) \circ \alpha, \gamma_1 z_1 \circ print_P(p, \lambda'(q'_1, b)), \omega_1 > \end{gathered}$$

Сега твърдението следва от индукционната хипотеза, приложена за $\widetilde{p}$ и $\widetilde{p_2}$. Наистина тъй като съществува извод

$$< \widetilde{p}, \beta\beta_2, \gamma, \omega_1 > \models < \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >,$$

то от индукционното предположение имаме, че

$$\begin{gathered} (\delta')^*(q', \beta) = q'_2 \text{ и} \\ < p, x(\lambda')^*(q', \beta)\alpha, \gamma z, \omega_1 > \models < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > . \end{gathered}$$

Но $z = \varepsilon$. Сега е ясно, че:

$$(\delta')^*(q_1', b\beta) = (\delta')^*(q', \beta) = q_2'.$$

Аналогично имаме, че $(\lambda')^*(q_1', b\beta) = \lambda'(q_1', b) \circ (\lambda')^*(q', \beta)$. Тогава от лема 3 получаваме, че съществува изводът:

$$\begin{gathered} < p_1, (\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \models \\ < p, rest_P(p_1, (\lambda'(q_1', b)) \circ (\lambda')^*(q', \beta) \circ \alpha, \gamma, \omega_1 >= \\ < p, x(\lambda')^*(q', \beta)\alpha, \gamma z, \omega_1 > \end{gathered}$$

Но от индукционното предполежение съществува изводът:

$$< p, x(\lambda')^*(q', \beta)\alpha, \gamma z, \omega_1 > \models < p_2, x_2\alpha, \gamma_2, \omega_2 >,$$

което доказва твърдението в този случай.

(б) Нека сега $p_1 \in Q$ и $\gamma_1 = b\gamma$, $\gamma \neq \varepsilon$. Тогава $\widetilde{p} = \widetilde{d}(\widetilde{p_1}, b, 1)$. Да забележим, че независимо от $d(p_1, b, 1)$ е в сила, че $q' = q_1'$, откъдето $(\delta')^*(q_1', \beta_1) = (\delta')^*(q', \beta_1)$.
Сега ако $d(p_1, b, 1) \in Q$, то имаме, че:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \vdash < d(p, b, 1), x_1\lambda^*(q', \beta_1)\alpha, \gamma z_1, \omega_1 \circ out(p_1, b, 1) > .$$

От друга страна от дефиницията на $\widetilde{d}$ в този случай знаем, че $p = d(q, b, 1)$, $x_1 = x$ и $z_1 = z$, а от дефиницията на $\widetilde{out}$ имаме, че $\widetilde{out}(\widetilde{p_1}, b, 1) = out(p_1, b, 1)$. Това означава, че в T всъщност имаме прехода:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \vdash < p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega >$$

Сега обаче в $\widetilde{T}$ е дефиниран изводът:

$$< \widetilde{p}, \beta_1\beta_2, \gamma, \omega > \models < \widetilde{p_2}, \beta_2, \gamma_1, \omega_2 >$$

Тогава от индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$ получаваме, че е дефиниран и изводът:

$$< p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega > \models < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > .$$

Като добавим и прехода:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \vdash < p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega >$$

получаваме желания резултат и за $\widetilde{p_1}$ и $\widetilde{p_2}$.
Нека сега $d(p_1, b, 1) \in P$. Тогава от дефиницията на $\widetilde{d}$ и от лема 3 следва, че:

$$\begin{gathered} p = walk_P(d(p_1, b, 1), x_1) \\ x = rest_P(d(p_1, b, 1), x_1) \\ z = z_1 \circ print_P(d(p, b, 1), x_1). \end{gathered}$$

Сега отново от лема 3 следва, че е дефиниран изводът:

$$\begin{gathered} < d(p_1, b, 1), x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma z_1, \omega \circ out(p_1, b, 1) > \models \\ < p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega > . \end{gathered}$$

От друга страна $\widetilde{out}(\widetilde{p_1}, b, 1) = out(p_1, b, 1)$. Тогава в T съществува преходът:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \vdash < d(p_1, b, 1), x_1(\lambda')^*(q', \beta_1)\alpha, \gamma z_1, \omega > .$$

Сега можем да заключим, че в T е дефиниран и изводът:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \models < p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega > .$$

Тъй като:

$$< \widetilde{p}, \beta_1\beta_2, \gamma, \omega > \models < \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >$$

можем да приложим индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$ и получаваме, че съществува и изводът:

$$< p, x(\lambda')^*(q', \beta_1)\alpha, \gamma z, \omega > \models < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > .$$

Това показва, че и:

$$< p_1, x_1(\lambda')^*(q_1', \beta_1)\alpha, \gamma_1 z_1, \omega_1 > \models < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > .$$

за всяко $\alpha \in \Sigma^*$.

(в) $\widetilde{p_1} \in \widetilde{Q}$ и $\gamma_1 = b \in \Gamma$. Тогава $\widetilde{p} = \widetilde{d}(p, b, 0)$ и $\omega = \omega_1 \circ \widetilde{out}(\widetilde{p}, b, 0)$. От дефиницията на $\widetilde{d}$ следва, че $q' = q'_1$, $p = walk_Q(p_1, x_1, bz_1)$, $z = print_Q(p_1, x, bz_1)$, $x = rest_Q(p_1, x_1, bz_1)$. От дефиницията на $\widetilde{out}$ пък следва, че $\omega = \omega_1 \circ print_Q(p_1, x_1, bz_1)$. Сега от лема 4 получаваме, че за всяко α:

$$< p_1, x_1\alpha, bz_1, \omega_1 > \models < p, x\alpha, z, \omega > .$$

Сега прилагаме индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$. Тъй като е в сила изводът:

$$< \widetilde{p}, \beta_1\beta_2, \varepsilon, \omega > \models < \widetilde{p_2}, \beta_2, \gamma_2, \omega_2 >,$$

то

$$\begin{gathered} (\delta')^*(q', \beta_1) = q'_2 \\ < p, x(\lambda')^*(q', \beta_1) \circ \alpha, z, \omega > \models < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > . \end{gathered}$$

Сега лесно получаваме, че и:

$$(\delta')^*(q'_1, \beta_1) = (\delta')^*(q', \beta_1) = q'_2 \text{ и}$$

$$\begin{gathered} < p_1, x_1 \circ (\lambda')^*(q'_1, \beta_1) \circ \alpha, bz_1, \omega_1 > \models \\ < p, x \circ (\lambda')^*(q'_1, \beta_1) \circ \alpha, z, \omega > \models \\ < p_2, x_2\alpha, \gamma_2 z_2, \omega_2 > . \end{gathered}$$

Това показва верността на твърдението и в този случай.

С това от принципа за математическа индукция, следва, че твърдението е вярно за произволни $\widetilde{p_1}$, $\widetilde{p_2}$ и β_1.

Аналогично твърдение е вярно ако и двете състояния са в $\widetilde{Q_1}$.

Твърдение 5 *Нека* $\widetilde{p_1} =< q'_1, p_1, x_1, z_1, 1 >$ *и* $\widetilde{q_2} =< q'_2, p_2, x_2, z_2, 1 >$ *са две състояния от* $\widetilde{Q_1}$ *като за тях е изпълнено, че:*

$$< \widetilde{p_1}, \varepsilon, \gamma_1, \omega_1 > \models < \widetilde{p_2}, \varepsilon, \gamma_2, \omega_2 > .$$

Тогава е в сила, че:

$$q'_1 = q'_2$$

и освен това:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

Доказателство: Индукция по дължината на извода.

1. Ако

   $$< \widetilde{p_1}, \varepsilon, \gamma_1, \omega_1 > = < \widetilde{p_2}, \varepsilon, \gamma_2, \omega_2 >,$$

   то директно получаваме, че и

   $$< p_1, x_1, \gamma_1 z_1, \omega_1 > = < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

   Освен това и $q'_1 = q'_2$.

2. Нека сега изводът се състои от прехода:

   $$< \widetilde{p_1}, \varepsilon, \gamma_1, \omega_1 > \vdash < \widetilde{p}, \varepsilon, \gamma, \omega >$$

   и извода:

   $$< \widetilde{p}, \varepsilon, \gamma, \omega > \models < \widetilde{p_2}, \varepsilon, \gamma_2, \omega_2 >$$

   Да забележим, че тъй като $\widetilde{d}$ и $\widetilde{\Delta}$ приемат стойности само от $\widetilde{P} \cup \widetilde{Q_0}$, когато аргументите им са от това множество, то вторият извод е възможен само ако $\widetilde{p} \in \widetilde{Q_1}$. Затова нека $\widetilde{p} = < q', p, x, z, 1 >$. Сега имаме три случая в зависимост от γ_1 и d.

   (a) $\gamma_1 = b\gamma$, $\gamma \neq \varepsilon$, $d(p_1, b, 1) \in Q$. Тогава от дефиницията на $\widetilde{d}$ получаваме, че:

   $$\begin{aligned} p &= d(p_1, b, 1) \\ x &= x_1 \\ z &= z_1 \\ q' &= q'_1. \end{aligned}$$

Освен това в този случай $\widetilde{out}(\widetilde{p_1}, b, 1) = out(p_1, b, 1)$. Сега да забележим, че в T е дефиниран преходът:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \vdash < p, x, \gamma z, \omega > .$$

Тогава, прилагайки индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$ получаваме, че съществува изводът:

$$< p, x, \gamma z, \omega > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

Обединявайки тези два резултата получаваме и изводът:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

Също от индукционното предположение следва, че $q'_2 = q' = q'_1$.

(б) $\gamma_1 = b\gamma$, $\gamma \neq \varepsilon$, $d(p_1, b, 1) \in P$. Тогава от дефиницията на $\widetilde{d}$ получаваме, че:

$$\begin{gathered} p = walk_P(d(p_1, b, 1), x_1) \\ x = rest_P(d(p_1, b, 1), x_1) \\ z = z_1 \circ print_P(d(p_1, b, 1), x_1) \\ q' = q'_1. \end{gathered}$$

Тогава според лема 3 съществува изводът:

$$< d(p, b, 1), x_1, \gamma z_1, \omega > \models < p, x, \gamma z, \omega >$$

Остана да забележим, че $\widetilde{out}(\widetilde{p_1}, b, 1) = out(p_1, b, 1)$ и, че имаме прехода:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \vdash < d(p, b, 1), x_1, \gamma_1 z_1, \omega > .$$

Събирайки тези два резултата заедно получаваме извода:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \models < p, x, \gamma z, \omega > .$$

Сега прилагаме индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$ и получаваме, че е дефиниран изводът:

$$< p, x, \gamma z, \omega > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

Това показва, че и:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

Освен това $q_1' = q' = q_2'$, като последното равенство следва от индукционното предположение.

(в) $\gamma_1 = b \in \Gamma$. Тогава по дефиницията на $\widetilde{d}$ имаме, че

$$\begin{gathered} p = walk_Q(p_1, x_1, bz_1) \\ x = rest_Q(p_1, x_1, bz_1) \\ z = print_Q(p_1, x_1, bz_1) \\ q' = q_1'. \end{gathered}$$

Сега от лема 4 следва, че съществува изводът:

$$< p_1, x_1, bz_1, \omega_1 > \models < p, x, z, \omega \circ output_Q(p_1, x_1, bz_1) > .$$

Сега като забележим, че $\widetilde{out}(\widetilde{p_1}, b, 0) = output_Q(p_1, x_1, bz_1)$ получаваме, че:

$$< p_1, x_1, bz_1, \omega_1 > \models < p, x, z, \omega > .$$

Сега можем да приложим индукционното предположение за $\widetilde{p}$ и $\widetilde{p_2}$, което ни дава, че съществува изводът:

$$< p, x, z, \omega > \models < p_2, x_2, \gamma_2 z_2, \omega_2 > .$$

С това получаваме, че и изводът:

$$< p_1, x_1, \gamma_1 z_1, \omega_1 > \models < p_2, x_2, \gamma_2 z_2, \omega_2 >$$

е определен и, че $q' = q_2'$, тоест $q_1' = q_2'$.

С това индукционната стъпка е направена, което доказва лемата.

Твърдение 6 *Нека* $q'_1, q'_2 \in Q'$, $\beta_1 \in \Omega^*$, $p_1, p_2 \in P$. *Тогава ако*

$$(\delta')^*(q'_1, \beta_1) = q'_2$$

и

$$< p_1, (\lambda')^*(q'_1, \beta_1), \varepsilon, \varepsilon > \models < p_2, \varepsilon, \gamma_1, \omega_1 >,$$

то съществува извод

$$< \widetilde{p_1}, \beta_1, \varepsilon, \varepsilon > \models < \widetilde{p_2}, \varepsilon, \gamma_2, \omega_1 >,$$

за който $\widetilde{p_1} =< q'_1, p_1, \varepsilon >$ и $\widetilde{p_2} =< q'_2, p_2, z >$ за някое z.

Доказателство: Да допуснем, че твърдението не е вярно и нека q'_1, q'_2, β_1, p_1, p_2, γ_1 и ω_1 удовлетворяват предпоставката на твърдението. Да разгледаме най-дългия възможен извод, започващ с конфигурацията $< \widetilde{p}, \beta_1, \varepsilon, \varepsilon >$, където $\widetilde{p} =< q'_1, p_1, \varepsilon >$. Нека това е:

$$< \widetilde{p_1}, \beta_1, \varepsilon, \varepsilon > \models < \widetilde{p}, \beta, \gamma, \omega > .$$

От дефиницията на $\widetilde{\Delta}$ и $\widetilde{d}$ лесно се вижда, че всяко състояние, получено по този извод е в множеството $\widetilde{P} \cup \widetilde{Q_0}$. В частност и $p \in \widetilde{P} \cup \widetilde{Q_0}$. Нека q', p, z и x са такива, че $\widetilde{p} =< q', p, x, z, 0 >$ ако $\widetilde{p} \in \widetilde{Q_0}$ и $\widetilde{p} =< q', p, z >$ и $x = \varepsilon$ иначе. Нека $\beta' = \beta_1 \beta^{-1}$. Тогава от твърдение 4 следва, че:

$$(\delta')^*(q'_1, \beta') = q'$$
$$< p, (\lambda')^*(q'_1, \beta_1), \varepsilon, \varepsilon > \models < p', x(\lambda')^*(q', \beta), \gamma z, \omega >$$

Ако допуснем, че $\beta = \varepsilon$ и $\widetilde{p} \in \widetilde{P}$ получаваме, че $q' = q'_2$ и $p_2 = p$, което е противоречие, с това, че твърдението не е изпълнено за p_1, q'_1, p_2, q'_2 и β_1. Сега от максималността на извода следва, че не съществува преход от $< \widetilde{p}, \beta, \gamma, \omega >$. Имаме три възможности:

1. $\widetilde{p} \in \widetilde{P}$, b е първата буква на β, но $\widetilde{\Delta}(\widetilde{p}, b)$ не е дефинирана.

   Тогава тъй като $q_2 = (\delta')^*(q'_1, \beta_1) = (\delta')^*(q', \beta)$, то $\delta'(q', b)$ е дефинирана. Тогава остава да не е дефинирана $walk_P(p, (\lambda')(q', b))$.

Според лема 3 в този случай не съществува извод от конфигурацията:

$$< p, \lambda'(q', b)\alpha, \gamma z, \omega >$$

с дължина по-голяма от $|\lambda'(q', b)| - 1$, което е противоречие с факта, че съществува изводът:

$$< p, (\lambda')^*(q', \beta), \gamma z, \omega > \models < p_2, \varepsilon, \gamma_2, \omega_2 > .$$

2. $\widetilde{p} \in \widetilde{Q_0}$, $\gamma = b\gamma'$, $\gamma' \neq \varepsilon$ и $\widetilde{d}(\widetilde{p}, b, 1)$ не е дефинирана. Тогава получаваме, че не е дефинирана и функцията $d(p, b, 1)$ или, че $d(p, b, 1) \in P$ и не е дефинирана $walk_P(d(p, b, 1), x)$. В първия случай получаваме, че няма преход и от конфигурацията:

$$< p, x(\lambda')^*(q', \beta), \gamma z, \omega > .$$

Във втория прилагаме лема 3 и получаваме, че не съществува извод с дължина по-голяма от $|x| - 1$ от конфигурацията

$$< d(p, b, 1), x\lambda(q', \beta), \gamma' z, \omega >,$$

което също е противоречие.

3. $\widetilde{p} \in \widetilde{Q}$, $\gamma = b$. Тогава от дефиницията на $\widetilde{d}$ получаваме, че не е дефинирана стойността $walk_Q(p, x, bz)$. От лема 4 следва, че най-дългият извод с начална конфигурация

$$< p, x\lambda(q', \beta), \gamma z, \omega >$$

или съдържа по-малко от $|x|$ прехода от тип 1, или завършва в състояние $q \in Q$. И в двата случая получаваме противоречие с факта, че:

$$< p, (\lambda')^*(q', \beta), \gamma z, \omega > \models < p_2, \varepsilon, \gamma_2, \omega_2 > .$$

Сега сме готови да докажем, че $f_{\widetilde{T}} = f_T \circ f_{T'}$.

Нека първо $\beta \in Dom(f_{\widetilde{T}})$. Това по дефиниция означава, че съществуват $\widetilde{p_1} \in \widetilde{P}$ и $\widetilde{p_2} \in \widetilde{Q}$, за които:

$$< \widetilde{s}, \beta, \varepsilon, \varepsilon > \models < \widetilde{p_1}, \varepsilon, \gamma, \omega_1 >$$
$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma, \omega_1 > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \omega_2 > .$$

Нека $\widetilde{p_1} =< q_1', p_1, z_1 >$, $\widetilde{p_2} =< q_2', p_2, x_2, z_2, 0 >$. Последното е в сила, защото $\widetilde{\psi}(\widetilde{p_2})$ е дефинирана. Тогава от твърдение 4, приложено за $\widetilde{s} =< s', s, \varepsilon >$ и $\widetilde{p_1}$ получаваме, че:

$$(\delta')^*(s', \beta) = q_1'$$
$$< s, (\lambda')^*(s', \beta)\phi'(q_1'), \varepsilon, \varepsilon > \models < p_1, \phi'(q_1'), \gamma z_1, \omega_1 > .$$

Това, че $\phi'(q_1')$ е дефинирана, означава, че $\beta \in Dom(f_{T'})$ и следователно $f_{T'}(\beta) = (\lambda')^*(s', \beta)\phi'(q_1')$.

Сега от дефиницията на $\widetilde{\phi}$ имаме два случая:

1. $walk_P(p_1, \phi'(q_1')) \in Q$. Тогава полагаме

$$p_3 = walk_P(p_1, \phi'(q_1'))$$
$$x_3 = rest_P(p_1, \phi'(q_1'))$$
$$z_3 = z_1 print_P(p_1, \phi'(q_1')).$$

Следователно $\widetilde{\phi}(\widetilde{p_1}) =< q_1', p_3, x_3, z_3, 1 >$. От друга страна от лема 3 получаваме, че съществува изводът:

$$< p_1, \phi'(q_1'), \gamma z_1, \omega_1 > \models < p_3, x_3, \gamma z_3, \omega_1 > \qquad (6)$$

Нека разгледаме сега извода:

$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma, \omega_1 > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \omega_2 > .$$

Нека $< \widetilde{p_4}, \varepsilon, \gamma_4, \omega_4 >$ е последната конфигурация от него, за която $\widetilde{p_4} \in \widetilde{Q_1}$. Нека $\widetilde{p_4} =< q_1', p_4, x_4, z_4, 1 >$. Тогава по лема 5 следва, че

$$< p_3, x_3, \gamma z_3, \omega_1 > \models < p_4, x_4, \gamma_4 z_4, \omega_4 > \qquad (7)$$

Тъй като $\widetilde{q_2} \in \widetilde{Q_0}$ съществува преход от $< \widetilde{p_4}, \varepsilon, \gamma_4, \omega_4 >$. Нека това е:

$$< \widetilde{p_4}, \varepsilon, \gamma_4, \omega_4 > \vdash < \widetilde{f}, \varepsilon, \gamma_f, \omega_f > .$$

От дефиницията на $\widetilde{d}$ и от лема 3 и лема 4 се вижда, че в този случай $\widetilde{f} =< q', \phi(p), \varepsilon, z, 0 >$, където $p \in P$ и освен това:

$$< p_4, x_4, \gamma_4 z_4, \omega_4 > \models < p, \varepsilon, \gamma_f z, \omega_f > . \qquad (8)$$

Сега като обединим резултатите 6, 7 и 8 получаваме, че съществува изводът:

$$< p_1, \phi'(q_1'), \gamma_1, \omega_1 > \models < p, \varepsilon, \gamma z, \omega_f > . \qquad (9)$$

Тъй като

$$< \widetilde{f}, \varepsilon, \gamma, \omega_f > \models < \widetilde{p_2}, \varepsilon, \varepsilon, \omega_2 >$$

и $\widetilde{f} \in \widetilde{Q_0}$ и $\widetilde{p_2} \in \widetilde{Q_0}$ от твърдение 4 следва, че е дефиниран и изводът:

$$< f, \varepsilon, \gamma z, \omega_f > \models < p_2, \varepsilon, z_2, \omega_2 > \qquad (10)$$

Накрая от дефиницията на $\widetilde{\psi}$ получаваме, че

$$f_{\widetilde{T}}(\beta) = \omega_2 \circ output_Q(p_2, \varepsilon, z_2) \circ \psi(walk_Q(p_2, \varepsilon, z_2)$$

и освен това

$$print_Q(p_2, \varepsilon, z_2) = \varepsilon.$$

Сега като приложим лема 4 получаваме, че:

$$< p_2, \varepsilon, z_2, \omega_2 > \models$$
$$< walk_Q(p_2, \varepsilon, z_2), \varepsilon, print_Q(p_2, \varepsilon, z_2), \omega_2 \circ print_Q(p_2, \varepsilon, z_2) > .$$

Сега като обединим 9, 10 и 11 получаваме, че:

$$< s, f_{T'}(\beta), \varepsilon, \varepsilon > \models < p, \varepsilon, \gamma z, \omega_f >$$
$$< \phi(p), \varepsilon, \gamma z, \omega_f > \models < walk_Q(p_2, \varepsilon, z_2), \varepsilon, \varepsilon, \omega_2 \circ output_Q(p_2, \varepsilon, z_2) > .$$

Сега по дефиниция имаме, че

$$f_T(f_{T'}(\beta)) = \omega_2 \circ output_Q(p_2, \varepsilon, z_2) \circ \psi(walk_Q(p_2, \varepsilon, z_2)) = f_{\widetilde{T}}(\beta).$$

2. Вторият случай се свежда до първия като вземем в предвид, че тогава $p = walk_P(p_1, \phi'(q'_1)) \in P$. Тогава направо получаваме, че $\widetilde{\phi}(\widetilde{p_1}) =< q'_1, f, \varepsilon, z, 0 >$ в същите означения като в първия случай. Сега обаче директно от лема 3 имаме, че:

$$< p_1, \phi'(q'_1), \gamma_1 z_1, \omega_1 > \vDash < p, \varepsilon, \gamma_1 z, \omega_f > .$$

Оттук нататък разсъжденията са същите.

Нека сега $\beta \in Dom(f_{T'})$ и $f_{T'}(\beta) \in Dom(f_T)$. Нека тогава:

$$\begin{gathered} q'_1 = (\delta')^*(s', \beta) \\ < s, (\lambda')^*(s', \beta)\phi'(q'), \varepsilon, \varepsilon > \vDash < p_1, \phi'(q'), \gamma_1, \omega_1 > \\ < p_1, \phi'(q'), \gamma_1, \omega_1 > \vDash < p_2, \varepsilon, \gamma_2, \omega_2 > \\ < \phi(p_2), \varepsilon, \gamma_2, \omega_2 > \vDash < p_3, \varepsilon, \varepsilon, \omega_3 >, \end{gathered}$$

където $p_1, p_2 \in P$, $p_3 \in Q$. Тогава от първите два извода и от твърдение 6 следва, че:

$$< \widetilde{s}, \beta, \varepsilon, \varepsilon > \vDash < \widetilde{p}, \varepsilon, \gamma'_1, \omega_1 >,$$

където $\widetilde{p} =< q'_1, p_1, z_1 >$ за някое z_1. Тогава от лема 6 получаваме, че $\gamma_1 = \gamma'_1 z_1$. Сега от дефиницията на $\widetilde{\phi}(\widetilde{p_1})$ следва, че тя е дефинирана ако е дефинирана и функцията $walk_P(p_1, \phi'(q'_1))$. Но от лема 3 това е така, защото иначе от конфигурацията $< p_1, \phi'(q'_1), \gamma_1, \omega_1 >$ нямаше да има извод с дължина по-голяма от $|\phi'(q'_1)| - 1$, което очевидно не е вярно.

Нека сега разгледаме най-дългия извод с начало

$$< \widetilde{\phi}(\widetilde{p_1}), \varepsilon, \gamma'_1, \omega_1 > \vDash < \widetilde{f}, \varepsilon, \gamma'_2, \omega'_2 >,$$

който минава само през състояния от $\widetilde{Q_1}$. Нека $\widetilde{f} =< q'_1, f, x, z, 1 >$. Тогава от твърдение 5 следва, че:

$$< p_1, \phi'(q'), \gamma'_1 z_1, \omega_1 > \vDash < f, x, \gamma'_2 z, \omega'_2 > .$$

От лема 4 и лема 3, тъй като от $< f, x, \gamma_2' z, \omega_2' >$ има извод до $< p_2, \varepsilon, \gamma_2, \omega_2 >$, следва, че от конфигурацията $< \widetilde{f}, \varepsilon, \gamma_2', \omega_2' >$ съществува преход (функцията $\widetilde{d}$ е дефинирана). Тогава отново от максималността на извода със състояния от $\widetilde{Q_1}$ следващата конфигурация трябва да е със състояние в $\widetilde{Q_0}$. Сега отново от лема 3 и лема 4 следва, че:

$$< \widetilde{f}, \varepsilon, \gamma_2', \omega_2' > \vdash < \widetilde{p_2}, \varepsilon, \gamma_3', \omega_2 >,$$

където $\widetilde{p_2} =< q_1', \phi(p_2), \varepsilon, z_2, 0 >$ и $\gamma_2 = \gamma_3' z_2$. Тогава

$$< \phi(p_2), \varepsilon, \gamma_2, \omega_2 > \models < p_3, \varepsilon, \varepsilon, \omega_3 >$$

и от твърдение 6 следва, че съществува и изводът:

$$< \widetilde{p_2}, \varepsilon, \gamma_3', \omega_2 > \models < \widetilde{p_3}, \varepsilon, \varepsilon, \omega_3' >,$$

за някое $\widetilde{p_3} =< q_1', p_3', \varepsilon, z_3 >$ и ω_3'. Сега от твърдение 4 следва, че:

$$< \phi(p_2), \varepsilon, \gamma_2, \omega_2 > \models < p_3', \varepsilon, z_3, \omega_3' >$$

Тъй като от последната конфигурация има преход до $< p_3, \varepsilon, \varepsilon, \omega_3 >$, то $walk_Q(p_3', \varepsilon, z_3)$ е дефинирана и от лема 4 следва, че $walk_Q(p_3', \varepsilon, z_3) = p_3$ $print_Q(p_3', \varepsilon, z_3) = \varepsilon$. Това означава, че $\widetilde{\psi}(\widetilde{p_3})$ е дефинирана. Това показва, че $\beta \in Dom(f_{\widetilde{T}})$. Тъй като β беше произволна от дефиниционната област на $f_T \circ f_{T'}$, то получаваме, че $Dom(f_T \circ f_{T'}) \subset Dom(f_{\widetilde{T}})$. По-рано видяхме и обратното включване заедно с равенството $f_{\widetilde{T}}(\beta) = f_T(f_{T'}(\beta))$. Следователно $f_{\widetilde{T}} = f_T \circ f_{T'}$. Това завършва и доказателството на теорема 6

4 Ω-представими функции

В тази част ще отделим клас от рационални функции, за които можем да построим FIFO-транддюсер. Всъщност за дефиниционната област на тези функции можем да си мислим като за регулярни множества над азбука Σ, които са разграничени с маркери (букви) на друга, чужда с Σ, азбука Ω. В отделните участъци на така накъсаната дума прилагаме подпоследователни преобразуватели. Това какъв точно преобразувател трябва да приложим се определя от прочетената до момента дума, която сме обработили, и краен брой сегмента от думата, която ни предстои да прочетем.

Първо ще дефинираме формално тези функции. След това ще построим FIFO-трансдюсер за всяка такава функция, а после и бимашина. Това ще означава, че действително сме отделили подклас от рационални функции. Накрая ще покажем и две свойства на Ω-представимите функции. Ще покажем устойчивост относно композиция отляво с подпоследователен преобразувател. Накрая ще разгледаме и достатъчни условия за композиция на две Ω-представими функции, които запазват представимостта.

В понататъшните ни разглеждания в тази част Σ и Ω са азбуки, които са взаимночужди:

$$\Sigma \cap \Omega = \varnothing.$$

4.1 Дефиниции

Дефиниция 13 $Pref(\Omega) = (\Sigma \cup \Omega)^*\Omega \cup \{\varepsilon\}$

Дефиниция 14 *За дума $\beta = b_1 \dots b_n \in (\Sigma \cup \Omega)^*$ с $|\beta|_\Omega$ означаваме броя на буквите $b_i \in \Omega$.*

Дефиниция 15 *За дума $\beta \in (\Sigma \cup \Omega)^*$ $j - \Omega$-представка на β наричаме представката $P_j(\beta)$, за която:*

1. *$P_j(\beta) = \beta$ ако $|\beta|_\Omega < j$ и*
2. *$P_j(\beta) \in Pref(\Omega)$ с $|P_j(\beta)|_\Omega = j$ иначе.*

Сега дефинираме $j-\Omega$*-наставка на* β *като* $S_j(\beta) = P_j(\beta)^{-1}\beta$ *и* $j-\Omega$*-инфикс на* β *като* $I_j(\beta) = P_{j-1}^{-1}(\beta)P_j(\beta)$.

Дефиниция 16 *Казваме, че една дясно-инвариантна релация на еквивалентност* $\sim_R$ *над думи от* $(\Sigma\cup\Omega)^*$ *е* Ω*-крайна ако съществува естествено число* k*, такова че за всеки* $\beta,\gamma \in (\Sigma\cup\Omega)^*$ *с* $|\beta|_\Omega \geq k$ *е в сила, че* $\beta \sim_R \beta\gamma$.

Дефиниция 17 *За релация на еквивалентност от краен индекс* $\sim_L$ *над двойки от думи от* $(Pref(\Omega)\times(\Sigma\cup\Omega)^*)\cup((\Sigma\cup\Omega)^*\times\{\varepsilon\})$ *казваме, че е* Ω*-крайно определима посредством* Ω*-крайната, дясно-инвариантна релация* $\sim_R$ *ако са в сила свойствата:*

1. $$\forall\alpha',\alpha'',(\varepsilon,\alpha')\sim_L(\varepsilon,\alpha'')$$

2. $$\begin{aligned}\forall\alpha',\beta',\alpha'',\beta'':(\alpha',\beta') &\sim_L (\alpha'',\beta'')\,\&\,\beta'\sim_R\beta''\Rightarrow\\ (\alpha'P_1(\beta'),S_1(\beta')) &\sim_L (\alpha''P_1(\beta''),S_1(\beta'')).\end{aligned}$$

Дефиниция 18 *Казваме, че* $f:(Pref(\Omega)\times(\Sigma\cup\Omega)^*\cup(\Sigma\cup\Omega)^*\times\{\varepsilon\})\to\Xi^*$ *е* Ω*-представима ако съществуват:*

1. Ω*-крайноопределима релация* $\sim_R$
2. Ω*-крайноопределима релация* $\sim_L$*, определена посредством* $\sim_R$ *и*
3. *подпоследователни трансдюсери* $T_{[\alpha][\beta]}$*, такива че* $T_{[\alpha'][\beta']} = T_{[\alpha''][\beta'']}$ *винаги когато* $(\alpha',\beta')\sim_L(\alpha'',\beta'')$ *и* $\beta'\sim_R\beta''$.

със свойството, че:

$$\begin{aligned}f(\alpha,\varepsilon) &= \varepsilon \text{ и}\\ f(\alpha,\beta) &= T_{[\alpha][\beta]}\quad(P_1(\alpha))f(\alpha P_1(\beta),S_1(\beta)) \text{ ако } \beta\neq\varepsilon.\end{aligned}$$

Дефиниция 19 *$f : (\Sigma \cup \Omega)^* \to \Xi^*$ е Ω-крайнопредставима, ако съществува Ω-представима $g : (Pref(\Omega) \times (\Sigma \cup \Omega)^*) \cup ((\Sigma \cup \Omega)^* \times \{\varepsilon\}) \to \Xi^*$ със свойството $f(\alpha) = g(\varepsilon, \alpha)$ за всяка дума α.*

Дефиниция 20 *За $\alpha \in (\Sigma \cup \Omega)^*$ дефинираме*

- *$base(\alpha)$ е $k - \Omega$-представка на α, където $k = |\alpha|_\Omega$.*
- *$last(\alpha) = base(\alpha)^{-1}\alpha$.*

Дефиниция 21 *Нека f е Ω-крайноопределима функция. Нека f се определя от подпоследователните трансдюсери*

$$T_{[\alpha][\beta]} = ((\Sigma \cup \Omega) \times \Xi^*, Q_{[\alpha][\beta]}, s_{[\alpha][\beta]}, \delta_{[\alpha][\beta]}, \lambda_{[\alpha][\beta]}, \phi_{[\alpha][\beta]}).$$

Тогава дефинираме:

1. *За $q \in Q_{[\alpha][\beta]}$ подпоследователния трансдюсер $T_{[\alpha][\beta]}(q)$:*

$$T_{[\alpha][\beta]}(q) = ((\Sigma \cup \Omega) \times \Xi^*, Q_{[\alpha][\beta]}, q, \delta_{[\alpha][\beta]}, \lambda_{[\alpha][\beta]}, \phi_{[\alpha][\beta]}).$$

2. *$state_f(\alpha, \beta)$ състояние на $T_{[base(\alpha)][last(\alpha)\beta]}$ като:*

$$state_f(\alpha, \beta) = \delta^*_{[base(\alpha)][last(\alpha)\beta]}(s_{[base(\alpha)][last(\alpha)\beta]}, last(\alpha)).$$

3. *$f^* : (\Sigma \cup \Omega)^* \times (\Sigma \cup \Omega)^* \to \Xi^*$ естествено разширение на f се дефинира като:*

$$f^*(\alpha, \beta) = T_{[base(\alpha)][last(\alpha)\beta]}(state_f(\alpha, \beta))(P_1(\beta)) \circ f(\alpha P_1(\beta), S_1(\beta)).$$

Забележка 5 *Лесно се вижда, че:*

$$f^*(\varepsilon, \beta) = T_{[\varepsilon][\beta]}(s_{[\varepsilon][\beta]})(P_1(\beta)) \circ f(P_1(\beta), S_1(\beta)) = f(\varepsilon, \beta).$$

Ще завършим този пораграф със следния факт, който ще използваме в следващите параграфи на тази част:

Факт 1 *Ако $\sim$ е дясно-инвариантна релация с индекс m над множество от думи X^*, а $Y \subset X^*$, то и релацията $x_1 \sim_P x_2$ дефинирана за всеки две $x_1, x_2 \in X^*$ чрез:*

$$\forall y_1, y_2 \in Y(y_1 \sim y_2 \Rightarrow y_1x_1 \sim y_2x_2)$$

е дясно-инвариантна релация на еквивалентност с индекс $ind(\sim_P) \leq m^m$.

Наистина това, че $\sim_P$ е рефлексивна следва от това, че $\sim$ е дясно-инвариантна. Тогава ако $y_1 \sim y_2$, то и $y_1x \sim y_2x$. Това, че $\sim_P$ е симетрична и транзитивна следва директно от съответните свойства на $\sim$. Дясната инваринтност също е очевидна. Наистина ако z е произволно и за всеки $y_1 \sim y_2$ е в сила:

$$y_1x_1 \sim y_2x_2,$$

то и $y_1x_1z \sim y_2x_2z$, защото $\sim$ е дясно-инвариантна. Това означава, че $x_1z \sim_P x_2z$, от където получаваме, че $\sim_P$ е дясно-инвариантна.

За да получим оценката за индекса на $\sim_P$ да разгледаме изображението $F([x]_{\sim_P})$, което на един клас на еквивалентност от $\sim_P$ съпоставя функция $F([x]_{\sim_P})([y]_\sim) = [yx]_\sim$, за всеки клас на еквивалентност, породен от $y \in Y$. От дефиницията на $\sim_P$ следва, че $F([x]_{\sim_P})$ е коректно дефинирана, защото ако $x_1 \sim_P x_2$ $y_1 \sim y_2$, то и $y_1x_1 \sim y_2x_2$, тоест $[y_1x_1]_\sim = [y_2x_2]_\sim$. Сега твърдението следва от това, че броят на функциите от едно множество с m елемента в друго множество с m елемента не надминава m^m.

4.2 Конструкция на FIFO-трансдюсер по дадена Ω-определима функция

Целта ни сега е да конструираме FIFO-трансдюсер по зададена Ω-(крайно)определима функция. Това ще направим в процеса на доказателство на следната:

Теорема 7 *За всяка Ω-крайноопределима функция $f : (\Sigma \cup \Omega)^* \to \Xi^*$ съществува FIFO-трансдюсер T, който разпознава f, тоест $f_T = f$.*

Доказателство: Нека $\sim_R$ е Ω-крайна релация на еквивалентност с параметър k, която Ω-определя релациятя $\sim_L$, а $T_{[\alpha][\beta]}$ са подпоследователните преобразуватели, които определят f, тоест:

$$f(\alpha) = g(\varepsilon, \alpha),$$

където $g : Pref(\Omega) \times (\Sigma \cup \Omega)^* \cup (\Sigma \cup \Omega)^* \times \{\varepsilon\} \to (\Sigma \cup \Omega)^*$ е дефинирана от равенствата:

$$g(\alpha, \beta) = \begin{cases} \varepsilon, \text{ ако } \beta = \varepsilon \\ T_{[\alpha][\beta]}(P_1(\beta) \circ g(\alpha P_1(\beta), S_1(\beta)) \end{cases}$$

Преди да преминем към доказателството на теоремата, ще ни трябват няколко помощни твърдения, които ще имат за цел да пренесем релацията $\sim_L$ само върху първата компонента от двойката. Започваме със следната:

Лема 5 *Нека релацията* $\sim_{GR} \subset (\Sigma \cup \Omega)^* \times (\Sigma \cup \Omega)^*$ *е дефинирана като:*

$$\beta' \sim_{GR} \beta'' \Leftrightarrow \forall \alpha', \alpha'' (\alpha' \sim_R \alpha'' \Rightarrow (\alpha'\beta' \sim_R \alpha''\beta'')).$$

Тогава $\sim_{GR}$ *е* Ω*-крайна, дясно-инвариантна, релация на еквивалентност с параметър* k.

Доказателство: Това, че $\sim_{GR}$ е дясно-инвариантна, крайна релация на еквивалентност следва директно от факт 1. Така че, трябва да покажем само, че ако $|\beta'|_\Omega \geq k$, то за всяко $\gamma \in (\Sigma \cup \Omega)^*$ в сила, че:

$$\beta'\gamma \sim_{GR} \beta'.$$

Наистина, нека $\alpha' \sim_R \alpha''$. Тогава имаме, че $\alpha'\beta' \sim_R \alpha''\beta'$. Тъй като $|\beta'|_\Omega \geq k$, то и $|\alpha'\beta'| \geq k$ и $|\alpha''\beta'|_\Omega \geq k$. Сега от дясната инвариантност на $\sim_R$ получаваме, че:

$$\alpha'\beta'\gamma \sim_R \alpha''\beta'\gamma.$$

Остана да използваме, че $\sim_R$ е Ω-крайна. Тогава от $|\alpha'\beta'| \geq k$ и $|\alpha''\beta'| \geq k$ получаваме, че:

$$\alpha'\beta' \sim_R \alpha'\beta'\gamma \text{ и}$$
$$\alpha''\beta' \sim_R \alpha''\beta'\gamma.$$

Оттук непосредствено следва, че:

$$\alpha'\beta'\gamma \sim_R \alpha''\beta'.$$

Но α' и α'' бяха произволни, откъдето заключаваме, че:

$$\beta' \sim_{GR} \beta'\gamma,$$

тоест $\sim_{GR}$ наистина е Ω-крайна с параметър k.

Следващата лема показва как можем да "прехвърлим" релацията $\sim_L$ само върху първата компонента на двойката:

Лема 6 *Нека релацията $\sim_{GL} \subset (\Sigma \cup \Omega)^*$ се определя по следния начин:*

$$\alpha' \sim_{GL} \alpha'',$$

точно когато са в сила следните условия:

1. $\alpha' \in Pref(\Omega) \Leftrightarrow \alpha'' \in Pref(\Omega)$.

2. *ако* $\alpha' \in Pref(\Omega)$ *и* $\alpha'' \in Pref(\Omega)$, *то:*

$$\alpha' \sim_R \alpha'' \text{ и}$$
$$\forall \beta' \sim_{GR} \beta'' \Rightarrow (\alpha', \beta') \sim_L (\alpha'', \beta'').$$

Тогава са в сила следните твърдения:

1. $\sim_{GL}$ *крайна релация на еквивалентност.*

2. *Нека* $\alpha' \sim_{GL} \alpha''$ *и* $\alpha' \in Pref(\Omega)$, *а* $S_1(\beta') \sim_{GR} S_1(\beta'')$ *и* $P_1(\beta') \sim_{GR} P_1(\beta'')$. *Тогава:*

$$\alpha' P_1(\beta') \sim_{GL} \alpha'' P_1(\beta'').$$

Доказателство:

1. Това, че $\sim_{GL}$ е симетрична и транзитивна е очевидно. Ще покажем, че $\sim_{GL}$ е рефлексивна. Ако $\alpha \notin Pref(\Omega)$ твърдението е очевидно. Затова ще разгледаме само случая, когато $\alpha \in Pref(\Omega)$.

И така нека $\alpha \in Pref(\Omega)$ и $\beta' \sim_{GR} \beta''$. Тогава е в сила, че:

$$\alpha\beta' \sim_R \alpha\beta''.$$

По дефиниция е вярно, че:

$$(\varepsilon, \alpha\beta') \sim_L (\varepsilon, \alpha\beta'').$$

С индукция по $i \leq |\alpha|_\Omega$ ще покажем, че:

$$(P_i(\alpha), S_i(\alpha)\beta') \sim_L (P_i(\alpha), S_i(\alpha)\beta'').$$

Вече проверихме твърдението при $i = 0$. Да допуснем, че за някое i твърдението е вярно. Тогава тъй като $\beta' \sim_{GR} \beta''$ получаваме, че:

$$S_i(\alpha)\beta' \sim_R S_i(\alpha)\beta''.$$

Това заедно с:

$$(P_i(\alpha), S_i(\alpha)\beta') \sim_L (P_i(\alpha), S_i(\alpha)\beta'').$$

и дефиницията на $\sim_L$ (Ω-крайноопределима от $\sim_R$) ни дава, че:

$$(P_{i+1}(\alpha), S_{i+1}\beta') \sim_L (P_{i+1}(\alpha), S_{i+1}(\alpha)\beta'').$$

Това завършва индукционната стъпка и следователно е в сила, че:

$$(P_i(\alpha), S_i(\alpha)\beta') \sim_L (P_i(\alpha), S_i(\alpha)\beta'')$$

за всяко $i \leq |\alpha|_\Omega$. В частонст при $i = |\alpha|_\Omega$ получаваме, че:

$$(\alpha, \beta') \sim_L (\alpha, \beta'').$$

Това, че $\alpha \sim_R \alpha$ е очевидно и следователно $\alpha \sim_{GL} \alpha$ а всяко α.

С това показахме, че $\sim_{GL}$ е релация на еквивалентност. Това, че $\sim_{GL}$ е крайна следва по същия начин, както и в доказателството на факта 1. Всъщност класовете на еквивалентност на $\sim_{GL}$ са изоморфни на функции с дефиниционна област крайно множество и множество от стойности класовете на еквивалентност на $\sim_L$, което също е крайно.

Сега преминаваме към доказателството на втората част на твърдението:

2. Тъй като $S_1(\beta') \sim_{GR} S_1(beta'')$, то в частност $S_1(\beta')$ е дефинирано точно когато $S_1(\beta'')$ е дефинирано. Тоест $\alpha' P_1(\beta) \in Pref(\Omega) \Leftrightarrow \alpha'' P_1(\beta'') \in Pref(\Omega)$. Ако $\alpha' P_1(\beta') \notin Pref(\Omega)$, то и $\alpha'' P_1(\beta'') \notin Pref(\Omega)$. Затова да разгледаме случая, когато $\alpha' P_1(\beta') \in pref(\Omega)$. Тогава както видяхме по горе имаме, че:

$$\alpha'' P_1(\beta'') \in Pref(\Omega).$$

Сега имаме, че:

$$\alpha' \sim_R \alpha''$$

и $P_1(\beta') \sim_{GR} P_1(\beta'')$. Тогава от дефиницията на $\sim_{GR}$ получаваме, че:

$$\alpha' P_1(\beta') \sim_R \alpha'' P_1(\beta'').$$

Сега остава да проверим, че ако $\alpha' P_1(\beta') \in Pref(\Omega)$ и $\alpha'' P_1(\beta'') \in Pref(\Omega)$, а $\gamma' \sim_{GR} \gamma''$, то е в сила, че:

$$(\alpha' P_1(\beta'), \gamma') \sim_L (\alpha'' P_1(\beta''), \gamma'').$$

Да забележим, че $P_1(\beta')\gamma' \sim_{GR} P_1(\beta'')\gamma''$. Наистина ако $x' \sim_R x''$, то от дефиницията на $\sim_{GR}$ получаваме, че:

$$x' P_1(\beta') \sim_R x'' P_1(\beta'').$$

Сега от $\gamma' \sim_{GR} \gamma''$ получаваме, че:

$$x' P_1(\beta')\gamma' \sim_R x'' P_1(\beta'')\gamma''.$$

Това означава, че $P_1(\beta')\gamma' \sim_{GR} P_1(\beta'')\gamma''$ и тъй като $\alpha' \sim_{GL} \alpha''$ получаваме, че:

$$(\alpha', P_1(\beta')\gamma') \sim_L (\alpha'', P_1(\beta'')\gamma'').$$

Сега обаче $P_1(\beta')\gamma' \sim_R P_1(\beta'')\gamma''$ и от условието, че $\sim_L$ се определя от $\sim_R$ заключаваме, че:

$$(\alpha' P_1(\beta'), \gamma') \sim_L (\alpha'' P_1(\beta''), \gamma'),$$

защото $P_1(\beta') \in \Sigma^*\Omega$ и $P_1(\beta'') \in \Sigma^*\Omega$. С това, тъй като $\gamma' \sim_{GR} \gamma''$ бяха произволни, следва, че:

$$\alpha' P_1(\beta') \sim_{GL} \alpha'' P_1(\beta'').$$

Лема 7 *Ако релацията $\sim_{\widetilde{R}}$ е дефинирана като:*

$$\beta' \sim_{\widetilde{R}} \beta'' \Leftrightarrow P_1(\beta') \sim_{GR} P_1(\beta'')\&S_1(\beta') \sim_{GR} S_1(\beta''),$$

то $\sim_{\widetilde{R}}$ е Ω-крайна с параметър $k+1$. Освен това за всеки $\alpha' \sim_{GL} \alpha''$ и всеки $\beta' \sim_{\widetilde{R}} \beta''$ е в сила:

$$\alpha' P_1(\beta') \sim_L \alpha'' P_1(\beta'') \text{ и}$$
$$T_{[\alpha'][\beta']} = T_{[\alpha''][\beta'']}.$$

Доказателство: Първата част следва непосредсвено от лема 5. Втората следва непосредствено от лема 6 и дефиницията на $T_{[\alpha][\beta]}$, която не зависи от избора на представители $(\alpha', \beta') \sim_L (\alpha'', \beta'')$ и $\beta' \sim_R \beta''$.

Сега пристъпваме към доказателството на теорема 7

С $[\beta]_{\widetilde{R}}$ ще означаваме класа на еквивалентност на β, определен от релацията $\sim_{\widetilde{R}}$, а с $[\alpha]_{GL}$ - класа на еквивалентност на α, определен от релацията $\sim_{GL}$. Нека още $k' = k + 1$.Накрая да фиксираме и компонентите на подпоследователните преобразуватели:

$$T_{[\alpha][\beta]} =< (\Sigma \cup \Omega) \times \Xi^*, Q_{[\alpha][\beta]}, s_{[\alpha][\beta]}, \delta_{[\alpha][\beta]}, \lambda_{[\alpha][\beta]}, \phi_{[\alpha][\beta]} >$$

Дефинираме $T =< (\Sigma \cup \Omega) \times \Xi^*, \Sigma \cup \Omega, P, Q, s, \Delta, d, \lambda, out, \phi, \psi >$ по следния начин:

1. $P = \{< [\alpha]_{GL}; [\beta_1]_{\widetilde{R}}, [\beta_2]_{\widetilde{R}} \dots [\beta_i]_{\widetilde{R}}, [], [], \dots [] > | 1 \leq i \leq k'\alpha, \beta, \beta_j \in (\Sigma \cup \Omega)^*\}$ е множество от наредени $k'+1$-орки, на които първият елемент е клас на еквивалентност на $\sim_{GL}$, следват $1 \leq i \leq k'$ класа на еквивалентност на $\sim_{\widetilde{R}}$, а останалите елементи са "празни"класове []. Първият елемент ще съответства на онази част от думата, която вече сме обработили, а $[\beta_j]_{\widetilde{R}}$ съответстват на класовете на еквивалентност на последните j Ω-наставки на прочетената до момента дума.

2. $Q = \{< q; [\alpha, \beta]_{GL}, [\beta]_{\widetilde{R}}; [\beta_1]_{\widetilde{R}}, [\beta_2]_{\widetilde{R}}, \dots [\beta_i]_{\widetilde{R}}, [], [], \dots, [] | 0 \leq i \leq k', \alpha, \beta, \beta_j \in (\Sigma \cup \Omega)\}^*, q \in Q_{[\alpha][\beta]}$.

3. $s =< [\varepsilon]_{GL}; [\varepsilon]_{\widetilde{R}}, [], [], \dots [] >$.

4. Функцията Δ дефинираме така:

$$\Delta(<[(\alpha]_{GL};[\beta_1]_{\widetilde{R}},\dots[\beta_i]_{\widetilde{R}},[]\dots[]>,\sigma) = \begin{cases} <[\alpha]_L; \\ [\beta_1\circ\sigma]_{\widetilde{R}},\dots[\beta_i\circ\sigma]_{\widetilde{R}},[],\dots,[]> \\ \text{ ако } \sigma\in\Sigma. \\ \\ <[\alpha]_{GL}; \\ [\beta_1\circ\sigma]_{\widetilde{R}},\dots[\beta_i\circ\sigma]_{\widetilde{R}},[\varepsilon]_{\widetilde{R}},[],\dots,[]> \\ \text{ ако } \sigma\in\Omega, i<k'. \\ \\ <s_{[\alpha][\beta_1\sigma]};[\alpha]_{GL},[\beta_1\circ\sigma]; \\ [\beta_2\circ\sigma]_{\widetilde{R}},\dots[\beta'_k\circ\sigma]_{\widetilde{R}},[\varepsilon]_{\widetilde{R}}> \\ \sigma\in\Omega \text{ и } i=k. \end{cases}$$

5. Функцията d дефинираме независимо от $j\in\{0,1\}$:

$$d(<q;[\alpha]_{GL},[\beta]_{\widetilde{R}};[\beta_1]_{\widetilde{R}},\dots[\beta_i]_{\widetilde{R}},[]\dots[]>,\sigma,j) = \begin{cases} <\delta_{[\alpha][\beta]}(q,\sigma);[\alpha]_{GL},[\beta]_{\widetilde{R}}; \\ [\beta_1]_{\widetilde{R}},\dots[\beta_i]_{\widetilde{R}},[]\dots[]>, \\ \text{ ако } \sigma\in\Sigma \text{ и } i\geq 1 \\ \\ <[\alpha P_1(\beta)]_{GL};[\beta_1]_{\widetilde{R}},\dots[\beta_k]_{\widetilde{R}}> \\ \text{ ако } i=k',\sigma\in\Omega \text{ и } !\phi(\delta_{[\alpha][\beta]}(q,\sigma)) \\ \\ <s_{[(\alpha P_1(\beta)][\beta_1]};[\alpha P_1(\beta)]_{GL}, \\ [\beta_2]_{\widetilde{R}};[\beta_2]_{\widetilde{R}},\dots[\beta_i]_{\widetilde{R}},[],\dots[]> \\ 1\geq i<k',\sigma\in\Omega \text{ и } !\phi(\delta_{[\alpha][\beta]}(q,\sigma)) \end{cases}$$

6. $\lambda(p,\sigma)=\sigma$, за всяко $p\in P$, $\sigma\in\Sigma\cup\Omega$.

7. Функцията out не зависи от $[\beta_i]_R$ за $i\geq 1$, тоест:

$$out(<q;[(\alpha]_{GL},[\beta]_{\widetilde{R}};.>,\sigma,j) = \begin{cases} \lambda_{[\alpha][\beta]}(q,\sigma), \text{ ако } \sigma\in\Sigma \text{ и } j=1 \\ out_{[\alpha][\beta]}(q,\sigma)\phi_{[\alpha][\beta]}(\delta_{[\alpha][\beta]},\sigma)) \text{ иначе} \end{cases}$$

8. $\phi([\alpha]_{GL};[\beta_1]_{\widetilde{R}}\ldots[\beta_i]_{\widetilde{R}},[]\ldots[]) = (s_{[\alpha][\beta_1]};[\alpha]_{GL},[\beta_1]_{\widetilde{R}};[\beta_2]_{\widetilde{R}},\ldots[\beta_i]_{\widetilde{R}},[],\ldots[]>$.

9. $$\psi(q) = \begin{cases} \varepsilon \text{ ако } q =< s_{[\alpha][\beta];[\alpha]_{GL},[];[],\ldots[]} > \\ \neg! \text{ иначе.} \end{cases}$$

Лема 8 *Нека $x,y,z \in (\Sigma \cup \Omega)^*$ а думи, а $q' \in Q$ и $q'' \in Q$ са две състояния. Нека q' има вида:*

$$q' = (< q;[(\alpha]_{GL},[\beta]_{\widetilde{R}};[\beta_1]_{\widetilde{R}},[\beta_2]_{\widetilde{R}}\cdots[\beta_i]_{\widetilde{R}},[],\ldots[]>$$

и нека:

$$(q',x,yz,\omega) \vDash (q'',x,z,\omega').$$

Тогава

- *ако $i = k'$, то*

$$q'' = (< \delta_{[\alpha][\beta]}(q,y);[\alpha]_{GL},[\beta]_{\widetilde{R}};[\beta_1]_{\widetilde{R}},[\beta_2]_{\widetilde{R}}\cdots[\beta_i]_{\widetilde{R}},[],\ldots[]>$$

  и

$$\omega' = \omega\lambda_{[\alpha][\beta]}(q',z)$$

- *ако $i < k'$ и $y = P_1(yz)$, то*

$$q'' = (s_{[\alpha P_1(\beta)][S_1(\beta)]};[\alpha P_1(\beta)]_{GL},[\beta_1]_{\widetilde{R}};[\beta_2]_{\widetilde{R}}\ldots[\beta_i]_{\widetilde{R}},[],\ldots,[])$$

  и

$$\omega' = \omega T_{[\alpha][\beta]}(q)(y).$$

Доказателство: Доказателството на 1 следва директно от дефиницията на d функцията и факта, че всички преходи в този извод са от вида 2 или 3 (d-преходи). Тогава фунцията d съвпада с $\delta_{[\alpha][\beta]}$, а out с $\lambda_{[\alpha][\beta]}$, прилагани съответно по първата компонента на състоянието. Останалите компоненти не се менят.

Втората част също следва лесно от дефиницията на d и out. Всъщност броят на незначещите [] расте само при 0-преход или преход със символ от Ω. При преходи със символ $\sigma \in \Sigma$ че d променя само първата компонента на състоянието като прилага към нея $\delta_{[\alpha][\beta]}$. В този случай out съвпада със функцията $\lambda_{[\alpha][\beta]}$, прилагана към първата компонента. При преход със символ от Ω d е дефинирана само ако $\delta_{[\alpha][\beta]}(q, \omega)$ води в заключително състояние и тогава приема стойност, независеща от q:

$$< s_{[\alpha P_1(\beta)][S_1(\beta)]}, [\alpha P_1(\beta)]_{GL}[\beta_1]_{\widetilde{R}}; [\beta_2]_{\widetilde{R}} \dots [\beta_i]_{\widetilde{R}}, [], \dots, [] > .$$

Допълнително $\lambda_{[\alpha][\beta]}$ приложена към предходното на q'' състояние добавя коректно и ϕ-функцията, което доказва твърдението.

Лема 9 *Нека $x, y, z \in (\Sigma \cup \Omega)^*$ и нека $p \in P$ е такова, че:*

$$< s, xy, \varepsilon, \varepsilon > \models < p, y, z, \omega > .$$

Нека $|x|_\Omega = n$. Тогава е в сила, че:

- *$z = S_{n-k'}(x)$ ако $n \geq k'$ и $z = x$ иначе.*
- *$p =< [\alpha]_{GL}; [\beta_1]_{\widetilde{R}}, \dots [\beta_i]_{\widetilde{R}}, [] \dots [] >$, където $|z|_\Omega + 1 = i$ и $\beta_j \sim_{\widetilde{R}} S_{j-1}(z)$ за $j \leq i$, а $\alpha \sim_{GL} xz^{-1}$.*
- *$\omega = f(\varepsilon, xy) f(xz^{-1}, zy)^{-1}$.*

Индукция по дължината на x. При $|x| = 0$, $p = s$, $z = \varepsilon$ и $\omega = \varepsilon$. Тъй като $|x|_\Omega < k$, всички твърдения следват от дефиницията на $\sim_{GL}$.

Нека твърдението е вярно за всички x с дължина $\leq n$. Нека

$$< s, x\sigma y, \varepsilon, \varepsilon > \models < p, \sigma y, z, \omega >,$$

където $p \in P$. Ако $\sigma \in Sigma$, то $|x\sigma|_\Omega = |x|_\Omega$ и лесно се вижда, че

$$< p, \sigma y, z, \omega > \vdash < p', y, z\sigma, \omega >,$$

като ако

$$\begin{aligned} p &=< [\alpha]_{LG}; [\beta_1]_{\widetilde{R}}, \dots [\beta_i]_{\widetilde{R}}, [] \dots [] > \text{ то} \\ p' &=< [\alpha]_{GL}; [\beta_1 \sigma]_{\widetilde{R}}, \dots [\beta_i \sigma]_{\widetilde{R}}, [] \dots [] > \end{aligned}$$

Тъй като $\sim_{\widetilde{R}}$ е дясно-инвариантна, то твърденията от условие 2 за p' следват от съответните условия за p. Свойствата 1 и 3 също се запазват.

Нека $\sigma \in \Omega$. Нека

$$p =< [\alpha]_{GL}; [\beta_1]_{\widetilde{R}}, \dots [\beta_i]_{\widetilde{R}}, [] \dots [] > .$$

В зависимост от i имаме два случая:

1. $i < k'$. Тогава $z = x$. Тогава

   $$< p, \sigma y, z, \omega > \vdash < p', y, z\sigma, \omega >,$$

   и $z\sigma = x\sigma$. Сега

   $$p' =< [\alpha]_{GL}; [\beta_1\sigma]_{\widetilde{R}}, \dots [\beta_i\sigma]_{\widetilde{R}}, [\varepsilon]_{\widetilde{R}}, [] \dots [] >,$$

   което отразява появата на последната $i+1$-а Ω-наставка в $x\sigma$. Тя е празна, така че и твърдението за нея следва непосредствено. Останалите условия 1-3 отново се запазват, поради дясната-инвариантност на $\sim_R$.

2. Нека $i = k'$. Тогава изводът може да се представи като:

   $$< p, \sigma y, z, \omega > \vdash < q, y, z\sigma, \omega > \models < q', y, \sigma' z', \omega' > \vdash < p', y, z', \omega'' >$$

   Нека

   $$p =< [\alpha]_{GL}; [\beta_1]_{\widetilde{R}}, \dots [\beta_{k'}]_{\widetilde{R}} > .$$

   Тогава от дефиницията на Δ следва, че

   $$q =< s_{[\alpha][\beta\sigma]}; [\alpha]_{GL}, [\beta_1\sigma]_{\widetilde{R}}; [\beta_2\sigma]_{\widetilde{R}}, \dots [\beta_{k'}\sigma]_{\widetilde{R}}, [\varepsilon]_{\widetilde{R}} > .$$

   От лема 8 тогава получаваме, че

   $$q' =< \delta^*_{[\alpha][\beta\sigma]}(s_{[\alpha][\beta\sigma]}, z(\sigma' z')^{-1}); [\alpha]_{GL}, [\beta_1\sigma]_{\widetilde{R}}; [\beta_2\sigma]_{\widetilde{R}}, \dots [\beta_{k'}\sigma]_{\widetilde{R}}, [\varepsilon]_{\widetilde{R}} > \text{ и}$$
   $$\omega' = \omega \circ \lambda^*_{[\alpha][\beta\sigma]}(s_{[\alpha][\beta\sigma]}, z(\sigma' z')^{-1})$$
   $$p' =< [\alpha P_1(\beta\sigma)]_{GL}; [\beta_2\sigma]_{\widetilde{R}} \dots [\beta_k\sigma]_{\widetilde{R}}, [\varepsilon]_{\widetilde{R}} > .$$

Но тъй като q не съдържа [] компоненти, то $\sigma' \in \Omega$ и $z(\sigma' z')^{-1} \in \Sigma^*$. Това означава, че $|z'|_\Omega = k'$ и $S_j(z') = S_{j+1}(z\sigma)$ за $j \leq k'-1$ и $S_{k'}(z') = \varepsilon$. Тъй като $\beta_{j+1}\sigma \sim_{\widetilde{R}} S_{j+1}(z\sigma) = S_j(z')$, а $\beta_{k'} = \varepsilon$, то второ условие се запазва в частта си за $\sim_R$. Това, че

$$(\alpha P_1(\beta_1), S_1(\beta_1)) \sim_L (x(z')^{-1}, z'),$$

следва от факта, че $z' = S_1(z\sigma)$, $z\sigma \sim_R \beta_1\sigma$ и от това, че $\sim_R$ определя $\sim_L$ следва, че $(\alpha P_1(\beta_1), S_1(\beta_1)) \sim_L (x(z')^{-1}, z')$.

От лема 8 заедно с дефиницията на функцията *out* следва, че $\omega'' = \omega \circ \lambda^*(s_{[\alpha][\beta_1\sigma]}, P_1(z\sigma)) = \omega T_{[\alpha][\beta_1\sigma]}(P_1(z\sigma))$. Но сега $xz^{-1} \sim_{GL} xz^{-1}$ и тъй като $|z\sigma|_\Omega = k'$, то:

$$\beta_1\sigma \sim_{\widetilde{R}} z\sigma \sim_{\widetilde{R}} z\sigma y.$$

Оттук и от индукционното предположение заедно с лема 7 получаваме, че:

$$T_{[xz^{-1}][z\sigma]} = T_{[xz^{-1}][z\sigma y]}$$

и освен това:

$$(xz^{-1})P_1(z\sigma) \sim_{GL} \alpha P_1(\beta_1\sigma).$$

Сега 1, 2 и 3 следват непосредствено.

С това показахме, верността на твърденията 1-3 и за произволно x с дължина $n+1$, от където от принципа за математеческа индукция твърдението следва и за произволно x.

Сега като използваме леми 8 и лема 9 лесно получаваме следната:

Теорема 8 $f_T = f$.

Наистина лесно се вижда, че $f(x)$ е дефинирана тогава само тогава, когато съществуват изводи:

$$< s, x, \varepsilon, \varepsilon > \models < p, \varepsilon, z, \omega >, \text{ с } p \in P$$
$$< \phi(p), \varepsilon, \varepsilon, \omega' >$$

и тогава $T_f(x) = \omega'$. От лема 9 имаме, че $\omega = f(\varepsilon, x) f(xz^-1, z)^{-1}$ и още:

$$p =< [xz^{-1}]_{GL}; [z]_{\widetilde{R}}, S_1(z)_{\widetilde{R}} \ldots [S_i(z)]_{\widetilde{R}}, [], \ldots [] > .$$

Тогава по дефиницията на ϕ следва, че:

$$q_0 = \phi(p) =< s_{[xz^{-1}][z]}; [xz^{-1}]_{GL}, [z]_{\widetilde{R}}; [S_1(z)]_{\widetilde{R}}, \ldots [S_i(z)]_{\widetilde{R}}, [] \ldots [] > .$$

Тогава от втората част на лема 8 получаваме, че ако:

$$< q_0, \varepsilon, z, \omega > \models < q_1, \varepsilon, S_1(z), \omega_1 >,$$

то всъщност:

$$q_1 =< s_{[x(S_1(z))^{-1}][S_1(z)]}; [x(S_1(z))^{-1}]_{GL}, [S_1(z)]_{\widetilde{R}}; [S_2(z)]_{\widetilde{R}}, \ldots [S_i(z)]_{\widetilde{R}}, [], \ldots [] >$$
$$\omega_1 = \omega\lambda^*_{[xz^{-1}][z]}(P_1(z)) = \omega T_{[xz^{-1}][z]}(P_1(z))$$

Оттук с индукция по дължината j следва, че ако:

$$< q_0, \varepsilon, z, \omega > \models < q_j, \varepsilon, S_j(z), \omega_j >,$$

то

$$q_j =< s_{[x(S_j(z))^{-1}][S_j(z)]}; [x(S_j(z))^{-1}]_{GL}, [S_j(z)]_{\widetilde{R}}; [S_{j+1}(z)]_{\widetilde{R}}, \ldots [S_i(z)]_{\widetilde{R}}, [], \ldots [] >$$
$$\omega_j = \omega_{j-1}\lambda^*_{[xz^{-1}][z]}(P_1(z)) = \omega \prod_{k=0}^{j} T_{[xP_k(z)][S_k(z)]}(I_{k+1}(z)).$$

Следователно $\omega' = \omega f(xz^{-1}, z) = f(\varepsilon, x)$. С това доказахме и верността на теорема 7.

4.3 Рационалност на Ω-представимите функции

В тази част ще покажем, че всяка Ω-представима функция е рационална. Това ще направим като по дадена Ω-представима функция построим бимашина, разпознаваща тази функция. Тъй като класът от езиците, разпознавани от бимашини и този на рационалните функции съвпадат [Eil74], то това ще бъде достатъчно.

Теорема 9 *За всяка Ω-крайноопределима функция $f : (\Sigma \cup \Omega)^* \to \Xi^*$ съществува бимашина:*

$$\mathcal{B} = (LA, RA, \Delta),$$

която определя рационална функция $f_{\mathcal{B}}$ със свойството:

$$f = f_{\mathcal{B}}.$$

Доказателство: Нека $g : (Pref(\Omega) \times (\Sigma \cup \Omega)^*) \cup ((\Sigma \cup \Omega)^* \times \{\varepsilon\}) \to \Xi^*$ е Ω-определима със свойството:

$$f(\alpha) = g(\varepsilon, \alpha)$$

за всяка дума $\alpha \in (\Sigma \cup \Omega)^*$. Нека g се определя от Ω-крайна релация $\sim_R$ с параметър k и Ω-определимата посредством $\sim_R$ релация $\sim_L$. С $[\beta]_R$ ще означаваме класа на еквивалентност на β, определен от релацията $\sim_R$, а с $[(\alpha, \beta)]_L$ - класа на еквивалентност на (α, β), определен от (α, β). Нека още k константата, с която $\sim_R$ определя $\sim_L$. Накрая с

$$T_{[\alpha][\beta]} =< (\Sigma \cup \Omega) \times \Xi^*, Q_{[\alpha][\beta]}, s_{[\alpha][\beta]}, \delta_{[\alpha][\beta]}, \lambda_{[\alpha][\beta]}, \phi_{[\alpha][\beta]} >$$

означаваме подпоследователните трансдюсери, които определят функцията g.

Първо ще определим десния автомат на бимашината:

$$\mathcal{B} = (LA, RA, \Delta).$$

Това правим като дефинираме, дясно-инвариантната релация на еквивалентност:

$$\beta' \sim_{RA} \beta'' \Leftrightarrow \forall \omega' \sim_R \omega'' \Rightarrow \omega'(\beta')^R \sim_R \omega''(\beta'')^R. \tag{11}$$

Лема 10 *$\sim_{RA}$, дефинирана чрез равенство 11 е крайна, дясно-инвариантна релация на еквивалентност.*

Наистина, лесно се вижда, че $\sim_{RA}$ е симетрична и транзитивна. Това, че $\sim_{RA}$ е рефлексивна, следва от факта, че $\sim_R$ е дясно-инвариантна. Тогава за всеки $\omega' \sim_R \omega''$ имаме, че за всяко β $\omega'(\beta)^R \sim_R \omega''(\beta)^R$.

Това, че $\sim_{RA}$ е крайна може да се види по следния начин. На всеки клас на еквивалентност на $\sim_{RA}$, $[\beta]_{RA}$ съпоставяме функция, която е дефинирана и приема стойности от множеството на класовете на еквивалентност на $\sim_R$, по следния начин:

$$F([\beta]_{RA})([\omega']_R) = [\omega'(\beta)^R]_R.$$

От дефиницията на $\sim_{RA}$ следва, че $F([\beta]_{RA})$ е коректно дефинирана функция. При това на различните класове на еквивалентност на $\sim_{RA}$ съответстват различни функции. Това показва, че броят на всички класове на еквивалентност на $\sim_{RA}$ ненадминава броя на функциите $g : \{[\omega]_R\} \to \{[\omega]_R\}$. Сега ако l е броят на класовете на еквивалентност на $\sim_R$, то броят на всички такива функции е l^l. Това показва, че и $\sim_{RA}$ е крайна.

Сега ще дефинираме дясно-инвариантната релация, която определя левия автомат LA на бимашината $\mathcal{B}$. Дефинираме релацията $\sim_{LA}$ по следния начин.

Дефиниция 22 $\alpha' \sim_{LA} \alpha''$ *точно, когато са изпълнени следните три условия:*

1. *За всеки β', β'', за които $(\beta')^R \sim_{RA} (\beta'')^R$ е в сила, че:*

$$(base(\alpha'), \beta') \sim_L (base(\alpha''), \beta'').$$

2. $(last(\alpha'))^R \sim_{RA} (last(\alpha''))^R$.

3. *За всеки трансдюсер T_u от дефиницията на f с начално състояние s_u и функция на преходите δ_u е всила равенството:*

$$\delta_u^*(s_u, last(\alpha')) = \delta_u^*(s_u, last(\alpha'')).$$

Забележка 6 *Условие 2 всъщност е еквивалентно на следното:*

$$\forall \omega', \omega''(\omega' \sim_R \omega'' \Rightarrow \omega' last(\alpha') \sim_R \omega'' last(\alpha'').$$

Лема 11 *Релацията $\sim_{LA}$ е крайна, дясно-инвариантна релация на еквивалентност.*

Да отбележим, че $last(\alpha\gamma) = last(\alpha)\gamma$ ако $|\gamma|_\Omega = 0$ и $last(\alpha\gamma) = last(\gamma)$ в противен случай.

Лесно се вижда, че всяко от условията 1–3 е симетрично и транзитивно. Това следва от съответните свойства на $\sim_L$, $\sim_{RA}$ и $=$. Сега ще покажем, че $\sim_{LA}$ е рефлексивна.

Това, че $(last(\alpha))^R \sim_{RA} (last(\alpha))^R$ е изпълнено за всяко α, защото $\sim_{RA}$ е релация на еквивалентност. По същия начин $\delta_u^*(s_u, last(\alpha)) = \delta_u^*(s_u, last(\alpha))$.

Сега ще покажем, че за всяко $\beta_1 \sim_{RA} \beta_2$ е в сила, че

$$(base(\alpha), \beta_1) \sim_L (base(\alpha), \beta_2).$$

Първо от дефиницията на $\sim_L$ е в сила, че:

$$(\varepsilon, base(\alpha)\beta_1) \sim_L (\varepsilon, base(\alpha)\beta_2)$$

Да допуснем, че за някое $l < |base(\alpha)|_\Omega$ е изпълнено, че:

$$(P_l(base(\alpha)), S_l(base(\alpha))\beta_1) \sim_L (P_l(base(\alpha)), S_l(base(\alpha))\beta_2)$$

Тогава от дефиницията на $\sim_{RA}$ следва:

$$S_l(base(\alpha))\beta_1 \sim_R S_l(base(\alpha))\beta_2.$$

Сега от дефиницията на $\sim_L$ получаваме, че:

$$(P_{l+1}(base(\alpha)), S_{l+1}(base(\alpha))\beta') \sim_L (P_{l+1}(base(\alpha)), S_{l+1}(base(\alpha))\beta''),$$

защото всъщност:

$$P_l(\alpha)P_1(S_l(\alpha)) = P_{l+1}(\alpha)$$

и

$$S_1(S_l(\alpha)) = S_{l+1}(\alpha)$$

за всяко α. В случая прилагаме това наблюдение за $base(\alpha)$, а останалото следва от дефиницията на $\sim_L$.

Сега ще покажем, че $\sim_{LA}$ е дясно-инвариантна. Нека $\alpha' \sim_{LA} \alpha''$ и нека γ е произволна дума. Да забележим, че $last(\alpha\gamma) = last(\alpha)\gamma$ ако $|\gamma|_\Omega = 0$ и $last(\alpha\gamma) = last(\gamma)$ в противен случай за всяка дума α. Освен това условията (2) и (3) са дясно-инвариантни. Сега тъй като (α', α'') удовлетворяват условие (3) лесно се вижда, че и $(\alpha'\gamma, \alpha''\gamma)$ го удовлетворяват. Условие (2) също се проверява лесно. Ако $|\gamma|_\Omega \geq 1$, то

$$last(\alpha'\gamma) = last(\gamma) = last(\alpha''\gamma)$$

и няма какво да доказваме. В противен случай:

$$last(\alpha'\gamma) = last(\alpha')\gamma \text{ и}$$
$$last(\alpha''\gamma) = last(\alpha'')\gamma.$$

Според забележката, която направихме след дефиницията на $\sim_{LA}$ достатъчно е да проверим, че за всеки $\omega' \sim_R \omega''$ е изпълнено, че $\omega' last(\alpha')\gamma \sim_R \omega'' last(\alpha'')\gamma$. От това, че $\alpha' \sim_{LA} \alpha''$ имаме, че $\omega' last(\alpha') \sim_R \omega'' last(\alpha'')$ в случай, че $\omega' \sim_R \omega''$. Сега $\sim_R$ е дясно-инвариантна, откъдето твърдението следва непосредствено.

Остава да проверим, че $(\alpha'\gamma, \alpha''\gamma)$ удовлетворяват и условие (1). Ако $|\gamma|_\Omega = 0$, то $base(\alpha'\gamma) = base(\alpha')$ и съответно $base(\alpha'\gamma) = base(\alpha')$. Тогава твърдението следва непосредствено.

Нека $|\gamma|_\Omega \geq 1$ и нека $(\beta')^R \sim_{RA} (\beta'')^R$ са произволни. Имаме, че:

$$base(\alpha'\gamma) = \alpha' base(\gamma)$$
$$base(\alpha''\gamma) = \alpha'' base(\gamma)$$

Сега $(last(\alpha'))^R \sim_{RA} (last(\alpha''))^R$. Ще покажем, че $(last(\alpha')base(\gamma)\beta')^R \sim_{RA} (last(\alpha'')base(\gamma)\beta'')^R$. Нека $\omega' \sim_R \omega''$ са произволни. Тогава $\omega' last(\alpha') \sim_R \omega'' last(\alpha'')$ и тъй като $\sim_R$ дясно-инвариантна, то и $\omega' last(\alpha')base(\gamma) \sim_R \omega'' last(\alpha'')base(\gamma)$. Сега от това, че $(\beta')^R \sim_{RA} (\beta'')^R$ получаваме, че $\omega' last(\alpha')base(\gamma)\beta' \sim_R \omega'' last(\alpha'')base(\gamma)\beta''$. Това показва, че:

$$(last(\alpha')base(\gamma)\beta')^R \sim_{RA} (last(\alpha'')base(\gamma)\beta'')^R.$$

Тогава от дефиницията на $\sim_{LA}$, тъй като $\alpha' \sim_{LA} \alpha''$, то получаваме, че:

$$(base(\alpha'), last(\alpha')base(\gamma)\beta') \sim_L (base(\alpha''), last(\alpha'')base(\gamma)\beta'').$$

Оттук с индукция по $|\gamma|_\Omega$ и прилагане на дефиницията на $\sim_L$ получаваме, че и:

$$(\alpha' base(\gamma), \beta') \sim_L (\alpha'' base(\gamma), \beta'')$$

Следователно $\sim_{LA}$ е дясно-инвариантна релация на еквивалентност. Остана да покажем, че $\sim_{LA}$ е крайна.

За всеки клас на еквивалентност $[\alpha]_{LA}$ дефинираме функциите:

$$F_1([\alpha]_{LA}) : \{[\beta]_{RA}\} \to \{[(\alpha, \beta)]_L\}$$
$$F_1([\alpha]_{LA})([\beta]_{RA}) = [(\alpha, (\beta)^R)]_L$$

$$F_2([\alpha]_{LA}) : \{[\omega]_R\} \to \{[\omega]_R\}$$
$$F_2([\alpha]_{LA})([\omega]) = [\omega last(\alpha)]_R$$

$$F_3([\alpha]_{LA}) : \{T_u \,|\, u = [(\alpha, \beta)][\beta]\,\} \to \cup_{u=[(\alpha,\beta)][\beta]} Q_u$$
$$F_3([\alpha])(u) = \delta^*_u(s_u, last(\alpha)).$$

Нека $\alpha_1 \sim_{LA} \alpha_2$. Тогава от условие (1) следва, че за всеки $(\beta')^R \sim_{RA} (\beta'')^R$ е в сила, че:

$$(\alpha_1, \beta') \sim_L (\alpha_2, \beta''),$$

което показва, че:

$$F_1([\alpha_1]_{LA})([(\beta')^R]_{RA}) = F_1([\alpha_2]_{LA})([(\beta'')^R]_{RA}).$$

Това означава, че F_1 е коректно дефинирана.

Аналогично, от условие (2) и дефиницията на $\sim_{RA}$ получаваме, че ако $\omega' \sim_R \omega''$, то $\omega' last(\alpha') \sim_R \omega'' last(\alpha_2)$, което показва, че и F_2 коректно дефинирана.

Накрая от условие (3) получаваме, че за всяко u е в сила равенството:

$$\delta^*_u(s_u, last(\alpha)) = \delta^*_u(s_u, last(\alpha_2)),$$

което ни дава, че и F_3 коректно дефинирана.

Сега да допуснем, че $\alpha_1 \not\sim_{LA} \alpha_2$. Тогава не е изпълнено едно от условията на дефиницията 22. Ако (1) не е изпълнено, то съществуват $(\beta')^R \sim_{RA} (\beta'')^R$, за които:

$$(\alpha_1, \beta') \not\sim_L (\alpha_2, \beta'').$$

Но тогава получаваме, че:

$$F_1([\alpha_1]_{LA})([(\beta')^R]_{RA}) \neq F_1([\alpha_2]_{LA})([(\beta'')^R]_{RA}) = F_1([\alpha_2]_{LA})([(\beta')^R]_{RA}).$$

Това показва, че $F_1([\alpha_1]_{LA}) \neq F_1([\alpha_2]_{LA})$

Аналогично, ако α_1 и α_2 нарушават условие (2), то съществуват $\omega' \sim_R \omega''$, за които $\omega' last(\alpha_1) \not\sim_R \omega'' last(\alpha_2)$. Следователно:

$$F_2([\alpha_1]_{LA})([\omega']_R) \neq F_2([\alpha_2]_{LA})([\omega'']_R) = F_2([\alpha_2]_{LA})([\omega']_R).$$

В частност $F_2([\alpha_1]_{LA}) \neq F_2([\alpha_2]_{LA})$.

Накрая ако α_1 и α_2 нарушават условие (3) на дефиниция 22 получаваме, че съществува u, за което:

$$\delta_u^*(s_u, last(\alpha_1)) \neq \delta_u^*(s_u, last(\alpha_2)).$$

Оттук непосредствено следва, че $F_3([\alpha_1]_{LA})(u) \neq F_3([\alpha_2]_{LA})(u)$, от където и $F_3([\alpha_1]) \neq F_3([\alpha_2]_{LA})$.

Така получихме, че ако $[\alpha_1]_{LA} \neq [\alpha_2]_{LA}$, то и

$$(F_1([\alpha_1]_{LA}), F_2([\alpha_1]_{LA}), F_3([\alpha_1]_{LA})) \neq (F_1([\alpha_2]_{LA}), F_2([\alpha_2]_{LA}), F_3([\alpha_2]_{LA})).$$

Тъй като всяка от функциите $F_1([\alpha]_{LA})$, $F_2([\alpha]_{LA})$ и $F_3([\alpha]_{LA})$ има крайна дефиниционна област и крайно множество на значенията, то всевъзможните функции $F_1([\alpha]_{LA})$, $F_2([\alpha]_{LA})$ $F_3([\alpha]_{LA})$ са краен брой. Оттук и взевъзможните тройки

$$(F_1([\alpha]_{LA}), F_2([\alpha]_{LA}), F_3([\alpha]_{LA}))$$

са краен брой. Тъй като, както проверихме, на различните класове на еквивалентност на $\sim_{LA}$ им съответстват различни тройки от функции, това показва, че и класовете на еквивалентност на $\sim_{LA}$ са краен брой, тоест $\sim_{LA}$ е крайна.

Остана да дефинираме функцията

$$\Delta : []_{LA} \times (\Sigma \cup \Omega) \times []_{RA} \to \Xi^*.$$

Това правим по следния начин:

$$\Delta([\alpha]_{LA}, a, [\beta]_{RA}) = \lambda_u(q_u, a), \text{ ако } a \in \Sigma \text{ и } [\beta]_{RA} \neq [\varepsilon]_{RA}. \quad (12)$$

$$\Delta([\alpha]_{LA}, a, [\beta]_{RA}) = \lambda_u(q_u, a)\phi_u(\delta_u(q_u, a)), \text{ ако } a \in \Omega \text{ или } [\beta]_{RA} = [\varepsilon]_{RA}. \quad (13)$$

където $T_u = T_{[base(\alpha)][last(\alpha)(\beta)^R]}$, а $q_u = \delta_u^*(s_u, last(\alpha)$.

Първо ще проверим, че дефиницията е коректна. Наистина нека $\alpha \sim_{LA} \alpha_2$ и $\beta_1 \sim_{RA} \beta_2$. Тогава за всяко $\omega' \sim_R \omega''$ маме, че $\omega' last(\alpha_1) \sim_R \omega'' last(\alpha_2)$. Тогава от дефиницията на $\sim_{RA}$ е в сила, че

$$\omega' last(\alpha_1)(\beta_1)^R \sim_R \omega'' last(\alpha_2)(\beta_2)^R.$$

Тъй като $\omega' \sim_R \omega''$ бяха произволни, това показва, че:

$$(last(\alpha_1)\beta_1^R)^R \sim_R (last(\alpha_2)\beta_2^R)^R.$$

Тогава от условие (1) на дефиниция 22 получаваме, че

$$(base(\alpha_1), last(\alpha_1)\beta_1^R) \sim_L (base(\alpha_2), last(\alpha_2)\beta_2^R).$$

От друга страна при $\omega' = \omega'' = \varepsilon$ от дефиницията на $\sim_{RA}$ получаваме, че:

$$last(\alpha_1)\beta_1 \sim_R last(\alpha_2)\beta_2$$

Сега от дефиницията за определима функция, следва, че:

$$T_{[base(\alpha_1][last(\alpha_1)(beta_1)^R]} = T_{[base(\alpha_2)][last(\alpha_2)(\beta_2)^R]}.$$

Накрая, прилагаме условие (3) на дефиниция 22 за:

$$T_u = T_{[base(\alpha_1][last(\alpha_1)(beta_1)^R]} = T_{[base(\alpha_2)][last(\alpha_2)(\beta_2)^R]},$$

от където получаваме, че и q_u е еднозначно определено.

Сега ще покажем, че функцията, която определя бимашината $\mathcal{B} = (LA, RA, \Delta)$, $f_B = f$.

Всъщност в сила е следното твърдение:

Твърдение 7 *За всеки α β е в сила, че:*

$$g^*(\alpha, \beta) = \Delta^*(\alpha, (\beta)^R).$$

Доказателство: Доказателството следва директно с индукция по дължината на α.

Ако $\alpha = \varepsilon$ няма какво да доказваме.

Нека $\alpha = \alpha_1\sigma$, то лесно се вижда, че:

$$state(\alpha_1, \sigma\beta) = q_u,$$

където

$$\begin{aligned} T_u &= T_{[base(\alpha_1)][last(\alpha_1)\sigma\beta]} \\ q_u &= \delta_u^*(s_u, last(\alpha_1)). \end{aligned}$$

Тогава като се разгледат различните случаи заключаваме, че

$$\begin{gathered} g^*(\alpha_1\sigma, \beta) = g^*(\alpha_1, \sigma\beta)\lambda_u(q_u, \sigma) = \\ g(\alpha_1, \sigma\beta)\Delta([\alpha_1], \sigma, [\beta^R\sigma]) = \Delta^*(\alpha, \beta^R), \end{gathered}$$

където използвахме, че по индукционната хипотеза $g(\alpha_1, \sigma\beta) = \Delta^*(\alpha_1, (\sigma\beta)^R)$.

Сега прилагаме твърдението за $\beta = \varepsilon$ и получаваме, че

$$f_{\mathcal{B}}(\alpha) = \Delta^*(\varepsilon, \alpha^R) = g(\varepsilon, \alpha) = f(\alpha),$$

което доказава и теорема 9.

Както отбелязахме в началото на този параграф, като директно следствие получаваме, че:

Следствие 3 *Всяка Ω-крайноопределима функция е рационална.*

4.4 Композиция отляво на Ω-представима функция с подпоследователен преобразувател

В този параграф ще докажем едно полезно свойство на Ω-представимите функции, което ще използваме при конструкцията на FIFO-трансдюсери за разпознаване на композиция от правила. Именно в сила е следната теорема, според която композицията на Ω-представима функция и функция, представена чрез подпоследователен преобразувател отново е Ω-представима. Въпреки че на идейно ниво такава теорема вече видяхме в част 3, параграф 3.2, в случая имаме допълнителни ограничения върху изходния FIFO-трансдюсер, което налага да изменим конструкцията. Въпреки това промените са несъществени и не възникват сериозни технически проблеми.

И така нека формулираме нещата формално:

Теорема 10 *Нека g е Ω-представима функция, а f_T е функцията, зададена от подпоследователния преобразувател:*

$$T =< \Sigma' \times \Xi^*, S, s_0, \delta, \lambda, \phi > .$$

Тогава можем да конструираме Ω-представима функция $\widetilde{g}$ със свойството:

$$\widetilde{g}(u,v) = \begin{cases} \varepsilon \text{ ако } v = \varepsilon \\ \lambda^*(q, g(u,v)) \circ \phi(\delta^*(q, g(u,v))) \text{ иначе,} \end{cases}$$

където:

$$q = \delta^*(s_0, g(\varepsilon, uv)(g(u,v))^{-1}).$$

Доказателство: Нека $\sim_R$, $\sim_L$ и $T_{[\alpha][\beta]}$ са компонентите, които определят функцията g. С други думи $\sim_R$ е дясно-инвариатна, Ω-крайна релация на еквивалентност, която определя $\sim_L$ с константа k, а $T_{[\alpha][\beta]}$ са подпоследователни преобразуватели, които се определят от класовете на еквивалентност $([(\alpha,\beta)]_{\sim_L}, [\beta]_{\sim_R})$ и са такива че:

$$g(\alpha,\beta) = T_{[\alpha][\beta]}(P_1(\beta)) \circ g(\alpha P_1(\beta), S_1(\beta)).$$

За да симулираме композицията на функцията g с подпоследователния трансдюсер, ясно е, че трябва да разглеждаме по-фини релации, които да отчитат поведението на T, върху онази част от думата, от която не се интересуваме, но която дефинира $q \in S$.

Нека за всяко $q \in S$ дефинираме подпоследователните преобразуватели:

$$T(q) =< \Sigma \times \Xi^*, S, q, \delta, \lambda, id >, \text{ където } id(p) = \varepsilon \text{ за } p \in S$$
$$T^0(q) =< \Sigma \times \Xi^*, S, q, \delta, \lambda, \phi > .$$

Идеята на $T(q)$ е, че отразяват поведението на преобразувателя след като вече ме стигнали до състояние $q \in S$ върху част от текста, като очакват и позволяват произволно продължение различно от празната дума. $T^0(q)$ симулира изпълнените на T върху суфикси на думи, чиито префикси водят в q. Интуитивно е ясно, че ни интересуват композицията на такива преобразуватели с преобразувателите $T_{[\alpha][\beta]}$. Така че да дефинираме и:

$$T_{[\alpha][\beta]}(q) = T(q) \circ T_{[\alpha][\beta]}$$
$$T^0_{[\alpha][\beta]}(q) = T^0(q) \circ T_{[\alpha][\beta]}.$$

Ще използваме следните означения за така дефинираните преобразуватели:

$$T_{[\alpha][\beta]} =< (\Sigma \cup \Omega) \times \Sigma^*, Q_{[\alpha][\beta]}, s_{[\alpha][\beta]}, \delta_{[\alpha][\beta]}, \lambda_{[\alpha][\beta]}, \phi_{[\alpha][\beta]} >$$
$$T_{[\alpha][\beta]}(q) =< (\Sigma \cup \Omega) \times \Xi^*, Q_{[\alpha][\beta]}(q), s_{[\alpha][\beta]}(q), \delta_{[\alpha][\beta]}(q), \lambda_{[\alpha][\beta]}(q), \phi_{[\alpha][\beta]}(q) >$$
$$T^0_{[\alpha][\beta]}(q) =< (\Sigma \cup \Omega) \times \Xi^*, Q^0_{[\alpha][\beta]}(q), s^0_{[\alpha][\beta]}(q), \delta^0_{[\alpha][\beta]}(q), \lambda^0_{[\alpha][\beta]}(q), \phi^0_{[\alpha][\beta]}(q) > .$$

Сега можем да преминем към дефиницията на релациите $\sim_{\widetilde{R}}$ и $\sim_{\widetilde{L}}$. И така започваме с $\sim_{\widetilde{R}}$.

Дефиниция 23 *Казваме, че $\beta' \sim_{\widetilde{R}} \beta''$ тогава и само тогава, когато:*

1. $\beta' \sim_R \beta''$.
2. $S_1(\beta') = \varepsilon \Leftrightarrow S_1(\beta'') = \varepsilon$ *и* $|\beta'|_\Omega \geq 1 \Leftrightarrow |\beta''|_\Omega \geq 1$.
3. *за всеки* α, β *и* $p \in Q_{[\alpha][\beta]}$ *е изпълнено, че:*

$$\delta^*_{[\alpha][\beta]}(p, P_1(\beta')) = \delta^*_{[\alpha][\beta]}(p, P_1(\beta''))$$

4. *за всеки* $q \in S$, $\alpha, \beta \in (\Sigma \cup \Omega)^*$.

$$\forall p \in Q_{[\alpha][\beta]}(q) \Rightarrow \delta^*(q, \lambda^*_{[\alpha][\beta]}(p, P_1(\beta')) = \delta^*(q, \lambda^*_{[\alpha][\beta]}(p, P_1(\beta'')).$$

Първо ще проверим, че $\sim_{\widetilde{R}}$ дефинира дясно-инвариантна релация на еквивалентност от краен индекс. Лесно се вижда, че всяко от условията дефинира крайна релация на еквивалнтност. За първото условие, това следва от дефиницията на $\sim_R$, второто условие се задава от релацията на регулярния език $\Sigma^*\Omega(\Sigma\cup\Omega)^+$. Трето и четвъртото условие се дефинират чрез равенство на едни и същи функции, което е релация на еквивалентност, а освен това тези функции, като функции на преходи приемат само краен брой стойности, тоест и те задават крайна релация на еквивалентност.

Сега ще проверим, че $\sim_{\widetilde{R}}$ е дясно-инвариантна. Наистина, нека $\beta' \sim_{\widetilde{R}} \beta''$ и γ е произволна. Тъй като $\sim_R$ е дясно-инвариантна следва, че $\beta'\gamma \sim_R \beta''\gamma$. Същото е в сила и за релацията, породена от условие 2. Както отбелязахме, тя е породената от регулярния език $\Sigma^*\Omega(\Sigma\cup\Omega)^+$ и следователно щом

$$S_1(\beta') = \varepsilon \Leftrightarrow S_1(\beta'') = \varepsilon,$$
$$|\beta'|_\Omega \geq 1 \Leftrightarrow |\beta''|_\Omega \geq 1,$$

то и:

$$S_1(\beta'\gamma) = \varepsilon \Leftrightarrow S_1(\beta''\gamma) = \varepsilon,$$
$$|\beta'\gamma|_\Omega \geq 1 \Leftrightarrow |\beta''\gamma|_\Omega \geq 1.$$

Преминаваме към условията 3 и 4. От условие 2 имаме, че $\beta' \in \Sigma^*$, точно когато и $\beta'' \in \Sigma^*$. Това показва, че $P_1(\beta'\gamma) = \beta' P_1(\gamma)$, точно когато $P_1(\beta''\gamma) = \beta'' P_1(\gamma)$ и съответно ако $P_1(\beta'\gamma) = P_1(\beta')$, то и $P_1(\beta''\gamma) = P_1(\beta'')$. Във втория случай няма какво да доказваме, така че предполагаме, че $P_1(\beta'\gamma) = \beta' P_1(\gamma)$. Тогава, както отбелязахме е в сила и $P_1(\beta''\gamma) = \beta'' P_1(\gamma)$. Нека α, β са произволни и $p \in T_{[\alpha][\beta]}$, а $q \in S$. Тогава имаме, че:

$$\delta^*_{[\alpha][\beta]}(p, \beta' P_1(\gamma)) = \delta^*_{[\alpha][\beta]}(\delta^*_{[\alpha][\beta]}(p, \beta'), P_1(\gamma)) =$$
$$\delta^*_{[\alpha][\beta]}(\delta^*_{[\alpha][\beta]}(p, \beta''), P_1(\gamma)) = \delta^*_{[\alpha][\beta]}(p, \beta'' P_1(\gamma)).$$

Тук използвахме, че $\beta' = P_1(\beta')$ и $\beta'' = P_1(\beta'')$ и съответно за β' и β'' условие 3 е налице. Полученото равенство показва, че условие 3 е изпълнено и за двойката думи $\beta'\gamma$, $\beta''\gamma$.

Да разгледаме равенството, което определя равенство 4. Имаме, че:

$$\delta^*(p, \lambda^*_{[\alpha][\beta]}(q, \beta' P_1(\gamma))) =$$
$$\delta^*(\delta^*(p, \lambda^*_{[\alpha][\beta]}(q, \beta')), \lambda^*(\delta^*(q, \beta'), P_1(\gamma))) =$$
$$\delta^*(\delta^*(p, \lambda^*_{[\alpha][\beta]}(q, \beta')), \lambda^*(\delta^*(q, \beta''), P_1(\gamma)) =$$
$$\delta^*(\delta^*(p, \lambda^*_{[\alpha][\beta]}(q, \beta'')), \lambda^*(\delta^*(q, \beta''), P_1(\gamma)) =$$
$$\delta^*(p, \lambda^*_{[\alpha][\beta]}(q, \beta'' P_1(\gamma))).$$

Тук първо използвахме условие 3 за $\beta' = P_1(\beta')$ и $\beta'' = P_1(\beta'')$, а след това и условие 4 за същите думи. Като резултат получаваме, че условие 4 е в сила и за думите $\beta'\gamma$ и $\beta''\gamma$. С това проверихме, че $\sim_{\widetilde{R}}$ е крайна, дясно-инвариантна релация на еквивалентност.

Остана да проверим, че тя е и Ω-крайна. За целта нека $|\beta|_\Omega \geq \max(2, k)$ и γ е произволна. Тогава от дефиницията на $\sim_R$ следва, че $\beta \sim_R \beta\gamma$. Освен това, тъй като $|\beta|_\Omega \geq 2$, то $S_1(\beta\gamma) = S_1(\beta)\gamma$ и $S_1(\beta) \neq \varepsilon$, от където и $S_1(\beta\gamma) \neq \varepsilon$. Накрая имаме, че и $P_1(\beta\gamma) = P_1(\beta)$, което показва и верността на последните две равенства от дефиницията. Така можем да заключим, че за $|\beta| \geq \max(2, k)$ и всяка дума γ е в сила, че $\beta \sim_{\widetilde{R}} \beta\gamma$. Това означава, че $\sim_{\widetilde{R}}$ е Ω-крайна, дясно-инвариантна релация на еквивалентност.

Сега можем да дефинираме и $\sim_{\widetilde{L}}$. Това правим както следва:

Дефиниция 24 *Казваме, че* $(\alpha', \beta') \sim_{\widetilde{L}} (\alpha'', \beta'')$ *тогава и само тогава, когато:*

1. $(\alpha', \beta') \sim_L (\alpha'', \beta'')$ *и*

2. *е в сила равенството:*

$$\delta^*(s, g(\varepsilon, \alpha'\beta')(g(\alpha', \beta'))^{-1}) = \delta^*(s, g(\varepsilon, \alpha''\beta'')(g(\alpha'', \beta''))^{-1}).$$

Ясно е, че като конюнкция на две крайни релации на еквивалентност и $\sim_{\widetilde{L}}$ се явява крайна релация на еквивалентност. Ще покажем, че $\sim_{\widetilde{R}}$ Ω-определя релацията $\sim_{\widetilde{L}}$. Наистина нека $(\alpha', \beta') \sim_{\widetilde{L}} (\alpha'', \beta'')$ и $\beta' \sim_{\widetilde{R}} \beta''$. Тогава, тъй като $\beta' \sim_R \beta''$ и $\sim_R$ Ω-определя релацията $\sim_L$, то е в сила, че $(\alpha' P_1(\beta'), S_1(\beta')) \sim_L (\alpha'' P_1(\beta''), S_1(\beta''))$. Нека

$$\begin{aligned} q &= \delta^*(s, g(\varepsilon, \alpha'\beta')(g(\alpha', \beta'))^{-1}) \\ &= \delta^*(s, g(\varepsilon, \alpha''\beta'')(g(\alpha'', \beta''))^{-1}) \end{aligned}$$

Тогава от дефиницията на $g(\alpha, \beta) = T_{[\alpha][\beta]}(P_1(\beta)) \circ g(\alpha P_1(\beta), S_1(\beta))$ следва, че:

$$\delta^*(s, g(\varepsilon, \alpha'\beta')(g(\alpha' P_1(\beta'), S_1(\beta')))^{-1}) = \delta^*(q, T_{[\alpha'][\beta']}(P_1(\beta'))) \text{ и аналогично}$$
$$\delta^*(s, g(\varepsilon, \alpha''\beta'')(g(\alpha'' P_1(\beta''), S_1(\beta')))^{-1}) = \delta^*(q, T_{[\alpha''][\beta'']}(P_1(\beta''))).$$

Тъй като $(\alpha', \beta') \sim_L (\alpha'', \beta'')$ и съответно $\beta' \sim_R \beta''$, то $T_{[\alpha'][\beta']} = T_{[\alpha''][\beta'']}$. Сега остана да забележим, че:

$$T_{[\alpha'][\beta']}(P_1(\beta')) = \lambda^*_{[\alpha'][\beta']}(s_{[\alpha'][\beta']}, P_1(\beta')) \circ \phi_{[\alpha'][\beta']}(p')$$
$$T_{[\alpha'][\beta']}(P_1(\beta'')) = \lambda^*_{[\alpha'][\beta']}(s_{[\alpha'][\beta']}, P_1(\beta'')) \circ \phi_{[\alpha''][\beta'']}(p''),$$

където:

$$p' = \delta^*_{[\alpha'][\beta']}(s_{[\alpha'][\beta']}, P_1(\beta')) \text{ и}$$
$$p' = \delta^*_{[\alpha'][\beta']}(s_{[\alpha'][\beta']}, P_1(\beta''))$$

Сега от условие 3 за релацията $\sim_{\widetilde{R}}$ следва, че $p' = p''$, а от условие 4 следва, че:

$$q' = \delta^*(q, \lambda^*_{[\alpha'][\beta']}(P_1(\beta'))) = \delta^*(q, \lambda^*_{[\alpha'][\beta']}(P_1(\beta''))),$$

за някое $q' \in S$. Тогава непосредствено получаваме, че:

$$\delta^*(q, T_{[\alpha'][\beta']}(P_1(\beta'))) = \delta^*(q', \phi_{[\alpha'][\beta']}(p')) = \\ \delta^*(q, T_{[\alpha'][\beta']}(P_1(\beta'')).$$

Това показва, че второто условие от дефиницията на $\sim_{\widetilde{L}}$ също е налице. Следователно $\sim_{\widetilde{R}}$ определя $\sim_{\widetilde{L}}$.

Сега можем да дефинираме и трансдюсерите $\widetilde{T_{[\alpha][\beta]}}$. Това правим по следния начин:

$$\widetilde{T}_{[\alpha][\beta]} = \begin{cases} T_{[\alpha][\beta]}(q) \text{ ако } S_1(\beta) \neq \varepsilon \\ T^0_{[\alpha][\beta]}(q) \text{ ако } S_1(\beta) = \varepsilon, \end{cases}$$

където и в двата случая:

$$q = \delta^*(s, g(\varepsilon, \alpha\beta)(g(\alpha, \beta))^{-1}).$$

От дефиницията на $\sim_{\widetilde{R}}$ следва че ако $\beta' \sim_{\widetilde{R}} \beta''$, то $\beta' \sim_R \beta''$ и съответно $S_1(\beta') = \varepsilon \Rightarrow S_1(\beta'') = \varepsilon$. От друга страна ако $(\alpha', \beta') \sim_{\widetilde{L}} (\alpha'', \beta'')$, то $(\alpha', \beta') \sim_L (\alpha'', \beta'')$ и освен това:

$$\delta^*(s, g(\varepsilon, \alpha'\beta')(g(\alpha', \beta'))^{-1}) = \delta^*(s, g(\varepsilon, \alpha''\beta'')(g(\alpha'', \beta''))^{-1}).$$

Това показва, че дефиницията на $\widetilde{T}_{[\alpha][\beta]}$ не зависи от избора на представителите от класовете на еквивалентност на $\sim_{\widetilde{L}}$ и $\sim_{\widetilde{R}}$.

Сега ще покажем, че Ω-представимата функция $\widetilde{g}$, определена от $\sim_{\widetilde{L}}$, $\sim_{\widetilde{R}}$ и подпоследоватлните преобразуватели $\widetilde{T_{[\alpha][\beta]}}$ има желаното свойство.

Доказателството е стандартно с индукция по дължината $|v|_\Omega$. Ако $v = \varepsilon$ няма какво да доказваме. Нека твърдението, че:

$$\widetilde{g}(u,v) = \begin{cases} \varepsilon, \text{ ако } v = \varepsilon \\ T(s)(g(\varepsilon, uv)(g(u,v))^{-1})^{-1}T(g(\varepsilon, uv)), \end{cases}$$

е вярно за всяко $|v|_\Omega \leq n$. Нека $|v|_\Omega = n+1$. Нека $q = \delta^*(s, g(\varepsilon, uv)(g(u,v))^{-1})$. Тогава от дефиницията на $\widetilde{g}$ имаме, че:

$$\widetilde{g}(u,v) = \begin{cases} (T^0(q) \circ T_{[u][v]})(P_1(v)) \circ \widetilde{g}(uv, \varepsilon) \text{ ако } S_1(v) = \varepsilon \\ (T(q) \circ T_{[u][v]})(P_1(v)) \circ \widetilde{g}(uP_1(v), S_1(v)) \text{ иначе.} \end{cases}$$

В първия случай директно следва, че:

$$T(s)(g(\varepsilon, uv)(g(u,v))^{-1})\widetilde{g}(u,v) = T(g(\varepsilon, uv)).$$

Във втория прилагаме индукционната хипотеза и отново заключаваме, че:

$$T(s)(g(\varepsilon, uv)(g(u,v)^{-1})) \circ T(q)(T_{[u][v]})(P_1(v))\widetilde{g}(uP_1(v), S_1(v)) =$$
$$T(s)(g(\varepsilon, uv)(g(uP_1(v), S_1(v)))^{-1})\widetilde{g}(uP_1(v), S_1(v)) = T(g(\varepsilon, uv)).$$

С това доказахме, че за всяко $v \neq \varepsilon$ е в сила:

$$\widetilde{g}(u,v) = T^0(q)(g(u,v)) = \lambda^*(q, g(u,v))\phi(\delta^*(g(u,v))),$$

което всъщност гласеше теорема 10. Да отбележим, че доказателството е конструктивно и показва как ефективно може да бъдат построени компонентите на функцията $\widetilde{g}$.

4.5 Композиция на Ω-представими функции

Би било хубаво ако класът на Ω-крайнопредставимите функции беше затворен относно композиция. За съжаление, както ще видим в следващата част, съществуват Ω-представими функции, чиято композиция не може да се определи с FIFO-трансдюсер, а от теорема 7 на параграф 4.2, това означава, че и получената функция не е Ω-представима.

Все пак при известни ограничения, композиция е възможна.

Ще докажем следната теорема за композиция на Ω-определими функции:

Теорема 11 *Нека $f : (Pref(\Omega)\times(\Sigma\cup\Omega)^*\cup(\Sigma\cup\Omega)^*\times\{\varepsilon\}) \to (\Sigma\cup\Omega')^*$ е Ω-представима и нека $g : (Pref(\Omega')\times(\Sigma\cup\Omega')^*\cup(\Sigma\cup\Omega')^*\times\{\varepsilon\}) \to \Xi^*$ е Ω'-представима, а $m, M \in \mathbb{N}$ са естествени числа. С $A_{m,M} \subset (\Sigma\cup\Omega)^*$ означаваме множеството от всички думи t със свойствата:*

1. *За всяко разбиване на $t = \alpha\beta\gamma$ с $\alpha \in Pref(\Omega)$ и $|\beta|_\Omega \geq M$ е в сила, че:*
$$\left|\prod_{i=0}^{|\beta|_\Omega-1} T_{[\alpha P_i(\beta)][S_i(\beta)\gamma]}(I_{i+1}(\beta))\right|_{\Omega'} \geq 1. \tag{14}$$
2. *За всяко разбиване на $t = \alpha\beta$, за които $\alpha \in Pref(\Omega)$ е в сила, че:*
$$|T_{[\alpha][\beta]}(I_1(\beta))|_{\Omega'} \leq m, \tag{15}$$

Тогава съществува Ω-представима функция $h : (Pref(\Omega)\times(\Sigma\cup\Omega)^*\cup(\Sigma\cup\Omega)^*\times\{\varepsilon\}) \to \Xi^*$, за която:

$$h(\alpha,\beta) = g^*(f(\varepsilon,\alpha)(f(\alpha,\beta))^{-1}, f(\alpha,\beta))$$

за всеки $\alpha\beta \in A_{m,M}$.

Доказателство: Нека $\sim_{Rf}$ е Ω-крайна с параметър k_f, $\sim_{Lf}$ е Ω-крайноопределима, посредством $\sim_{Rf}$ и $T^f_{[\alpha][\beta]}$ са параметрите, които описват функцията f. Аналогични означения ще използваме за параметрите, задаващи g. Сега ще дефинираме релациите и трансдюсерите, задаващи h.

Започваме с $\sim_{Rh}$ релация над думи от $(\Sigma \cup \Omega)^*$, която ще бъде Ω-крайната релация за h:

Полагаме:

$$k_h = M(m + k_g) + k_f.$$

$\beta' \sim_{Rh} \beta''$ ако са изпълнени едновременно следните условия:

1. $|\beta'|_\Omega = |\beta''|_\Omega \leq k_h$ или $|\beta'|_\Omega > k_h$ и $|\beta''|_\Omega > k_h$.

2. $S_j(\beta') \sim_{Rf} S_j(\beta'')$ за $j \leq M(k_g + m)$.

3. За всеки две думи $\omega' \sim_{Rg} \omega''$, за които $|\omega'|_{\Omega'} = |\omega''|_{\Omega'} = 0$ за всяко $j \leq M(m + k_g)$ за всяка j-орка от трансдюсери $T^f_{i_1}, T^f_{i_2} \dots T^f_{i_j}$ и за всяко $l \leq m$

   (а)
   $$\left|\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta'))\right|_{\Omega'} = \left|\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta''))\right|_{\Omega'} \leq m + k_g \text{ или}$$
   $$\left|\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta'))\right|_{\Omega'} > m + k_g \text{ и } \left|\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta''))\right|_{\Omega'} > m + k_g$$

   (б)
   $$S_l\left(\omega' \prod_{t=1}^{j} T^f_{i_t}(I_t(\beta'))\right) \sim_{Rg} S_l\left(\omega'' \prod_{t=1}^{j} T^f_{i_t}(I_t(\beta''))\right)$$

   (в)
   $$(\delta^f_{i_t})^*(s^f_{i_t}, last(I_t(\beta'))) = (\delta^f_{i_t})^*(s^f_{i_t}, last(I_t(\beta'')))$$

   (г) За всеки трансдюсер T^g_u и всяко $q \in Q^g_u$ е в сила:
   $$(\delta^g_u)^*\left(q, last\left(\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta'))\right)\right) = (\delta^g_u)^*\left(q, last\left(\prod_{t=1}^{j} T^f_{i_t}(I_t(\beta''))\right)\right)$$

Ще покажем, че $\sim_{Rh}$ е дясно-инвариантна. Наистина нека $\beta' \sim_{Rh} \beta''$ и γ са произволни. Ще проверим, че и $\beta'\gamma \sim_{Rh} \beta''\gamma$.

1. Ако $|\beta'\gamma|_\Omega = |\beta'|_\Omega + |\gamma|_\Omega$ и $|\beta''\gamma|_\Omega = |\beta''|_\Omega + |\gamma|_\Omega$, то ако $|\beta'|_\Omega = |\beta''|_\Omega$, то същото е в сила и за $|\beta'\gamma|_\Omega = |\beta''\gamma|_\Omega$ и ако $|\beta'|_\Omega > k_h$, то и $|\beta'\gamma|_\Omega > k_h$. Сега е ясно, че и $\beta'\gamma$ и $\beta''\gamma$ удовлетворяват това условие.

2. Нека $|\beta'|_\Omega = x$ и $|\beta''|_\Omega = y$. Нека $j \leq M(k_g + m) = k_h - k_f$ е произволно. Сега, тъй като $j < k_h$, то ако $x < j$ получаваме, че $x < k_h$, от където получаваме, че $x = y$. Сега ако $j \leq x$, то и $j \leq y$ и тогава $S_j(\beta'\gamma) = S_j(\beta')\gamma$ и аналогично $S_j(\beta''\gamma) = S_j(\beta'')\gamma$. Тъй като $\sim_{Rf}$ е дясно-инвариантна и $S_j(\beta') \sim_{Rf} S_j(\beta'')$ получаваме, че и $S_j(\beta'\gamma) \sim_{Rf} S_j(\beta''\gamma)$. Ако пък $j > x$ то от забележката по горе имаме, че $x = y$ и тогава $S_j(\beta'\gamma) = S_{j-x}(\gamma)$ и $S_j(\beta''\gamma) = S_{j-y}(\gamma) = S_{j-x}(\gamma)$. Което означава, че $S_j(\beta'\gamma) = S_j(\beta''\gamma)$ в този случай и отново твърдението следва.

3. Тъй като за $j \leq M(k_g + m)$ и $j \leq |\beta'|_\Omega$ имаме, че и $j \leq |\beta''|$ то

   $$\omega' \prod_{t=1}^{j} T_{i_t}^{f}(I_t(\beta'\gamma)) = \omega' \prod_{t=1}^{j} T_{i_t}^{f}(I_t(\beta')) \text{ и}$$
   $$\omega' \prod_{t=1}^{j} T_{i_t}^{f}(I_t(\beta''\gamma)) = \omega' \prod_{t=1}^{j} T_{i_t}^{f}(I_t(\beta'')).$$

   Тогава твърденията следват непосредствено от съответните твърдения за $\beta' \sim_{Rh} \beta''$.

   И така нека $j \leq M(k_g + m)$, но $j > |\beta'|_\Omega = x$. Тогава от забележката от предния пункт имаме, че и $|\beta''|_\Omega = x$. Тогава получаваме, че

$$\omega' \prod_{t=1}^{j} T_{i_t}(I_t(\beta'\gamma)) = \omega' \prod_{t=1}^{x} T^f_{i_t}(I_t(\beta'\gamma)) \prod_{t=x+1}^{j} (T^f_{i_t}(\beta'\gamma)) =$$

$$\omega' \prod_{t=1}^{x} T^f_{i_t}(\beta') T^f_{i_{x+1}}(last(\beta')I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \text{ и аналогично}$$

$$\omega' \prod_{t=1}^{j} T_{i_t}(I_t(\beta''\gamma)) = \omega' \prod_{t=1}^{x} T^f_{i_t}(I_t(\beta''\gamma)) \prod_{t=x+1}^{j} (T^f_{i_t}(\beta''\gamma)) =$$

$$\omega' \prod_{t=1}^{x} T^f_{i_t}(\beta'') T^f_{i_{x+1}}(last(\beta'')I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma))$$

Сега тъй като

$$(\delta^f_{i_t})^*(s^f_{i_t}, last(\beta')) = (\delta^f_{i_t})^*(s^f_{i_t}, last(\beta'')),$$

то ако означим:

$$q = (\delta^f_{i_t})^*(s^f_{i_t}, last(\beta'))$$

получаваме, че

$$\omega' \prod_{t=1}^{j} T_{i_t}(I_t(\beta'\gamma)) = \omega' \prod_{t=1}^{x} T^f_{i_t}(\beta') T^f_{i_{x+1}}(last(\beta')I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) =$$

$$\omega' \prod_{t=1}^{x} T^f_{i_t}(\beta') T^f_{i_{x+1}}(last(\beta'))\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) =$$

$$\omega' \prod_{t=1}^{x+1} T^f_{i_t}(\beta')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)).$$

Аналогично получаваме, че:

$$\omega'' \prod_{t=1}^{j} T_{i_t}(I_t(\beta''\gamma)) = \omega'' \prod_{t=1}^{x+1} T^f_{i_t}(\beta'')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)).$$

Сега нека положим:

$$x' = \left| \omega' \prod_{t=1}^{x+1} T^f_{i_t}(\beta')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right|_{\Omega'} \text{ и}$$

$$y' = \left| \omega'' \prod_{t=1}^{x+1} T^f_{i_t}(\beta'')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right|_{\Omega'}.$$

Сега отново лесно се вижда, че ако $l \leq k_g + m$ и $l \geq x'$, то $x' = y'$ и тогава получаваме:

$$S_l\left(\omega' \prod_{t=1}^{x+1} T^f_{i_t}(\beta')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right) =$$

$$S_{l-x'}\left(\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right) =$$

$$S_{l-y'}\left(\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right) =$$

$$S_l\left(\omega'' \prod_{t=1}^{x+1} T^f_{i_t}(\beta'')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right).$$

Тоест двете наставки съвпадат. В противен случай:

$$S_l\left(\omega' \prod_{t=1}^{x+1} T^f_{i_t}(\beta')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right) =$$

$$S_l\left(\omega' \prod_{t=1}^{x+1} T^f_{i_t}(\beta') \right) \lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \text{ и}$$

$$S_l\left(\omega'' \prod_{t=1}^{x+1} T^f_{i_t}(\beta'')\lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma)) \right) =$$

$$S_l\left(\omega'' \prod_{t=1}^{x+1} T^f_{i_t}(\beta'') \right) \lambda^*_{i_{x+1}}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{t-x}(\gamma))$$

Сега $\sim_{Rg}$ е дясно-инвараинтна и твърдението следва непосредствено. Тъй като функцията $last$ е всъщност дава последната наставка на дадена дума, то и условието в) и г) следват директно. Наистина ако $|\gamma|_\Omega = 0$, то

$$
\begin{gathered}
last(\beta'\gamma) = last(\beta')\gamma \text{ и} \\
last(\beta''\gamma) = last(\beta'')\gamma,
\end{gathered}
$$

от където получаваме:

$$
\begin{aligned}
(\delta^f_{i_t})^*(s_{i_t}, last(\beta'\gamma)) = \\
(\delta^f_{i_t})^*((\delta^f_{i_t})^*(s_{i_t}, last(\beta'), \gamma)) = \\
(\delta^f_{i_t})^*((\delta^f_{i_t})^*(s_{i_t}, last(\beta''), \gamma)) = \\
(\delta^f_{i_t})^*(s_{i_t}, last(\beta')\gamma) = \\
(\delta^f_{i_t})^*(s_{i_t}, last(\beta'\gamma)).
\end{aligned}
$$

Ако пък $|\gamma|_\Omega > 0$, то $last(\beta'\gamma) = last(\gamma) = last(\beta''\gamma)$ и няма какво да доказваме.

Аналогично стоят нещата в г). Там ролята на γ се играе от

$$
\lambda^*_{i_t}(q, I_1(\gamma)) \prod_{t=x+2}^{j} (T^f_{i_t}(I_{t-x}(\gamma)),
$$

а пък на β' и β'' им съответстват:

$$
\begin{gathered}
\prod_{t=1}^{x+1} (T^f_{i_t}(I_t(\beta')) \text{ и} \\
\prod_{t=1}^{x+1} (T^f_{i_t}(I_t(\beta'')).
\end{gathered}
$$

И така установихме, че $\sim_{Rh}$ е дясно-инвариантна релация на еквивалентност. Остана да проверим, че е Ω-крайна. Ще покажем, че k_h има желаното свойство:

$$
\forall \beta \text{ с } |\beta|_\Omega > k_h \text{ и } \forall \gamma \Rightarrow \beta \sim_{Rh} \beta\gamma.
$$

За целта трябва да проверим, че условията 1-3 са изпълнени за β и $\beta\gamma$.

(а) От $|\beta|_\Omega > k_h$ получаваме, че и $|\beta\gamma|_\Omega > k_h$.

(б) От това, че $|\beta|_\Omega > k_h > k_h - k_f \geq j$ получаваме, че $S_j(\beta\gamma) = S_j(\beta)\gamma$. Сега обаче $\sim_{Rf}$ е Ω-райна с параметър k_f тъй като $|S_j(\beta)|_\Omega = |\beta|_\Omega - j > k_h - j \geq k_f$, то получаваме, че:

$$S_j(\beta) \sim_{Rf} S_j(\beta)\gamma,$$

откъдето следва, че за всяко $j \leq M(k_g + m)$:

$$S_j(\beta) \sim_{Rf} S_j(\beta\gamma).$$

(в) От условието, че $|\beta'|_\Omega > k_h > M(k_g + m)$ получаваме, че за всяко $j \leq M(k_g + m)$ в сила, че

$$I_j(\beta\gamma) = I_j(\beta),$$

от където следват всички твърдения.

И така $\sim_{Rh}$ е Ω-крайна дясно-инвариантна с параметър k_h.

Преди да дефинираме релацията на $\sim_{Lh}$ ще ни бъде полезна следната:

Дефиниция 25 *Множеството $X_{m,M}$ се състои от онези двойки $(\alpha, \beta) \in Pref(\Omega) \times (\Sigma \cup \Omega)^* \cup (\Sigma \cup \Omega)^* \times \{\varepsilon\}$, за които:*

1. *$\exists i < |\alpha|_\Omega$, за което $|S_i(\alpha)\beta|_\Omega > M(k_g + m) + k_f = k_h$ и има свойството, че:*

$$\left|\prod_{j=0}^{M(k_g+m)} T_{[P_{i+j}(\alpha\beta)][S_{i+j}(\alpha\beta)]}(I_{i+j+1}(\alpha\beta))\right| < m + k_g \text{ или}$$
$$\neg! \left(\prod_{j=0}^{k_h} T_{[P_{i+j}(\alpha\beta)][S_{i+j}(\alpha\beta)]}(I_{i+j+1}(\alpha\beta))\right).$$

2. *или $\exists i < |\alpha|_\Omega$, за което:*

$$T_{[P_i(\alpha)][S_i(\alpha)\beta]}(I_{i+1}(\alpha)) > m \text{ или}$$
$$\neg! T_{[P_i(\alpha)][S_i(\alpha)\beta]}(I_{i+1}(\alpha)).$$

Сега ще дефинираме релацията L_h. Това правим по следния начин. За две двойки думи, (α', β') и (α'', β'') означаваме с $base'$, $last'$ и $rest'$, съответно $base''$, $last''$ и $rest''$:

$$\begin{aligned} rest' &= f(\alpha', \beta') \\ base' &= base(f(\varepsilon, \alpha'\beta')rest'^{-1}, rest') \\ last' &= last(f(\varepsilon, \alpha'\beta')rest'^{-1}, rest') \end{aligned}$$

$$\begin{aligned} rest'' &= f(\alpha'', \beta'') \\ base'' &= base(f(\varepsilon, \alpha''\beta'')rest''^{-1}, rest'') \\ last'' &= last(f(\varepsilon, \alpha''\beta'')rest''^{-1}, rest'') \end{aligned}$$

При тези означения казваме, че

$$(\alpha', \beta') \sim_{Lh} (\alpha'', \beta'')$$

тогава и само тогава, когато са в сила следните условия:

1. $(\alpha', \beta') \in X_{m,M} \Leftrightarrow (\alpha'', \beta'') \in X_{m,M}$.
2. Ако $(\alpha', \beta') \notin X_{m,M}$ и $(\alpha'', \beta'') \notin X_{m,M}$, то:
   (а) $(\alpha', \beta') \sim_{Lf} (\alpha'', \beta'')$.
   (б) $base' \sim_{Lg} base''$.
   (в) $last' \sim_{Rg} last''$.
   (г) $state_g(base'last', rest') = state_g(base''last'', rest'')$ или в случай, че $\alpha' = \alpha'' = \varepsilon$.

Лема 12 *Релацията $\sim_{Lh}$ е крайна релация на еквивалентност, Ω-определена посредством $\sim_{Rh}$.*

Това, че $\sim_{Lh}$ е крайна релация на еквивалентност следва от факта, че всяко от условията l в дефиницията задава крайна релация на еквивалентност, а $\sim_{Lh}$ е тяхна конюнкция.

Освен това лесно се вижда, че:

$$(\varepsilon, \beta) \sim_{Lh} (\varepsilon, \beta'').$$

Наистина тъй като $|\varepsilon|_\Omega = 0$, то $(\varepsilon, \beta') \notin X_{m,M}$, защото не съществува естествено число $i < 0$. Аналогично и $(\varepsilon, \beta'') \notin X_{m,M}$. Първите две свойства следват от съответните свойства за $\sim_{Lf}$ и $\sim_{Lg}$, третото също е ясно $base(\varepsilon, f(\varepsilon, \beta)) = \varepsilon$, а четвъртото следва от самата дефиниция. За да проверим, че $\sim_{Rh}$ определя $\sim_{Lh}$ трябва да проверим, че

$$\begin{gathered}\forall \alpha', \alpha'', \beta', \beta'' \\ (\alpha', \beta') \sim_{Lh} (\alpha'', \beta'') \& \beta' \sim_{Rh} \beta'' \Rightarrow \\ (\alpha' P_1(\beta'), S_1(\beta')) \sim_{Lh} (\alpha'' P_1(\beta''), S_1(\beta'')).\end{gathered}$$

И така нека α', α'', β' и β'' са произволни като

$$\begin{gathered}(\alpha', \beta') \sim_{Lh} (\alpha'', \beta'') \& \\ \beta' \sim_{Rh} \beta''.\end{gathered}$$

Ако $(\alpha', \beta') \in X_{m,M}$, то съществува $i < |\alpha'|$, за което:

$$P_i(\alpha')\beta' > M(k_g + m) + k_f$$

и:

$$\begin{gathered}\left|\prod_{j=0}^{k_h} T_{[P_{i+j}(\alpha\beta)][S_{i+j}(\alpha\beta)]}(I_{i+j+1}(\alpha\beta))\right| < m + k_g \text{ или} \\ \neg! \left(\prod_{j=0}^{k_h} T_{[P_{i+j}(\alpha\beta)][S_{i+j}(\alpha\beta)]}(I_{i+j+1}(\alpha\beta))\right).\end{gathered}$$

или $\exists i < |\alpha'|_\Omega$, за което:

$$\begin{gathered}T_{[P_i(\alpha)][S_i(\alpha)\beta]}(I_{i+1}(\alpha)) > m \text{ или} \\ \neg! T_{[P_i(\alpha)][S_i(\alpha)\beta]}(I_{i+1}(\alpha))\end{gathered}$$

И в двата случая $i < |\alpha' P_1(\beta')|_\Omega$ и $\alpha' P_1(\beta') S_1(\beta') = \alpha' \beta'$. Оттук следва, че $(\alpha' P_1(\beta'), S_1(\beta')) \in X_{m,M}$. Аналогично се показва, че ако $(\alpha'', \beta'') \in X_{m,M}$, то и $(\alpha'' P_1(\beta''), S_1(\beta'')) \in X_{m,M}$. Сега, тъй като $(\alpha', \beta') \sim_{Lh} (\alpha'', \beta'')$, получаваме, че ако $(\alpha', \beta') \in X_{m,M}$, то и $(\alpha'', \beta'') \in X_{m,M}$, откъдето и:

$$(\alpha' P_1(\beta'), S_1(\beta')) \in X_{m,M} \text{ и}$$
$$(\alpha'' P_1(\beta''), S_1(\beta'')) \in X_{m,M}.$$

Това показва, че:

$$(\alpha' P_1(\beta'), S_1(\beta')) \sim_{Lh} (\alpha'' P_1(\beta''), S_1(\beta'')).$$

Остава да разгледаме случая, когато $(\alpha', \beta') \notin X_{m,M}$. Тогава и $(\alpha'', \beta'') \notin X_{m,M}$.

Сега получаваме, че $(\alpha', \beta') \sim_{Lf} (\alpha'', \beta'')$. Тъй като $\beta' \sim_{Rh} \beta''$, то е в сила, че:

$$S_j(\beta') \sim_{Rf} S_j(\beta'') \text{ за } j \leq M(k_g + m)$$

Но $\sim_{Rf}$ определя $\sim_{Lf}$, откъдето получаваме, че:

$$(\alpha' P_j(\beta'), S_j(\beta')) \sim_{Lf} (\alpha'' P_j(\beta''), S_j(\beta'') \text{ за } j \leq M(m + g_k).$$

Нека положим:

$$T'_x = T^f_{[\alpha' P_x(\beta)][S_x(\beta)]} \text{ и}$$
$$T''_x = T^f_{[\alpha'' P_x(\beta'')][S_x(\beta'')]},$$

за $x \leq M(m + k_g) + k_f$. Тогава $T'_i = T''_i$, за $i \leq M(k_g + m)$. Да допуснем, че $(\alpha' P_1(\beta'), S_1(\beta')) \in X_{m,M}$. Тъй като $(\alpha', \beta') \notin X_{m,M}$, а $\alpha'\beta' = \alpha' P_1(\beta') S_1(\beta')$ това е възможно само ако $|\beta'| > M(k_g+m)+k_f = k_h$ и е в сила, че:

$$\left| \prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta')) \right|_{\Omega'} < m + k_g \text{ или}$$
$$\neg! \left(\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta')) \right).$$

или ако:

$$T'_0(I_1(\beta')) > m \text{ или}$$
$$\neg! T'_0(I_1(\beta')).$$

Сега обаче $\beta' \sim_{Rh} \beta''$. Тогава в първия случай получаваме, че $|\beta''|_\Omega > M(k_g+m)+k_f$ и освен това, прилагайки условие 3 от $\sim_{Rh}$ получаваме, че ако:

$$\left|\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta'))\right|_{\Omega'} < m+k_g, \text{ то}$$
$$\left|\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta''))\right|_{\Omega'} = \left|\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta'))\right|_{\Omega'} < m+k_g,$$

и съответно ако

$$\neg!\left(\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta'))\right) \text{ то}$$
$$\neg!\left(\prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta''))\right).$$

Сега обаче $T'_i = T''_i$, което показва, че $(\alpha'' P_1(\beta''), S_1(\beta'')) \in X_{m,M}$.

Аналогично във втория случай имаме, че:

$$|T'_0(I_1(\beta'))|_{\Omega'} > m \Leftrightarrow |T'_0(I_1(\beta''))|_{\Omega'} > m \text{ и}$$
$$\neg!T'_0(I_1(\beta')) \Leftrightarrow T'_0(I_1(\beta'')).$$

Сега обаче $T'_0 = T''_0$, откъдето получаваме, че $(\alpha'' P_1(\beta''), S_1(\beta'')) \in X_{m,M}$.

Така остава да разгледаме случая, когато $(\alpha' P_1(\beta') \notin X_{m,M}$. Тогава от разсъжденията по горе следва, че и $(\alpha'' P_1(\beta''), S_1(\beta'') \notin X_{m,M})$.

Имаме, че

$$f(\alpha',\beta') = \prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta')) f(\alpha' P_{M(k_g+m)+1}(\beta'), S_{M(k_g+m)+1}(\beta'))$$
$$f(\alpha'',\beta'') = \prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta'')) f(\alpha' P_{M(k_g+m)+1}(\beta''), S_{M(k_g+m)+1}(\beta'')).$$

От разсъжденията по горе и от факта, че $\beta' \sim_{Rh} \beta''$, получаваме, че:

$$\left|\prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))\right|_{\Omega'} = \left|\prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))\right|_{\Omega'}$$

или,

$$\left|\prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))\right|_{\Omega'} > k_g + m \text{ и}$$
$$\left|\prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))\right|_{\Omega'} > k_g + m.$$

При това ако и двете стойности са не по-големи от $(k_g + m)$ то, тъй като $(\alpha' P_1(\beta'), S_1(\beta')) \notin X_{m,M}$ и $(\alpha'' P_1(\beta''), S_1(\beta'')) \notin X_{m,M}$ следва, че $|\beta'|_\Omega = |\beta''|_\Omega \leq M(k_g+m)$. Сега тъй като $\sim_{Rg}$ е Ω'-крайна, то за всяко $\omega' \sim_{Rg} \omega''$ и всяко $l \leq m$:

$$S_l(\omega' f(\alpha', \beta')) \sim_{Rg} S_l(\omega' \prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))) \text{ и}$$
$$S_l(\omega'' f(\alpha'', \beta'')) \sim_{Rg} S_l(\omega'' \prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'')))$$

Сега от това, че $\beta' \sim_{Rh} \beta''$ следва, че:

$$S_l(\omega' \prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta'))) \sim_{Rg} S_l(\omega'' \prod_{i=0}^{M(k_g+m)} T'_i(I_{i+1}(\beta''))),$$

от където и

$$S_l(\omega' f(\alpha', \beta')) \sim_{Rg} S_l(\omega'' f(\alpha'', \beta''))$$

Сега ако в ролята на ω' и ω'' изберем

$$\omega' = last(f(\varepsilon, \alpha'\beta') f(\alpha', \beta')^{-1}) \text{ и}$$
$$\omega'' = last(f(\varepsilon, \alpha'\beta') f(\alpha', \beta')^{-1})$$

получаваме, че:

$$S_l(last(f(\varepsilon, \alpha'\beta') f(\alpha', \beta')^{-1}) f(\alpha', \beta')) \sim_{Rg} S_l(last(f(\varepsilon, \alpha''\beta'') f(\alpha'', \beta'')^{-1}) f(\alpha'', \beta'')).$$

Следователно:

$$(base(f((\varepsilon, \alpha'\beta') f(\alpha', \beta')^{-1}) P_l(f(\alpha', \beta'))), S_l(f(\alpha', \beta'))) \sim_{Lg}$$
$$(base(f((\varepsilon, \alpha''\beta'') f(\alpha'', \beta'')^{-1}) P_l(f(\alpha'', \beta''))), S_l(f(\alpha'', \beta'')))$$

От условие $(\alpha' P_1(\beta'), S_1(\beta') \notin X_{m,M}$ и $(\alpha'' P_1(\beta''), S_1(\beta'') \notin X_{m,M}$, то получаваме, че:

$$|T_0'(I_1(\beta')|_{\Omega'} \leq m \text{ и} \tag{16}$$

$$|T_0'(I_1(\beta'')|_{\Omega'} \leq m \tag{17}$$

Сега от това, че $\beta' \sim_{Rh} \beta''$ следва, че

$$|T_0'(I_1(\beta'))|_{\Omega'} = |T_0'(I_1(\beta''))|_{\Omega'}.$$

Да положим:

$$\begin{aligned} base' &= (base(f((\varepsilon, \alpha'\beta')f(\alpha', \beta')^{-1}T_0'(P_1(\beta')))), \\ base'' &= (base(f((\varepsilon, \alpha''\beta'')f(\alpha'', \beta'')^{-1}T_0'(P_1(\beta'')))), \\ last' &= last((f((\varepsilon, \alpha'\beta')f(\alpha', \beta')^{-1}T_0'(P_1(\beta')))), \\ last'' &= last((f((\varepsilon, \alpha''\beta'')f(\alpha'', \beta'')^{-1}T_0'(P_1(\beta'')))). \end{aligned}$$

Тогава:

$$\begin{gathered} (base', last' f(\alpha' P_1(\beta'), S_1(\beta'))) \sim_{Lg} \\ (base'', last'' f(\alpha'' P_1(\beta''), S_1(\beta''))) \text{ и} \\ last' \sim_{Rg} last''. \end{gathered}$$

Накрая имаме, че за всеки от трансдюсерите T_u^g :

$$(\delta_u^g)^*(s_u, last') = (\delta_u^g)^*(s_u, last'').$$

Като приложим това равенство за:

$$T_{[base'][last' f(\alpha' P_1(\beta'), S_1(\beta')))]} = T_{[base''][last'' f(\alpha'' P_1(\beta''), S_1(\beta'')))]},$$

получаваме точно, че

$$state_g(base' last', f(\alpha' P_1(\beta'), S_1(\beta'))) = state_g(base'' last'', f(\alpha'' P_1(\beta'') S_1(\beta''))).$$

Сега остава да отчетем, че всъщност:

$$\begin{aligned} base' &= base(f(\varepsilon, \alpha'\beta')f(\alpha' P_1(\beta'), S_1(\beta'))^{-1}) \\ last' &= last(f(\varepsilon, \alpha'\beta')f(\alpha' P_1(\beta'), S_1(\beta'))^{-1}) \\ base'' &= base(f(\varepsilon, \alpha''\beta'')f(\alpha'' P_1(\beta''), S_1(\beta''))^{-1}) \text{ и} \\ last'' &= last(f(\varepsilon, \alpha''\beta'')f(\alpha'' P_1(\beta''), S_1(\beta''))^{-1}) \end{aligned}$$

получаваме, че

$$(\alpha' P_1(\beta'), S_1(\beta')) \sim_{Lh} (\alpha'' P_1(\beta''), S_1(\beta''))$$

във всички случаи.

Сега остана да определим трансдюсерите $T^h_{[\alpha][\beta]}$. Това правим по следния начин.

1. Определяме
$$q = state_g(f(\varepsilon, \alpha\beta) f(\alpha, \beta)^{-1}, f(\alpha, \beta))$$

2. Определяме
$$l = \left| T^f_{[\alpha][\beta]}(P_1(\beta)) \right|_{\Omega'}.$$

3. Определяме трансдюсерите:
$$T^g_i = T^g_{[base \circ P_i(last \circ rest)][S_i(last \circ rest)]} \text{ за } 0 \leq i \leq l,$$
   където
$$\begin{gathered} base = base(f(\varepsilon, \alpha\beta) f(\alpha, \beta)^{-1}) \\ last = last(f(\varepsilon, \alpha\beta) f(\alpha, \beta)^{-1}) \text{ и} \\ rest = f(\alpha, \beta), \end{gathered}$$
   а дефиниционните области на T^g_i са ограничени до $\Sigma^*\Omega'$ за $i < l$.

Сега полагаме:

$$T^h_{[\alpha][\beta]} = \begin{cases} (T^g_0(q) \prod_{j=1}^{l} T^g_j)(T^f_{[\alpha][\beta]}) \text{ ако } (\alpha, \beta) \notin X_{m,M} \\ \neg! \text{ ако } (\alpha, \beta) \in X_{m,M} \end{cases}$$

Лема 13 *Дефиницията* $T^h_{[\alpha][\beta]}$ *е коректна, тоест за всеки:*

$$\begin{gathered} (\alpha', \beta') \sim_{Lh} (\alpha'', \beta'') \& \beta' \sim_{Rg} \beta'' \Rightarrow \\ T^h_{[\alpha'][\beta']} = T^h_{[\alpha''][\beta'']}. \end{gathered}$$

и освен това ако $(\alpha\beta) \in A_{m,M}$*, то*

$$T^h_{[\alpha][\beta]}(P_1(\beta)) = g^*(base \circ last, rest) g^*(base_1 \circ last_1, rest_1)^{-1},$$

където:

$$\begin{aligned} base &= base(f(\varepsilon, \alpha\beta) f(\alpha, \beta)^{-1}, f(\alpha, \beta)), \\ last &= last(f(\varepsilon, \alpha\beta) f(\alpha, \beta)^{-1}, f(\alpha, \beta)), \\ rest &= f(\alpha, \beta), \ \textit{и аналогично} \end{aligned}$$

$$\begin{aligned} base_1 &= base(f(\varepsilon, \alpha\beta) f(\alpha P_1(\beta), S_1(\beta))^{-1}, f(\alpha P_1(\beta), S_1(\beta))), \\ last_1 &= last(f(\varepsilon, \alpha\beta) f(\alpha P_1(\beta), S_1(\beta))^{-1}, f(\alpha P_1(\beta), S_1(\beta))), \\ rest_1 &= f(\alpha P_1(\beta), S_1(\beta). \end{aligned}$$

Доказателство: Първо ще покажем коректността. Нека $(\alpha', \beta') \sim_{Lh} (\alpha'', \beta'')$. Ако $(\alpha', \beta') \in X_{m,M}$, то и $(\alpha'', \beta'') \in X_{m,M}$ и следователно

$$!\neg T_{[\alpha'][\beta']} \text{ и } \neg ! T_{[\alpha''][\beta'']}.$$

Така че интересен е случаят, когато $(\alpha', \beta') \notin X_{m,M}$ и $(\alpha'', \beta'') \notin X_{m,M}$. Тогава Първите два пункта следват директно от дефинициите на $\sim_{Lh}$ и $\sim_{Rh}$. Това, че изборът на T^g_i не зависи от конкретния избор на α и β, а само от техните класове на еквивалентност в $\sim_{Lh}$ и $\sim_{Rh}$ може да се види по същия начин както в предишната лема установихме, че класовете

$$S_l\left(\omega' \prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta'))\right) \sim_{Rg} S_l\left(\omega'' \prod_{i=1}^{M(k_g+m)} T'_i(I_i(\beta''))\right)$$

за $l \leq m$. Накрая $\sim_{Lg}$-еквивалентността на основите $base' \sim_{Lg} base''$ ни дава, че за всяко $j \leq m$ е в сила, че

$$(base' P_j(last' rest'), S_j(last' rest') \sim_{Lg} (base'' P_j(last'' rest''), S_j(last'' rest''),$$

а от $\sim_{Rh}$ еквивалентността на β' и β'' получаваме, че

$$S_j(last' rest') \sim_{Rg} S_j(last'' rest'').$$

За да проверим втората част на лемата нека $\alpha\beta = t \in A_{m,M}$. Тогава лесно се вижда, че за всяко разбиване на $t = \alpha'\beta'$ е в сила, че $(\alpha', \beta') \notin X_{m,M}$. Тогава от дефиницията на $T^h_{[\alpha][\beta]}$ следва, че:

$$T^h_{[\alpha][\beta]}(P_1(\beta)) = \left(T^g_0(q)\prod_{j=1}^{l} T^g_j\right)(T^f_{[\alpha][\beta]}(P_1(\beta))) =$$
$$\left(T^g_0(q)\prod_{j=1}^{l} T^g_j\right)(rest \circ (rest_1)^{-1}) =$$
$$\lambda^*_0(q, I_1(rest \circ (rest_1)^{-1})\prod_{j=1}^{l} T^g_j(I_{j+1}(rest \circ (rest_1)^{-1}) =$$
$$g^*(base \circ last, rest)g^*(base_1 \circ last_1, rest_1)^{-1},$$

защото $|rest \circ (rest_1)^{-1})|_{\Omega'} = l$.

С това построихме релациите $\sim_{Lh}$ $\sim_{Rh}$ и $T^h_{[\alpha][\beta]}$, които определят функция h:

$$h(\alpha, \beta) = g^*(f(\varepsilon, \alpha\beta)f(\alpha, \beta)^{-1}, f(\alpha, \beta))$$

за всеки (α, β) със свойството $\alpha\beta \in A_{m,M}$.

Сега от теорема 11 лесно получаваме, че:

Следствие 4 *Ако* $f : Pref(\Omega) \times (\Sigma \cup \Omega)^* \cup (\Sigma \cup \Omega)^* \times \{\varepsilon\} \to (\Sigma \cup \Omega')^*$ *и* $g : Pref(\Omega') \times (\Sigma \cup \Omega)^* \cup (\Sigma \cup \Omega')^* \times \{\varepsilon\} \to\to \Xi^*$ *са съответно* Ω- *и* Ω'-*представими и ако допълнително съществуват константи* M *и* m *със свойството, че:*

$$\{\alpha\beta \,|\, (\alpha, \beta) \in Dom(f)\} \subset A_{m,M},$$

където $A_{m,M}$ *се дефинира като в условието на теорема 11, то съществува* Ω-*представима функция* h *със свойството:*

$$h(\alpha, \beta) = g^*(f(\varepsilon, \alpha\beta)f(\alpha, \beta)^{-1}, f(\alpha, \beta)).$$

Доказателство: Конструираме h както в доказателството на теорема 11. Тогава за всеки α и β със свойството, че $\alpha\beta \in A_{m,M}$ е в сила, че:

$$h(\alpha, \beta) = g^*(f(\varepsilon, \alpha\beta)f(\alpha, \beta)^{-1}, f(\alpha, \beta)).$$

Сега твърдението следва от факта, че ако $(\alpha, \beta) \in Range(f)$, то $\alpha\beta \in A_{m,M}$.

Следствие 5 *Ако* $f : (\Sigma \cup \Omega)^* \to (\Sigma \cup \Omega')^*$ *и* $g : (\Sigma \cup \Omega')^* \to \Xi^*$ *се представят чрез* f_1 *и* g_1 *съответно и* f_1 *и* g_1 *удовлетворяват и ако съществуват естествени числа* M *и* m*, за които*

$$\{\alpha\beta \,|\, (\alpha, \beta) \in Dom(f_1)\} \subset A_{m,M},$$

и функцията $h = g \circ f$ *е представима.*

Доказателство: От следствие 4 намираме функция h_1 със свойството:

$$h_1(\alpha, \beta) = g_1^*(f_1(\varepsilon, \alpha\beta)f_1(\alpha, \beta)^{-1}, f_1(\alpha, \beta).$$

Тогава:

$$h_1(\varepsilon, \alpha) = g_1^*(\varepsilon, f_1(\varepsilon, \alpha)) = g(f(\alpha)),$$

което и означава, че h_1 е представяне за h.

4.6 Доказателство на теорема 2 от част 2

След като въведохме Ω-представимите функции и изследвахме част от техните свойства, сега можем да се възползваме от тях и сравнително лесно да докажем теорема 2, свързана с примерите от част 2 - $ins_{0,1}$ и $rep_{0,1}$. За целта първо ще покажем следното по-общо твърдение:

Твърдение 8 1. *Функцията* $ins_{0,1}$ *е* Ω*-представима за* $\Omega = \varnothing$.

2. *Функцията* $rep_{0,1}$ *е* Ω*-представима за* $\Omega = \{0, 1\}$

Доказателство:

1. Дефинираме дясно-инвариантната релация R_1 като:

$$\beta_1 \sim_{R_1} \beta_2 \Leftrightarrow |\beta_1|_{\{a\}} \equiv |\beta_2|_{\{a\}}$$
$$\beta_1 = \varepsilon \Leftrightarrow \beta_2 = \varepsilon,$$

тоест две думи са R_1 еквивалентни тогава и само тогава, когато имат сравним по модул 2 брой букви a и не са празната дума. Ясно е, че R_1 е крайна, дясно-инвариатна релация на еквивалентност. Тъй като $\Omega = \varnothing$, то за всяка дума е в сила, че $|\alpha|_\Omega = 0$, откъдето следва, че R_1 е и Ω крайна.

Дефинираме Ω-определимата релация на еквивалентност L_1 по следния начин:

$$(\alpha_1, \beta_1) \sim_{L_1} (\alpha_2, \beta_2) \Leftrightarrow \alpha_1 = \alpha_2 = \varepsilon \text{ или}$$
$$\beta_1 = \beta_2 = \varepsilon, \text{ и } \alpha_1 \sim_{R_1} \alpha_2.$$

Ясно е, че L_1 е рефлексивна и симетрична. Трябва да проверим, че е транзитивна. Наистина нека $(\alpha_1, \beta_1) \sim_{L_1} (\alpha_2, \beta_2)$ и $(\alpha_2, \beta_2) \sim_{L_1} (\alpha_3, \beta_3)$. Ако $\alpha_1 = \alpha_3 = \varepsilon$, то очевидно $(\alpha_1, \beta_1) \sim_{L_1} (\alpha_3, \beta_3)$. Затова да допуснем, че например $\alpha_3 \neq \varepsilon$. Тогава от дефиницията нарелацията L_1 следва, че $\beta_3 = \varepsilon$, защото L_1 е дефинирана само върху двойки от вида Ω-представка, Ω-наставка и защото $\Omega = \varnothing$. Сега от дефиницията на L_1 следва, че $\beta_2 = \varepsilon$ и $\alpha_2 \sim_{R_1} \alpha_3$. Тогава обаче от дефиницията на R_1 получаваме, че $\alpha_2 \neq \varepsilon$. Като приложим същите разсъждения и за двойките $(\alpha_1, \beta_1) \sim_{L_1} (\alpha_2, \beta_2)$ получаваме, че $\beta_1 = \varepsilon$ и съответно $\alpha_1 \sim_{R_1} \alpha_2$. Сега от транзитивността на R_1 заключаваме, че $\alpha_1 \sim_{R_1} \alpha_3$, а отделно получихме, че $\beta_1 = \beta_3 = \varepsilon$. Това показва, че:

$$(\alpha_1, \beta_1) \sim_{L_1} (\alpha_3, \beta_3),$$

тоест L_1 е транзитивна, а с това и релация на еквивалентност. Сега ще покажем, че R_1 Ω-определя L_1. Наистина нека $(\alpha_1, \beta_1) \sim_{L_1} (\alpha_2, \beta_2)$ и $\beta_1 \sim_{R_1} \beta_2$. Имаме два случая. Ако $\beta_1 = \beta_2 = \varepsilon$ то и $P_1(\beta_1) = S_1(\beta_1) = \varepsilon$ и $P_1(\beta_2) = S_1(\beta_2) =$

ε. Тогава всъщност $(\alpha_1 P_1(\beta_1), S_1(\beta_1)) = (\alpha_1, \beta_1)$. Аналогично $(\alpha_2 P_1(\beta_2, S_1(\beta_2) = (\alpha_2, \beta_2)$. Това показва, че:

$$(\alpha_1 P_1(\beta_1), S_1(\beta_1), \alpha_2 P_1(\beta_2), S_1(\beta_2)).$$

Във втория случай имаме, че $\alpha_1 = \alpha_2 = \varepsilon$. Тогава от $\beta_1 \sim_{R_1} \beta_2$ и $\Omega = \varnothing$ получаваме, че:

$$P_1(\beta_1) = \beta_1 \text{ и } S_1(\beta_1) = \varepsilon$$
$$P_1(\beta_2) = \beta_2 \text{ и } S_1(\beta_2) = \varepsilon.$$

Това означава обаче и $(\beta_1, \varepsilon) \sim_{L_1} (\beta_2, \varepsilon)$ и тъй като $\alpha_1 = \alpha_2 = \varepsilon$, то:

$$(\alpha_1 P_1(\beta_1), S_1(\beta_1)) \sim_{L_1} (\alpha_2 P_1(\beta_2), S_1(\beta_2))$$

Накрая дефинираме трансдюсерите:

$$T_{[\varepsilon][\beta]} = \begin{cases} pref_{0,1} \text{ ако } |\beta|_{\{a\}} \equiv 0(\mod 2) \\ pref_{1,0} \text{ ако } |\beta|_{\{a\}} \equiv 0(\mod 2), \end{cases}$$

където $pref_{0,1}$ и $pref_{1,0}$ са както в параграф 2.2 на част 2. Тогава имаме, че $\varnothing$-определимата функция, зададена от L_1, R_1 и трансдюсерите $pref_{0,1}$ и $pref_{1,0}$ е функцията:

$$f(t) = \begin{cases} pref_{0,1}(t) \text{ ако } |t|_{\{a\}} \equiv 0(\mod 2) \\ pref_{1,0}(t) \text{ ако } |t|_{\{a\}} \equiv 1(\mod 2) \end{cases}$$

Нека $t = \prod_{i=0}^{k}(a^{m_i} b^{n_i})$. Тогава имаме, че:

$$pref_{0,1}(t) = \prod_{i=0}^{k}(c_i a^{m_i} b^{n_i}) c_{k+1}, \text{ където}$$
$$c_i \in \{0, 1\} \text{ и } c_i \equiv \sum_{j=0}^{i-1} m_i(\mod 2).$$

Сега ако $\sum_{j=0}^{k} m_j = |t|_{\{a\}} \equiv 0(\mod 2)$, то и:

$$\sum_{j=i}^{k} m_i \equiv c_i(\mod 2),$$

за всяко i, откъдето директно следва, че $f(t) = ins_{0,1}(t)$.

Нека сега разгледаме случая, когато $|t|_{\{a\}} \equiv 1(\mod 2)$. Тогава имаме, че:

$$\sum_{j=0}^{k} m_j \equiv 1(\mod 2) \Leftrightarrow \sum_{j=0}^{i-1} m_j \equiv 1 - \sum_{j=i}^{k}(\mod 2)$$

за всяко i. Оттук получаваме, че $ins_{0,1}(t) = pref_{1,0}(t)$. С това показахме, че дейтвително $f = ins_{0,1}$. С това доказахме, че $ins_{0,1}$ е $\varnothing$-представима.

2. Дефинираме R_2 по следния начин:

$$\beta_1 \sim_{R_2} \beta_2 \Leftrightarrow \forall\gamma(\beta_1\gamma \in \{a,b\}^*0\{a,b,0,1\}^* \Leftrightarrow \beta_2\gamma \in \{a,b\}^*0\{a,b,0,1\}^*) \text{ и } \forall\gamma(\beta_1\gamma \in \{a,b\}^*1\{a,b,0,1\}^* \Leftrightarrow \beta_2\gamma \in \{a,b\}^*1\{a,b,0,1\}^*).$$

Като композиция на две дясно-инвариантни, крайни релации на еквивалентност, R_2 също е дясно-инвариантна, крайна релация на еквивалентност. Освен това ако $|\beta|_{\{0,1\}} \geq 1$, то $\beta = xcy$, където $x \in \{a,b\}^*$, а $c \in \{0,1\}$. Сега ако γ е произволна, то $\beta\gamma = xcy\gamma$ и следователно ако $\beta\gamma \in \{a,b\}^*c\{a,b,0,1\}^*$, както и $\beta \in \{a,b\}^*c\{a,b,0,1\}^*$. Аналогично $\beta \notin \{a,b\}^*\overline{c}\{a,b,0,1\}^*$ и $\beta\gamma \notin \{a,b\}^*\overline{c}\{a,b,0,1\}^*$, където $\overline{c} = 1 - c$. С това показахме, че $\beta \sim_{R_2} \beta\gamma$, тоест R_2 е $\{0,1\}$-крайна релация.

Сега дефинираме релацията L_2 като:

$$(\alpha_1, \beta_1) \sim_{L_2} (\alpha_2, \beta_2) \Leftrightarrow \alpha_1 = \alpha_2 = \varepsilon \text{ или } \beta_1 \sim_{R_2} \beta_2$$

Както и в първия случай лесно се вижда, че R_2 $\{0,1\}$ определя L_2. Сега за всеки α, β дефинираме:

$$T_{[\alpha][\beta]} = \begin{cases} T_A, \text{ ако } \beta \in \{a,b\}^*1\{a,b,0,1\}^* \\ T_a, \text{ ако } \beta \in \{a,b\}^*0\{a,b,0,1\}^* \\ \neg! \text{ ако } \beta \in \{a,b\}^*. \end{cases}$$

където $T_A(\alpha 1) = \alpha_A$, а $T_a(\alpha 0) = \alpha$, за всяко $\alpha \in \{a, b\}^*$. Сега да разгледаме $\{0, 1\}$-представимата функция, зададена от L_2, R_2, T_A и T_a лесно получаваме, че:

$$f(\alpha c \beta) = \begin{cases} T_A(\alpha c) f(\beta) \text{ ако } c = 1 \\ T_a(\alpha c) f(\beta) \text{ ако } c = 0 \\ \neg! \text{ иначе} \end{cases}$$

Това обаче е точно дефиницията на $rep_{0,1}$. С това показахме, че и $rep_{0,1}$ е $\{0, 1\}$-редставима.

Като следствие от твърдение 8 получаваме, че:

Следствие 6
1. *Всяка от функциите $ins_{0,1}$ и $rep_{0,1}$ е рационална.*
2. *За всяка от функциите $ins_{0,1}$ и $rep_{0,1}$ съществува FIFO-трансдюсер, който я разпознава.*

Доказателство: Тъй като $ins_{0,1}$ и $rep_{0,1}$ са Ω-представими, то теорема 7 следва, че за тях съществува FIFO-трансдюсер, който ги разпознава, а от теорема 9 следва, че те са рационални.

С това доказахме в частност и теорема 2.

5 Пример на Ω-представими функции, за чиято композиция не съществува FIFO-трансдюсер. Следствия

В тази част ще покажем, че не съществува FIFO-трансдюсер, който да разпознава $rep_{0,1} \circ ins_{0,1}$. Както видяхме в края на предишната част това са две рационални функции, за които съществуват FIFO-трасдюсери, които ги разпознават. Като резултат от това тяхната композиция също е рационална функция [Eil74]. Така, ако не съществува FIFO-трансдюсер, който да разпознава $rep_{o,1} \circ ins_{0,1}$ ще получим, че:

1. Класът от функции, разпознаван от FIFO-трансдюсери не е затворен относно композиция.

2. Класът на рационалните функции не е подмножество на класа от функции, които се описват с FIFO-трансдюсер.

Преди да пристъпим към основния резултат в тази част, ще докажем следната:

Лема 14 *Ако $m_0 \geq 0$, $m_i > 0$ и $n_{i-1} > 0$, $n_k \geq 0$ са естествени числа, то*

$$rep_{0,1}(ins_{0,1}\left(\prod_{i=0}^{k}(a^{m_i}b^{n_i})\right)) = \prod_{i=0}^{k}(c_i^{m_i}b^{n_i}),$$

където

$$c_i = A \Leftrightarrow \sum_{j=i+1}^{k} m_j \equiv 1(\mod 2)$$

$$c_i = a \Leftrightarrow \sum_{j=i+1}^{k} m_j \equiv 0(\mod 2).$$

Доказателство: Нека

$$ins_{0,1}\left(\prod_{i=0}^{k}(a^{m_i}b^{n_i})\right) = \prod_{i=0}^{k}(d_i a^{m_i}b^{n_i})d_{k+1},$$

където $d_i \in \{0,1\}$. Тогава от дефиницията на $ins_{0,1}$ имаме, че:

$$d_i \equiv \sum_{j=i}^{k} m_j (\mod 2).$$

От друга страна от дефиницията на $rep_{0,1}$ и от равенството:

$$rep_{0,1}\left(\prod_{i=0}^{k}(d_i a^{m_i} b^{n_i}) d_{k+1}\right) = \prod_{i=0}^{k}(c_i^{m_i} b^{n_i}),$$

получаваме, че:

$$c_i = a \Leftrightarrow d_{i+1} = 0 \text{ и}$$
$$c_i = A \Leftrightarrow d_{i+1} = 1.$$

Сега като вземем в предвид дефиницията на d_i получаваме, че:

$$c_i = \begin{cases} A, \text{ ако } \sum_{j=i+1}^{k} m_j \equiv 1(\mod 2) \\ a, \text{ ако } \sum_{j=i+1}^{k} m_j \equiv 0(\mod 2) \end{cases}$$

С това лемата е доказана.

Лема 14 ни дава явния вид на композицията $rep_{0,1} \circ ins_{0,1}$. Така всъщност проблемът, който разискваме в тази част се свежда до:

Теорема 12 *Нека $f : \{a,b\}^* \to \{a,b,A\}^*$ е функцията, която за произволно k и произволни $m_i > 0$ за $i > 1$ и $n_i > 0$ за $i < k$ е дефинирана чрез равенството:*

$$f\left(\prod_{i=1}^{k} a^{m_i} b^{n_i}\right) = \prod_{i=1}^{k} c_i^{m_i} b^{n_i},$$

$c_i \in \{a, A\}$ и $c_i = a$ тогава и само тогава, когато

$$\sum_{j=i+1}^{k} m_j \equiv 1(\mod 2).$$

Тогава не съществува FIFO-трансдюсер T, за който $f_T = f$.

Да допуснем, че твърдението не е вярно. Тогава съществува FIFO-трансдюсер, разпознаващ f, а от теорема 3 следва, че съществува и каноничен FIFO-трансдюсер $T =< \{a,b\}\times\{a,b,A\}^*, P, Q, s, \Delta, d, out, \phi, \psi)$ за който $f_T = f$.

Нека

$$l = \max_{q\in Q, p\in P, c\in\{a,b\}} (|out(q, < p, a >)|, |\psi(q)|).$$

Тъй като $Dom(f) = \{a,b\}^+$ е тотална, то и $Dom(f_T) = \{a,b\}^+$.

Лема 15 *Нека $\alpha \in a^+\{a,b\}^*$. Тогава:*

1. *ако съществува извод от вида:*

$$(s, \alpha, \varepsilon, \varepsilon) \vDash (p, \beta, \gamma, \omega), \tag{18}$$

   то $\omega = \varepsilon$.

2. *ако*

$$(s, \alpha, \varepsilon, \varepsilon) \vDash (p_0, \varepsilon, \gamma_0, \omega_0)$$

   за някое $p_0 \in P$, то $l(|\gamma_0| + 1) \geq |\alpha|$.

1. Действително от дефиницията за f_T и от твърдение 4 следва, че $f_T(\alpha \circ b) = \omega\omega_1$ и $f_T(\alpha \circ ba) = \omega\omega_2$. От друга страна лесно се вижда, че броят на символите a в блоковете след първия в думите $\alpha \circ ba$ и $\alpha \circ b$ се различава с 1. Тогава от дефиницията на f получаваме, че $f(\alpha \circ b)$ и $f(\alpha \circ ba)$ нямат обща представка - едната започва с a, докато другата с A. Но по допускане $f = f_T$, от където следва, че $f_T(\alpha \circ b) = f(\alpha \circ b)$ и $f_T(\alpha \circ ba) = f(\alpha \circ ba)$. Това означава, че ω е обща представка за $f(\alpha \circ b)$ и $f(\alpha \circ ba)$, което е възможно само за $\omega = \varepsilon$.

2. Имаме, че $\alpha \in Dom(f) = Dom(f_T)$. Това по дефиниция означава, че съществуват $p \in P$ и $q \in Q$, за които:

$$(s, \alpha, \varepsilon, \varepsilon) \vDash (p, \varepsilon, \gamma, \omega_1), \tag{19}$$

$$(\phi(p), \varepsilon, \gamma, \omega_1) \vDash (q, \varepsilon, \varepsilon, \omega_1\omega_2) \text{ и} \tag{20}$$

$$f_T(\alpha) = \omega_1\omega_2\psi(q). \tag{21}$$

От твърдение 4 следва, че $p_0 = p$, $\gamma = \gamma_0$ и $\omega_1 = \omega_0$. Сега от първата част на лемата следва, че $\omega_1 = \varepsilon$. Освен това от първата част на твърдение 4 получаваме, че дължината на извода

$$(\phi(p), \varepsilon, \gamma, \omega_1) \vDash (q, \varepsilon, \varepsilon, \omega_1\omega_2)$$

е равна на $|\gamma|$. Следовтелно тъй като при всеки преход дължината на изходната дума се увеличава най-много с

$$|out(q', < p', c >)| \leq l$$

за някои $p' \in P$, $q' \in Q$ и $c \in \{a, b\}$. Тогава $|\omega_1\omega_2| - |\omega_1| \leq l|\gamma|$. От тук можем да зключим, че $|f_T(\alpha)| \leq |\gamma|l + |\psi(q)| \leq (|\gamma| + 1)l$. От друга страна от дефиницията на f имаме, че $|f(\alpha)| = |\alpha|$. По допускане имаме, че $f_T(\alpha) = f(\alpha)$ и следователно $l(|\gamma|+1) \geq |\alpha|$.

Нека положим $M = (l + 1)|Q| + 1$. Дефинираме редицата $\{m_i\}_{i=0}^M$ индуктивно по следния начин:

$$m_M = 1 \tag{22}$$

$$m_{i-1} = l(\textstyle\sum_{j=i}^M (m_j + 1) + 1) \text{ за всяко } M \geq i \geq 1. \tag{23}$$

Сега разглеждаме думите $\{\alpha_j\}_{j=1}^M$, дефинирани като:

$$\alpha_j = \left(\prod_{i=0}^{j-1} a^{2m_i} b\right) a^{2m_j+1} \left(\prod_{i=j+1}^{M} ba^{2m_i}\right). \tag{24}$$

За всяко $1 \leq j \leq M$ полагаме:

1. β_j, така че:

$$\beta_j = a^{2m_0} \prod_{i=1}^{j} ba^{2m_i}$$

2. γ_j и $p_j \in P$, за които съществува извод:

$$(s, \alpha_j, \varepsilon, \varepsilon) \vDash (p_j, \varepsilon, \gamma_j, \varepsilon)$$

и от твърдение 4 се определят еднозначно.

3. $\gamma_{i,r}$ наставка на γ_i с дължина $|\gamma_{i,r}| = \frac{2rm_0}{l} \leq |\gamma_i|$.

4. $q_{i,r} \in Q$ и $\omega_{i,r}$ се определят от извода:

$$(\phi(p_i), \varepsilon, \gamma_i, \varepsilon) \models (q_{i,r}, \varepsilon, \gamma_{i,r}, \omega_{i,r}).$$

Да отбележим следните две свойства, които обединяваме в следната:

Лема 16 *За всеки две $1 \leq i < j \leq M$ са в сила следните две твърдения:*

1. $|\alpha_i| = |\alpha_j| = \frac{2(l+1)m_0}{l} - (M+1)$ и $\gamma_{i,1}$ и $\gamma_{j,1}$ са добре определени.

2. Ако $\gamma_{i,r}$ и $\gamma_{j,r}$ са определени, то $\gamma_{j,r}$ и $\gamma_{i,r}$ съвпадат в първите си

$$\frac{2rm_0}{l} - \sum_{k=i+1}^{M}(2m_k+1)$$

позиции.

1. Първата част следва директно от дефиницията на α_j. Имаме, че:

$$|\alpha_j| = \textstyle\sum_{i=0}^{j-1}(2m_i+1) + (2m_j+1) + \sum_{i=j+1}^{M}(2m_j+1) = \\ 2m_0 + 2\sum_{i=1}^{M} m_i + (M+1).$$

По дефиниция обаче $m_0 = l(\sum_{i=1}^{M}(m_i+1)+1)$. От тук следва, че

$$\sum_{i=1}^{M} m_i = \frac{m_0}{l} - (M+1).$$

Сега като заместим получаваме, че:

$$|\alpha_j| = 2m_0 + 2\tfrac{m_0}{l} - 2(M+1) + (M+1) = \\ \tfrac{2m_0(l+1)}{l} - (M+1).$$

Втората част следва от първата като приложим лема 15. Наистина от лема 15 имаме, че:

$$l(|\gamma_i| + 1) \geq |\alpha_i|.$$

Сега както вече видяхме $|\alpha_i| = 2m_0 + \frac{2m_0}{l} - (M+1)$. Но $m_j \geq 1$ за всяко j, от където:

$$2m_0 = 2l(\sum_{k=1}^{M}(m_k + 1) + 1) \geq 2l(2M + 1).$$

Следователно:

$$\frac{2m_0}{l} - (M+1) \geq 4M + 2 - (M+1) = 3M + 1.$$

От тук тъй като $M > l$ заключаваме, че

$$|\gamma_i| + 1 \geq \frac{|\alpha|}{l} \geq \\ \frac{2m_0}{l} + \frac{3M+1}{l} \geq \frac{2m_0}{l} + 3.$$

Това означава, че $|\gamma_i| \geq \frac{2m_0}{l} + 1$.

2. Тъй като $\beta_i = a^{2m_0} \prod_{k=1}^{M} ba^{2m_k}$ е представка и на α_i, и на α_j, защото $i < j$, то и двата извода

$$(s, \alpha_i, \varepsilon, \varepsilon) \vDash (p_i, \varepsilon, \gamma_i, \varepsilon) \text{ и} \\ (s, \alpha_j, \varepsilon, \varepsilon) \vDash (p_j, \varepsilon, \gamma_j, \varepsilon)$$

съществуват състояние $p \in P$ и дума $\gamma \in \Gamma^*$, за които:

$$(s, \beta_i(\beta_i^{-1}\alpha_i), \varepsilon, \varepsilon) \vDash (p, (\beta_i^{-1}\alpha_i), \gamma, \varepsilon),$$

и съответно:

$$(s, \beta_i(\beta_i^{-1}\alpha_j), \varepsilon, \varepsilon) \vDash (p, (\beta_i^{-1}\alpha_j), \gamma, \varepsilon),$$

се определят еднозначно от β_i. Останалата част от двата извода съдържат точно $\sum_{k=i+1}^{M}(2m_k+1)$ прехода от тип 1 и следователно толкова пъти се добавя символ като наставка на γ. Тогава всеки символ в γ_i на позиция $|\gamma_i| - \sum_{k=i+1}^{M}(2m_k+1) - t \geq 0$ е равен на символа на позиция $|\gamma| - t$ γ. Същото е в сила и за γ_j. Това означава, че $\gamma_{i,r} = \gamma_{j,r}$ се различават евентуално само в някой от последните си $\sum_{k=i+1}^{M}(2m_k+1)$ на брой символа.

С това лемата е доказана.

За всяко i определяме r_i да бъде максималното r, за което $\gamma_{i,r}$ е определено. Тъй като T е каноничен за всеки преход от тип 1 записваме точно един символ в опашката. Тогава големината $|\gamma_i| \leq |\alpha_i| < \frac{2(l+1)m_0}{l}$. Следователно $r_i \leq l$ за всяко i. Полагаме δ_i да бъде наставката на γ_{i,r_i} с дължина:

$$|\delta_i| = \sum_{k=i+1}^{M}(2m_k+1).$$

Накрая нека $\omega_i = \omega_{i,r_i}$ и $q_i = q_{i,r_i}$, тоест:

$$(\phi(p_i), \varepsilon, \gamma_i, \varepsilon) \vDash (q_i, \varepsilon, \gamma_{i,r_i}\delta_i, \omega_i).$$

Тъй като $1 \leq r_i \leq l$ и $q_i \in Q$ то съществуват неповече от $l|Q|$ различни двойки (q_i, r_i). Тъй като $M > l|Q|$, то съществуват $1 \leq i < j \leq M$, за които $q_i = q_j$ и $r_i = r_j$. Тъй като γ_{i,r_i+1} не е дефинирана, то $|\gamma_i\gamma_{i,r_i}^{-1}| < \frac{2m_0}{l}$. Сега тъй като при всеки d-преход дължината на ω се увеличава най-много с l, а дължината на изводите:

$$(\phi(p_i), \varepsilon, \gamma_i, \varepsilon) \vDash (q_i, \varepsilon, \gamma_{i,r_i}, \omega_i) \text{ и}$$
$$(\phi(p_j), \varepsilon, \gamma_j, \varepsilon) \vDash (q_i, \varepsilon, \gamma_{j,r_j}, \omega_j).$$

е по-малка от $\frac{2m_0}{l}$, то $|\omega_i| < 2m_0$ и $\omega_j < 2m_0$. От лема 16 имаме, че $\gamma_{i,r_i}\delta_i^{-1}$ е представка на γ_{j,r_j}. Да разгледаме изводите:

$$(q_i, \varepsilon, \gamma_{i,r_i}, \omega_i) \vDash (q_i', \varepsilon, \delta_i, \omega_i\omega)$$
$$(q_i', \varepsilon, \delta_i, \omega_i\omega) \vDash (q_i'', \varepsilon, \varepsilon, \omega_i\omega\omega_i')$$

и

$$(q_i, \varepsilon, \gamma_{i,r_i}, \omega_j) \vDash (q_i', \varepsilon, (\gamma_{i,r_i}\delta_i^{-1})^{-1}\gamma_{j,r_j}), \omega_j\omega)$$
$$(q_i', \varepsilon, (\gamma_{i,r_i}\delta_i^{-1})^{-1}\gamma_{j,r_j}), \omega_j\omega) \vDash (q_j'', \varepsilon, \varepsilon, \omega_j\omega\omega_j').$$

Вторите изводи от двете двойки са с дължина $|\delta_i| = \sum_{k=i+1}^{M}(2m_k+1)$. На всеки преход дължината на изходната дума се увеличава най-много с l и следователно

$$|\omega_i'| \leq l \sum_{k=i+1}^{M}(2m_k+1) =$$

$$2l \sum_{k=i+1}^{M}(m_k+1) - l(M-i) = 2m_i - l(M-i+1).$$

Тогава дължината на $\omega_i'\psi(p_i'')$ ненадминава $2m_i - l(M-i) \leq 2m_i$. Аналогично получаваме, че $|\omega_j'\psi(q_j'')| \leq 2m_i$. По предположение имаме, че

$$f(\alpha_j) = f_T(\alpha_j) = \omega_j\omega\omega_j'\psi(q_j'') \text{ и}$$
$$f(\alpha_i) = f_T(\alpha_i) = \omega_i\omega\omega_i'\psi(q_i'').$$

От друга страна от дефиницията на f получаваме:

$$f(\alpha_i) = f\left(\left(\prod_{k=0}^{i-1} a^{2m_k}b\right) a^{2m_i+1} \left(\prod_{k=i+1}^{M} ba^{2m_k}\right)\right) =$$
$$\left(\prod_{k=0}^{i-1} a^{2m_k}b\right) A^{2m_i+1} \left(\prod_{k=i+1}^{M} bA^{2m_k}\right)$$

и

$$f(\alpha_j) = f\left(\left(\prod_{k=0}^{j-1} a^{2m_k}b\right) a^{2m_j+1} \left(\prod_{k=j+1}^{M} ba^{2m_k}\right)\right) =$$
$$\left(\prod_{k=0}^{j-1} a^{2m_k}b\right) A^{2m_j+1} \left(\prod_{k=j+1}^{M} bA^{2m_k}\right) =$$
$$\left(\prod_{k=0}^{i-1} a^{2m_k}b\right) a^{2m_i}b \left(\prod_{k=i+1}^{j-1} ba^{2m_k}\right) A^{2m_j+1} \left(\prod_{k=j+1}^{M} bA^{2m_k}\right).$$

Сега тъй като $|\omega_i'| \leq 2m_i$ и $|\omega_j'| \leq 2m_j$, то

$$\left(\prod_{k=0}^{i-1} a^{2m_k}b\right) A \text{ е представка на } \omega_i\omega$$
$$\left(\prod_{k=0}^{i-1} a^{2m_k}b\right) a \text{ е представка на } \omega_j\omega.$$

Единствената възможност това да се случи е $\omega = \varepsilon$, но тогава $a^{2m_0}b$ се съдържа в ω_i, докато ние видяхме, че $|\omega_i| < 2m_0$.

И така стигнахме до противоречие, което се дължи на допускането, че $f = f_T$. Следователно не съществува, FIFO-трансдюсер, чиито език да е графиката на f.

Да обобщим резултатите, получени в тази и предходната част:

Следствие 7 *1. Не съществува FIFO-трансдюсер, който да разпознава функцията:*

$$rep_{0,1} \circ ins_{0,1}$$

2. Класът от Ω-представими функции не е затворен относно композиция.

3. Класът от функции, разпознавани от FIFO-трансдюсери не е затворен относно композиция.

4. Класът от функции, разпознавани от FIFO-трансдюсер, не съдържа класа от всички рационални функции.

Доказателство: Доказателството на първото твърдение е директно следствие от лема 14 и теорема 12. В част 4, параграф 4.6 видяхме, че $rep_{0,1}$ и $ins_{0,1}$ са Ω-представими и рационални функции, за които може да се построи FIFO-трансдюсер. Това показва, че тяхната композиция е рационална функция [Eil74]. А тъй като не съществува FIFO-трансдюсер за тяхната композиция, то тя не е Ω-представима, както следва от теорема 7.

6 Ω-крайни и регулярни правила за заместване.

От предходната част 5 разбрахме, че с помощта на FIFO-трансдюсери не можем да опишем класа от всички рационални функции. Това означава, че не за всяко регулярно правило за заместване ще ни се удаде да конструираме FIFO-трансдюсер, който да извършва замяната. В тази част целта ни е да отделим клас от регулярни правила, за които, все пак, това е възможно. Ограниченията, които ще наложим, ще бъдат алгоритмично проверими. Това ще ни позволи по дадено правило да проверим дали то може да бъде представено с FIFO-трансдюсер и ако това е така да го конструираме алгоритмично.

Ще реализираме тази цел на стъпки. Първо ще разгледаме един специален клас правила, в които предварително са поставени маркери. Тези маркери ще имат функцията да ограждат потенциалния фокус на правилото и ще играят съществена роля при разрешаването на конфликти. Ще покажем, че можем да разпознаваме такива правила, като за всяко едно такова правило ефективно ще построим еквивалентна на него Ω-представима функция.

Във втория параграф на тази част ще разгледаме произволни правила и ще наложим такива ограничения върху тях, че да можем да ги сведем към правила с маркери. Във фазата на замяна ще използваме подпоследователен преобразувател, който маркира текста, след което върху получения маркиран текст ще трябва да приложим правилото с маркери, и накрая ще остане да изтрием ненужните маркери, което отново ще направим с подпоследователен преобразувател. Тогава изходното правило ще можем да представим като композиция на подпоследвотелни преобразуватели с FIFO-трансдюсер, която както видяхме в част 3 ефективно може да се трансформира в един единствен FIFO-трансдюсер.

В последния параграф ще обърнем внимание на композицията на регулярни правила. И тук ще наложим алторитмично проверими условия, които да гарантират, че такава композиция е възможна да бъде представена посредствам FIFO-трансдюсер и ще опишем процедура за неговото построяване.

Започваме с дефинициите, свързани с регулярни правила и правила с ограничен брой маркери.

6.1 Регулярни правила с ограничен брой маркери Ω

В този параграф, както и в част 4 азбуките Σ и Ω са крайни азбуки, за които:

$$\Sigma \cap \Omega = \varnothing.$$

Следващите дефинции се отнасят до произволни регулярни правила за заместване.

Дефиниция 26 *Регулярно правило за заместване това е израз $\mathcal{R}$ от вида $E \to T(E)\backslash L\_R$, където E, R и L са регулярни езици над азбука Σ, а $T : E \to \Xi^*$ е рационална функция.*

Дефиниция 27 *Нека $\mathcal{R}$, $E \to T(E)\backslash L\_R$ е регулярно правило за заместване. Казваме, че $longest_{\mathcal{R}}(\alpha, \beta) =< \alpha, v, w >$ е най-дългото срещане на това правило в думата $t = \alpha\beta$ относно α ако:*

1. *$\alpha \in \Sigma^* L$, $v \in E$ и $w \in R\Sigma^*$ и $vw = \beta$.*

2. *ако $v' \in E$ и $w' \in R\Sigma^*$ са такива, че $v'w' = \beta$, то $|v'| \le |v|$.*

Дефиниция 28 *Заместване на регулярно правило за заместване $\mathcal{R}$, $E \to T(E)\backslash L\_R$, наричаме функцията $replace : \Sigma^* \times \Sigma^* \to \Xi^*$, която се дефинира по следния начин:*

1. *$replace_{\mathcal{R}}(\alpha, \varepsilon) = \varepsilon$.*

2. *ако $\beta = a\beta'$, то*

$$replace_{\mathcal{R}}(\alpha, a\beta') = \begin{cases} a \circ replace(\alpha a, \beta'), \\ \text{ако } longest_{\mathcal{R}}(\alpha, \beta) \text{ не е дефинирана} \\ \\ T(v) \circ replace(\alpha \circ v, w), \text{ ако } longest_{\mathcal{R}} =< \alpha, v, w > \end{cases}$$

Дефиниция 29 *Заместване на текст по регулярно правило $\mathcal{R}$, $E \to T(E)\backslash L\_R$ по стратегията най-ляво, най-дълго, наричаме функцията $rewrite_{\mathcal{R}} : \Sigma^* \to \Xi^*$, която се дефинира чрез:*

$$rewrite_{\mathcal{R}}(t) = replace_{\mathcal{R}}(\varepsilon, t).$$

за всяко $t \in \Sigma^*$.

Когато казваме, че искаме да представим дадено правило $\mathcal{R}$ обикновено разбираме да представим неговата $rewrite_{\mathcal{R}}$.

Сега пристъпваме към дефиницията на един клас от правила, които ще играят важна роля в нашите разглеждания по-нататък.

Дефиниция 30 *Едно регулярно правило за заместване $\mathcal{R}$ $E \to T(E)\backslash L\_R$, където E, L и R са регулярни езици над азбука $\Sigma \cup \Omega$ с $\Sigma \cap \Omega = \varnothing$, а $T : E \to \Xi^*$ е рационална функция наричаме правило с краен брой маркери Ω ако:*

1. *$L \subset (\Sigma \cup \Omega)^*\Omega$ и $R \subset \Omega(\Sigma \cup \Omega)^*$.*

2. *Съществуват константи $e, r \in \mathbb{N}$, че*

$$\begin{aligned} \forall v \in E \quad & |v|_\Omega \leq e \\ \forall w \in R \quad & |w|_\Omega \leq r. \end{aligned}$$

3. *Функцията T може да се представи чрез подпоследователен трансдюсер.*

Целта ни ще бъде да покажем, че всяко правило с краен брой маркери е Ω-представимо. За това ще ни трябват някои означения и твърдения, които да ни позволят да дефинираме съответните $\sim_L$, $\sim_R$ и трансдюсери.

Нека

$$T = < (\Sigma \cup \Omega) \times \Xi^*, Q, s, \delta, \lambda, \phi >$$

са компонентите на подпоследователния трансдюсер. С $T(q) \restriction \Sigma^*\Omega$ ще означаваме трансдюсера получен от изходния, с начално състояние q, който работи и разпознава думи от $\Sigma * \Omega$, запазвайки δ- и

λ- функциите на изходния, но с тривиална ϕ-функция - всички състояния са финални и образът им е ε. Ще използваме и $T_f(q) \restriction \Sigma^*\Omega$ и $T_f(q) \circ id(\Omega) \restriction \Sigma^*\Omega$. $T_f(q) \restriction \Sigma^*\Omega$ е трансдюсер, получен от T с начално състояние q, чиято дефиниционно област е ограничена до $\Sigma^*\Omega$. $T_f(q) \circ id(\Omega) \restriction \Sigma^*\Omega$ е трансдюсерът, получен от T с начално състояние q, в който всички Ω-преходи са отстранени, а от всяко финално състояние на T са добавени преходи с всяко $\omega \in \Omega$ към едно единствено финално състояние, с етикети $\phi(q) \circ w$.

Формално това може да се дефинира така:

$$T(q) \restriction \Sigma^*\Omega =< (\Sigma \cup \Omega) \times \Xi^*, Q, q, \delta, \lambda, \phi' >, \text{ където } \phi'(p) = \varepsilon, \forall p \in Q.$$
$$T_f(q) \restriction \Sigma^*\Omega =< (\Sigma \cup \Omega) \times \Xi^*, Q, q, \delta, \lambda, \phi > \restriction \Sigma^*\Omega$$

$$T_f(q) \circ id(\Omega) \restriction \Sigma^*\Omega =< (\Sigma \cup \Omega) \times \Xi^*, Q \cup \{f\}, q, \delta', \lambda', \phi'' >,$$
$$\text{където} f \notin Q \text{ е ново състояние}$$
$$\delta'(p, \sigma) = \begin{cases} \delta(p, \sigma) \text{ ако } \sigma \in \Sigma \\ f, \text{ ако } !\phi(p) \& \sigma \in \Omega \\ \neg! \text{иначе.} \end{cases}$$
$$\lambda'(p, \sigma) = \begin{cases} \lambda(p, \sigma), \text{ ако } \sigma \in \Sigma \\ \phi(p) \circ \sigma \text{ ако } !\phi(p) \& \sigma \in \Omega \\ \neg! \text{ иначе.} \end{cases}$$
$$\phi''(p) = \begin{cases} \varepsilon \text{ ако } p = f, \\ \neg! \text{ иначе.} \end{cases}$$

Дефиниция 31 $count : (Pref(\Omega) \times (\Sigma \cup \Omega)^*) \cup ((\Sigma \cup \Omega)^* \times \{\varepsilon\}) :\to \mathbb{Z}$

се дефинира рекурсивно по първия аргумент:

$$count(\varepsilon, \beta) = -1$$

$$count(\alpha P_1(\beta), S_1(\beta)) = \begin{cases} count(\alpha, \beta) - 2 \text{ ако } count(\alpha, \beta) > 1 \\ \\ -1 \text{ ако } count(\alpha, \beta) \leq 1 \text{ и} \\ longest_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta)) \text{ не е дефинирана,} \\ \\ 2|v|_\Omega \text{ ако } count(\alpha, \beta) \leq 1 \\ < \alpha P_1(\beta), v, w >= longest_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta) \\ \text{ и } v \in (\Sigma \cup \Omega)^* \Sigma^+ \\ \\ 2|v|_\Omega - 1 \text{ ако } count(\alpha, \beta) \leq 1 \\ < \alpha P_1(\beta), v, w >= longest_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta) \\ \text{ и } v \in (\Sigma \cup \Omega)^* \Omega \end{cases}$$

Идеята на тази функция е да брои колко символа от Ω още трябва да заместим преди да проверим за следващо срещане на правилото $\mathcal{R}$. Тя разграничава двата случая, когато трябва да бъде заместен последния маркер и когато това е нежелателно.

Дефиниция 32 *Дефинираме функцията state* : $(Pref(\Omega) \times (\Sigma \cup \Omega)^*) \cup (\Sigma \cup \Omega)^*) \times \{\varepsilon\} :\rightarrow Q \cup \{q_\varnothing\}$, *където* $q_\varnothing \notin Q$.

$$state(\varepsilon, \beta) = q_\varnothing$$

$$state(\alpha P_1(\beta), S_1(\beta)) = \begin{cases} \delta^*(state(\alpha, \beta), P_1(\beta)) \text{ ако } count(\alpha, \beta) > 1 \\ s \text{ ако } count(\alpha, \beta) \leq 1, \text{ но } count(\alpha P_1(\beta), S_1(\beta)) \geq 0 \\ q_\varnothing \text{ иначе.} \end{cases}$$

Дефиниция 33 *Накрая дефинираме и фамилия от трансдюсери.*

Нека α, β са две думи и $q = state(\alpha, \beta)$. Тогава:

$$T_{[\alpha][\beta]} = \begin{cases} T(q) \upharpoonright \Sigma^*\Omega \text{ ако } q \in Q \text{ и } count(\alpha, \beta) > 1 \\ T_f(q) \upharpoonright \Sigma^*\Omega \text{ ако } q \in Q \text{ и } count(\alpha, \beta) = 1 \\ T_f(q) \circ id(\Omega) \upharpoonright \Sigma^*\Omega \text{ ако } q \in Q \text{ и } count(\alpha, \beta) = 0 \\ id \upharpoonright \Sigma^*\Omega \text{ ако } q = q_\varnothing \end{cases}$$

При тези означения е в сила следното:

Твърдение 9 *Ако $count(\alpha P_1(\beta), S_1(\beta)) = -1$ или ако $count(\alpha P_1(\beta), S_1(\beta)) \neq count(\alpha, \beta) - 2$, то:*

$$\prod_{k=1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) = replace_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta)).$$

Доказателство: Доказателството е с индукция по $|\beta|_\Omega$. Ако $|\beta|_\Omega =$ 0 няма какво да доказваме.

В противен случай трябва да разгледаме двете възможности:

- $count(\alpha P_1(\beta), S_1(\beta)) = -1$. Това е възможно само ако $longets_{\mathcal{R}}(\alpha P_1(\beta), \S_1(\beta))$ не е дефинирана. Тогава тъй като $L \subset (\Sigma \cup \Omega)^*\Omega$, то за всяко ta представка на $I_2(\beta)$

  $$replace(\alpha P_1(\beta)t, t^{-1}S_1(\beta)) = a \circ replace(\alpha P_1(\beta)ta, (ta)^{-1}S_1(\beta)),$$

  откъдето следва, че:

  $$replace_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta)) = I_2(\beta) replace_{\mathcal{R}}(\alpha P_2(\beta), S_2(\beta)),$$

  Сега твърдението следва като използваме индукционната хипотеза и това, че при $count(\alpha P_1(\beta), S_1(\beta)) = -1$ имаме, че $state(\alpha P_1(\beta), \S_1(\beta)) = q_\varnothing$.

- Нека сега $count(\alpha P_1(\beta), S_1(\beta)) > -1$, но $count(\alpha P_1(\beta), S_1(\beta)) \neq count(\alpha, \beta) - 2$. От дефиницията на *count* това е възможно само ако имаме срещане на правилото $< \alpha P_1(\beta), v', w' >$. Нека

$longest_{\mathcal{R}}(\alpha P_1(\beta), S_1(\beta)) < \alpha\ P_1(\beta), v, w >$ е най-дългото такова. Тогава:

$$count(\alpha P_1(\beta), S_1(\beta)) = \begin{cases} 2|v|_\Omega, \text{ ако } v \in (\Omega \cup \Sigma)^*\Sigma^+ \\ 2|v|_\Omega - 1 \text{ иначе} \end{cases}$$
$$state(\alpha P_1(\beta), S_1(\beta)) = s.$$

Да означим $q_1 = s$ и $q_{i+1} = \delta^*(q_i, I_{i+1}(\beta))$. Тогава лесно се вижда, че $state(\alpha P_i(\beta), S_i(\beta) = q_i$ и също така, че $count(\alpha P_i(\beta), S_i(\beta)) = 2|v|_\Omega - 1 - 2(i-1)$ ако v завършва на символ от Ω и $count(\alpha P_i(\beta), S_i(\beta)) = 2|v|_\Omega - 2(i-1)$ иначе. Оттук следва, че ако v завършва на символ от Ω, то:

$$\prod_{k=1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\prod_{k=1}^{|v|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) \prod_{k=|v|+1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\prod_{i=1}^{|v|_\Omega} \lambda^*(q_i, I_{i+1}(\beta)) \prod_{k=|v|+1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\lambda^*(s, v) \prod_{k=|v|+1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$T(v) \prod_{k=|v|+1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)).$$

В алтернативния случай получаваме, че $vb = \prod_{k=2}^{|v|_\Omega+1} I_k(\beta)$, защото w започва със символ от Ω. Тогава:

$$\prod_{k=1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\prod_{k=1}^{|v|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) \prod_{k=|v|_\Omega+1}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\prod_{i=1}^{|v|_\Omega} \lambda^*(q_i, I_{i+1}(\beta)) T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) \prod_{k=|v|_\Omega+2}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$\lambda^*(s, v) b \prod_{k=|v|_\Omega+2}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta)) =$$
$$T(v) b \prod_{k=|v|_\Omega+2}^{|\beta|_\Omega} T_{[\alpha P_k(\beta)][S_k(\beta)]}(I_{k+1}(\beta))$$

Остана да забележим, че във втория случай:

$$replace_{\mathcal{R}}(\alpha P_1(\beta) v, b S_{|v|_\Omega+1}(\beta)) = b \circ replace_{\mathcal{R}}(\alpha P_1(\beta) v, S_{|v|_\Omega+1}(\beta)),$$

защото $v \in (\Sigma \cup \Omega)^* \Sigma^+$ и следователно $\alpha P_1(\beta) v \notin \Sigma^* L$. И в двата случая $count(\alpha P_{|v|_\Omega+1}(\beta), S_{|v|_\Omega+1}(\beta))$ и $count(\alpha P_{|v|_\Omega}(\beta), S_{|v|_\Omega}(\beta))$ удоврлетворяват предпоставките на индукционната хипотеза, което завършва доказателството.

Сега идеята е да построим представяне с трансдюсери точно $T_{[\alpha][\beta]}$. Това ни подсказва да дефинираме релацията $\sim_1$ по следния начин:

$$(\alpha', \beta') \sim_1 (\alpha'', \beta'')$$

точно когато:

1. $\alpha' \sim_{(\Sigma \cup \Omega)^* L} \alpha''$
2. $state(\alpha', \beta') = state(\alpha'', \beta'')$
3. $count(\alpha', \beta') = count(\alpha'', \beta'')$.

Ясно е, че за дефиницията на трансдюсерите е достатъчно само условие 2, но за неговата дефиниция ние използвахме *count*, а за дефиницията на *count* от значение е дали $\alpha \in (\Sigma \cup \Omega)^* L$.

Всяко от условията дефинира релация на еквивалентност. При това релациите, определени от 1 и 2 очевидно са крайни, а от това, че всяка дума $\alpha \in E$ има неповече от $2e$ символа от Ω следва, че $-1 \leq count(\alpha', \beta') \leq 2e$. Така заключаваме, че и 3 задава крайна релация на еквивалентност и следователно $\sim_1$ е крайна релация на еквивалентност.

Остава да построим и крайна дясно-инвариантна релация, която да определя $\sim_1$. Това правим по следния начин. Казваме, че

$$\beta' \sim_2 \beta''$$

тогава и сама тогава, когато:

1. за всеки $0 \leq i \leq e$ за всяко $\gamma \in (\Sigma \cup \Omega)^*$

$$S_1(\beta')\gamma \in (E \cap (\Sigma^* \Omega)^i \Omega) R (\Sigma \cup \Omega)^* \Leftrightarrow$$
$$S_1(\beta'')\gamma \in (E \cap (\Sigma^* \Omega)^i \Omega) R (\Sigma \cup \Omega)^*$$

и

$$S(\beta')\gamma \in (E \cap (\Sigma^* \Omega)^i \Sigma^+) R (\Sigma \cup \Omega)^* \Leftrightarrow$$
$$S_1(\beta'')\gamma \in (E \cap (\Sigma^* \Omega)^i \Sigma^+) R (\Sigma \cup \Omega)^*$$

2. За всеки $\omega' \sim_{(\Sigma\cup\Omega)^*L} \omega''$ е в сила, че $\omega' P_1(\beta') \sim_{(\Sigma\cup\Omega)^*L} \omega'' P_1(\beta'')$.

3. За всяко $q \in Q$:

$$\delta^*(q, P_1(\beta')) = \delta^*(q, P_1(\beta'')).$$

Лема 17 *Релацията $\sim_2$ е Ω-крайна, дясно-инвариантна релация на еквивалентност, която Ω-определя $\sim_1$.*

Доказателство: Тъй като релацията $\sim_{(\Sigma\cup\Omega)^*L}$ е дясно-инвариантна, то условието 2 задава релация на еквивалентност. Сега е ясно, че всяко от условията 1-3 за $\sim_2$ определя дясно-инвариантна, крайна релация на еквивалентност. Тогава и самата $\sim_2$ като тяхна конюнкция се явява крайна, дясна-инвариантна релация на еквивалентност. Ще покажем, че за всеки $\beta\gamma \in (\Sigma\cup\Omega)^*$, за което $|\beta|_\Omega > e+r$ е в сила, че

$$\beta \sim_2 \beta\gamma$$

Тъй като $r \geq 1$, то $|\beta|_\Omega \geq 1$. Тогава $P_1(\beta\gamma) = P_1(\beta)$, от където условията 2 и 3 следват непосредствено. Условието 1 също е тривиално в едната посока, тоест ако

$$S_1(\beta) \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma\cup\Omega)^* \Rightarrow$$
$$S_1(\beta)\gamma \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma\cup\Omega)^*$$

за всяко γ. Да допуснем, че

$$S_1(\beta) \notin (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma\cup\Omega)^* \text{ но}$$
$$S_1(\beta)\gamma \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma\cup\Omega)^*$$

Тогава за някоя представка α на $S_1(\beta)\gamma$ ще бъде изпълнено, че е от езика:

$$(E \cap (\Sigma^*\Omega)^i\Omega)R.$$

Но всяка такава дума има дължина $|\alpha|_\Omega \leq r + i \leq r + e = |S_1(\beta)|_\Omega$. Това означава, че α е представка и на $S_1(\beta)$ и следователно,

$$S_1(\beta) \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma\cup\Omega)^*,$$

което е противоречие с нашето допускане. Аналогично се доказва, че за $|\beta| > e + r$ е изпълнено, че:

$$\beta \in (E \cup (\Sigma^*\Omega)^i\Sigma^+)R(\Sigma \cup \Omega)^* \Leftrightarrow \beta\gamma \in (E \cup (\Sigma^*\Omega)^i\Sigma^+)R(\Sigma \cup \Omega)^*.$$

С това, тъй като всяка от релациите е дясно-инвариантна, твърдението следва директно.

Преминаваме към доказателството на втората част от лемата. Именно, ще покажем, че за всеки α', α'', β' и β'' в сила импликацията:

$$\begin{gathered}(\alpha', \beta') \sim_1 (\alpha'', \beta'') \& \beta' \sim_2 \beta'' \Leftarrow \\ (\alpha' P_1(\beta), S_1(\beta)) \sim_1 (\alpha'' P_1(\beta''), S_1(\beta'')\end{gathered}$$

1. Това, че $\alpha' P_1(\beta') \sim_{(\Sigma\cup\Omega)^*L} \alpha'' P_1(\beta'')$ следва от факта, че $\alpha' \sim_{(\Sigma\cup\Omega)^*L} \alpha''$ и от условието 2 на $\sim_2$, според което:

$$\forall \omega' \sim_{(\Sigma\cup\Omega)^*L} \omega'' \Rightarrow \omega' P_1(\beta') \sim_{(\Sigma\cup\Omega)^*L} \omega'' P_1(\beta).$$

2. Ако $count(\alpha', \beta') > 1$, то получаваме:

$$\begin{gathered}count(\alpha' P_1(\beta'), S_1(\beta')) = count(\alpha', \beta') - 2 = \\ count(\alpha'', \beta'') - 2 = count(\alpha'' P_1(\beta''), S_1(\beta''))\end{gathered}$$

   Аналогично:

$$\begin{aligned}state(\alpha' P_1(\beta'), S_1(\beta')) = \delta^*(state(\alpha', \beta'), P_1(\beta')) \\ \delta^*(state(\alpha'', \beta''), P_1(\beta')),\end{aligned}$$

   защото $state(\alpha', \beta') = state(\alpha'', \beta'')$. Накрая тъй като $\beta' \sim_2 \beta''$, то от 3 на $\sim_2$ заключаваме, че:

$$\delta^*(state(\alpha'', \beta''), P_1(\beta')) = \delta^*(state(\alpha', \beta'), P_1(\beta'')) = state(\alpha'' P_1(\beta''), S_1(\beta'')).$$

   Сега като обединим последните две равенства получаваме, че

$$state(\alpha' P_1(\beta'), S_1(\beta')) = state(\alpha'' P_1(\beta''), S_1(\beta'')).$$

Така проверихме, че

$$(\alpha' P_1(\beta'), S_1(\beta')) \sim_1 (\alpha'' P_1(\beta''), S_1(\beta''))$$

в случай, че $count(\alpha', \beta') > 1$.

Нека $count(\alpha', \beta') \leq 1$. Тогава и $count(\alpha'', \beta'') \leq 1$. Както вече видяхме $\alpha' P_1(\beta') \sim_{(\Sigma\cup\Omega)^* L} \alpha'' P_1(\beta'')$. В частност или $\alpha' P_1(\beta')$ и $\alpha'' P_1(\beta'')$ едновременно принадлежат на $(\Sigma \cup \Omega)^* L$, или и двете не принадлежат на $(\Sigma \cup \Omega)^* L$. Във втория случай не е възможно срещане на правилото нито в $(\alpha' P_1(\beta'), S_1(\beta'))$, нито в $(\alpha'' P_1(\beta''), S_1(\beta''))$. Тогава

$$count(\alpha' P_1(\beta'), S_1(\beta')) = -1,$$
$$count(\alpha'' P_1(\beta''), S_1(\beta'') = -1$$

и също така

$$state(\alpha' P_1(\beta'), S_1(\beta')) = q_\varnothing$$
$$state(\alpha'' P_1(\beta''), S_1(\beta'')) = q_\varnothing.$$

Оттук получаваме, че:

$$(\alpha' P_1(\beta'), S_1(\beta')) \sim_1 (\alpha'' P_1(\beta''), S_1(\beta'')).$$

Остана да разгледаме случая, когато $\alpha' P_1(\beta')$ и $\alpha'' P_1(\beta'')$ са коректен ляв контекст на правилото. Тогава дали има срещане на правилото $\mathcal{R}$ се определя от това дали съществува i, за което $S_1(\beta') = v'w'$ с $v' \in E$ $w' \in R(\Sigma \cup \Omega)^*$ и $|v'|_\Omega = i$. От условието за регулярния език E следва, че $i \leq e$. Сега тъй като $\beta' \sim_2 \beta''$, от условие 1 съществува разбиване на $S_1(\beta') = v'w'$ с горните условия, точно когато съществува разбиване на $S_1(\beta'') = v''w''$, изпълняващо същите условия. При това v' завършва на символ от Ω точно когато и v'' има това свойство. Сега като приложим тези наблюдения за най-голямото i, за което такова разбиване съществува получаваме, че:

$$count(\alpha' P_1(\beta'), S_1(\beta')) = count(\alpha'' P_1(\beta''), S_1(\beta''))$$

откъдето и

$$state(\alpha' P_1(\beta'), S_1(\beta')) = state(\alpha'' P_1(\beta''), S_1(\beta''))$$

С това лемата е доказана.

Като следствие от тази лема получаваме следната:

Теорема 13 *За всяко правило с ограничено срещане на маркери* $\mathcal{R}$, $E \to T(E) \backslash L\_R$ *функцията* $rewrite_{\mathcal{R}}$ *е* Ω*-крайнопредставима функция. При това компонентите на функцията, която представя* $rewrite_{\mathcal{R}}$ *могат да бъдат намерени алгоритмично.*

Доказателство: Дефинираме $f : Pref(\Omega) \times (\Sigma \cup \Omega)^* \cup (\Sigma \cup \Omega)^* \times \{\varepsilon\}$ Ω-крайнопредставима функция чрез релациите $\sim_1$, $\sim_2$ и равенствата:

$$f(\varepsilon, \alpha) = \varepsilon$$
$$f(\alpha, \beta) = T_{[\alpha][\beta]}(P_1(\beta))f(\alpha P_1(\beta), S_1(\beta)).$$

Лесно се вижда, че $count(\varepsilon, \alpha) = -1$ за всяко α и освен това:

$$f(\varepsilon, \alpha) = \prod_{k=0}^{|\alpha|_\Omega} T_{[P_k(\alpha)][S_k(\alpha)]}(I_{k+1}(\alpha)).$$

Тогава от твърдение 9 получаваме, че:

$$f(\varepsilon, \alpha) = replace_{\mathcal{R}}(\varepsilon, \alpha).$$

Сега твърдението следва непосредствено от дефиницията на $rewrite$:

$$rewrite_{\mathcal{R}}(t) = replace_{\mathcal{R}}(\varepsilon, t)$$

и дефиницията за Ω-определима функция. С това теорема 13 е доказана.

6.2 Представяне на регулярни правила за заместване чрез FIFO-трансдюсери. Достатъчни условия

В този параграф ще се спрем на въпроса, как да трансформираме дадено регулярно правило за заместване в еквивалентно на него правило с маркери. Крайната ни цел е да отделим клас от правила, които ще можем да разпознаваме с помощта на FIFO-трансдюсер. Това означава, че освен конструкцията за правила с ограничен брой маркери, можем да използваме и композиция отляво и отдясно с подпоследователни трансдюсери.

И така нека $\mathcal{R}$, $E \to T(E)\backslash L\_R$ е регулярно правило за заместване над крайна азбука Σ. Идеята ни е да представим функцията на заместване, дефинирана от $\mathcal{R}$ като композиция от функция f_1, представена от подпоследователен трансдюсер, функция на заместване, породена от правило с ограничен брой маркери $\mathcal{R}_\Omega$ и функция f_2, също представена чрез подпоследователен трансдюсер. Тук с Ω сме означили крайно множество от маркери, които не са в Σ. Нека с $h : (\Sigma \cup \Omega)^* \to \Sigma^*$ означим хомоморфизма, породен от:

$$h(\sigma) = \begin{cases} \sigma, & \sigma \in \Sigma \\ \varepsilon, & \sigma \in \Omega. \end{cases}$$

Ролята на f_1 ще бъде единствено да поставя маркери, тоест:

$$h(f_1(t)) = t,$$

за всяка дума $t \in \Sigma^*$. Ролята на f_2 ще бъде да отстранява маркерите, тоест $f_2 = h$. Да забележим, че h може да се представи посредством подпоследователен преобразувател, така че изискването за f_2 е изпълнено. От друга страна съществуват безброй много начини да изберем f_1. Независимо, обаче, от избора на f_1 съществува универсално правило $\mathcal{U}(\mathcal{R})$ което може да се представи като:

$$E_U \to T(h(E_U))\backslash L_U\_R_U,$$

със свойството:

$$rewrite_{\mathcal{R}}(t) = h(rewrite_{\mathcal{U}(\mathcal{R})}(f_1(t))).$$

Наистина достатъчно е да положим:

$$E_U = \{u \,|\, h(u) \in E\}$$
$$L_U = \{u \,|\, h(u) \in L\}$$
$$R_U = \{u \,|\, h(u) \in R\}$$

Тогава лесно се вижда, че са в сила следните твърдения:

Твърдение 10 *1. Ако $u \in L_U$ $/R_U, E_U/$, то $h(u) \in L$, съответно $/R, E/$.*

2. Ако $< u, v, w >$ е най-дългото срещане на правилото $\mathcal{R}(\mathcal{U})$ в $< u, vw >$, то $< h(u), h(v), h(w) >$ е най-дългото срещане на правилото $\mathcal{R}$ в $< h(u), h(v)h(w) >$.

3. $h(replace_{\mathcal{U}(\mathcal{R})}(u, v)) = replace_{\mathcal{R}}(h(u), h(v))$.

Доказателство: Първото твърдение следва непосредствено от дефиницията на h и множествата L_U, съответно $/R_U, E_U/$.

За твърдение 2 можем да разсъждаваме така. Първо щом $< u, v, w >$ е срещане на правилото $\mathcal{U}(\mathcal{R})$, то прилагайки твърдение 1 получаваме, че $< h(u), h(v), h(w) >$ е срещане на правилото $\mathcal{R}$. Да допуснем, че съществува по-дълго срещане на $\mathcal{R}$ в $< h(u), h(v)h(w) >=< h(u), h(vw) >$. Нека например $< h(u), \alpha, \beta >$ е едно такова срещане. Тъй като $|h(a)| \leq 1$, за всяка буква $a \in \Sigma \cup \Omega$ и $h(v) \prec \alpha \prec h(vw)$, то съществува $v \prec v_1 \prec vw$ със свойството $h(v_1) = \alpha$ съответно $h(v_1^{-1}(vw)) = h(w_1)$. Това показва, че $< u, v_1, v_1^{-1}(vw) >$ също е срещане на правилото $\mathcal{U}(\mathcal{R})$, за което $|v_1| > |v|$. Това обаче е противоречие с допускането, че $< u, v, w >$ е най-дългото.

Сега твърдение 3 следва лесно с индукция по дължината на думата v.

При $v = \varepsilon$ имаме, че:

$$h(replace_{\mathcal{U}(\mathcal{R})}(u, \varepsilon)) = h(\varepsilon) = \varepsilon$$
$$replace_{\mathcal{R}}(h(u), h(\varepsilon)) = replace_{\mathcal{R}}(h(u), \varepsilon) = \varepsilon,$$

което показва твърдението в базовия случай.

Нека твърдението е вярно за всяка дума v с дължина по-малка от n. Нека $u \in (\Sigma \cup \Omega)^*$, $a \in \Sigma \cup \Omega$ са произволни и нека $v = av' \in (\Sigma \cup \Omega)^*$ е с дължина $n + 1$. Тогава ако имаме най-дълго срещане $< u, x, w >$ на правилото $\mathcal{U}(\mathcal{R})$ в (u, v), то от твърдение 2 следва, че $< h(u), h(x), h(w) >$ е най-дълго срещане на правилото $\mathcal{R}$ в $(h(u), h(v))$.

Това означава, че:

$$h(replace_{\mathcal{U}(\mathcal{R})}(u,v)) = h(T(h(x)) \circ replace_{\mathcal{U}(\mathcal{R})}(ux,w)) = \\ T(h(x)) \circ h(replace_{\mathcal{U}(\mathcal{R})}(ux,w))$$

От друга страна:

$$replace_{\mathcal{R}}(h(u),h(v)) = T(h(x)) \circ replace_{\mathcal{R}}(h(u)h(x),h(v)).$$

Сега тъй като $h(ux) = h(u)h(x)$ и $x \neq \varepsilon$ можем да приложим индукционното предположение.

Ако нямаме срещане на $\mathcal{U}(\mathcal{R})$ в (u,v), то:

$$h(replace_{\mathcal{U}(\mathcal{R})}(u,av')) = h(a) \circ h(replace_{\mathcal{U}(\mathcal{R})}(ua,v')).$$

В този случай нямаме срещане на правилото $\mathcal{R}$ $(h(u),h(av')$. В зависимост от a имаме два случая. В първия $a \in \Sigma$. Тогава $h(a) = a$ и също така:

$$replace_{\mathcal{R}}(h(u),h(a)h(v')) = a \circ replace_{\mathcal{R}}(h(u)h(a),h(v')), \\ h(replace_{\mathcal{U}(\mathcal{R})}(u,av')) = a \circ h(replace_{\mathcal{U}(\mathcal{R})}(ua,v')).$$

Тогава можем да приложим индукционното предположение за v' , откъдето получаваме твърдението и за v. Във втория случай $a \in \Omega$ и тогава $h(a) = \varepsilon$. Следователно $h(ua) = h(u)$ и $h(av') = h(v')$. Тогава е ясно, че:

$$h(replace_{\mathcal{U}(\mathcal{R})}(u,av')) = h(replace_{\mathcal{U}(\mathcal{R})}(ua,v'), \\ replace_{\mathcal{R}}(h(u),h(av')) = replace_{\mathcal{R}}(h(ua),h(v')).$$

Сега отново сме в положение да приложим индукционната хипотеза за v', от където получаваме верността на твърдениеато и за v.

Сега като вземем под внимание, че:

$$h(f_1(t)) = t,$$

и приложим твърдение 3, получаваме, че:

$$h(replace_{\mathcal{U}(\mathcal{R})}(\varepsilon,f_1(t))) = replace_{\mathcal{R}}(\varepsilon,t_1).$$

Това непосредствено води и до:

$$h(rewrite_{\mathcal{U}(\mathcal{R})}(f_1(t))) = rewrite_{\mathcal{R}}(t).$$

За съжаление обаче, $\mathcal{U}(\mathcal{R})$ не е правило с краен брой маркери, за да можем да построим за него FIFO-трансдюсер. Всъщност то не спазва нито едно от условията за правило с краен брой маркери. Действително изправени сме пред следните проблеми:

1. В общия случай $T \circ h$ не е подпоследователен трансдюсер.

2. Не е задължително всяка дума от фокуса да е оградена с маркери.

3. Възможно е за всяко k да съществува дума от $E_U R_U$ с повече от k маркера.

За да се справим с тези проблеми ще ни се наложи да ограничим правилата $\mathcal{R}$.

Първият проблем решаваме тривиално като оттук нататък ще изискваме, T да може да се представи с подпоследователен трансдюсер. Тогава и композицията $T \circ h$ се представя с подпоследователен трансдюсер.

Идеята за конструкцията на правило с краен брой маркери $\mathcal{R}_\Omega$ и функция f_1 се основава на следното твърдение, което в известен смисъл обобщава твърдение 10. Всъщност не са ни необходими целите множества L_U, E_U и R_U, а само техни подмножества L_Ω, E_Ω и R_Ω, стига f_1 да запазва срещанията на множествата и да гарантира тяхната еднозначност в известен смисъл. Тогава композицията на $f_2 \circ rewrite_{\mathcal{R}_\Omega} \circ f_1$ е еквивалентна на $rewrite_R$. Формално това може да се формулира така:

Твърдение 11 *Нека $\mathcal{R}$ $E \to T(E)\backslash L\_R$ е правило за заместване, а $f_1 : \Sigma^* \to (\Sigma \cup \Omega)^*$ има свойствата, че:*

$$\forall t \in \Sigma^* h(f_1(t)) = t \text{ и} \tag{25}$$

$$Range(f_1) \cap (\Sigma \cup \Omega)^* \Omega^2 (\Sigma \cup \Omega)^* = \varnothing. \tag{26}$$

Нека още $L_\Omega \subset L_U$*,* $E_\Omega \subset E$ *и* $R \subset R_\Omega$ *задават правилото* $\mathcal{R}_\Omega$*:*

$$E_\Omega \to T(h(E_\Omega)) \backslash L_\Omega\_R_\Omega.$$

като $L_\Omega \subset (\Sigma \cup \Omega)^* \Omega$*. Ако за всяка срещане* $< u, v, w >$ *на правилото* $\mathcal{R}$ *съществува срещане* $< u', v', w' >$ *на* $\mathcal{R}_\Omega$ *със свойствата:*

$$\begin{aligned} f(uvw) &= u'v'w', \\ h(u') &= u, \\ h(v') &= v, \\ h(w') &= w. \end{aligned}$$

то са в сила следните твърдения:

1. *Нека* $f_1(uvw) = u'v'w'$ *и* $h(u') = u$*,* $h(v') = v$*,* $h(w') = w$*. Тогава ако* $< u', v', w' >$ *е най-дългото срещане на* $\mathcal{R}_\Omega$ *относно* $(u', v'w')$*, то* $< u, v, w >$ *е най-дългото срещане на* $\mathcal{R}$ *спрямо* (u, vw)*.*

2. *За всеки* $u, v \in \Sigma^*$ *и* $u', v' \in (\Sigma \cup \Omega)^*$*, за които* $f_1(uv) = u'v'$ *е в сила равенството:*

$$h(replace_{\mathcal{R}_\Omega}(u', v') = replace_R(u, v).$$

3. *За всяко* $t \in \Sigma^*$*.*

$$h(rewrite_{\mathcal{R}_\Omega}(f_1(t))) = rewrite_{\mathcal{R}}(t).$$

Доказателство: Нека $< u', v', w' >$ е най-дългото срещане на $\mathcal{R}_\Omega$ относно $(u', v'w')$. Тогава $u' = y'a$, където $a \in \Omega$. Тъй като $L_\Omega \subset L_U$, $E_\Omega \subset E_U$ и $R_\Omega \subset R_U$ от дефиницията на h следва, че $< h(u'), h(v'), h(w') >$ е срещане на $\mathcal{R}$ в $(h(u'), h(v'w'))$. Това означава, че $< u, v, w >$ е срещане на правилото $\mathcal{R}$ в (u, vw). Да допуснем, че то не е най-дълго. Тогава съществува срещане $< u, \alpha, \beta >$ на $\mathcal{R}$ в (u, vw) с $|\alpha| > |v|$. Тъй като $u'v'w' = f_1(u\alpha\beta)$ от условието за f_1 получаваме, че съществуват x', α' и β', за които $h(x') = u$, $h(\alpha') = \alpha$, $h(\beta') = \beta$ и $< x', \alpha', \beta' >$ е срещане в $< x', \alpha'\beta' >$.

Първо ще покажем, че $x' = u'$. Наистина да допуснем, че $x' \neq u'$. Тогава или $x' \prec u'$, или $x' \succ u'$. Освен това, тъй като $x' \in (\Sigma \cup \Omega)^* L_\Omega$, то последната буква на x' трябва да е $\omega \in \Omega$. Нека първо $x' \prec u'$. Тогава имаме, че:

$$h((x')^{-1}u') = h(x')^{-1}h(u') = h(u)^{-1}h(u) = \varepsilon.$$

Сега от дефиницията на h следва, че $(x')^{-1}u' \in \Omega^*$. Тогава е ясно, че $\omega((x')^{-1}u') \in \Omega\Omega^*\Omega$ е подстринг на $f_1(uvw)$, което противоречи на условие 25 за f_1. Така показахме, че $x' \not\prec u'$.

Аналогично се показва, че не е възможно $x' \succ u'$. Иначе получаваме равенството:

$$h((u')^{-1}x') = h(u')^{-1}h(x') = u^{-1}u = \varepsilon.$$

Тогава, както и по горе, получаваме, че $(u'^{-1})x' \in \Omega^*\Omega$. Тогава лесно се забелязва, че $a(u')^{-1}x' \in \Omega\Omega^*\Omega$ е поддума на $f_1(uvw)$, което отново е противоречие с условие 25 за f_1.

С това показахме, че $x' = u'$. Сега имаме, че щом $|h(\alpha')| > |v| = |h(v')|$ и тъй като $|h(\sigma)| \leq 1$ за всеки символ от $\sigma \in \Sigma \cup \Omega$, то $|\alpha'| > |v'|$. Следователно намерихме по-дълго срещане $< u', \alpha', \beta' >$ относно $< u', v'w' >$. Това е противоречие с максималността на $< u', v', w' >$.

Второто твърдение ще покажем с индукция по дължината на v'. В базовия случай, $v' = \varepsilon$, имаме, че:

$$h(replace_{\mathcal{R}_\Omega}(u', \varepsilon)) = h(\varepsilon) = \varepsilon$$
$$replace_{\mathcal{R}}(u, h(\varepsilon)) = replace_{\mathcal{R}}(u, \varepsilon) = \varepsilon.$$

Това показва и верността на твдентието. Да допуснем, че за всеки две думи u', v' с $|v'| \leq n$ и u, v, за които $f_1(uv) = u'v'$ е в сила желаното равенство. Нека $v' = aw'$, където $|w'| = n$. Да допуснем, че $< u', x', y' >$ е най-дългото срещане на $\mathcal{R}_\Omega$ в (u', v'). Тогава от първата част на твърдението следва, че $< h(u'), h(x'), h(y') >$ е най-дългото срещане в $< u, v >$. Тогава лесно получаваме, че:

$$h(replace_{\mathcal{R}_\Omega}(u', v')) = h(T(h(x'))) \circ h(replace_{\mathcal{R}_\Omega}(u'x', y')) =$$
$$T(h(x') \circ h(replace_{\mathcal{R}_\Omega}(u'x', y'))$$

и съответно:

$$replace_{\mathcal{R}}(u, v) = T(h(x')) \circ replace_{\mathcal{R}}(uh(x'), h(y')).$$

Сега твърдението следва като приложим индукционната хипотеза за y'.

Да допуснем, че $< u', av' >$ не допуска срещане на $\mathcal{R}$. Тогава:

$$h(replace_{\mathcal{R}_\Omega}(u', aw')) = h(a) \circ h(replace_{\mathcal{R}_\Omega}(u'a, w'))$$

Имаме два случая. Ако $a \in \Sigma$, то няма срещане на $\mathcal{R}$ в (u, v). Действително в противен случай щяхме да имаме срещане от вида $< u, x, y >$ и от условието за $\mathcal{R}_\Omega$ щяхме да имаме и срещане $\widetilde{u}, x', y'$, където $h(\widetilde{u}) = u$. Сега обаче $h(u'a) = h(u')a = ua$ и следователно $\widetilde{u}$ е представка на u' като $h(\widetilde{u}) = u$. Това е възможно само ако $\widetilde{u} = u'$ или $u' \in \widetilde{u}\Omega^+$. Първият случай е невъзможен, защото тогава щяхме да имаме срещане на $\mathcal{R}_\Omega$ и в (u', v'). Вторият също е недопустим. Наистина от условието за f_1 следва, че $u' = \widetilde{u}\sigma$, където $\sigma \in \Omega$ и $\widetilde{u} \in (\Sigma \cup \Omega)^*\Sigma$. Това обаче означава, че $\widetilde{u} \notin (\Sigma \cup \Omega)^*\Omega$. Следователно $\widetilde{u} \notin (\Sigma \cup \Omega)^* L_\Omega$, откъдето и $< \widetilde{u}, x', y' >$ не може да е срещане на правилото $\mathcal{R}_\Omega$. Тогава получаваме, че:

$$replace_{\mathcal{R}}(u, h(aw')) = replace_{\mathcal{R}}(u, ah(w')) = a \circ replace_{\mathcal{R}}(ua, h(w'))$$

и съответно:

$$h(replace_{\mathcal{R}_\Omega}(u', aw')) = a \circ h(replace_{\mathcal{R}_\Omega}(u'a, w')).$$

Сега можем да приложим индукционната хипотеза за w'.

Накрая ако $a \in \Omega$, то:

$$h(replace_{\mathcal{R}_\Omega}(u', aw')) = h(a)h(replace_{\mathcal{R}_\Omega}(u'a, w')) = h(replace_{\mathcal{R}_\Omega}(u'a, w')).$$

От друга страна е ясно, че $h(u'a) = h(u')h(a) = h(u') = u$ и $v = h(aw') = h(a)h(w') = h(w')$ и следователно отново можем да приложим индукционната хипотеза за w'.

Третата част на твърдението е директно следствие от втората като вземем в предвид, че:

$$h(rewrite_{\mathcal{R}_\Omega}(f_1(t))) = h(replace_{\mathcal{R}_\Omega}(\varepsilon, f_1(t))) =$$
$$replace_{\mathcal{R}}(\varepsilon, t) = rewrite_{\mathcal{R}}(t)$$

След това въведение ще поставим достатъчни условия за правилата $\mathcal{R}$, така че за тях да съществуват функции f_1 и правила $\mathcal{R}_\Omega$, които да удовлетворяват условията на твърдение 11. Допълнително налагаме изискването $\mathcal{R}_\Omega$ да бъде с краен брой маркери Ω и f_1 да се представя чрез подпоследователен трансдюсер. Тъй като функцията T за правилото $\mathcal{R}$ е зададена от подпоследователен преобразувател, то остава да удовлетворим другите две изисквания за правило с краен брой маркери, именно:

1. $L_\Omega \subset (\Sigma \cup \Omega)^*\Omega$, $R \subset \Omega(\Sigma \cup \Omega)^*$ и

2. да намерим константи e и r, за които са в сила импликациите:

$$\alpha \in E_\Omega \Rightarrow |\alpha|_\Omega \leq e$$
$$\beta \in R_\Omega \Rightarrow |\beta|_\Omega \leq r.$$

Условие 1 означава, че всъщност трябва да можем да разделяме левия контекст от фокуса и съответно фокуса от десния контекст. Тъй като това трябва да става независимо от избора на думи в тези три езика, то трябва да подберем функцията f_1 така, че тя да отбелязва всички краища на думи от L или всички възможни начала на думи от E. Симетрично f_1 трябва да поставя маркери или след всички краища на думи от E, или преди всяко начало на дума от R. Ако f_1 реализира тази идея, то тя ще удовлетворява изискванията на твърдение 11.

За даден език $\mathcal{L}$ с $Suf(\mathcal{L})$ и $Pref(\mathcal{L})$ ще означаваме множеството от всички наставки, съответно представки на думи от езика $\mathcal{L}$, тоест:

Дефиниция 34

$$Suf(\mathcal{L}) = \{y \mid \exists x, xy \in \mathcal{L}\}$$
$$Pref(\mathcal{L}) = \{x \mid \exists y, xy \in \mathcal{L}\}.$$

Тогава ние искаме да маркираме подмножества $L_0 \subset Suf(L)$ или $PE_0 \subset Pref(E)$, съответно $SE_0 \subset Suf(E)$ или $R_0 \subset Pref(R)$ със свойствата, че:

$$L \subset \Sigma^* L_0 \text{ или } E \subset PE_0\Sigma^* \text{ и}$$
$$E \subset \Sigma^* SE_0 \text{ или } R \subset R_0\Sigma^*.$$

От друга страна, за да съблюдаваме второто условие за правила с краен брой маркери, думите в тези езици не трябва да се срещат "твърде много" пъти в ER.

Накрая, понеже f_1 трябва да поставя всички маркери в произволен текст $t \in \Sigma^*$, тоест да поставя маркери например пред всяка поддума на t от PE_0 и в същото време f_1 трябва да се представя чрез подпоследователен преобразувател, то ще искаме множествата в последния етап да бъдат крайни.

И така задачата, която си поставяме сега, е следната:

Проблем 1 *Даден е регулярен език $\mathcal{L}$ и регулярени език A. Съществува ли краен език $\mathcal{L}_0 \subset Suf(\mathcal{L})$ със следните свойства:*

1. *$\mathcal{L}_0 \subset Suf(\mathcal{L})$.*
2. *$\mathcal{L} \subset \Sigma^*\mathcal{L}_0$.*
3. *съществува константа k, че за всяка дума $\alpha \in \mathcal{L}_0$ и дума $\beta \in A$ думата α се среща най-много k пъти като поддума на β.*

Ако такъв език съществува, да се намери един такъв език $\mathcal{L}_0$ и за него да се намери подходяща константа k.

Едновременно с това можем да поставим и дуалната задача:

Проблем 2 *Даден е регулярен език $\mathcal{L}$ и регулярени език A. Съществува ли краен език $\mathcal{L}_0 \subset Pref(\mathcal{L})$ със следните свойства:*

1. *$\mathcal{L}_0 \subset Pref(\mathcal{L})$.*

2. $\mathcal{L} \subset \mathcal{L}_0\Sigma^*$.

3. съществува константа k*, че за всяка дума* $\alpha \in \mathcal{L}_0$ *и дума* $\beta \in A$ *думата* α *се среща най-много* k *пъти като поддума на* β.

Ако такъв език съществува, да се намери един такъв език $\mathcal{L}_0$ *и за него да се намери подходяща константа* k.

Ясно е, че ако можем да решаваме първата задача, по симетричен начин ще можем да решаваме и втората.

Като резултат, ще можем да намираме ефективно крайни множества S_0, PE_0, SE_0 и R_0, които се срещат ограничен брой пъти във всяка дума от ER, или да установяваме, че такива множества не съществуват.

За целта ще ни бъдат полезни някои помощни твърдения.

Започваме със следната дефиниция:

Дефиниция 35 *Ако* M *и* N *са две множества от думи, то с* $fin(M, N)$ *означаваме онези думи от* M*, които се срещат ограничен брой пъти в коя да е дума от* N*, по-точно:*

$$fin(M, N) = \{\alpha \in M \mid \exists k, \text{ за което всяка дума } \beta \in N \text{ съдържа неповече от } k \text{ срещания на } \alpha\}.$$

Следващото твърдение показва, как може алгоритмично да се намери множеството $fin(M, N)$ в случая, когато M и N са регулярни.

Твърдение 12 *Нека* M *и* N *са регулярни множества и нека* $A =< \Sigma, Q, s, \delta, F >$ *е автомат, разпознаващ* N*. Нека с* $C(q)$ *означим езика на автомата* $A(q) =< \Sigma, Q, q, \delta, \{q\} >$ *за всяко състояние* $q \in Q$*. Тогава са в сила следните две твърдения:*

1.

$$fin(M, N) = M \cap \overline{\cup_q Pref(C(q))},$$

където обединението е по всички достижими състояния $q \in Q$*, от които се достига състояние от* F.

2. *За всяка дума* $\alpha \in fin(M, N)$ *и всяка дума* $\beta \in N$, β *съдържа неповече от* $|Q||\alpha|$ *срещания на* α.

Доказателство: Първо ще покажем, че всяка $\alpha \in Pref(C(q))$, за което q е достижимо и от q се достига F не е в множеството $fin(M, N)$. Наистина нека $\beta \in C(q)$ е дума с представка α, тоест $\alpha\gamma \in C(q)$. Тъй като q е достижимо, то съществува дума u, за която:

$$\delta^*(s, u) = q.$$

Освен това, от факта, че от q е достижимо F, следва, че съществува дума v, за която:

$$\delta^*(q, v) \in F.$$

Сега нека l е произволно естествено число. Имаме, че:

$$\delta^*(q, \alpha\gamma) = q,$$

от където и:

$$\delta^*(q, (\alpha\gamma)^l) = q.$$

Оттук следва, че:

$$\delta^*(s, u(\alpha\gamma)^l v) = \delta^*(q, (\alpha\gamma)^l v) = \delta^*(q, v) \in F.$$

Това показва, че всяка от думите $u(\alpha\gamma)^l v$ се разпознава от A и следователно е дума от N. Сега непосредствено следва, че $\alpha \notin fin(M, N)$.

Обратно нека

$$\alpha \in M \cap \overline{\cup_q Pref(C(q))},$$

където обединението е по достижимите състояния в A, от които има път до F. Нека $\beta = b_1 b_2 \dots b_n \in N$. Ще покажем, че не съществуват повече от $|Q||\alpha|$ срещания на α в β. Тогава ще следва и втората част на твърдението, а в ролята на k в дефиницията на $fin(M, N)$ ще можем да изберем точно $|Q||\alpha|$.

И така нека $i_1 < i_2 < \dots < i_k$ са всички позиции в β, от които започва срещане на α. Нека с $q_j \in Q$ означим:

$$q_j = \delta^*(s, b_1 \dots b_{i_j - 1}).$$

От дефиницията на q_j следва, че q_j е достижимо от s. Освен това имаме, че:

$$\delta^*(q_j, b_{i_j} \dots b_n) = \delta^*(s, \beta) \in F.$$

Тоест q_j участва в обединението

$$M \cap \overline{\cup_q Pref(C(q))}.$$

В частност получаваме, че:

$$\alpha \notin Pref(C(q_j)).$$

Нека сега $j' \geq j + |\alpha|$. Тогава, тъй като i_j е начало на срещане на α в β, а $i_{j'} \geq i_j + |\alpha|$, то α е представка на $b_{i_j} \dots b_{i'_j}$. Това показва, че

$$q_{j'} = \delta^*(s, b_1 \dots b_{i'_j}) = \delta^*(q_j, b_j \dots b_{i'_j - 1}).$$

Сега тъй като $\alpha \notin Pref(C(q_j)$ следва, че $q_{j'} \neq q_j$. Прилагайки това наблюдение за всеки две от състоянията от $Q' = \{q_{1+r|\alpha|}\}_r$, получаваме, че състоянията $q_{1+r|\alpha|} \neq q_{1+t|\alpha|}$ за $r \neq t$. Следователно тъй като съществуват най-много $|Q|$ на брой различни състояния, то $k < 1 + |Q||\alpha|$, което завършва доказателството.

Твърдение 12 ни дава алгоритъм, по който от всички думи в $Suf(\mathcal{L})$ можем да намерим онези, които се срещат ограничен брой пъти в A. От тях трябва да отделим крайното подмножество $\mathcal{L}_0$, ако такова съществува.

Следващото твърдение ни дава отговор как можем да извършим това ефективно.

Твърдение 13 *Нека L е регулярен език, а $S \subset Suf(L)$ е множество от наставки на L със свойството:*

$$L \subset \Sigma^* S.$$

Тогава са в сила твърденията:

1. *Съществува крайно подмножество $S_0 \subset S$ със свойството:*

$$L \subset \Sigma^* S_0,$$

тогава и само тогава, когато $Suf(S) \backslash \Sigma^* S$ е крайно.

2. *В случай, че $Suf(S)\backslash\Sigma^*S$ е крайно и $l = \max\{|v| \mid v \in Suf(S)\backslash\Sigma^*S\}$, то*

$$S_0 = \{u \in S \mid |u| \leq l+1\}$$

има свойството, че:

$$L \subset \Sigma^* S_0.$$

Доказателство: Да допуснем първо, че $Suf(S)\backslash\Sigma^*S$ е крайно и нека l и S_0 са както във втората част на твърдението. Да разгледаме произволна дума $\alpha \in L$. Тъй като, по условие, $L \subset \Sigma^*S$, то съществуват думи u и $v \in S$ със свойството $\alpha = uv$. Сега ако $|v| \leq l+1$, то по дефиницията на S_0 имаме, че $v \in S_0$ и следователно $uv \in \Sigma^*S_0$. В противен случай $v = w_1w_2$ има наставка w_2 с дължина $l+1$. Тъй като $w_2 \in Suf(S)$ и $|w_2| > l$, то $w_2 \in \Sigma^*S$. Тогава $w_2 = st$, където $t \in S$. Тъй като $|w_2| = l+1$, то и $|t| \leq l+1$, тоест $t \in S$ и $|t| \leq l+1$, което означава, че $t \in S_0$. Окончателно получаваме, че:

$$\alpha = uw_1st \in \Sigma^*S_0.$$

Обратно да допуснем, че $Suf(S)\backslash\Sigma^*S$ е безкрайно, но въпреки това съществува крайно подмножество $S_0 \subset S$, за което:

$$L \subset \Sigma^*S_0.$$

Нека $m = \max\{|v| \mid v \in S_0\}$. Тогава тъй като $Suf(S)\backslash\Sigma^*S$ е безкрайно, то съществува $v \in Suf(S)$, за което $v \notin \Sigma^*S$ и $|v| > m$. Тъй като $S_0 \subset S$ това означава, че $v \notin \Sigma^*S_0$. Сега v е наставка в S, което означава, че съществува u, за което $uv \in S$. Но $S \subset Suf(L)$, следователно съществува дума $\alpha = xuv \in L$. Тъй като, по допускане, $L \subset \Sigma^*S_0$, то съществува дума $s \in S_0$, която е наставка на xuv. Но $|s| \leq m$ и следователно s е наставка на v. Това показва, че $v \in \Sigma^*S_0 \subset \Sigma^*S$. Последното обаче е противоречие с факта, че $v \notin \Sigma^*S$.

Това показва, че допускането не е вярно, с което твърдението е доказано.

Ще отбележим, че всъщност наличието на крайно множество S_0 със свойствата $\Sigma^*S = \Sigma^*S_0$, както лесно следва от доказателството на твърдение 13, показва, че Σ^*S, са езици свободни от звезди - свойство, изследвано от Schuetzenberger [Sch65]. За разлика от общия случай, разглеждания от нас случай е значително по-прост за разрешаване от алгоритмична гледна точка.

Сега сме готови да опишем алгоритъма, който разрешава поставения от нас проблем 1 за крайно множество от наставки на даден регулярен език, които се срещат ограничен брой пъти във всяка от думите на друг регулярен език A.

Именно имаме следното:

Твърдение 14 *Проблемът 1 е алгоритмично разрешим.*

Действително, нека $\mathcal{L}$ и A са регулярни езици, зададени с крайни автомати. Първо построяваме автомат S за езика $Suf(\mathcal{L})$. След това от автомата за A и за S построяваме автомат за езика:

$$fin(S, A) = S \cap \overline{\cup_q Pref(C(q))},$$

Според твърдение 13 това е автомат точно за онези наставки на $\mathcal{L}$, които се срещат ограничен брой пъти в A. Сега проверяваме дали:

$$\mathcal{L} \backslash \Sigma^* fin(S, A) = \varnothing.$$

Това отново е автоматна конструкция. При това ако тази разлика не дава празното множество, то съществува дума в $\mathcal{L}$, която се среща неограничен брой пъти в думи от A и съответно множество $\mathcal{L}_0$ с желаните свойства не съществува.

Накрая, ако:

$$\mathcal{L} \backslash \Sigma^* fin(S, A) = \varnothing,$$

построяваме автомат за регулярния език:

$$Suf(fin(S, A)) \backslash \Sigma^* fin(S, A).$$

Проверяваме дали този език е краен или безкраен, което съответства на свойството на автомата да не съдържа цикъл, който е достижим от началното състояние, от който можем да стигнем до финално.

Ако езикът е безкраен, то според твърдение 12 крайно множество $\mathcal{L}_0$ с търсените свойства не съществува. В првотивен случай намираме дължината n на най-дългата дума в $Suf(fin(S, A)) \backslash \Sigma^* fin(S, A)$.

Това съответства на това да намерим дължината на най-дългия път от началното до някое финално състояние в автомата за този език. Сега според твърдение 12 в качеството на $\mathcal{L}_0$ можем да изберем всички думи от $fin(S, A)$ с дължина ненадминаваща $n+1$, това са точно думите, изписани по пътищата от началното до някое финално състояние на автомата за $fin(S, A)$ с дължина ненадминаваща $n+1$.

Накрая, от втората част на твърдение 13 знаем, че всяка дума $\alpha \in fin(S, A)$ и всяка дума $\beta \in A$ думата β съдържа неповече от $|Q||\alpha|$ срещания на α, където $|Q|$ е броят на състоянията на автомата за A. Тъй като всяка дума от $\mathcal{L}_0$ има дължина не по-голяма от $n+1$, то всяка дума β съдържа неповече от $(n+1)|Q||\mathcal{L}_0|$ позиции на срещания на думи от езика $\mathcal{L}_0$.

Сега можем да формулираме и достатъчни условия, за това да можем да построим FIFO-трансдюсер за правилото $\mathcal{R}$.

Дефиниция 36 *Нека $P \cup S$ са две крайни множества от думи над азбука Σ. Дефинираме $g_{P,S} : \Sigma^* \times \Sigma^* \to (\Sigma \cup \{\omega\})^*$, където $\omega \notin \Sigma$ по следния начин:*

$$g_{P,S}(u, \varepsilon) = \begin{cases} \omega, & \text{ако } u \in \Sigma^* S \\ \varepsilon, & \text{иначе.} \end{cases}$$

$$g_{P,S}(u, av) = \begin{cases} \omega a \circ g_{P,S}(ua, v), & \text{ако } u \in \Sigma^* S \text{ или } av \in P\Sigma^*. \\ a \circ g_{P,S}(ua, v), & \text{иначе.} \end{cases}$$

Дефиниция 37 *Нека $P \cup S$ са две крайни множества от думи над азбука Σ. Дефинираме $f_{P,S} : \Sigma^* \to (\Sigma \cup \{\omega\})^*$, където $\omega \notin \Sigma$ като:*

$$f_{P,S}(t) = g_{P,S}(\varepsilon, t).$$

Имаме следните свойства на $f_{P,S}$:

1. $Range(f) \cap (\Sigma \cup \{\omega\})^* \omega\omega (\Sigma \cup \{\omega\})^* = \varnothing$.

2. Ако v е поддума на $f_{P,S}(t)$ и $h(v)$ съдържа неповече от p срещания на думи от P и s срещания на думи от S, а най-дългата дума от $P \cup S$ има дължина m, то v съдържа неповече от $p + s + 2m$ символа ω.

3. Съществува подпоследователен трансдюсер, който представя $f_{P,S}$.

Първото свойство следва директно от дефиницията на $g_{P,S}$.

Второто също е ясно. Нека v е поддума на $f_{P,S}(t)$. Тогава v е представка на някоя дума:

$$g_{P,S}(x, y) = vz,$$

където $xy = t$. Тогава е ясно, че $h(v) \prec y$. Нека предположим, че във v има повече от $2m+p+s$ срещания на символа ω. Да разгледаме онази поддума v' на $v = \alpha v' \beta$, където α и β съдържат точно по m символа ω (ако такива няма твърдението е ясно). Тогава от първото свойство следва, че α и β съдържат поне по m символа от Σ. Тоест $|h(\alpha)| \geq m$ и $|h(\beta) \geq m$. Да разгледаме някой от символите ω в v'. Той разбива думата на две части $v' = v_1 \omega v_2$. От дефиницията на $g_{P,S}$ следва, че $h(v_2)h(\beta)h(z) \in P\Sigma^*$ или $xh(\alpha)h(v_1') \in \Sigma^* S$. Тъй като обаче всяка от думите от $P \cup S$ е с дължина поне m, то всъщност $h(\alpha)v_1' \in \Sigma^* S$ или $v_2' h(\beta) \in P\Sigma^*$. Тоест това е позиция, на която завършва дума от S в $h(v)$, или позиция от която започва срещане на дума от P в $h(v)$. При това на всеки различен символ ω във v' му съответства различна позиция на срещане в $h(v)$. Тъй като последните са неповече от $p+s$, то и броят на символите ω във v' ненадминава $p + s$. Така общият брой на символи ω във v е неповече от $2m + p + s$.

Конструкцията на подпоследователен преоброзувател за $f_{P,S}$ може да се направи по начин, подобен на алгоритъма на Михов и Schulz [SM06] за замяна на всички срещания на думи от даден речник в текст. Отново започваме с дървото на Ахо-Корасик за множеството $P \cup S$ и след това го обхождаме в дълбочина, поставяйки необходимите изходи по ребрата. За да облекчим изложението, отлагаме самата конструкция, заедно с доказателството за нейната коректност за част 7.

Сега можем да покажем следното твърдение, което ще ни даде необходимо условие за регулярните правила, които могат да се разпознават от FIFO-трансдюсери.

Теорема 14 *Нека $\mathcal{R}$, $E \to T(E)\backslash L\_R$ е регулярно правило за заместване, като T е подпоследоватлен трансдюсер. Нека (L_0, l_0), (SE_0, s_0) са резултатите, които връща алгоритъма за проблема 1 при вход съответно (L, ER) и (E, ER), а пък (PE_0, p_0) и (R_0, r_0) са резултатите, които връща дуалният алгоритъм, за проблема 2 при вход (E, ER) и (R, ER), съответно. Ако някой от поставените въпроси получи отрицателен отговор, смятаме, че съответното множество е празно.*

Ако поне едно от множествата L_0 и PE_0 не е празно и поне едно от множествата SE_0 и R_0 не е празно, то можем ефективно да конструираме FIFO-трансдюсер, разпознаващ правилото $\mathcal{R}$.

Доказателство: Нека $M = L_0$, ако $L_0 \neq \varnothing$ и $M = PE_0$ в противен случай. Полагаме $m = l_0$ в първия случай и $m = p_0$ във втория. Аналогично дефинираме $N = SE_0$ и $n = s_0$ ако $SE_0 \neq \varnothing$ и $N = R_0$ и $n = r_0$, иначе. Дефинираме множеството P като:

$$P = \begin{cases} M, \text{ ако } M = PE_0, N = SE_0 \\ N, \text{ ако } M = L_0, N = R_0 \\ M \cup N, \text{ ако } M = PE_0, N = R_0 \\ \varnothing \text{ иначе.} \end{cases}$$

С S бележим $(M \cup N)\backslash P$. Накрая нека l е дължината на най-дългата дума в $M \cup N$. Сега построяваме $f_{P,S}$ и полагаме:

$$L_{\{\omega\}} = \mathcal{U}(L)\omega,$$
$$E_{\{\omega\}} = \mathcal{U}(E) \cap (\Sigma^*\omega\Sigma^*)^{\leq m+n+2l}$$
$$R_{\{\omega\}} = \omega\mathcal{U}(R) \cap (\Sigma^*\omega\Sigma^*)^{\leq m+n+2l}$$

Както в началото на тази част правилото $\mathcal{R}_{\{\omega\}}$ се определя от:

$$E_{\{\omega\}} \to T(h(E_{\{\omega\}}))\backslash L_{\{\omega\}}\_R_{\{\omega\}}$$

От дефиницията на $L_{\{\omega\}}$, $E_{\{\omega\}}$ и $R_{\{\omega\}}$ непосредствено следва, че $\mathcal{R}_{\{\omega\}}$ е правило с ограничен брой маркери.

Ще покажем, че $f_{P,S}$ и $\mathcal{R}_{\{\omega\}}$ удовлетворяват условията на твърдение 11.

Наистина от свойствата на $f_{P,S}$ следва, че $Range(f_{P,S}) \cap (\Sigma \cup \{\omega\})^*\omega\omega(\Sigma \cup \{\omega\})^* = \varnothing$. $L_{\{\omega\}}$ също удовлетворява изискването, че всяка дума от $L_{\{\omega\}}$ завършва на ω. Така остава да проверим, че за всеки $u \in \Sigma^* L$, $v \in E$ и $w \in R\Sigma^*$ съществуват u', v' и w' със свойствата:

$$\begin{aligned} f_{P,S}(uvw) &= u'v'w', \\ h(u') &= u, \\ h(v') &= v, \\ h(w') &= w. \end{aligned}$$

Ще покажем, че:

$$\begin{aligned} u' &= g_{P,S}(\varepsilon, uvw)(g_{P,S}(u, vw))^{-1}\omega \\ v' &= (\omega)^{-1} g_{P,S}(u, vw)(g_{P,S}(uv, w)) \\ w' &= g_{P,S}(uv, w) \end{aligned}$$

притежават желаното свойство. Наистина тъй като $u \in \Sigma^* L$ и $v \in E$, то или $u \in \Sigma^* L_0$ или $v \in PE_0\Sigma^*$. Това означава, че $u \in \Sigma^* S$ или $v \in P\Sigma^*$, от където и $vw \in P\Sigma^*$. По дефиницията на $g_{P,S}$ тогава $g_{P,S}(u, vw)$ започва с ω. Аналогично, тъй като $v \in E$ и $w \in R\Sigma^*$, то или $v \in \Sigma^* SE_0$ или $w \in R_0\Sigma^*$. Това показва, че $v \in \Sigma^* S$ или $w \in P\Sigma^*$. И в двата случая $g_{P,S}(uv, w)$ започва с ω. Така получихме, че u', v' и w' са дефинирани коректно като освен това:

$$f_{P,S}(uvw) = u'v'w', h(u') = u, h(v') = v, h(w') = w,$$

както лесно следва от дефиницията на $g_{P,S}$ и $f_{P,S}$. Остана да покажем, че $< u', v', w' >$ е срещане на правилото $\mathcal{R}_{\{\omega\}}$. Тъй като $h(u') = u$ и $u' \in (\Sigma \cup \{\omega\})^*\omega$, то очевидно имаме, че $u' \in (\Sigma \cup \{\omega\})^* L_{\{\omega\}}$. За v' и w' ще използваме второто свойство на $f_{P,S}$. Тъй като $h(v') = v \in E$, то от дефиницята на M и N никоя дума $\alpha \in ER$ не допуска повече от m срещания на думи от M. В частност v не съдържа повече от m срещания на думи от M. Аналогично v не съдържа повече от n

срещания на думи от N. Сега обаче $M \cup N = P \cup S$, което показва, че v не съдържа повече от $m + n$ срещания на думи от $P \cup S$. Сега най-дългата дума от $P \cup S$ има дължина l и следователно от второто свойство на $f_{P,S}$, тъй като $h(v') = v$, то v' не съдържа повече от $m+n+2l$ символа ω. Това, заедно с факта, че $h(v') = v \in E$ ни дава, че:

$$v' \in \mathcal{U}(E) \cup (\Sigma^* \omega \Sigma^*)^{\leq m+n+2l} = E_{\{\omega\}}$$

Аналогично можем да разсъждаваме, зе представката w_0 на w, за която $w_0 \in R$. Тъй като $h(w') = w$, то съществува представка на w'_0 на w', за която $h(w'_0) = w_0$. Тук използвахме, че $|h(a)| \leq 1$ за всяко $a \in \Sigma \cup \{\omega\}$. Сега от дефиницията на M и N w_0 не съдържа повече от $m+n$ поддуми от $M \cup N = P \cup S$ и тъй като дължината на най-дългата дума в $P \cup S$ е l, то w'_0 не съдържа повече от $m + n + 2l$ символа ω. Сега тъй като $w' \in \omega(\Sigma \cup \{\omega\})^*$ и $h(w'_0) \in R$, то $w'_0 \in R_{\{\omega\}}$, откъдето непосредствено следва, че $w' \in R_{\{\omega\}}(\Sigma \cup \{\omega\})^*$. С това показахме, че предпоставките на твърдение 11 са изпълнени и следователно:

$$h(rewrite_{\mathcal{R}_{\{\omega\}}}(f_{P,S}(t))) = rewrite_{\mathcal{R}}(t).$$

Тъй като $\mathcal{R}_{\{\omega\}}$ е правило с ограничен брой маркери, то имаме ефективен алгоритъм за построяването на FIFO-трансдюсер за $\mathcal{R}_{\{\omega\}}$ ако са зададени неговите компоненти и константите, които ограничават броя на маркерите. В случая ние показахме алгоритмично как можем да намерим константите m, n и l, което ни дава и директната конструкция на множествата $L_{\{\omega\}}$, $E_{\{\omega\}}$ и $R_{\{\omega\}}$. Условието, че T е подпоследователен ни гарантира, че ефективно можем да намерим и подпоследователен преобразувател и за $T \circ h$. Накрая $f_{P,S}$ и h могат да се представят ефективно с подпоследователни трансдюсери, което ни дава, че можем да приложим алгоритмите за композиция отляво и отдясно на подпоследовтелен преоброзувател с FIFO-трансдюсер.

В допълнение ще отбележим, че както видяхме проблемът 1 се разрешава ефективно, тоест можем алгоритмично да проверяваме, дали предпоставките на теорема 14 са налице и в случай, че това е така да определим съответните множества M, N, P, S и константи m, n, l.

6.3 Разпознаване на композиция от регулярни правила с FIFO-трансдюсер. Достатъчни условия

В тази част ще отделим клас от двойки правила за заместване, които позволяват да бъдат композирани, посредством FIFO-трансдюсер. Както видяхме, класът на рационалните функции, разпознавани от FIFO-трансдюсери не е затворен относно композиция и дори сравнително прости правила за заместване, които могат да се представят чрез FIFO-трансдюсер, не позволяват да бъдат композирани с този формализъм. Затова към правилата, които разгледахме в предишната част и за които намерихме достатъчни условия за тяхното представяне, ще трябва да наложим допълнителни ограничения, за да получим теорема за композиция. Тя ще се възползва съществено от теоремата за композиция на Ω-крайнопредставими функции, която разгледахме в параграф 4.5 на част 4.

Нека

$$\mathcal{R}_1 : E_1 \to T_1(E_1)\backslash L_1\_R_1 \text{ и}$$
$$\mathcal{R}_2 : E_2 \to T_2(E_2)\backslash L_2\_R_2$$

са две регулярни правила за заместване. Целта ни е да построим FIFO-трансдюсер, който да представя рационалната функция:

$$rewrite_{\mathcal{R}_2} \circ rewrite_{\mathcal{R}_2}.$$

Идеята ни отново ще бъде да ограждаме с маркери части от текста, които локално приличат на начало на фокуса, или край на фокуса на едно от двете правила. С други думи отново ще отбелязваме наставките на левия контекст или представките на фокуса и съответно наствакиte на фокуса или представките на десния контекст.

По този начин, както вече видяхме, ще можем да установим дали всяко едно от правилата може да се замести с еквивалентно на него правило с ограничен брой маркери:

$$\mathcal{R}'_{\Omega_1} : E'_{\Omega_1} \to T_{\Omega_1}(E'_{\Omega_1})\backslash L'_{\Omega_1}\_R'_{\Omega_1}$$
$$\mathcal{R}''_{\Omega_2} : E''_{\Omega_2} \to T_{\Omega_2}(E''_{\Omega_2})\backslash L''_{\Omega_2}\_R''_{\Omega_2}$$

За да можем обаче да ги композираме, в условията на теорема 11 налага допълнителни изисквания, които ние ще трябва да удовлетворим.

За да дефинираме основния резултат в този параграф ще ни трябва следната дефиниция:

Дефиниция 38 *Нека $\mathcal{R} : E \to T(E)\backslash L\_R$ е регулярно правило за заместване. Нека $\kappa \subset \{L, E, R\}$, A е регулярно множество. Казваме, че $\mathcal{R}$ е разрешимо от конфигурацията κ при ограничения A ако:*

1. *Ако $L \in \kappa$, то проблемът 1 има положително решение при вход L и $ER \cup A$, а в противен случай има положително решение дуалният проблем 2 при вход E и $ER \cup A$.*

2. *Ако $R \in \kappa$, то проблемът 2 има положително решение при вход R и $ER \cup A$, а в противен случай има положително решение проблемът 1 при вход E и $ER \cup A$.*

3. *Ако $E \in \kappa$, то поне един от проблемите 1 или 2 имат положителен отговор при вход E и $ER \cup A$.*

Забележка 7 *От горната дефиниция и от дефиницията на проблема 1 непосредствено следва, че ако $\mathcal{R}$ се разрешава от конфигурация κ при ограничения A, то съществуват крайни множества P и S и константа k, които могат да бъдат намерени ефективно със следните свойства:*

1. *$u \in L$ $v \in E$, то $uv \in uP\Sigma^*$ или $uv \in \Sigma^*Sv$.*

2. *$v \in E$ $w \in R$, то $vw \in vP\Sigma^*$ или $vw \in \Sigma^*Sw$.*

3. *Всяка дума $\beta \in (ER \cup A)$ съдържа неповече от k срещания на думи от $P \cup S$.*

Така получаваме следната:

Теорема 15 *Нека*

$$\mathcal{R}_1 : E_1 :\to T_1(E_1)\backslash L_1\_R_1 \text{ и}$$
$$\mathcal{R}_2 : E_2 :\to T_2(E_2)\backslash L_2\_R_2$$

са две регулярни правила за заместване, където T_1 *и* T_2 *са представени чрез подпоследователни преобразуватели. Нека* $\kappa_i \subset \{L_i, E_i, R_i\}$ *като* κ_i *съдържа поне едно от множествата* L_i *и* E_i*, и поне едно от множествата* E_i *и* R_i *за* $i = 1, 2$*. С* $A_i = \cup\kappa_i$ *означава обединението на множествата, които съдържа* κ_i*.*

Ако $\mathcal{R}_1$ *е разрешимо от конфигурацията* κ_1 *при ограничения* $\overline{\Sigma^* A_2 \Sigma^*}$*, а* $\mathcal{R}_2$ *е разрешимо от конфигурацията* κ_2 *при ограничения* $\overline{\Sigma^* A_1 \Sigma^*} \cup Range(T_1)$ *и освен това:*

$$Range(T_1) \subset \Sigma^* A_2 \Sigma^*,$$

то можем ефективно да построим FIFO-трансдюсер за композицита:

$$rewrite_{\mathcal{R}_2} \circ rewrite_{\mathcal{R}_1}.$$

Доказателство: От забележка 7 следва, че можем ефективно да намерим крайни множества P_1, S_1 и P_2, S_2 и константи k_1 и k_2 със свойствата:

1. $u \in L_i$, $v \in E_i$, то $uv \in uP_i\Sigma^*$ или $uv \in \Sigma^* Sv$ за $i = 1, 2$.

2. $v \in E_i$, $v \in R_i$, то $vw \in vP_i\Sigma^*$ или $vw \in \Sigma^* Sw$ а $i = 1, 2$.

3. Всяка дума от $\overline{\Sigma^* A_2 \Sigma^*} \cup E_1 R_1$ съдържа неповече от k_1 срещания на думи от $P_1 \cup S_1$.

4. Всяка дума от $\overline{\Sigma^* A_1 \Sigma^*} \cup Range(T_1) \cup E_2 R_2$ съдържа неповече от k_2 думи от $P_2 \cup S_2$.

Да отбележим още, че тъй като за всяко множество от конфигурацията κ_1, $M \in \kappa_1$ е в сила, че:

$$M \subset \Sigma^* S_1 \cup P_1 \Sigma^*,$$

то е изпълнено, че:

$$\Sigma^* M \Sigma^* \subset \Sigma^*(P_1 \cup S_1)\Sigma^*.$$

Оттук непосредствено следва, че:

$$\Sigma^* A_1 \Sigma^* \subset \Sigma^*(P_1 \cup S_1)\Sigma^*.$$

Аналогично получаваме, че:

$$\Sigma^* A_2 \Sigma^* \subset \Sigma^*(P_2 \cup S_2)\Sigma^*.$$

Сега като използваме теорема 14 от предишня параграф, можем да построим правила $\mathcal{R}_{\Omega_1}$ и $\mathcal{R}_{\Omega_2}$ с краен брой маркери със свойствата:

$$\begin{gathered} h \circ rewrite_{\mathcal{R}_{\Omega_1}} \circ f_{P_1,S_1} = rewrite_{\mathcal{R}_1} \text{ и} \\ h \circ rewrite_{\mathcal{R}_{\Omega_2}} \circ f_{P_2,S_2} = rewrite_{\mathcal{R}_2}. \end{gathered}$$

Нека сега g_1 е Ω-представимата функция, която ни дава тероема 13 за правилото $\mathcal{R}_{\Omega_1}$. Тогава от консрукцията в доказателството на теорема 13, следва, че :

$$\begin{aligned} g_1(u,v) = replace_{\mathcal{R}_{\Omega_1}}(u,v), &\text{ при} \\ count(u,v) = -1. & \end{aligned}$$

за $u \in (\Sigma \cup \Omega_1)^*\Omega_1$ или $v = \varepsilon$ Съответно с g_2 е Ω-представимата функция, която ни дава тероема 13 за правилото $\mathcal{R}_{\Omega_2}$. Тогава от консрукцията в доказателството на теорема 13, следва, че :

$$\begin{aligned} g_2(u,v) = replace_{\mathcal{R}_{\Omega_2}}(u,v), &\text{ при} \\ count(u,v) = -1. & \end{aligned}$$

за $u \in (\Sigma \cup \Omega_2)^*\Omega_2$ или $v = \varepsilon$

Нека T е каноничният трансдюсер, който задава $f_{P_2,S_2} \circ h$. Използваме резултата от теорема 10, част 4, параграф 4.4, за да конструираме Ω_1-представимата функция $\widetilde{g_1}$, за която:

$$\widetilde{g_1}(u,v) = \begin{cases} (\bigwedge_x f_T((g_1(\varepsilon,uv)g_1(u,v)^{-1})x))^{-1} f_T(g_1(\varepsilon,uv)) & \text{ако } v \neq \varepsilon \\ f_T(g_1(\varepsilon,uv)) & \text{иначе.} \end{cases}$$

Целта ни ще бъде да покажем, че можем да композираме $\widetilde{g_1}$ и g_2 върху множеството $Range(f_{P_1,S_1})$, като използваме конструкцията от теорема 11 от част 4, параграф 4.5. Това ще направим като намерим подходящи константи M и m със свойствата:

1. За всяко $uv = f_{P_1,S_1}(t)$, $u \in (\Sigma \cup \Omega_1)^*\Omega_1 \cup \{\varepsilon\}$ е в сила, че:
$$|\widetilde{g_1}(u,v)\widetilde{g_1}(uP_1(v), S_1(v))^{-1}|_{\Omega_2} \leq m.$$

2. За всяко $uv = f_{P_1,S_1}(t)$, за което $u \in (\Sigma\cup\Omega_1)^*\Omega_1\cup\{\varepsilon\}$ и $|v|_{\Omega_1} \geq M$ е изпълнено, че:
$$|\widetilde{g_1}(u,v)\widetilde{g_1}(uP_M(v), S_M(v))^{-1}|_{\Omega_2} \geq 1.$$

Ако успеем да намерим ефективно такива константи, ще можем да приложим конструкцията, която ни дава композицията на g_2 и $\widetilde{g_1}$, тоест ще можем да намерим Ω-определима функция g със свойството:

$$g(u) = g_2(\varepsilon, \widetilde{g_1}(\varepsilon, u))$$

за всяко $u \in Range(f_{P_1,S_1})$. Тогава ще имаме, че за всяко $t \in \Sigma^*$ е изпълнено:

$$\begin{aligned}
h(g(f_{P_1,S_1}(t))) = h(g_2(\varepsilon, \widetilde{g_1}(\varepsilon, f_{P_1,S_1}(t)))) = \\
h(g_2(\varepsilon, f_{P_2,S_2}(h(g_1(\varepsilon, f_{P_1,S_1}(t)))))) = \\
h(g_2(\varepsilon, f_{P_2,S_2}(h(replace_{\mathcal{R}_{\Omega_1}}(\varepsilon, f_{P_1,S_1}(t)))))) = \\
h(g_2(\varepsilon, f_{P_2,S_2}(h(rewrite_{\mathcal{R}_{\Omega_1}}(f_{P_1,S_1}(t)))))) = \\
h(g_2(\varepsilon, f_{P_2,S_2}(rewrite_{\mathcal{R}_1}(t)))) = \\
h(rewrite_{\mathcal{R}_{\Omega_2}}(f_{P_2,S_2}(rewrite_{\mathcal{R}_1(t)}))) = rewrite_{R_2}(rewrite_{R_1}(t)),
\end{aligned}$$

за всяко $t \in \Sigma^*$.

Тъй като g е Ω-определима, то за нея ще можем ефективно да построим FIFO-трансдюсер, а след това ще можем да композираме с f_{P_1,S_1} отляво и h отдясно, за да получим FIFO-трансдюсер, разпознаващ точно композицията на двете правила.

Следващата лема ще ни даде отговор на въпроса какви константи M и m можем да използваме за желаната от нас конструкция.

Лема 18 *Нека $t \in \Sigma^*$ е произволна дума, а $uv = f_{P_1,S_1}(t)$, като $u \in (\Sigma^* \cup \Omega_1)^*\Omega_1 \cup \{\varepsilon\}$.*

1. *Нека l_2 е дължината на най-дългата дума в $P_2 \cup S_2$. Тогава:*

$$|\widetilde{g_1}(u,v)(\widetilde{g_1}(uP_1(v),S_1(v)))^{-1}|_{\Omega_2} \leq k_2 + 4l_2.$$

2. *Нека l_1 е дължината на най-дългата дума в $P_1 \cup S_1$ $|v|_{\Omega_1} > l_2(k_1+2l_1)+4(k_1+2l_1) = M$. Тогава е в сила, че:*

$$\widetilde{g_1}(u,v)(\widetilde{g_1}(uP_M,S_M(v)))^{-1}|_{\Omega_2} \geq 1.$$

Ако $v = \varepsilon$ няма какво да доказваме. Иначе имаме, че:

$$\widetilde{g_1}(\varepsilon,uv)\widetilde{g_1}(u,v)^{-1} = \bigwedge_x f_{P_2,S_2}(h((g_1(\varepsilon,uv)(g_1(u,v))^{-1})x)).$$

Нека $\alpha\gamma = h(g_1(\varepsilon,uv)(g_1(u,v))^{-1})$, където γ е най-дългата наставка с дължина ненадминавеща l_2. Тогава лесно се вижда, че:

$$\widetilde{g_1}(\varepsilon,uv)\widetilde{g_1}(u,v)^{-1} = \bigwedge_x f_{P_2,S_2}(\alpha\gamma x)).$$

Но за всяко $x \in \Sigma^*$ имаме, че ако $|\beta| \geq l_2$ $\beta x \in P_2\Sigma^*$, то и $\beta \in P_2\Sigma^*$. Оттук следва, че:

$$f_{P_2,S_2}(\alpha\gamma x) = g_{P_2,S_2}(\varepsilon,\alpha\gamma x) = (g_{P_2,S_2}(\varepsilon,\alpha\gamma x)g_{P_2,S_2}(\alpha,\gamma x)^{-1})g_{P_2,S_2}(\alpha,\gamma x) = g_{P_2,S_2}(\varepsilon,\alpha\gamma)g_{P_2,S_2}(\alpha,\gamma)^{-1}g_{P_2,S_2}(\alpha,\gamma x).$$

Това показва, че:

$$\widetilde{g_1}(\varepsilon,uv)\widetilde{g_1}(u,v)^{-1} = f_{P_2,S_2}(\alpha\gamma)g_{P_2,S_2}(\alpha,\gamma)^{-1}\bigwedge_x g_{P_2,S_2}(\alpha,\gamma x).$$

Сега ако $g_1(u,v) = v_0 \circ g_1(uP_1(v),S_1(v))$ аналогично получаваме, че:

$$\widetilde{g_1}(\varepsilon,uv)\widetilde{g_1}(uP_1(v),S_1(v)^{-1} = f_{P_2,S_2}(\alpha\gamma)g_{P_2,S_2}(\alpha,\gamma)^{-1})\bigwedge_x g_{P_2,S_2}(\alpha,\gamma h(v_0)x).$$

Оттук правим извода, че:

$$\widetilde{g_1}(u,v)\widetilde{g_1}(uP_1(v),S_1(v))^{-1} = (\bigwedge_x g_{P_2,S_2}(\alpha,\gamma x))^{-1} \bigwedge_x g_{P_2,S_2}(\alpha,\gamma h(v_0)x).$$

Сега е ясно, че:

$$|\widetilde{g_1}(u,v)\widetilde{g_1}(uP_1(v),S_1(v))^{-1}|_{\Omega_2} \leq |g_{P_2,S_2}(\alpha,\gamma v_0)|_{\Omega_2}.$$

Сега да използваме, че $g_1(u,v) = replace_{\Omega_1}(u,v)$. Имаме два случая:

1. Съществува срещане $< u, v', w' >$ на правилото $\mathcal{R}_{\Omega_1}$ в (u,v). Тогава знаем, че $P_1(v) \preceq v'$ и следователно $v_0 \preceq T_1(h(v))$, тоест v_0 е представка на дума от $Range(T_1)$ и следователно не съдържа повече от k_2 позиции на срещане на дума от $P_2 \cup S_2$.

2. Не съществува срещане $< u, v', w' >$ в (u,v), тогава $v_0 = h(P_1(v))$. От дефиницията на $P_1(v)$ следва, че $P_1(v) = v_0$ или $P_1(v) = v_0\omega_1$. Нека $v_0 = bw_0a$. Тогава $a, b \in \Sigma$ и $w_0 \in \Sigma^*$ е поддума на $f_{P_1,S_1}(t)$. Да допуснем, че $w_0 \in \Sigma^* A_1 \Sigma^*$, тогава $w_0 \in \Sigma^*(P_1 \cup S_1)\Sigma^*$. Тогава обаче

   $$g_{P_2,S_2}(h(u)a, w_0 b \circ h(S_1(v)))(g_{P_2,S_2}(h(u)a \circ w_0, bh(S_1(v))))^{-1} \prec w_0$$

   щеше да съдържа поне един символ ω_1, което е противоречие. Следователно $w_0 \notin \Sigma^* A_1 \Sigma^*$. Тогава от дефиницията на k_2 w_0 не съдържа повече от k_2 позиции на срещане на думи от $P_2 \cup S_2$. Тогава $v_0 = aw_0b$ не съдържа повече от $2l_2 + k_2$ позиции на срещания на думи от $P_2 \cup S_2$. Наистина, всяка дума $x \in P_2 \cup S_2$, която не е поддума на w_0 започва с представка на v_0 и тъй като $|x| \leq l_2$, то завършва в една от първите l_2 позиции на v_0, или завършва с наставка на v_0 и съответно започва на една от последните l_2 позиции.

Нека $v_0 = x_0 y_0$, където x_0 е представка на v_0 с дължина l_2. Тогава за всяко z е в сила, че $zx_0\beta \in \Sigma^* S_2$, то е изпълнено и $x_0\beta \in \Sigma^* S$ Сега от дефиницията на g_{P_2,S_2} следва, че:

$$|g_{P_2,S_2}(\alpha,\gamma x_0 y_0)|_{\Omega_2} \leq |\gamma| + |x_0| + |g_{P_2,S_2}(\alpha\gamma x_0, y_0)|_{\Omega_2} \leq 2l_2 + |g_{P_2,S_2}(x_0,y_0)| \leq$$
$$2l_2 + |g_{P_2,S_2}(\varepsilon, v_0)| \leq 2l_2 + k_2 + 2l_2 = k_2 + 4l_2.$$

Ако пък $|v_0| < l_2$, то очевидно $g_{P_2,S_2}(u, \gamma v_0) \leq |\gamma v_0| = 2l_2$.

Отново полагаме:

$$\alpha\gamma = g_1(\varepsilon, uv) g_1(u, v)^{-1},$$

където γ е най-дългата наставка с дължина ненадминаваща l_2. От дефиницията на g_1 имаме, че:

$$g_1(u, v) = replace_{\mathcal{R}_{\Omega_1}}(u, v).$$

Имаме два случая:

1. Да допуснем, че съществува срещане в $uP_i(v), S_i(v)$ за някое $i \leq k_1 + 2l_1 + 1$ и нека i_0 е най-малкото с това свойство и нека най-дългото срещане е $< uP_{i_0}(v), x, y >$. Тогава тъй като $\mathcal{R}_{\Omega_1}$ е съ ограничен брой маркери, а от канструкцията в предишната част видяхме, че за всяка дума $\beta \in E_{\Omega_1}$ $|\beta|_{\Omega_1} \leq k_1 + 2l_1$, то и $|x| \leq k_1 + 2l.$:

$$replace_{\mathcal{R}_{\Omega_1}}(u, v) = P_{i_0}(v) T_1(h(x)) replace_{\mathcal{R}}(uP_{i_0}(v)x, y) = \\ P_{i_0}(v) T_1(h(x)) g_1(uP_{i_0}(v)x, y).$$

   Сега тъй като $|x|_{\Omega_1} \leq k_1 + 2l_1$, то $P_{i_0}(v)x \prec P_{i_0+k_1+2l_1}(v)$. Следователно тъй като $i_0 \leq k_1$ то получаваме, че $i_0 + k_1 + 2l_1 \leq 2k_1 + 4l_1 \leq 2k_1 + 2l_1 + 2l_2 = M$, което означава, че $P_{i_0}(v)x \prec P_M(v)$. Това показва, че $P_{i_0}(v)T_1(h(x))$ е представка на:

$$g_1(u, v) g_1(uP_M(v), S_M(v))^{-1}.$$

   Тъй като $T_1(h(x)) \in \Sigma^* A_2 \Sigma^*$, то и:

$$g_1(u, v) g_1(uP_M(v), S_M(v))^{-1} \in \Sigma^* A_2 \Sigma^*.$$

   Следователно, тъй като $\Sigma^* A_2 \Sigma^* \subset \Sigma^*(P_2 \cup S_2)\Sigma^*$, то съществува поддума $z \in P_2 \cup S_2$ на

$$g_1(u, v) g_1(uP_{2k_1+4l_1}(v), S_{2k_1+4l_1}(v))^{-1}.$$

2. В противен случай отново, тъй като $replace_{\mathcal{R}_{\Omega_1}}(u,v) = g_1(u,v)$ и няма срещане за никое $i \leq 2k_1 + 2l_1 + 1$, то

   $$replace_{\mathcal{R}_{\Omega_1}}(u,v) = P_{k_1+2l_1+1}(v) replace_{\mathcal{R}_{\Omega_1}}(uP_{k_1+2l_1+1}(v), S_{k_1+2l_1+1}(v)).$$

   Сега тъй като $P_{k_1+2l_1+1}(v)$ е поддума на $f_{P_1,S_1}(t)$, то $h(P_{k_1+2l_1+1}(v))$ съдържа поне $2k_1+1$ позиции на срешания на думи от $P_1 \cup S_1$ и от дефиницията на k_1 $h(P_{k_1+2l_1+1}) \in \Sigma^* A_2 \Sigma^*$ и следователно $h(P_{k_1+2l_1+1}) \in \Sigma^*(P_2 \cup S_2)\Sigma^*$.

И така, и в двата случая получихме, че:

$$h(g_1(u,v)g_1(uP_M(v), S_M(v))^{-1}) = z_0 h(g_1(uP_j(v), S_j(v)), g_1(uP_M(v), S_M(v))^{-1}),$$

където $j \leq 2l_1 + 2k_1$. Нека $N = M - j \geq (k_1 + 2l_1)l_2$. Да означим с p броят на заместванията, които се правят от $replace_{\mathcal{R}_{\Omega_1}}(uP_j(v), S_j(v))$ преди да бъде прочетен $P_M(v)$. Тъй като при всяко заместване се прочитат неповече от $k_1 + 2l_1$ символа от Ω_1 и се отпечатват поне един, защото изходната дума съдържа поддума от $P_2 \cup S_2$, то след изпълнението на тези p замествания резултатния текст има дължина поне $p + (l_2 - p)(k_1 + 2l_1) \geq l_2$. Така получаваме, че:

$$h(g_1(u,v)g_1(uP_M(v), S_M(v))^{-1}) = z_0 h(g_1(uP_j(v), S_j(v)), g_1(uP_M(v), S_M(v))^{-1}) = z_0 y_0,$$

където $z_0 \in \Sigma^*(P_2 \cup S_2)\Sigma^*$, $|y_0| \geq l_2$. За всяко β имаме, че ако $z_0 y_0 \beta \in P\Sigma^*$, то $y_0\beta \in \Sigma^*$. Това означава, че:

$$\bigwedge_x g_{P_2,S_2}(\alpha, \gamma z_0 y_0 x) =$$

$$\bigwedge_x g_{P_2,S_2}(\alpha, \gamma z_0 y_0) g_{P_2,S_2}(\alpha\gamma z_0, y_0)^{-1} g_{P_2,S_2}(\alpha\gamma z_0, y_0 x) =$$

$$g_{P_2,S_2}(\alpha, \gamma z_0 y_0) g_{P_2,S_2}(\alpha\gamma z_0, y_0)^{-1} \bigwedge_x g_{P_2,S_2}(\alpha\gamma z_0, y_0 x).$$

Сега от дефиницията на $\widetilde{g_1}$ виждаме, че това е точно:

$$\widetilde{g_1}(u,v)\widetilde{g_1}(uP_M(v), S_M(v))^{-1} = g_{P_2,S_2}(\alpha, \gamma z_0 y_0) g_{P_2,S_2}(\alpha\gamma z_0, y_0)^{-1} \bigwedge_x g_{P_2,S_2}(\alpha\gamma z_0, y_0 x).$$

Тъй като $z_0 \in \Sigma^*(P_2 \cup S_2)\Sigma^*$, то получаваме, че:

$$|g_{P_2,S_2}(\alpha, \gamma z_0 y_0) g_{P_2,S_2}(\alpha\gamma z_0, y_0)^{-1}|_{\Omega_2} \geq 1,$$

което завършва доказателството на това твърдение.

Сега можем да композираме g_2 и $\widetilde{g_1}$ с константи $m = 2k_2 + 4l_2$ и $M = 4(k_1 + 2l_1) + l_2(k_1 + 2l_1)$. Така конструираме:

$$g(u) = g_2(\varepsilon, \widetilde{g_1}(\varepsilon, u)).$$

След това намираме пресмятаме и композицията:

$$h \circ g \circ f_{P_1,S_1},$$

която според разсъжденията, направени преди доказателството на лемата е точно:

$$h \circ g \circ f_{P_1,S_1} = rewrite_{\mathcal{R}_\in} \circ rewrite_{\mathcal{R}_\infty}.$$

7 Алгоритмична реализация на основните конструкции

В тази част ще изложим примерна реализация на по-голяма част от конструкциите, които описахме в частите 3, 4 и 6.

Ще започнем с описанието на алгоритъм за функцията $f_{P,S}$, което пропуснахме в предходната част, но което съществено използвахме при доказателството и на двата основни резултата в параграфите 6.2 и 6.3. Всъщност ще опишем алгоритми за маркирането на всички наставки и всички представки и ще получим последователен преобразувател за $f_{P,S}$ като композиция от двата. За да можем обаче да го използваме в конструкциите от предишната част ще трябва да го приведем в канонична форма, за което можем да се обърнем към алгоритъма на Mohri [Moh94].

Във втория параграф ще разтледаме ефективни алгоритми за намирането на необходимите ни стойности на функциите $walk_P$, $print_P$, $rest_P$, съответно на $walk_Q$, $print_Q$, $output_Q$, които дефинирахме при конструкцията в теорема 6 в част 3, параграф 3.2. Същите алгоритми, с малки изменения, могат да се приложат и за конструкцията на лявата композиция на FIFO-трансдюсер с подпоследоватлен преобразувател и при намирането на каноничен FIFO-трансдюсер, еквивалентен на даден.

След това ще разгледаме алгоритмите, свързани с построяването на FIFO-трансдюсер по дадено Ω-представима функция, като се основаваме изцяло на конструкцията в теорема 7. Ще дадем алгоритъм и за композицията на две Ω-представими функции, в случая, когато предпоставките на тероема 11 са налице. Накрая ще дадем и алгоритъм, който по дадено правило с ограничен брой маркери намира компонентите на съответстващата му според теорема 13 Ω-представима функция.

Ще завършим тази част като скицираме алгоритмите, които разработихме в параграфите 6.2 и 6.3 на предходната част 6.

7.1 Алгоритъм за маркиране на префикси и суфикси по даден речник

В тази част ще опишем алгоритъм, който по зададен речник от думи P конструира подпоследователен трансдюсер, който маркира всички позиции в даден текст, които са начало на срещане на дума от P. Формално това означава да построим преборазувател за следната функция $f_P : \Sigma^* \to (\Sigma \cup \{\omega\})^*$:

$$f_P(\varepsilon) = \varepsilon$$

$$f_P(c\alpha) = \begin{cases} \omega c \circ f_P(\alpha) \text{ ако } c\alpha \in P\Sigma^* \\ c \circ f_P(\alpha) \text{ иначе.} \end{cases}$$

Ясно е, че ако P съдържа две думи $\alpha \prec \beta$, то за всеки текст $T = t_0 t_1 \ldots t_n$ и всяка позиция i в този текст, от която започва срещане на думата β е изпълнено, че $t_i t_{i+1} \ldots t_n$ започва и с думата α. Оттук лесно следва, че $f_{P\setminus\{\beta\}} = f_P$. Затова можем да считаме, че никоя дума на P не се явява представка на друга дума от P.

Идеята за конструкцията е подобна на тази на алгоритъма на Михов и Schulz [SM06] за конструиране на подпоследователен преобразувател, който заменя всяко срещане на дума от речника с нейното съответствие. Това в общия случай не може да стане еднозначно и затова те прилагат стратегията за най-ляво, най-дълго срещане на дума от речника в текста. Това им гарантира еднозначност при замяната. Ясно е обаче, че в общия случай, няма да бъдат заменени всички думи в речника. Това е и разликата с нашия случай - ние искаме да маркираме всички срещания на думи, независимо от това дали се застъпват, или не.

Както Михов и Schulz [SM06], и ние ще започнем от trie за речника P, но тъй като нямаме "преводи няма да ни се налага да имаме изходи по финалните състояния. След това ще извършим същото обхождане в широчина, както в [SM06], за да попълним функцията на преходите и, за да дефинираме подходящо функциите на изхода, ϕ и λ.

И така нека $T =< \Sigma, Q, s, \delta, F >$ е trie за крайното множество от думи P. В понататъшното изложение ще ни бъдат нужни следните дефиниции:

Дефиниция 39 *За всяко състояние $p \in Q$, $\alpha(p) \in \Sigma^*$ е думата, определена от единствения път от s до p в дървото T, тоест:*

$$\delta^*(s, \alpha(p)) = p.$$

Дефиниция 40 *За дума α с $tail(\alpha)$ означаваме най-дългата собствена наставка на α, тоест различна от α, за която $!\delta^*(s, tail(\alpha))$.*

Дефиницията е коректна за всяка непразна дума α, доколкото $\delta^*(s, \varepsilon) = s$ е дефинирана. Казано иначе, $tail(\alpha)$ е най-дългата наствака на α, различна от α, която е представка на някоя дума от P.

Дефиниция 41 *За всяко състояние $p \in Q \backslash \{s\}$, $fail(p)$ се дефинира от равенството:*

$$fail(p) = \delta^*(s, tail(\alpha(p))).$$

Следващото просто наблюдение е в основата на алгоритъма, който ще опишем:

Твърдение 15 *Нека $\alpha \in \Sigma^*$ и нека β е най-дългата наставка на α, за която $!\delta^*(s, \beta)$, тоест β е предстваkа на дума от P. Тогава за всяка буква $a \in \Sigma$ е в сила, че:*

1. $$f_P(\alpha)(f_P(\beta))^{-1} = f_P(\alpha \circ a)(f_P(\beta \circ a))^{-1}.$$

2. *ако $\alpha \notin P\Sigma^*$, но $\alpha \circ a \in P$, то е в сила, че:*

$$f_P(\alpha \circ a)(f_P(tail(\alpha) \circ a))^{-1} = \omega \circ f_P(\alpha)(f_P(tail(\alpha)))^{-1}.$$

Доказателство:

1. Първото твърдение ще докажем с индукция по дължината на α', където $\alpha = \alpha'\beta$.

   (a) $\alpha' = \varepsilon$. Тогава $\beta = \alpha$ и твърдението е очевидно.

(б) Нека твърдението е в сила за всяка дума α'' с дължина по-малка от n. Нека $\alpha' = c \circ \alpha''$, $|\alpha'| = n$. Тогава лесно се вижда, че $\alpha \in P\Sigma^*$ точно тогава, когато $\alpha \circ a \in P\Sigma^*$. Наистина, ако $\alpha \in P\Sigma^*$, то е ясно, че и $\alpha \circ a \in P\Sigma^*$. Обратно, нека $\alpha \circ a \in P\Sigma^*$. Да допуснем, че $\alpha \notin P\Sigma^*$. Тогава следва, че $\alpha \circ a \in P$ и от дефиницията на T получаваме, че $\delta^*(s, \alpha \circ a) \in F$. В частност $!\delta^*(s, \alpha)$. Последното показва, че $\beta = \alpha$ и следователно $\alpha' = \varepsilon$, което е противоречие.

Оттук можем да заключим, че:

$$f_P(\alpha'\beta) = c \circ f_P(\alpha''\beta) \Leftrightarrow f_P(\alpha'\beta \circ a) = c \circ f_P(\alpha''\beta \circ a)$$

и съответно:

$$f_P(\alpha'\beta) = \omega c \circ f_P(\alpha''\beta) \Leftrightarrow f_P(\alpha'\beta \circ a) = \omega c \circ f_P(\alpha''\beta \circ a)$$

Сега β е най-дългата наставка на $\alpha'\beta$, за която $!\delta^*(s, \beta)$, следователно β е най-дългата наставка на $\alpha''\beta$ с това свойство. Тъй като $|\alpha''| < n$ то можем да приложим индукционното предположение за $\alpha''\beta$ и да получим, че:

$$f_P(\alpha''\beta)(f_P(\beta))^{-1} = f(\alpha''\beta \circ a)(f_P(\alpha''\beta \circ a))^{-1}.$$

Сега остава да заместим в горните равенства, за да се убедим, че:

$$f_P(\alpha'\beta)(f_P(\beta))^{-1} = f_P(\alpha'\beta \circ a)(f_P(\alpha'\beta \circ a))^{-1}.$$

2. Нека сега $\alpha \notin P\Sigma^*$, но $\alpha \circ a \in P$. Нека $\alpha = c\alpha'$. Тогава е в сила, че:

$$f_P(\alpha) = c \circ f_P(\alpha')$$
$$f_P(\alpha \circ a) = \omega c f_P(\alpha' \circ a).$$

Сега остана да забележим, че $tail(\alpha)$ е най-дългата наставка на α', за която $!\delta^*(s, tail(\alpha))$. Тогава от първото твърдение, директно следва, че:

$$f_P(\alpha)(f_P(tail(\alpha)))^{-1} = c \circ f_P(\alpha')(f_P(tail(\alpha)))^{-1} \text{ и}$$
$$f_P(\alpha \circ a)(f_P(tail(\alpha) \circ a))^{-1} = \omega c f_P(\alpha')(f_P(tail(\alpha)))^{-1},$$

което завършва доказателството.

Сега сме готови да представим нашия алгоритъм и да докажем неговата коректност:

$T \; =< \Sigma, Q, s, \delta, F >$
$\widetilde{\phi}(s) = \varepsilon$
$L_1 \;\leftarrow\; \varnothing$
for $\forall \; a \;\in\; \Sigma$ **do**

    if $(\neg \; ! \; \delta(s,a))$
        $\widetilde{\delta}(s,a) \;\leftarrow\; s$
        $\widetilde{\lambda}(s,a) \leftarrow\; a$
    else
        $\widetilde{\delta}(s,a) \leftarrow\; \delta(s,a)$
        $\widetilde{\lambda}(s,a) \leftarrow\; \varepsilon$
        if $(\delta(s,a) \;\in\; F)$
            $\gamma \leftarrow\; \omega \;\circ\; a$
        else
            $\gamma \leftarrow\; a$
        $L_1 \;\leftarrow\; L_1 \;\cup\; \{(\delta(s,a), s, \gamma)\}$
done

$i \leftarrow\; 1$
while$(L_i \;\;/\!\!=\;\; \varnothing)$ **do**
    $L_{i+1} \leftarrow\; \varnothing$
    for $\forall \; (p,q,\gamma) \;\in\; L_i$ **do**
        $\widetilde{\phi}(p) \leftarrow\; \gamma \;\circ\; \widetilde{\phi}(q)$
        for $(\forall \; a \;\in\; \Sigma)$ **do**
            if $(\neg \; !\delta(p,a))$
                $\widetilde{\delta}(p,a) \leftarrow\; \widetilde{\delta}(q,a)$
                $\widetilde{\lambda}(p,a) \leftarrow\; \gamma \circ \widetilde{\lambda}(q,a)$
            else
                $\widetilde{\delta}(p,a) \leftarrow \delta(p,a)$
                $\widetilde{\lambda}(p,a) \leftarrow \varepsilon$
                if $(\delta(p,a) \;\in\; F)$
                    $L_{i+1} \leftarrow\; L_{i+1} \;\cup \{(\delta(p,a), \widetilde{\delta}(q,a), \omega \;\circ\; \gamma \circ \widetilde{\lambda}(q,a))\}$

```
            else
                $L_{i+1} \leftarrow L_{i+1} \cup \{\delta(p,a), \widetilde{\delta}(q,a), \gamma \circ \widetilde{\lambda}(q,a)\}$
        done
    done
    $i \leftarrow i+1$
done
```

Твърдение 16 *Нека за всяко $p \in Q$ с $\gamma(p)$ да означим:*

$$\gamma(p) = f_P(\alpha(p))(f_P(tail(\alpha(p))))^{-1}.$$

Да допуснем, че L_i е конструирано и още, че:

$$L_i = \{(p, fail(p), \gamma(p)) \mid |\alpha(p)| = i\}$$

Нека още всяко $p \in Q$, за което $|\alpha(p)| < i$ и за всяко $a \in \Sigma$, за което β е най-дългата наставка на $\alpha(p) \circ a$ със свойството $!\delta^*(s, \beta)$ са в сила следните твърдения:

1. $\widetilde{\delta}(p,a) = \delta^*(s, \beta)$.
2. $\widetilde{\lambda}(p,a) = f_P(\alpha(p) \circ a)(f_P(\beta))^{-1}$.
3. $\widetilde{\phi}(p) = f_P(\alpha(p))$.

При тези предположения след конструирането на L_{i+1} са в сила твърденията:

1. $L_{i+1} = \{(p, fail(p), \gamma(p)) \mid |\alpha(p)| = i+1\}$.
2. *Твърденията 1-3 са в сила за всяко $p \in Q$ с $\alpha(p) \leq i$.*

Доказателство: Наистина, тъй като за всяко $p \neq s$ е изпълнено, че $\alpha(p) = \alpha(p')a$ за някоя буква $a \in \Sigma$, то очевидно е, че ако в L_i има по един единствен запис за всяко p с $|\alpha(p)| = i$, то в L_{i+1} ще има по един единствен запис за всяко от състоянията p', за които $|\alpha(p')| = i+1$. При това този запис ще бъде добавен точно, когато разглеждаме съответния запис за p в L_i и $a \in \Sigma$, за които $\alpha(p') = \alpha(p) \circ a$.

Нека (p', q', γ') е добавеният на тази стъпка запис. Тогава имаме, че:

$$q' = \widetilde{\delta}(fail(p), a)$$
$$\gamma' = \begin{cases} \gamma(p)\widetilde{\lambda}(fail(p), a) \text{ ако } p' = \delta(p, a) \notin F \\ \omega \circ \gamma(p)\widetilde{\lambda}(fail(p), a) \text{ иначе.} \end{cases}$$

От дефиницията за $fail(p)$ имаме, че $|\alpha(fail(p))| < |\alpha(p)| = i$. Тогава, от условието на твърдението е в сила, че:

$$q' = \widetilde{\delta}(fail(p), a) = \delta^*(s, \beta),$$

където β е най-дългата наставка на $\alpha(fail(p)) \circ a$, за която $!\delta^*(s, \beta)$. Тогава обаче, лесно се вижда, че β е най-дългата собствена наставка на $\alpha(p')$, за която $!\delta^*(s, \beta)$. В противен случай щеше да съществува наставка β' на $\alpha(p')$, която да съдържа $\alpha(fail(p)) \circ a$ като собствена наставка и щеше да е дефинирана $!\delta^*(s, \beta_0)$. Тогава е ясно, че ще е дефинирана и $!\delta^*(s, \beta_0 a^{-1})$, което показва, че $\beta_0 a^{-1}$ е по-дълга собствена наставка на $\alpha(p)$ от $\alpha(fail(p))$, за която $!\delta^*(s, \beta_0 a^{-1})$. Това обаче е противоречие с дефиницията на $fail(p)$. С това доказахме, че:

$$q' = fail(p').$$

Нека β' е най-дългата наставка на $\alpha(fail(p))$, за която $!\delta^*(s, \beta')$. По дефиниция имаме, че:

$$\gamma(p') = f_P(\alpha(p'))(f_P(\alpha(q')))^{-1}.$$

Тъй като $\alpha(q')$ е наставка на $\beta' a$, то лесно се вижда, че:

$$\gamma(p') = f_P(\alpha(p'))(f_P(\alpha(q)a))^{-1} \circ f_P(\alpha(q)a)(f_P(\alpha(q')))^{-1} =$$
$$f_P(\alpha(p)a)(f_P(tail(\alpha(p))a))^{-1} \circ f_P(\alpha(q)a)(f_P(\alpha(q')))^{-1}.$$

Сега остана да забележим, че тъй като P не съдържа двойка думи, едната, от които е собствена представка на другата, то $\alpha(p) \notin P$. Тогава ако $p' \in F$, то $\alpha(p)a \in P$ и следователно от твърдение 15:

$$f_P(\alpha(p)a)(f_P(tail(\alpha(p))a))^{-1} = \omega f_P(\alpha(p)) f_P(tail(\alpha(p)))^{-1} = \omega \circ \gamma(p),$$

а в противен случай е ясно, че:

$$f_P(\alpha(p)a)(f_P(tail(\alpha(p))a))^{-1} = f_P(\alpha(p))f_P(tail(\alpha(p)))^{-1} = \gamma(p),$$

Накрая остана да забележим, че:

$$\widetilde{\lambda}(fail(p), a) = f_P(\alpha(fail(p)))(f_P(\alpha(q')))^{-1} = \\ f_P(tail(\alpha(p)))(f_P(\alpha(q')))^{-1}.$$

Сега остана да заместим, за да получим, че:

$$\gamma(p') = \begin{cases} \omega\gamma(p)\widetilde{\lambda}(fail(p), a) \text{ ако } p' \in F \\ \gamma(p)\widetilde{\lambda}(fail(p), a) \text{ иначе.} \end{cases}$$

С това показахме, че записът за p' в L_{i+1} действително има вида $(p', fail(p'), \gamma(p'))$.

Сега ще докажем, че твърденията 1-3 са в сила за всяко $p \in Q$, за което $|\alpha(p)| \leq i$. В рамките на цикъла, в който се конструира L_{i+1} се променят стойности на функциите с първи аргумент p, където $(p, fail(p), \gamma(p)) \in L_i$ следователно останалите стойности остават непроменени, в частност за всяко p $|\alpha(p)| < i$ свойствата 1-3 остават в сила.

Всяко от състоянията p, за които $|\alpha(p)| = i$ се разглежда точно веднъж в рамките на цикъла. При това получаваме:

$$\widetilde{\phi}(p) = \gamma(p)\phi(fail(p))$$

$$\widetilde{\delta}(p, a) = \begin{cases} \delta(p, a) \text{ ако } !\delta(p, a) \\ \widetilde{\delta}(fail(p), a) \text{ иначе.} \end{cases}$$

$$\widetilde{\lambda}(p, a) = \begin{cases} \varepsilon \text{ ако } !\delta(p, a) \\ \widetilde{\lambda}(p, a) = \gamma(p) \circ \widetilde{\lambda}(p, a) \text{ иначе.} \end{cases}$$

Сега обаче $|\alpha(fail(p))| < i$ и следователно условията 1-3 за $fail(p)$ са налице. Тогава $\widetilde{\phi}(fail(p)) = f_P(\alpha(fail(p))$. От друга страна $\gamma(p) = f_P(\alpha(p))(f_P(\alpha(fail(p))))^{-1}$, което означава, че:

$$\widetilde{\phi}(p) = f_P(\alpha(p)).$$

Аналогично прилагаме, съответните свойствата на $fail(p)$, за да получим твърденията 2 и 3 за p. Ако $!\delta(p, a)$, то най-дългата наставка β на $\alpha(p)a$, за която $!\delta^*(s, \beta)$ е самата дума $\alpha(p)a$ и тогава:

$$\delta^*(s, \alpha(p)a) = \delta(\delta^*(s, \alpha(p)), a) = \delta(p, a).$$

Тогава имаме и че:

$$f_P(\alpha(p)a)(f_P(\alpha(p)a))^{-1} = \varepsilon,$$

което заедно с дефиницията на $\widetilde{\delta}$ и $\widetilde{\lambda}$ доказва верността на твърдения 2 и 3.

В алтернативния случай $\neg!\delta(p, a)$. Тогава най-дългата наставка β на $\alpha(p)a$, за която $!\delta^*(s, \beta)$ е собствена наставка на $\alpha(p)a$, откъдето получаваме, че β е наставка на $\alpha(fail(p))a$. Сега от дефиницията на β следва, че β е най-дългата наставка на $\alpha(fail(p))a$, за която $!\delta^*(s, \beta)$. Тогава от допускането за $fail(p)$ следва, че:

$$\delta^*(s, \beta) = \widetilde{\delta}(fail(p), a).$$

Това показва твърдение 2 за състоянието p в този случай. За да проверим и твърдение 3 използваме твърдение 15 и:

$$f_P(\alpha(p)a)(f_P(\beta))^{-1} = f_P(\alpha(p)a)(f_P(tail(\alpha(p))a))^{-1} f_P(tail(\alpha(p))a)(f_P(\beta))^{-1}.$$

Нека $\alpha(p) = c\alpha'$. Тъй като $\neg!\delta^*(s, c\alpha' a)$, то

$$f_P(c\alpha' a) = f_P(c\alpha') f_P(\alpha')^{-1} f_P(\alpha' a).$$

Като заместим получаваме, че:

$$f_P(\alpha(p)a)(f_P(tail(\alpha(p))))^{-1} = f_P(\alpha)(f_P(\alpha'))^{-1} f_P(\alpha' a)(f_P(tail(\alpha(p))a))^{-1},$$

но $tail(\alpha)$ е най-дългата наставка на α', която е представка на дума от P и да нея можем да приложем твърдение 15. Тогава получаваме, че:

$$f_P(\alpha(p))(f_P(\alpha'))^{-1}(f_P(\alpha')(f_P(tail(\alpha(p))))^{-1}) = \\ f_P(\alpha(p))(f_P(tail(\alpha(p))))^{-1} = \gamma(p).$$

Накрая остана да забележим, че от допускането за $fail(p)$ следва, че:

$$f_P(\alpha(fail(p))a)(f_P(\beta'))^{-1} = \widetilde{\lambda}(fail(p), a).$$

С това показахме, че действително:

$$\widetilde{\lambda}(p,a) = \gamma(p)\widetilde{\lambda}(fail(p),a) = f_P(\alpha(p)a)(f_P(\beta'))^{-1}.$$

Сега лесно се вижда, че L_1 изпълнява условията на твърдение 16 и тогава с индукция по i, прилагайки твърдение 16 лесно се вижда, че L_i има тези свойства за всяко i. В частност ако $i > \max\{|u| \mid u \in P\}$ получаваме, че $L_i = \varnothing$. Това означава, че алгоритъмът спира след $\max\{|u| \mid u \in P\}$ итерации на основния цикъл.

Твърдение 17 *Нека $\alpha \in \Sigma^*$, а β е най-дългата наставка на α, за която $!\delta^*(s,\beta)$. Тогава са в сила твърденията:*

1. $\widetilde{\delta}^*(s,\alpha) = \delta^*(s,\beta)$ *и*
2. $\widetilde{\lambda}^*(s,\alpha) = f_P(\alpha)(f_P(\beta))^{-1}$.

Доказателство: Индукция по дължината на α. За $\alpha = \varepsilon$ всичко е ясно. Нека твърденията са доказани за всяка дума α', за която $|\alpha'| < n$ и нека $|\alpha| = n$. Тогава $\alpha = \alpha' a$. Нека β' е най-дългата наставка на α', за която $!\delta^*(s,\beta')$. Нека $p = \delta^*(s,\beta')$. Тогава от индукционната хипотеза имаме, че:

$$\widetilde{\delta}^*(s,\alpha' a) = \widetilde{\delta}(\delta^*(s,\beta'),a) = \widetilde{\delta}(p,a) \text{ и}$$
$$\widetilde{\lambda}^*(s,\alpha' a) = f_P(\alpha')(f_P(\beta'))^{-1} \circ \widetilde{\lambda}(p,a).$$

Сега е ясно, че $\alpha(p) = \beta'$. Освен това $\alpha(p) \leq \max\{|u| \mid u \in P\}$ и следователно от твърдение 16 получаваме, че:

$$\widetilde{\delta}(p,a) = \delta^*(s,\beta) \text{ и}$$
$$\widetilde{\lambda}(p,a) = f_P(\beta' a)(f_P(\beta))^{-1},$$

където β е най-дългата наставка на $\beta' a$, за която е дефинирана $\delta^*(s,\beta' a)$. Лесно се вижда, че β е най-дългата наставка и на α с това свойство.

Наистина, ако съществуваше по-дълга такава наставка β'', то тя ще съдържа и $\beta' a$, което означава, че $!\delta^*(s, \beta'')$, в частност ще е дефинирана и $\delta^*(s, \beta'' a^{-1})$. Тъй като $\beta'' a^{-1} \prec \alpha'$, то β'' не може да е по-дълга от $\beta' a$. С това първото твърдение е доказано, именно:

$$\widetilde{\delta}^*(s, \alpha) = \delta^*(s, \beta) = \widetilde{\delta}(p, a),$$

където β е най-дългата наставка на α, за която $!\delta^*(s, \beta)$. За втората част на твърдението да забележим, че от твърдение 15 имаме, че:

$$f_P(\alpha')(f_P(\beta'))^{-1} = f_P(\alpha' a)(f_P(\beta' a))^{-1},$$

откъдето заедно със свойството на $\widetilde{\lambda}$ можем да заключим, че:

$$\widetilde{\lambda}^*(s, \alpha) = f_P(\alpha')(f_P(\beta'))^{-1} f_P(\beta' a)(f_P(\beta))^{-1} =$$
$$f_P(\alpha' a)(f_P(\beta' a))^{-1} f_P(\beta' a)(f_P(\beta))^{-1} = f_P(\alpha)(f_P(\beta))^{-1}.$$

С това индукционната стъпка е завършена и твърдението е доказано.

Накрая остана да забележим, че от твърдение 16, приложено за най-дългата наставка β на α, за която $!\delta^*(s, \beta)$ и за състоянието $p = \delta^*(s, \beta)$ е изпълнено, че:

$$\widetilde{\phi}(p) = f_P(\beta).$$

Сега обединяваме резултатът от последното твърдение с последното равенство и получаваме, че:

$$f_P(\alpha) = \widetilde{\lambda}^*(s, \alpha)\widetilde{\phi}(\widetilde{\delta}^*(s, \alpha)).$$

Това показва, че подпоследоватлният преобразувател $T_P =< \Sigma \times (\Sigma \cup \{\omega\})^*, Q, s, \widetilde{\delta}, \widetilde{\lambda}, \widetilde{\phi} >$ разпознава точно функцията f_P.

Ще отбележим, че времето за изпълнение на представения алгоритъм е $O(|T|)$, тоест линейно по броя на върховете на дървото T. Действително, всеки връх на T попада точно веднъж в някое от множествата L_i и за всеки връх на T извършваме $O(|\Sigma|)$ операции по насочване на преходите и по определяне на функцията на преходите.

Всяка от първите се изпълнява за константно време. Втората операция е по-трудоемка, защото изисква конкатенация на две думи. Ние обаче, можем да използваме подхода на Михов и Schulz [SM06], които използват дървовидно представяне на тази конкатенация. Всяка стойност на λ функцията се представя с два указателя. Ако $\lambda(q, a)$ е получена при конкатенацията на $\gamma\beta$ правим следното. Ако и двете думи γ и β са непразни, поставяме двата указателя на $\lambda(q, a)$ към корените на дърветата, които представляват β и γ. Ако точно една от двете думи е празната, утъждествяваме първият указател на $\lambda(q, a)$ с непразната дума, а втория нулираме. В случай, че и двете думи са празни – $\lambda(q, a)$ също е празната дума. Тогава и двата указателя се нулират. Ясно е, че по този начин всяка дума се представя като двоично дърво, всеки връх на което с изключение на листата има точно два наследника. Това означава, че броят на върховете му е два пъти повече от броя на листатата -1. По листата му се намират буквите на съответния етикет, тоест при обхождане в дълбочина получаваме точно желаната стойност на $\lambda(q, a)$ без това да влияе на сложността на алгоритъма за траверсиране на преобразувателя.

Крайната ни цел е да построим подпоследователен преобразувател за функцията $f_{P,S}$. Функцията f_P ще бъде базата, на която ще стъпим. Идеята е сега да конструираме f_S със свойството:

$$f_S \circ f_P = f_{P,S}.$$

Затова дефинираме $f_S : (\Sigma \cup \{\omega\})^* \to (\Sigma \cup \{\omega\})^*$ по следния начин:

$$f_S(\omega^k) = \omega^{\min(k,1)}$$

$$f_S(\alpha \circ a\omega^k) = \begin{cases} f_S(\alpha)a\omega \text{ ако } h(\alpha) \circ a \in \Sigma^* S \\ f_S(\alpha)a\omega^{\min(k,1)} \text{ иначе.} \end{cases}$$

Първо ще покажем, че така дефинираната f_S действително има свойството:

$$f_S \circ f_P = f_{P,S},$$

а след това ще построим и подпоследователен преобразувател за нея.

Първо да си припомним, че:

$$g_{P,S}(u,\varepsilon) = \begin{cases} \omega u \in \Sigma^* S \\ \varepsilon \ . \end{cases}$$

$$g_{P,S}(u,av) = \begin{cases} \omega a \circ g_{P,S}(ua,v) \text{ ако } u \in \Sigma^* S \text{ или } av \in P\Sigma^*, \\ a \circ g_{P,S}(ua,v) \text{ иначе.} \end{cases}$$

Тогава дефинирахме $f_{P,S}$ като:

$$f_{P,S}(v) = g_{P,S}(\varepsilon, v).$$

И така започваме със следната лема:

Лема 19 *Нека $u \in (\Sigma \cup \{\omega\})^*$ и $v \in \Sigma^*$ са произволни думи. Тогава:*

$$f_S(uf_P(v)) = \begin{cases} f_S(u)\omega^{-1} g_{P,S}(h(u),v) \text{ ако } g_{P,S} \in \omega(\Sigma \cup \{\omega\})^* \text{ и } f_S(u) \in (\Sigma \cup \{\omega\})^* \omega, \\ f_S(u) g_{P,S}(h(u),v) \text{ иначе.} \end{cases}$$

Доказателство: Доказателството е с индукция по дължината на v.

1. $v = \varepsilon$. Тогава по дефиниция имаме, че $f_P(\varepsilon) = \varepsilon$. Ако $g_{P,S}(h(u),\varepsilon) = \varepsilon$ твърдението е очевидно. В противен случмай имаме, че $h(u) \in \Sigma^* S$. Тогава от дефиницията на f_S имаме, че $f_S(u)$ завършва на ω и следователно действително:

$$f_S(u) = f_S(u)\omega^{-1}\omega = f_S(u)\omega^{-1} g_{P,S}(h(u),\varepsilon).$$

2. Нека сега твърдението е вярно за произволни u' и v', за които $|v'| < n$. Нека u и $v = av'$ са произволни като $|v| = n$. В зависимост от това дали $v \in P\Sigma^*$ или не имаме два случая.

   (a) $v \in P\Sigma^*$, тогава $f_P(av') = \omega a \circ f_P(v')$ и $g_{P,S}(h(u),av') = \omega a \circ g_{P,S}(h(u)a,v')$. Сега от дефиницията на f_P получаваме, че:

$$f_S(u \circ f_P(v)) = f_S(u\omega a \circ f_P(v')),$$

откъдето като използваме индукционното предположение заключаваме, че:

$$f_S(u \circ f_P(v)) = \begin{cases} f_S(u\omega a) \circ \omega^{-1} \circ g_{P,S}(h(u)a, v') \\ \text{ако } f_S(u\omega a) \in (\Sigma^* \cup \{\omega\})^*\omega \text{ и } g_{P,S}(h(u)a, v') \in \omega(\Sigma \cup \{\omega\})^*, \\ \\ f_S(u\omega a) \circ g_{P,S}(h(u)a, v') \text{ иначе.} \end{cases}$$

Тъй като $a \in \Sigma$, от дефиницията на f_S лесно се вижда, че първият случай е възможен само ако $h(u)a \in \Sigma^* S$. Но тогава $f_S(u\omega a) = f_S(u\omega)a\omega$. Във втория случай имаме, че $h(u)a \notin \Sigma^* S$, защото иначе $f_S(u\omega a)$ завършва с ω и $g_{P,S}(h(u)a, v')$ започва с ω. И така и в двата случая получаваме, че:

$$f_S(u \circ f_P(v)) = f_S(u\omega) \circ a \circ g_{P,S}(h(u)a, v').$$

Сега остана да забележим, че:

$$f_S(u\omega) = \begin{cases} f_S(u)\omega^{-1}\omega \text{ ако } f_S(u) \in (\Sigma \cup \{\omega\})^*\omega, \\ f_S(u)\omega \text{ иначе.} \end{cases}$$

Този факт лесно се проверява като вземем под внимание дефиницията на f_S. Всъщност ако $f_S(u)$ завършва с ω, то $f_S(u\omega) = f_S(u)$, докато в противен случай нито $h(u) \in \Sigma^* S$, нито u завършва на ω, откъдето следва и втория случай.

Сега, като бединим тези две равенства получаваме, че:

$$f_S(uf_P(v)) = \begin{cases} f_S(u)\omega^{-1}\omega \circ a \circ g_{P,S}(h(u)a, v'), \\ \text{ ако } f_S(u) \in (\Sigma \cup \{\omega\})^*\omega \\ \\ f_S(u)\omega \circ a \circ g_{P,S}(h(u)a, v'), \text{ иначе.} \end{cases}$$

Тъй като в този случай $g_{P,S}(h(u), v) = \omega a \circ g_{P,S}(h(u)a, v')$

започва с ω, то непосредствено получаваме и желаното:

$$f_S(uf_P(v)) = \begin{cases} f_S(u)\omega^{-1}g_{P,S}(h(u),v) \\ \text{ ако } g_{P,S} \in \omega(\Sigma \cup \{\omega\})^* \text{ и } f_S(u) \in (\Sigma \cup \{\omega\})^*\omega, \\ \\ f_S(u)g_{P,S}(h(u),v) \text{ иначе.} \end{cases}$$

(б) Нека сега $v \notin P\Sigma^*$. Тогава $f_P(v) = af_P(v')$. Нещо повече в този случай ако $g_{P,S}(h(u),v)$ започва с ω то $h(u) \in S$, което означава, че $f_S(u)$ завършва с ω. Отново прилагаме индукционното предположение за v' и както и в първия случай получаваме, че:

$$f_S(uaf_P(v')) = f_S(u)a \circ g_{P,S}(h(u)a, v').$$

Остана да приложим наблюдението, което отбелязахме по-рано. Именно:

$$g_{P,S}(h(u), av') = \omega a \circ g_{P,S}(h(u)a, v') \Rightarrow f_S(u) \in (\Sigma \cup \{\omega\})^*\omega$$
$$g_{P,S}(h(u), av') = a \circ g_{P,S}(h(u)a, v'), \text{ иначе.}$$

Оттук следва и верността на твърдението. Наистина ако:

$$g_{P,S}(h(u), av') = \omega a \circ g_{P,S}(h(u)a, v'),$$

то $f_S(u) \in (\Sigma \cup \{\omega\})^*\omega$ и

$$f_S(uaf_P(v)) = f_S(u)\omega^{-1}\omega a g_{P,S}(h(u)a, v) = f_S(u)\omega^{-1}g_{P,S}(h(u), av').$$

С това индукционната стъпка е направена, което показва, че за всеки две думи $u \in (\Sigma \cup \{\omega\})^*$ и $v\Sigma^*$ е в сила равенството:

$$f_S(uf_P(v)) = \begin{cases} f_S(u)\omega^{-1}g_{P,S}(h(u),v) \\ \text{ако } g_{P,S} \in \omega(\Sigma \cup \{\omega\})^* \text{ и } f_S(u) \in (\Sigma \cup \{\omega\})^*\omega, \\ \\ f_S(u)g_{P,S}(h(u),v) \text{ иначе.} \end{cases}$$

Следствие 8 *За всяко $v \in \Sigma^*$ е в сила, че:*

$$f_S(f_P(v)) = f_{P,S}(v).$$

Наистина тъй като $f_S(\varepsilon) = \varepsilon$ не завършва на ω, то от горното твърдение получаваме, че:

$$f_S(f_P(v)) = f_S(\varepsilon \circ f_P(v)) =$$
$$f_S(\varepsilon) g_{P,S}(\varepsilon, v) = \varepsilon \circ f_{P,S}(v) = f_{P,S}(v).$$

Сега ще покажем как можем да построим подпоследователен трансдюсер за f_S. Тогава използвайки конструкцията за композиция на два подпоследователни преобразувателя и подпоследователния преобразувател за f_P, ще можем да конструрираме и подпоследоватлен преобразувател за $f_S \circ f_P = f_{P,S}$.

Конструкцията е много подобна на тази за f_P, но тук трябва да отчетем и специалната роля на символа ω. Всъщност трябва да разграничаваме двата случая блок от произволен брой ω^k за $k > 0$ и буква от Σ. В зависимост от тези две ситуации ще имаме по две копия на всяко състояние от дървото за речника S.

Първо ще изложим алгоритъма и след това ще докажем неговата коректност.

$T \;=< \Sigma, Q, s, \delta, F >$ `-trie за речника` S
$Q_i = \{(q,i) \;\mid\; q \;\in\; Q\}$`,` $i = 1, 2$`.`
$A =< \Sigma \cup \{\omega\} \times (\Sigma \cup \{\omega\})^*, Q_1 \cup Q_2, (s,1), \widetilde{\delta}, \widetilde{\lambda}, \widetilde{\phi} >$
`- преобразувателят, който ще строим.`

`for` $\forall\; q \;\in\; Q$ `do`
    $\widetilde{\delta}((q,i),\omega) \leftarrow \;(q,2)$
    $\widetilde{\phi}((q,i)) \leftarrow \;\varepsilon$
    $\widetilde{\lambda}((q,2),\omega) \leftarrow \;\varepsilon$
    `if` $(q \;\in\; F)$
        $\widetilde{\lambda}((q,1),\omega) \leftarrow \;\varepsilon$
    `else`
        $\widetilde{\lambda}((q,1),\omega) \leftarrow \;\omega$

```
done

L₁ ← ∅
for ∀ a ∈ Σ do
    if (¬ !δ(s,a) )
        δ̃((s,j),a) ← (s,1), j = 1,2
        λ̃((s,j),a) ← a
    else
        p ← δ(s,a)
        δ̃((s,j),a) ← p, j = 1,2
        if (p ∈ F)
            λ̃((s,j),a) ← aω, j = 1,2
        else
            λ̃((s,j),a) ← a, j = 1,2
        L₁ ← L₁ ∪ {(p,s)}
done

i ← 1
while (Lᵢ ≠ ∅)do
    Lᵢ₊₁ ← ∅
    for ∀ (p,q) ∈ Lᵢ do
        if (p ∈ F) or (λ̃((q,1),ω) = ε)
            λ̃((p,1),ω) ← ε
        else
            λ̃((p,1),ω) ← ω
        for ∀ a ∈ Σ do
            if (¬ !δ(p,a))
                δ̃((p,j),a) ← δ̃((q,j),a), j = 1,2
                λ̃((p,j),a) ← λ̃((q,j),a), j = 1,2
            else
                p′ ← δ(p,a)
                δ̃((p,j),a) ← p′, j = 1,2
                if (p′ ∈ F) or (λ̃((q,1),a) = aω)
                    λ̃((p,j),a) ← aω, j = 1,2
```

```
                else
                      λ̃((p,j),a) ← a, j = 1,2
                L_{i+1} ← L_{i+1} ∪ {(p', δ̃(q,a))}
        done
    done
    i ← i + 1
done
```

Твърдение 18 *Нека за някое* i

$$L_i = \{(p, fail(p)) \,|\, |\alpha(p)| = i\}$$

и нека за всяко $q \in Q$, $a \in \Sigma$, *за които* $|\alpha(q)| < i$ *е в сила, че:*

$$\begin{aligned} \widetilde{\lambda}((q,1),\omega) = \omega &\Leftrightarrow f_S(\alpha(q)) \text{ не завършва с } \omega, \\ \widetilde{\lambda}((q,1),\omega) = \varepsilon &\Leftrightarrow f_S(\alpha(q)) \text{ завършва с } \omega, \\ \widetilde{\lambda}((q,j),a) &= f_S(\alpha(q))^{-1} f_S(\alpha(q)a), \end{aligned}$$

$$\widetilde{\delta}((q,j),a) = \delta^*(s,\beta),$$

където β *е най-дългата наставка на* $\alpha(q)$, *за която* $!\delta^*(s,\beta)$. *Тогава след конструирането на* L_{i+1} *е в сила, че:*

1. $L_{i+1} = \{(p, fail(p)) \,|\, \alpha(p) = i + 1\}$.

2. *За всяко* $q \in Q$, $a \in \Sigma$, *за които* $|\alpha(q)| < i + 1$ *е в сила, че:*

$$\begin{aligned} \widetilde{\lambda}((q,1),\omega) = \omega &\Leftrightarrow f_S(\alpha(q)) \text{ не завършва с } \omega, \\ \widetilde{\lambda}((q,1),\omega) = \varepsilon &\Leftrightarrow f_S(\alpha(q)) \text{ завършва с } \omega, \\ \widetilde{\lambda}((q,j),a) &= f_S(\alpha(q))^{-1} f_S(\alpha(q)a), \\ \widetilde{\delta}((q,j),a) &= \delta^*(s,\beta), \end{aligned}$$

където β *е най-дългата наставка на* $\alpha(q)$, *за която* $!\delta^*(s,\beta)$.

Доказателство: Доказателството на първата част на твърдението не се различава по нищо от доказателството на първата част на твърдение 16. Затова пристъпваме направо към доказателството на втората част.

Нека $p \in Q$ и $|\alpha(p)| = i$. Тогава по условие $(p, fail(p)) \in L_i$. Да разгледаме момента, в който алгоритъмът разглежда двойката $(p, fail(p))$. Тогава $f_S(\alpha(p))$ завършва с ω, точно когато $\alpha(p) \in \Sigma^* S$, тоест или когато $\alpha(p) \in S$ или когато съществува собствена наставка β на $\alpha(p)$, за която $\beta \in S$. Тогава обаче $!\delta^*(s, \beta)$, откъдето получаваме, че β е наставка и на $tail(\alpha(p)) = \alpha(fail(p))$. Тоест $f_S(\alpha(p))$ завършва с ω точно когато $\alpha(p) \in S$, което означава, че $p \in F$, или, когато $f_S(\alpha(fail(p)))$ завършва на ω. Сега поставяйки в контра позиция получаваме, че:

$$f_S(\alpha(p)) \in (\Sigma \cup \{\omega\})^* \omega \Leftrightarrow p \in F \text{ или } f_S(\alpha(fail(p))) \in (\Sigma \cup \{\omega\})^* \omega,$$

но според даденото, тъй като $|\alpha(fail(p))| < i$, последното е еквивалентно на:

$$f_S(\alpha(p)) \in (\Sigma \cup \{\omega\})^* \omega \Leftrightarrow p \in F \text{ или } \widetilde{\lambda}((fail(p), 1), \omega) = \varepsilon,$$

точно проверката, която извършва и алгоритъмът преди да определи дали $\widetilde{\lambda}((p, 1), \omega) = \varepsilon$ или $\widetilde{\lambda}((p, 1), \omega) = \omega$.

Нека сега $a \in \Sigma$ тогава, разсъждавайки както при доказателството на твърдение 16, лесно се вижда, че най дългата наставка β на $\alpha(p)a$ е или $\alpha(p)a$, или най-дългата наставка на $\alpha(fail(p))a$. В първия случай имаме, че:

$$\delta^*(s, \beta) = \delta(\delta^*(s, \alpha(p)), a) = \delta(p, a).$$

Оттук получаваме, че наистина $\widetilde{\delta}((p, j), a) = (\delta(p, a), 1) = (\delta^*(s, \beta), 1)$. Във втория случай, тъй като $\delta^*(s, \alpha(p)a)$ не е дефинирана, то не е дефинирана и $\delta(p, a)$. Тогава алгоритъмът определя $\widetilde{\delta}((p, j), a) = \widetilde{\delta}((fail(p), 1), a) = (\delta^*(s, \beta), 1)$ според предположението за $fail(p)$, защото $|\alpha(fail(p))| < i$.

Накрая ще покажем, че е в сила и равенството:

$$\widetilde{\lambda}((p, j), a) = f_S(\alpha(p))^{-1} f_S(\alpha(p)a).$$

Наистина по определение:

$$f_S(\alpha(p)a) = \begin{cases} f_S(\alpha(p))a \text{ ако } \alpha(p)a \notin \Sigma^*S \\ f_S(\alpha(p))a\omega \text{ иначе.} \end{cases}$$

Както и по горе $\alpha(p)a \in \Sigma^*S$ точно когато $\alpha(p)a \in S$, тоест $\delta(p,a) \in F$, или $\alpha(fail(p))a \in \Sigma^*S$. Алгоритъмът разглежда първия случай и ако той е налице, то правилно поставя $\widetilde{\lambda}((p,j),a) = a\omega$. В противен случай алгоритъмът проверява дали $\widetilde{\lambda}(fail(p),1),a) = a\omega$. Според предположението на твърдението имаме, че:

$$\widetilde{\lambda}((fail(p),1),a) = f_S(\alpha(fail(p)))^{-1} f_S(\alpha(fail(p))a),$$

което е точно $f_S(\alpha(p))^{-1} f_S(\alpha(p)a)$, когато $\alpha(p)a \notin S$. Оттук следва, че:

$$\widetilde{\lambda}((p,j),a) = f_S(\alpha(p))^{-1} f_S(\alpha(p)a).$$

С това твърдението е доказано.

Както при преобразувателя за представките, така и тук, лесно се вижда, че предпоставките на твърдение 18 са изпълнени за L_1 веднага след инициализиращия цикъл. Сега с индукция по i и прилагане на твърдение 18 виждаме, че твърдението е налице и за $l = \max\{|u| \mid u \in S\} + 1$, тоест:

$$L_l = \{(p, fail(p)) \mid |\alpha(p)| = l\}$$

и съответно за всяко $i < l$ и за всяко $q \in Q$, за което $|\alpha(q)| < l$ и за всяко $a \in \Sigma$:

$$\widetilde{\delta}((q,j),a) = \delta^*(s,\beta) \text{ за } j = 1,2$$
$$\widetilde{\lambda}((q,j),a) = f_S(\alpha(q))^{-1} f_S(\alpha(q)a) \text{ за } j = 1,2,$$

където β е най-дългата наставка на $\alpha(p)a$, за която $!\delta^*(s,\beta)$. Накрая получаваме и, че:

$$\widetilde{\lambda}((q,1),\omega) = \begin{cases} \omega \text{ ако } f_S(\alpha(p)) \notin \Sigma^*S \\ \varepsilon \text{ иначе.} \end{cases}$$

От дефиницията на l обаче имаме, че не съществуват думи в S с дължина по-голяма от $l-1$. Това означава, че $L_l = \varnothing$, тоест алгоритъмът спира. Поради същата причина останалите твърдения всъщност обхващат всички състояния $q \in Q$, защото за всяко $q \in Q$ е в сила, че $|\alpha(q)| < l$.

Сега ще докажем и коректност на алгоритъма:

Твърдение 19 *Нека $\alpha \in (\Sigma \cup \{\omega\})^*$ и нека β е най-дългата наставка на $h(\alpha)$, за която $!\delta^*(s, \beta)$. Тогава:*

1. *ако $\alpha \in (\Sigma \cup \{\omega\})^* \Sigma^+ \{\varepsilon\}$, то*
$$\widetilde{\delta}((s,1), \alpha) = (\delta^*(s, \beta), 1).$$
2. *ако $\alpha \in (\Sigma \cup \{\omega\})^* \omega$, то*
$$\widetilde{\delta}((s,1), \alpha) = (\delta^*(s, \beta), 2).$$
3. *$\lambda^*((s,1), \alpha) = f_S(\alpha)$.*

Доказателство: Доказателството е с индукция по дължината на α. При $\alpha = \varepsilon$ твърдението е очевидно.

Нека твърденията 1-3 са изпълнени за всяка дума с дължина ненадминаваща n и $\alpha = \alpha' \circ a$ е дума с дължина $n+1$. Нека β' е най-дългата наставка на $h(\alpha')$, за която $!\delta^*(s, \beta')$.

1. $a = \omega$. Тогава $\beta = \beta'$. Ако $\alpha' \notin (\Sigma \cup \{\omega\})^* \omega$, то от индукционната хипотеза имаме, че:
$$\widetilde{\delta}((s,1), \alpha') = (\delta^*(s, \beta'), 1) = (p, 1).$$
Тогава от дефиницията на $\widetilde{\delta}$ имаме, че $\widetilde{\delta}((p,1), \omega) = (p, 2)$. Това показва, че
$$\widetilde{\delta}((s,1), \alpha'\omega) = (p, 2),$$
точно условие 2 от твърдението. Ако пък $\alpha' \in (\Sigma^* \cup \{\omega\})^* \omega$, то отново от индукционното предположение:
$$\widetilde{\delta}((s,1), \alpha') = (\delta^*(s, \beta'), 1) = (p, 2).$$

Сега отново от дефиницията на $\widetilde{\delta}$, $\widetilde{\delta}((p,2),\omega) = (p,2)$. Сега, като заместим, получаваме, че:

$$\widetilde{\delta}((s,1),\alpha' a) = \widetilde{\delta}((\delta^*(s,\beta'),1),a) = (p,2).$$

Но от твърдение 18 знаем, че $\widetilde{\lambda}((p,1),\omega) = \omega$ само ако $\beta \in \Sigma^* S$. Тъй като β е най-дългата наставка на $h(\alpha)$, за която $\delta^*(s,\beta)$, то $h(\alpha) \in \Sigma^* S$ е еквивалентно на $\beta \in \Sigma^* S$. Това показва, че:

$$f_S(\alpha'\omega) = f_S(\alpha')\omega \Leftrightarrow f_S(\beta'\omega) = f_S(\beta')\omega.$$

Оттук получаваме, че ако $\alpha' \notin (\Sigma \cup \{\omega\})^*\omega$, то:

$$f_S(\alpha')^{-1} f_S(\alpha) = \widetilde{\lambda}((p,1),\omega),$$

откъдето заедно с индукционното предположение заключаваме, че:

$$\begin{aligned} f_S(\alpha) = f_S(\alpha')\widetilde{\lambda}((p,1),\omega) = \\ \widetilde{\lambda}^*((s,1),\alpha') \circ \widetilde{\lambda}((\widetilde{\delta}^*((s,1),\alpha')),\omega) = \\ \widetilde{\lambda}^*((s,1),\alpha). \end{aligned}$$

Ако пък $\alpha' \in (\Sigma \cup \{\omega\})^*\omega$, то лесно се забелязва, че:

$$f_S(\alpha) = f_S(\alpha').$$

Но тогава и $\widetilde{\lambda}((p,2),\omega) = \varepsilon$, откъдето получаваме, че

$$\begin{aligned} \widetilde{\lambda}^*((s,1),\alpha'\omega) = \widetilde{\lambda}^*((s,1),\alpha') \circ \widetilde{\lambda}((p,2),\omega) = \\ \widetilde{\lambda}^*((s,1),\alpha') = f_S(\alpha') = f_S(\alpha). \end{aligned}$$

С това този случай е изчерпан. Остана да разгледаме и случая, когато $a \in \Sigma$.

2. $a \in \Sigma$. Тогава $\alpha \in (\Sigma \cup \{\omega\})^*\Sigma^+$ и следователно β е най-дългата наставка на $\beta' a$, за която $!\delta^*(s,\beta')$. От индукционната хипотеза имаме, че:

$$\widetilde{\delta}((s,1),\alpha') = (\delta^*(s,\beta),j) = (p,j)$$

за $j = 1$ или $j = 2$. Но от свойствата на $\widetilde{\delta}$ имаме, че $\widetilde{\delta}((p, j), a) = (q, 1)$, където $q = \delta^*(s, \beta)$. Това показва първата част на твърдението. Втората също следва лесно. Именно:

$$f_S(\alpha')^{-1} f_S(\alpha' a) = a \Leftrightarrow f_S(\beta')^{-1} f_S(\beta' a) = a \text{ и}$$
$$f_S(\alpha')^{-1} f_S(\alpha' a) = a\omega \Leftrightarrow f_S(\beta')^{-1} f_S(\beta' a) = a\omega.$$

Но от твърдение 18, тъй като $|\beta'| < l$ следва, че:

$$\widetilde{\lambda}((p, j), a) = f_S(\beta')^{-1} f_S(\beta' a).$$

Тогава директно получаваме, че:

$$f_S(\alpha) = f_S(\alpha')\widetilde{\lambda}((p, j), a) = \widetilde{\lambda}^*((s, 1), \alpha') \circ \widetilde{\lambda}^*(\widetilde{\delta}^*((s, 1), \alpha'), a) = \widetilde{\lambda}((s, 1), \alpha).$$

С това твърдението е доказано, а с това и коректността на алгоритъма, защото $\phi((p, j)) = \varepsilon$, за всяко състояние на преобразувателя.

С това построихме и подпоследователен преобразувател за f_S, а както видяхме, при композицията на $f_S \circ f_P$ получаваме и подпоследователен преобразувател за $f_{P,S}$. Тоест с това показахме и алгоритмично как да построим подпоследователен преобразувател за $f_{P,S}$.

7.2 Алгоритъм за ефективно пресмятане на функциите $walk_P$, $print_P$, $rest_P$, $walk_Q$, $print_Q$ и $output_Q$

7.2.1 Функциите $walk_P$, $print_P$ и $rest_P$

Целта ни в тази част е да опишем алгоритми, които ефективно да пресмятат функциите $walk_P$, $print_P$, $rest_P$ и съответно $walk_Q$, $ouput_Q$, $print_Q$, дефинирани в частта за композиция на подпоследователен преобразувател с FIFO-трансдюсер.

Първо ще припомним тяхната дефиниция.

$$walk_P : P \times \Sigma^* \to P \cup Q$$
$$walk_P(p, \varepsilon) = p$$
$$walk_P(p, a \circ \alpha) = \begin{cases} \Delta(p, a) \text{ ако } \Delta(p, a) \in Q \\ walk_P(\Delta(p, a), \alpha) \text{ иначе.} \end{cases}$$

Съответно:

$$rest_P : P \times \Sigma^* \to \Sigma^*$$
$$rest_P(p, \varepsilon) = \varepsilon$$
$$rest_P(p, a \circ \alpha) = \begin{cases} \alpha \text{ ако } \Delta(p, a) \in Q \\ rest_P(\Delta(p, a), \alpha) \text{ иначе.} \end{cases}$$

и накрая:

$$print_P : P \times \Sigma^* \to \Gamma^*$$
$$print_P(p, \varepsilon) = \varepsilon$$
$$print_P(p, a \circ \alpha) = \begin{cases} \lambda(p, a) \text{ ако } \Delta(p, a) \in Q \\ \lambda(p, a) \circ print_P(\Delta(p, a), \alpha) \text{ иначе.} \end{cases}$$

Тук Δ и λ са дадени функции - които са част от FIFO-трансдюсера. Затова считаме, че обръщенията за тях могат да бъдат извършени за константно време.

Всъщност проблемът е да пресметнем стойностите на тези функции за всяко $p \in P$ и всяка дума от речник $D(= Out(\lambda'))$ - в нашия случай това беше областта от стойности на функциите на преходите на подпоследователния трансдюсер.

Тъй като първо ще разгледаме по-внимателно алгоритъм, свързан с пресмятането на тези три функции, дефиницията на останалите за момента ще отложим. Ще отбележим обаче, че параметрите, от които зависят на $walk_Q$, $ouput_Q$ и $print_Q$ се определят като:

$$\begin{aligned} \Lambda = \quad & \{rest_P(p_k, \alpha_k), \textstyle\prod_{i=1}^k print_P(p_i, \alpha_i)) \,|\, \alpha_0 \in D, p_i \in P \\ & \qquad\qquad \alpha_{i+1} = rest_P(p_i, \alpha_i)\} \end{aligned}$$

Единственото, което можем да кажем от тази дефиниция е, че $rest_P(p_i, \alpha_i)$ са наставки на някоя дума от D. Ние ще трябва да намерим и множеството Λ, затова не можем да се ограничим просто да намерим стойностите на $walk_P$, $print_P$, $rest_P$ върху множеството D, а ще ни трябват техните стойности върху множеството:

$$Suf(D) = \{y \,|\, \exists x\, xy \in D\}.$$

Разбира се можем да приложим директно дефиницията на трите функциии за всяка наставка на всяка дума от D, но това ще доведе до множество изчисления, които ще бъдат правени по повече от веднъж. Тук ще обясним как можем да се справим с този проблем. Идеята е подобна на тази на Breslauer,[Bre96]. Както в [Bre96] построяваме дърво(trie) за обърнатите думи от D - $T(D)$. Това ще ни помогне да представим по ефективно наставките на думите в D, като на всяка такава наставка ще й съответства еднозначно връх от дървото. След това ще приложим схемата на динамичното програмиране.

Сега ще формализираме тази идея. Нека за всеки връх $v \in T(D)$ с $\rho(v)$ означим етикетът на пътя от върха v до корена на дървото $T(D)$. Това, което искаме да направим е да намерим:

$$walk_P(p, \rho(v))$$
$$print_P(p, \rho(v))$$
$$rest_P(p, ro(v)).$$

От дефиницията на $T(D)$ е ясно, че ако $u \neq v$, то и $\rho(u) \neq \rho(v)$. Освен това за всяка дума $\alpha \in Suf(D)$ съществува връх в $T(D)$ със свойството, че $\rho(v) = \alpha$. Това показва, че броят на върховете в дървото $T(D)$ е точно равен на броя на различните наставки на думи от D. Сега желаните от нас стойности могат да бъдат пресметнати чрез обхождане в широчина на дървото $T(D)$, като за всеки възел $v \in T(D)$ намираме стойностите:

$$walk_P(p, \rho(v))$$
$$print_P(p, \rho(v))$$
$$rest_P(p, \rho(v))$$

за всяко $p \in P$. Поради естествения изоморфизъм, който съществува между върховете на $T(D)$ и наставките на D можем да си мислим, че $rest_P(p, \rho(v)) \in T(D)$. Поради същите съображения можем да си мислим, че вторият аргумент на $walk_P$, $print_P$ и $rest_P$ е връх на дървото $T(D)$.

Сега алгоритъмът може да се представи по следния начин:

$T(D)$ - дървото за обърнатите думи от D

r - коренът на $T(D)$

δ - функцията на преходите на $T(D)$

$parent(u)$ - предшественикът на u - по пътя му
към r в $T(D)$
$label(u, v)$ - етикетът по дъгата (u, v) в $T(D)$

```
for v ∈ T(D) do
     for p ∈ P do
        walk_P(p,v) ← undef
        print_P(p,v) ← undef
        rest_P(p,v) ← undef
    done
done

L_1 ← ∅
for p ∈ P do
    walk_P(p,r) ← p
    print_P(p,r) ← ε
    rest_P(p,r) ← r
    for ∀ a ∈ Σ do
        if (!δ(r,a))
            L_1 ← L_1 ∪ {δ(r,a)}
    done
done

i ← 1
while (L_i ≠ ∅) do
    L_{i+1} ← ∅
    for ∀ u ∈ L_i do
        father ← parent(u)
```

```
a ← label(father, u)
for ∀ p ∈ P do
    if (!Δ(p, a))
        q ← Δ(p, a)
    if (q ∈ Q)
        walk_P(p, u) ← q
        print_P(p, u) ← λ(p, a)
        rest_P(p, u) ← father
    else
        walk_P(p, u) ← walk_P(q, v)
        print_P(p, u) ← λ(p, a) ∘ print(q, father)
        rest_P(p, u) ← rest_P(q, father)
done
for a ∈ Σ do
    if (!δ(u, a))
        L_{i+1} ← L_{i+1} ∪ {δ(u, a)}
done
done
i ← i + 1
done
```

Коректността на алгоритъма следва директно от дефиницята на функциите и представянето на наставките на D. Освен това за всеки връх v на $T(D)$ пресмятаме $|P|$ на брой стойности, всяка от които изисква константно време - използваме, че стойностите за предшественика $parent(v)$ на v всички стойности са вече пресметнати и, че $\rho(v) = label(parent(v), v) \circ \rho(parent(v))$. Освен това всеки връх попада точно веднъж в някой от списъците L_i, защото има единствен предшественик, който може да го добави. Това означава, че времето необходимо за изпълнението на алгоритъма е $O(|P||T(D)|$. Сега обаче $|T(D)|$ е точно броят на различните наставки в D, и тъй като трябва да пресметнем $|P||T(D)|$ стойности, то това време е оптимално. Тук отново ще отбележим, че проблемът с конкатенацията може да бъде решена с подхода, който използват Михов и Schulz [SM06], тоест времето за конкатенеция не зависи от големината на думите и е $O(1)$.

Ще отбележим само, че времето за конструирането на $T(D)$ е пропорционално на $\sum_{\alpha \in D} |\alpha|$. Това означава, че времето за изпълнение на целия алгоретъм е:

$$O(\sum_{\alpha \in D} |\alpha| + |P||Suf(D)|).$$

7.2.2 Функциите $walk_Q$, $ouput_Q$ и $print_Q$

Първото нещо, което ще направим е да представим множеството Λ. Това ще направим в термините на последователните преобразуватели, въпреки че това не е съвсем точно. Ние няма да имаме единствено начално състояние, но въпреки това от всеки връх ще излиза единствена дъга със съответен символ от входната азбука. Тъй като ние няма да го използваме като разпознавател, това няма да играе съществена роля.

Идеята е състоянията на нашия автомат A да съответстват на онези наставки от D, които се получават при итерация на функцията $rest_P(p, \alpha)$ започвайки от някоя дума $\alpha \in D$, а думите $\prod_{i=1}^{k} print_P(p_i, \alpha_i)$ да се получават като думи прочетени по изходната лента, достигайки до състоянието $rest(p_k, \alpha_k)$. Това ни подсеща да дефинираме автомата:

$$A =< P, T(D), F(T(D)), \delta_A, \lambda_A >,$$

където P е входната азбука, $F(T(D))$ са финалните състояния в $T(D)$ - тоест онези състояния v, за които $\rho(v) \in D$, а функциите δ_A и λ_A се дефинират като:

$$\delta_A(v, p) = rest_P(p, v)$$
$$\lambda_A(v, p) = print_P(p, v).$$

При тази дефиниция ние искаме да разглеждаме само достижимите от $F(T(D))$ състояния. Това можем да направим с търсене в широчина. И така нека в резултат сме получили автомата:

$$A_R =< P, R(D), F(T(D)), \delta_A, \lambda_A >,$$

където $R(D)$ са достижимите от $F(T(D))$ върхове в A, а δ_A и λ_A са съответните рестрикции на двете изходни функции върху множеството $R(D)$.

Сега знаем следното:

$$\begin{aligned}(\rho(v), \gamma) \in \Lambda \Leftrightarrow \exists u \in F(T(D)) \text{ и } \pi \in P^* :\\ \delta_A^*(u, \pi) = v\\ \lambda_A^*(u, \pi) = \gamma\end{aligned}$$

Този факт е очевиден за $R(D)$, а след това лесно се разпространява с индукция по броя на прилаганията на $rest_P$ върху всички състояния от $R(D)$ и съответно двойки думи от Λ.

Преди да видим как ще използваме автомата A_R да си припомним дефиницията на функциите $walk_Q$, $ouput_Q$ и $print_Q$.

$$walk_Q : Q \times \Sigma^* \times \Gamma^* \to P \cup Q$$
$$walk_Q(q, \alpha, \varepsilon) = q$$

$$walk_Q(q, \alpha, b) = \begin{cases} d(q, b, 0) \text{ ако } d(q, b, 0) \in Q \\ \\ walk_P(d(q, b, 0), \alpha) \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in P \\ \\ walk_Q(q', \alpha', \gamma') \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 0), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 0), \alpha)), \\ \gamma' = print_P(d(q, b, 0), \alpha) \end{cases}$$

$$walk_Q(q, \alpha, b\gamma) = \begin{cases} walk_Q(d(q, b, 1), \alpha, \gamma) \text{ ако } d(q, b, 1) \in Q \\ \\ walk_P(d(q, b, 1), \alpha) \text{ ако } d(q, b, 1) \in P \text{ и} \\ walk_P(d(q, b, 1), \alpha) \in P \\ \\ walk_Q(q', \alpha', \gamma\gamma') \text{ ако } d(q, b, 1) \in P \text{ и} \\ walk_P(d(q, b, 1), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 1), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 1), \alpha)), \\ \gamma' = print_P(d(q, b, 1), \alpha) \end{cases}$$

Сътоветно функцията $ouput_Q$ се задава като:

$$ouput_Q : Q \times \Sigma^*\Gamma^* \to \Omega^*$$
$$ouput_Q(q, \alpha, \varepsilon) = \varepsilon$$

$$ouput_Q(q, \alpha, b) = \begin{cases} out(q, b, 0) \text{ ако } d(q, b, 0) \in Q \\ \\ out(q, b, 0) \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in P \\ \\ out(q, b, 0) \circ ouput_Q(q', \alpha', \gamma') \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 0), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 0), \alpha)), \\ \gamma' = print_P(d(q, b, 0), \alpha) \end{cases}$$

$$ouput_Q(q, \alpha, b\gamma) = \begin{cases} out(q, b, 1) \circ ouput_Q(d(q, b, 1), \alpha, \gamma) \text{ ако } d(q, b, 1) \in Q \\ \\ out(q, b, 1) \text{ ако } d(q, b, 1) \in P \text{ и} \\ walk_P(d(q, b, 1), \alpha) \in P \\ \\ out(q, b, 1) \circ ouput_Q(q', \alpha', \gamma\gamma') \text{ ако } d(q, b, 1) \in P \text{ и} \\ walk_P(d(q, b, 1), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 1), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 1), \alpha)), \\ \gamma' = print_P(d(q, b, 1), \alpha) \end{cases}$$

Накрая функцията $print_Q$ е дефинирана като:

$$print_Q : Q \times \Sigma^*\Gamma^* \to \Gamma^*$$
$$print_Q(q, \alpha, \varepsilon) = \varepsilon$$

$$print_Q(q, \alpha, b) = \begin{cases} \varepsilon \text{ ако } d(q, b, 0) \in Q \\ \\ print_P((q, b, 0), \alpha) \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in P \\ \\ print_Q(q', \alpha', \gamma') \text{ ако } d(q, b, 0) \in P \text{ и} \\ walk_P(d(q, b, 0), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 0), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 0), \alpha)), \\ \gamma' = print_P(d(q, b, 0), \alpha) \end{cases}$$

$$print_Q(q, \alpha, b\gamma) = \begin{cases} print_Q(d(q, b, 1), \alpha, \gamma) \text{ ако } d(q, b, 1) \in Q \\ \\ \gamma \circ print_P(d(q, b, 1), \alpha) \text{ ако } d(q, b, 1) \in P\text{и} \\ walk_P(d(q, b, 1), \alpha) \in P \\ \\ print_Q(q', \alpha', \gamma\gamma') \text{ ако } d(q, b, 1) \in P \text{ и } walk_P(d(q, b, 1), \alpha) \in Q, \\ \text{където } q' = walk_P(d(q, b, 1), \alpha), \\ \alpha' = rest_P(walk_P(d(q, b, 1), \alpha)), \\ \gamma' = print_P(d(q, b, 1), \alpha) \end{cases}$$

Първо ще покажем аналогията, която можем да направим между тези три функции и $walk_P$, $print_P$ и $rest_P$, съответно. Да разгледаме тройката (q, α, γ), където можем да смятаме, че $\alpha = \rho(v)$, $v \in R(D)$, а $\gamma = b_0 b_1 \dots b_n$ като на двойката (q, v) и думата $\widetilde{\gamma} = (b_0, 1)(b_1, 1) \dots (b_{n-1}, 1)(b_n, 0)$. Всяка буква b_i на γ се заменя с $(b_i, 1)$ с изключение на последната, която се заменя с $(b_n, 0)$. Да погледнем на двойката (q, v) като на "състояние". Тогава функциите $walk_Q$, $ouput_Q$

и $print_Q$ напомнят на функциите $walk_P$, $print_P$ и $rest_P$. Именно, намирайки се в дадено състояние (q, v) и буква (b, j) знаем точно в кое състояние (q', v') да преминем. В случая, когато $d(q, b, j) \in Q$ това е $(d(q, b, j), v)$ иначе това е $(walk_P(d(q, b, j), v), rest_P(d(q, b, j), v))$ и двете стойности вече са пресметнати и зависят само локално от буквата (b, j).

Аналогията щеше е да е пълна и щяхме да можем да решим по същия начин този проблем, както се справихме с $walk_P$, $print_P$ и $rest_P$ с подхода на Breslauer [Bre96], ако не трябваше да добавяме дума след γ. Тогава губим тази монотонност, с която разполагахме в първия случай и благодерение, на която успяхме да реализираме динамичното програмиране.

Все пак с малки промени ще успеем да приложим подобна стратегия на динамичното програмиране и в тази ситуация, като съществено ще използваме, че думите γ могат да се конкатенират само с думи от вида $print_P(p, v)$, тоест крайно множество от думи, което ние знаем.

И така както и в първата част, отново прилагаме идеята от [Bre96] и първо ще построим дърво за обърнатите думи, които са втори компоненти на двойка от Λ. Това можем да направим с обхождане в широчина, започвайки от всички върхове на A_R и четем етикетите от v до $F(D(T))$ отзад напред. По този начин получаваме едно дърво $\Gamma(T)$, което представя всички наставки на думи, които са втори аргументи в Λ. След това всички етикети b на ребро излизащо от корена на на $\Gamma(T)$ заменяме с $(b, 0)$, а всички останали с $(b, 1)$. Така ако за някой връх $\gamma(v) \in \Gamma(T)$ сме имали, че $\rho(\gamma(v)) = b_0 b_1 \dots b_n$ преди модификацията, след направената замяна ще имаме, че $\rho(\gamma(v)) = (b_0, 1)(b_1, 1) \dots (b_{n-1}, 1)(b_n, 0)$, което съответства точно на думите, които ни интересуват в схемата, която описахме.

Сега основната трудност, която трябва да преодолеем е за всяка наставка γ, представена в $\Gamma(T)$ и всяка дума $print_P(p, v)$ за $v \in T(D)$ ефективно да определяме наставката $\gamma print_P(p, v)$ ако такава съществува. Това означава за всеки връх $\gamma(v) \in \Gamma(T)$ и всяка дума $\{w_0, w_1, \dots w_n\} = \{print_P(p, v) \,|\, p \in P, v \in T(D)\}$ да можем да намерим върхове $\gamma(v_i) = \gamma(v)$ със свойството:

1. ако $\rho(\gamma(v)) = (b, 0)$, то $\rho(\gamma(v)(i)) = (b, 1)w_i$, ако $w_i \neq \varepsilon$ и $\rho(\gamma(v)) = (b_0)$ иначе.

2. ако $\rho(\gamma(v)) = a\gamma'(b, 0)$, то $\rho(\gamma(v, i)) = a\gamma'(b, 1)w_i$ ако $w_i \neq \varepsilon$ и $\rho(\gamma(v, i)) = a\gamma'(b, 0)$ иначе.

И тук можем да използваме обхождане в широчина. Всъщност тъй като $w_i = b_0^{(i)} b_1^{(i)} \ldots b_{k_i}^{(i)}$ са думи от езика, който разглеждаме, то в дървото $\Gamma(T)$ - което представя обърнатите думи, заменяйки последната буква b с $(b, 0)$, а всяка от останалите с $(b, 1)$ то в $\Gamma(T)$ съществуват върхове $r(0), r(1) \ldots r(k)$ със свойството:

$$\rho(r(i)) = (b_0^{(i)}, 1)(b_1^{(i)}, 1) \ldots (b_{k_i-1}^{(i)}, 1)(b_{k_i}^{(i)}, 0).$$

След като намерим върховете r_i можем да приложим търсене в широчина. От разсъждението по горе имаме, че ако $father = parent(u)$, а (b, j) е етикетът на дъгата $(father, u)$, то:

$$u(i) = \begin{cases} \delta(father(i), (b, 1)) \text{ ако } w_i \neq \varepsilon \\ u \text{ иначе.} \end{cases}$$

Така лесно получаваме алгоритъм, който за всяко v намира $v(i)$, ако те съществуват.

$\Gamma(T)$ `- дървото дефинирано по горе`

r `- коренът на` $\Gamma(T)$

$\{w_i \,|\, i = 0 \ldots\ k\} = \{print_P(p, v) \,|\, v \ \in\ R(D)\}$
δ `- функцията на преходите в` $\Gamma(T)$

$parent(u)$`- предшественикът на` u `в` $\Gamma(T)$`.`
$label(u, v)$ `- етикетът на реброто` (u, v)`,`
$\delta(u, (label(u, v))) = v$
`for` $i = 0$ `to` k `do`
$w \ \leftarrow\ w_i$

```
    l ← |w_i|
    p ← r
    if (l = 0)
        p ← δ(r, (w[l], 0))
    while l ≥ 1 do
        p ← δ(p, (w[l], 1))
        l ← l − 1
    done
    r[i] ← p
done

L_1 ← ∅
for ∀ ∈ Γ do
    if (!δ(r, (a, 0)))
        L_1 ← L_1 ∪ {δ(r, (a, 0))}
done

i ← 1 while L_i ≠ ∅ do
    L_{i+1} ← ∅
    for ∀ u ∈ L_i do
        father ← parent(u)
        (b, j) ← label(father, u)
        for count = 0 to k do
            if (w_count = ε)
                u[count] ← u
            else
                if (!father[count])
                    u[count] ← δ(father[count], (b, 1))
        done
        for ∀ a ∈ Γ do
            if (!δ(u, (a, 1)))
                L_{i+1} ← L_{i+1} ∪ {δ(u, (a, 1))}
        done
    done
    i ← i + 1
```

```
done
```

Коректността на алгоритъма следва от разсъжденията по горе. Неговата сложност също се изчислява лесно. Подобно на алгоритъма в предишната част се вижда, че времето за неговота изпълнение е $O(k|\Gamma(T)| + \sum_{i=0}^{k} |w_i|)$.

Сега можем да опишем и алгоритъма и който пресмята функциите $walk_Q$, $ouput_Q$, $print_Q$. Както в предишната част можем да отъждествим всяка дума $\gamma = b_0 b_1 \dots b_n \in \Gamma^*$ с връх $\gamma(v) \in \Gamma(T)$, за който $\rho(\gamma(v)) = (b_0, 1) \dots (b_{n-1}, 1)(b_n, 0)$. Както и преди отъждествяваме наставките на D с върховете на $T(D)$. Тогава можем да намерим на стойностите на функциите $walk_Q(q, u, v)$, $ouput_Q(q, u, v)$ и $print_Q(q, u, v)$ рекурсивно за всяко $q \in Q$ и всяка двойка (u, v), за която $(\rho(u), \rho(v)) \in \Lambda$.

```
Find(q, u, v)
    begin
        if (marked(q, u, v))
            return
        marked(q, u, v) ← true
        if (v = r)
            walk_Q(q, u, v) ← q
            print_Q(q, u, v) ← v
            output_Q(q, u, v) ← ε
            return
        (b, j) ← label(parent(v), v)
        q' ← d(q, b, j)
        if (q' ∈ Q)
            Find(q', u, parent(v))
            walk_Q(q, u, v) ← walk_Q(q', u, parent(v))
            print_Q(q, u, v) ← print_Q(q', u, parent(v))
            output_Q(q, u, v) ← out(q, b, j) ∘ output_Q(q', u, v')
        else
            p ← walk_P(q', u)
            u' ← rest_P(q', u)
            w ← print_P(q', u)
```

```
        i е такова, че w_i = w

        v' ← v[i]
        if (p ∈ P)
            walk_Q(q,u,v) ← p
            output_Q(q,u,v) ← out(q,b,j)
            print_Q(q,u,v) ← v'
        else
            Find(p,u',v')
            walk_Q(q,u,v) ← walk_Q(p,u',v')
            output_Q(q,u,v) ← out(q,b,j) ∘ output_Q(p,u',v')
            rest_Q(q,u,v) ← rest_Q(p,u',v')
    end
```

Коректността на горната процедура следва непосредствено от дефиницията на съответните функции заедно с разсъжденията направени по горе. Първоначално инициализираме масива *marked* със стойности $false$ за всички q и стойности $(u, v) \in \Lambda$. След това стартираме процедурата Find(q,u,v) за всяка такава тройка. Тъй като Find(q,u,v) се изпълнява само ако тройката е немаркирана, и съответно маркира тройката, то Find(q,u,v) изпълнява неповече от веднъж нетривиалната част. Но действията извън рекурсивната част отнемат константно време, а като използваме подхода на Михов и Schulz [SM06] конкатенацията за конструирането на $ouput_Q$ също може да бъде реализирана за константно време. Така общото време за изпълнение на алгоритъма е ограничено отгоре с $O(|Q||T(D)||\Gamma(T)|)$.

7.3 Алгоритъм за построяване на FIFO-трансдюсер по дадена Ω-представима функция

Сега ще дадем примерна реализация на алгоритъм, който по дадена Ω-орледелима функция f конструира FIFO-трансдюсер, който представя f.

Нека f се задава от R – Ω-крайна, дясно-инвариантна релация, за която имаме автомат:

$$A_R =< \Sigma \cup \Omega, Q_R, s_R, \delta_R, F_R > .$$

Релацията R Ω-определя с параметър k релацията L, за която p_0 е класът на всички думи $[(\varepsilon, \beta)]$. Накрая Rel и T са две матрици с размер $id(L) \times id(R)$, за които:

$$Rel[p][q] = r,$$

тогава и само тогава, когато за всеки елемент $p = [(\alpha, \beta)]$, а $q = [\beta']$, то $r = [\alpha P_1(\beta'), S_1(\beta')]$. Матрицата T е матрицата от трансдюсери:

$$T[p][q] =< (\Sigma \cup \Omega) \times \Xi^*, Q_{p,q}, s_{p,q}, \delta_{p,q}, \lambda_{p,q}, \phi_{p,q} > .$$

Алгоритъмът следва конструкцията, която описахме по-рано в теорема 7 и дефинира компонентите на FIFO-трансдюсера:

$$\widetilde{T} =< (\Sigma \cup \Omega) \times \Xi^*, \Sigma, P, Q, s, \Delta, d, \lambda, out, \phi, \psi >,$$

със свойството:

$$f_{\widetilde{T}} = f.$$

Всяко състояние $p \in P$ ще се описва със следните полета:

1. $p.l$ състояние на релацията L,

2. $1 \leq p.count \leq k$

3. $p.R_j$ състояния на автомата A_R а всяко $j \leq p.count$.

Състоянията $q \in Q$ ще описваме в сходни термини:

1. $q.l$ състояние на релацията L,

2. $q.r$ състояние на автомата A_R,

3. $q.t$ състояние на подпоследователния преобразувател $T[q.l][q.r]$,

4. $0 \leq q.count \leq k$ и

5. $q.R_j$ за $1 \leq j \leq q.count$ състояния на автомата A_R.

```
Build_FIFO_Transducer(A_R, L, p_0, Rel, T, k)
    begin
        P[0] ← newstate_P
        P[0].l ← p_0
        P[0].count ← 1
        P[0].R_1 ← s_R

        QueueP ← ∅
        QueueQ ← ∅
        EnQueue(QueueP, P[0])
        M ← 1
        N ← 0
        while (QueueP ≠ ∅ or QueueQ ≠ ∅ do
            if(QueueP ≠ ∅)
                P[k] ← DeQueue(QueueP)
                FixStateP(P[k], P, Q, QueueP, QueueQ, M, N, A_R, L, Rel, T, k)
            else
                Q[k] ← DeQueue(QueueQ)
                FixStateQ(Q[k], P, Q, QueueP, QueueQ, M, N, A_R, L, Rel, T, k)
        done
        T̃ ← < P, Q, P[0], Δ, d, λ, out, φ, ψ >
        return T̃
    end

FixStateP(p, P, Q, QueueP, QueueQ, M, N, A_R, L, Rel, T, k)
    begin
        for ∀ a ∈ Σ do
            p' ← newstateP()
            p'.l ← p.l
            p'.count ← p.count
            for i = 1 to p.count do
                p'.R_i ← δ_R(p.R_i, a)
            done
            if ∃ P[j] = p'
```

```
        Δ(p,a) ← P[j]
        free(p')
    else
        P[M] ← p'
        EnQueueP(QueueP, P[M])
        Δ(p,a) ← P[M]
        M ← M + 1
    λ(p,a) ← a
done
for ∀ a ∈ Ω do
    if(p.count < k)
        p' ← newstateP()
        p'.l ← p.l
        p'.count ← p.count + 1
        for i = 1 to p.count do
            p'.R_i ← δ_R(p.R_i, a)
        done
        p'.R_{p.count+1} ← s_R
        if ∃ P[j] = p'
            Δ(p,a) ← P[j]
            free(p')
        else
            P[M] ← p'
            EnQueueP(QueueP, P[M])
            M ← M + 1
    else
        q' ← newstateQ()
        q'.l ← p.l
        q'.r ← p.R_1
        q'.t ← s[q'.l, q'.r]
        q'.count ← k
        for i = 1 to k − 1
            q'.R_i ← p.R_{i+1}
        done
        q'.R_k ← s_R
```

```
                if ∃ Q[j] = q'
                    Δ(p, a)  ←  Q[j]
                    free(q')
                else
                    Q[N]  ←  q'
                    EnQueueQ(QueueQ, Q[N])
                    Δ(p, a) ←  Q[N]
                    N  ←  N + 1

            λ(p, a)  ←  a
        done
        q'  ←  newstateQ()
        q'.l  ←  p.l
        q'.r  ←  p.R_1
        q'.t  ←  s[q'.l, q'.r]
        q'.count  ←  p.count − 1
        for i = 1  to p.count − 1 do
            q'.R_i  ←  p.R_{i+1}
        done
        if ∃ Q[j]  =  q'
            φ(p)  ←  Q[j]
            free(q')
        else
            Q[N]  ←  q'
            φ(p)  ←  Q[N]
            EnQueue(QueueQ, Q[N])
            N  ←  N + 1
    end

FixStateQ(q, P, Q, QueueP, QueueQ, M, N, A_R, L, Rel, T, k)
    begin
        for ∀ a ∈ Σ do
            q'  ←  newstateQ()
            q'.l  ←  q.l
            q'.r  ←  q.r
```

```
    q'.t  ←  δ[q.l][q.r](q.t, a)
    q'.count  ←  q.count
    for i = 1 to q.count do
        q'.R_i  ←  q.R_i
    done
    if ∃ Q[j]  =  q'
        d(q, a, 1)  ←  Q[j]
        d(q, a, 0) ←  Q[j]
        free(q')
    else
        Q[N]  ←  q'
        EnQueueP(QueueP, Q[N])
        d(q, a, 0) ←  Q[N]
        d(q, a, 1) ←  Q[N]
        N  ←  N + 1
    out(q, a, 1)  ←  λ[q.l][q.r](q.t, a)
    out(q, a, 0)  ←  out(q, a, 1)  ∘ φ[q.l][q.r](λ[q.l][q.r](q.t, a))
done
for ∀ a  ∈  Ω do
    if (q.count = k)
        p  ←  newstateP()
        p.l  ←  Rel[q.l][q.r]
        p.count  ←  k
        for i = 1 to k do
            p.R_i  ←  q.R_i
        done
        st'  ←  δ[q.r][q.l](q.t, a)
        if (!φ[q.l][q.r](st')
             if ∃ P[j]  =  p
                d(q, a, 1)  ←  P[j]
                free(p)
            else
                P[M]  ←  p
                EnQueueP(QueueP, P[M])
                d(q, a, 1) ←  P[M]
```

```
                        $M \leftarrow M + 1$
                    $out(q, a, 1) \leftarrow \lambda[q.l][q.r](q.t, a) \circ \phi(t')$
                else
                    $free(p)$
            else
                $t' \leftarrow \delta[q.l][q.r](q.t, a)$
                if$(!\phi[q.l][q.r](t'))$ and $(q'.count > 0)$
                    $q' \leftarrow newstateQ()$
                    $q'.l \leftarrow Rel[q.l][q.r]$
                    $q'.r \leftarrow q.R_1$
                    $q'.t \leftarrow s[q'.l][q'.r]$
                    $q'.count \leftarrow q.count - 1$
                    for $i =$ 1 to $q.count - 1$
                        $q'.R_i \leftarrow q.R_{i+1}$
                    done
                    if $\exists\ Q[j] = q'$
                        $d(q, a, 1) \leftarrow Q[j]$
                        $d(q, a, 0) \leftarrow Q[j]$
                        $free(q')$
                    else
                        $Q[N] \leftarrow q'$
                        $EnQueue(QueueQ, Q[N])$
                        $d(q, a, 1) \leftarrow Q[N]$
                        $d(q, a, 0) \leftarrow Q[N]$
                        $N \leftarrow N + 1$
                    $out(q, a, 1) \leftarrow \lambda[q.l][q.r](q.t, a) \circ \phi[q.l][q.r](t')$
                    $out(q, a, 0) \leftarrow out(q, a, 1)$
        done
        if$(q.count = 0)$
            $\psi(q) \leftarrow \varepsilon$
    end
```

7.4 Композициа на две Ω-представими функции с параметри M и m. Алгоритъм

Нека f и g са две Ω-крайно представими функции. Нека M и m са константите, които определят възможността да конструираме $g \circ f$. Тук ще представим алгоритъм, който по зададени k_f, $\sim_{R_f}$ и $\sim_{L_f}$ и подпоследователни преобразуватели $T^f_{[\alpha][\beta]}$, съответно k_g, R_g и $\sim_{L_g}$ и $T^g_{[\alpha][\beta]}$, конструира съответните компоненти на:

$$h(u,v) = g^*(f(\varepsilon, uv)f(\alpha,\beta)^{-1}, f(\alpha,\beta)),$$

така както са описани при доказателството на теоремата за композиция.

Първо ще фиксираме вида на входните данни. Регулярните релации ще бъдат представени чрез краен детерминиран автомат, чиито състояния са точно класовете на еквивалентност на съответната релация. Релациите $\sim_L$ ще се задават с техните класове на еквивалентност $\{p_1, p_2 \dots p_n\}$ и матрица на Ω-преходите $\sim_L \times \sim_R$, която на двойката класове на еквивалентност $[(\alpha,\beta)]_{\sim_L}, [\beta_1]_{\sim_R}$, съпоставя $[(\alpha P_1(\beta_1), S_1(\beta_1)]_{\sim_L}$. За всеки елемент на тази матрица имаме и по един подпоследователен преобразувател $T_{[(\alpha,\beta)][\beta_1]}$.

Започваме с алгоритъма, който намира класовете на еквивалентност на $\sim_{R_h}$. Следваме директно дефиницията на релацията $\sim_{R_h}$. За всяко състояние Q ще имаме следните полета:

1. $Q.l$, което ще бъде Ω-дължината на думите от този клас ако, тя е $< M(m+k_g)+k_f+1$ и $M(m+k_g)+k_f+1$ в противен случай.

2. $Q.l$ на брой състояния $Q.q^f_0, Q.q^f_1, \dots Q.q^f_l$ състояния на автомата на $\sim_{R_f}$, за съответната наставка $S_j(\beta)$, $S_0(\beta) = \beta$. С други думи това ще означава, че за всяка дума β от класа на Q, $S_j(\beta)$ е в класа q^f_j на релацията $\sim_{R_f}$.

3. $Q.Set$ множество, чиито $Q.Set.s$ елементи имат следната структура:

   (a) $s.j \leq Q.l+1$ и съответно масив от $s.j$ трансдюсера: $s.\overline{T} = \{T^f_{i_1}, T^f_{i_2} \dots T^f_{i_j}\}$, които участват в дефиницията на f.

(б) $s.l_g \leq k_g + m + 1$ Ω'-дължината на думата $\prod_{k=1}^{j} T_{i_k}^f(I_k(\beta))$ за всяка дума от класа на Q ако тази дължина е по-малка от $k_g + m$ и $k_g + m + 1$ в противен случай.

(в) $s.start^g$, състояние от автомата на $\sim_{R_g}$ и състояния $s.q_0^g, s.q_1^g, \dots s.q_{l_g}$, които съответстват на класовете на еквивалентност на

$$S_l(\omega \prod_{k=1}^{s.l_g} T_k^f(I_k(\beta)))$$

относно релацията $\sim_{R_g}$, където ω е произволна дума от класа на еквивалентност на $s.start^g$, а за β можем да си мислим като за дума от класа на Q.

(г) $s.finish^f$, това е състоянието в $T_{s.l_g}^f$, до което стигаме след като сме прочели $I_{s.l_g}(\beta)$, тоест:

$$(\delta_{s.l_g}^f)^*(s_{s.l_g}^f, I_{s.l_g}(\beta)).$$

(д) $s.T_u^g$ и $s.q_u^g$, подпоследователен преобразувател от дефиницията на g и съответно състояние в този преобразувател със свойството:

$$(\delta_u^g)^*(s_u^g, I_{s.l_g+1}(\beta)).$$

Това са компонентите на всяко състояние. Те съответстват точно на дефиницията на релацията $\sim_{R_h}$, така че трябва да е ясно, че съществуват краен брой такива. Все пак, за да подсилим тази интуиция, можем да отбележим, че параметрите, които определят състоянието са краен брой. Имаме два брояча $Q.l$, който определя най много n^l възможности за множеството $Q.q_i^f$. За всеки елемент на $Q.set$ също имаме горна граница $Q.set.s.l_g \leq M(m + k_g) + 1$. При фиксирано $s.l_g$ има краен брой наредени $s.l_g$-орки от трансдюсери, съответно от състояния $s.q_i^g$. Елементът s се определя от още три параметъра - $s.start^g$, $s.finish^f$ и $s.q_u^g$, всяко от които може да приема краен брой стойности. Това означава, че възможните елементи са краен брой, а оттук и всички възможни множества $Q.Set$.

За да построим автомат за релацията $\sim_{R_h}$ трябва да определим началното състояние и след това да направим търсене в ширина, генерирайки новите състояния Q и поставяйки необходимите преходи.

Началното състояние трябва да отговаря на празната дума ε. Така получаваме, че:

$Q[0] \;\leftarrow\; makestate()\;\; Q[0].l = 0$
$Q[0].q_0^f = s_0$ `началното състояние на`
$sim_{R_f}\;\; Q.Set \leftarrow\; \varnothing$
`for` $\forall\; T_{p,q}^f$ `-`
    `преобразувател на` f
     `for` $\forall\; st \;\in\; StatesOf(\sim_{R_g})$ `do`
        `for` $\forall\; T_u^g$
            $s \;\leftarrow\; new()$
            $s.j \;\leftarrow\; 1$
            $s.l_g \;\leftarrow\; 0$
            $s.T_1 \;\leftarrow\; T_{p,q}^f$
            $s.start_g \;\leftarrow\; st$
            $s.q_0^g \;\leftarrow\; st$
            $s.finish^f \;\leftarrow\; s_{p,q}^f$
            $s.q_u^g \;\leftarrow\; s_u^g$ `- начално състояние на` T_u^g`.`
            $Q[0].S \;\leftarrow\; Q[0].S \;\cup\; \{s\}$
        `done`
    `done`
`done`

Сега конструираме останалите състояния и Δ функцията на преходите в широчина:

$Queue \;\leftarrow\; \varnothing\;\; EnQueue(Queue, Q[0])$
$N \leftarrow\; 1$ `while` $Queue \;\neq\; \varnothing$ `do`
    $Q \leftarrow\; DeQueue(Queue)$
    $l \;\leftarrow\; Q.l$
    `for` $a \;\in\; \Sigma$ `do`
        `set_ordinary`$(Q, a, \Delta, Queue, N)$
    `done`

```
    for  a ∈ Ω  do
        if (l ≤ M(k_g + m) + k_f)
            set_omega(Q, a, Δ, Queue, N)
        else
            set_ordinary(Q, a, Δ, Queue, N)
    done
done

set_ordinary (Q, a, Δ, Queue, N)
    begin
        P ← newstate()
        P.l ← Q.l
        for  j = 0  to  Q.l  do
            P.q_j^f ← δ_{R_f}(Q.q_j^f, a)
        done
        P.Set ← ∅
        for ∀ s ∈ Q.Set  do
            if (s.j < Q.l + 1)
                P.Set ← P.Set ∪ {s}
            else
                t ← new()
                t.j ← s.j
                t.T̄ ← s.T̄
                t.start^g = s.start^g
                t.finish^f ← δ^f_{t.j}(s.finish^f, a)
                α ← λ^f_{t.j}(s.finish^f, a)
                len ← |α|_{Ω'}
                t.l_g ← min{s.l_g + len, m + k_g}
                for count = 0 to t.l_g do
                    if ( count ≤ s.l_g)
                        t.q^g_count ← δ*_{R_g}(s.q^g_count, α)
                    else
                        t.q^g_count ← δ*_{R_g}(s_g, S_{count−s.l_g}(α))
                done
                if (len = 0)
```

```
                $t.q^g_u \leftarrow (\delta^g_u)^*(q^g_u, \alpha)$
            else
                $t.q^g_u \leftarrow (\delta^g_u)^*(s^g_u, I_{t.l_g\ -s.l_g}(\alpha))$
            if  $t$ е дефинирано коректно
            $P.Set \leftarrow P.Set \cup \{t\}$
        done
        if $\exists\ Q[i] = P$
            $\Delta(Q,a) \leftarrow Q[i]$
            $free(P)$
        else
            $Q[N] \leftarrow P$
            $\Delta(Q,a) \leftarrow Q[N]$
            $EnQueue(Queue, Q[N])$
            $N \leftarrow N+1$
    end
```

По аналогичен начин се дефинира и функцията, която даобавя преход със символ от Ω. Сега обаче трябва да добавим и нови елементи и с всеки възможен преобразувател, дефиниращ f и да отчетем ϕ функцията на последния преобразувател във всяко финално състояние:

```
set_ordinary $(Q, a, \Delta, Queue, N)$
    begin
        $P \leftarrow newstate()$
        $P.l \leftarrow Q.l + 1$
        for  $j = 0$ to $Q.l$ do
            $P.q^f_j \leftarrow \delta_{R_f}(Q.q^f_j, a)$
            $P.q_{P.l} \leftarrow s_{R_f}$
        done
        $P.Set \leftarrow \varnothing$
        for $\forall\ s \in Q.Set$ do
            if( $(s.j < Q.l+1)$)
                $P.Set \leftarrow P.Set \cup \{s\}$
            else
                $t \leftarrow new()$
```

$t.j \leftarrow s.j$
$t.\overline{T} \leftarrow s.\overline{T}$
$t.start^g = s.start^g$
$t.finish^f \leftarrow \delta^f_{t.j}(s.finish^f, a)$
$\alpha \leftarrow \lambda^f_{t.j}(s.finish^f, a) \circ \phi_{t.j} spf(t.finish^f)$
$len \leftarrow |\alpha|_{\Omega'}$
$t.l_g \leftarrow \min\{s.l_g + len, m + k_g\}$
`for` $count = 0$ `to` $t.l_g$ `do`
  `if` $(count \leq s.l_g)$
    $t.q^g_{count} \leftarrow \delta^*_{R_g}(s.q^g_{count}, \alpha)$
  `else`
    $t.q^g_{count} \leftarrow \delta^*_{R_g}(s_g, S_{count-s.l_g}(\alpha))$

`done`
`if` $(len = 0)$
  $t.q^g_u \leftarrow (\delta^g_u)^*(q^g_u, \alpha)$
`else`
  $t.q^g_u \leftarrow (\delta^g_u)^*(s^g_u, I_{t.l_g \; -s.l_g}(\alpha))$
`if` t `е добре дефинирано`
  $P.Set \leftarrow P.Set \cup \{t\}$
  `for` $\forall\ T^f$ `do`
    $t' \leftarrow copy(t)$
    $t'.j \leftarrow t.j + 1$
    $t'.T_{t'.j} = T^f$
    $t'.finish\hat{\ }f = s\_t'.j$
    $P.Set \leftarrow P.Set \cup \{t'\}$
  `done`
`if` $\exists\ Q[i] = P$
  $\Delta(Q, a) \leftarrow Q[i]$
  $free(P)$
`else`
  $Q[N] \leftarrow P$
  $\Delta(Q, a) \leftarrow Q[N]$
  $EnQueue(Queue, Q[N])$
  $N \leftarrow N + 1$

```
end
```

Сега ще покажем как се конструира релацията $\sim_{L_h}$. За нея също ще се ръководим от конструкцията в теоремата. За да я определим ни трябват два класа на еквивалентност от L_f и L_g заедно със състояние от автомата за $\sim_{R_g}$, което да съответства на еквивалентността на наставките и състояние от някой от подпоследователните преобразуватели за g. Тоест всяко състояние или клас на еквивалентност (ние утъждествяваме двете) на L_h - P може да се представи чриз следните четири полета:

1. $P.p_f$ - едно състояние от релацията $\sim_{L_f}$
2. $P.p_g$ - едно състояние от релацията $\sim_{L_g}$
3. $P.q^g_u$ - състояние от подпоследователен трансдюсер за g и
4. $P.q^g_0$ - състояние на релацията $\sim_{R_g}$.

Следващата процедура създава всички необходими състояния на $\sim_{L_h}$, таблицата $\sim_{L_h} \times \sim_{R_h}$ и съответните трансдюсери.

```
N ← 0 for ∀ p_f ∈ L_f do
    for∀ p_g ∈ L_g do
        for∀ q^g_u ∈ T^g_u do
            for∀ q^g_0 ∈ R_g do
                P[N] ← newstate()
                P[N].p_f ← p_f
                P[N].p_g ← p_g
                P[N].q^g_u ← q^g_u
                P[N].q^g_0 ← q^g_0
                N ← N + 1
            done
        done
    done
done
```

```
for  ∀ Q ∈ R_h  do
    for  k = 0  to  N  do
        for ∀ s ∈ Q.Set  do
            if (s.j = 1)  and  (s.start^g = P[k].q_0^g) and  s.T_1 = T_{[P.p_f][Q.q_0^f]}
                l ← s.l_g
                T ← T_u^g(P[k].q_u^g)
                p_g ← P[k].p_g
                for  count = 1  to  l
                    p_g ← (L_g × R_g)(p_g, s.q_count^g)
                    T ← concat(T, T^g_{[p_g][s.q^g_count]})
                done

            s_0 ∈ Q.Set  за което
            s_0  се различава от  s  само по
            s_0.q_u^g   и
            s_0.T_u^g = T^g_{[p_g][s.q^g_count]}

            P' е състоянието в L_h, за което:
            P'.p_g = p_g
            P'.p_f = (L_f × R_f)[P[k].p_f][Q.q_0^f]
            P'.q_u^g = s_0.q_u^g
            P'.q_0^g = s.q_0^g

            (L_h × R_h)[P[k]][Q] = P'
            T_{[P[k]][Q]} = T^f_{[P[k].p_f][Q.q_0^f]} ∘ T
        done
    done
done
```

7.5 Намиране на компонентите на Ω-представима функция, еквивалентна на дадено правило с ограничен брой маркери Ω. Алгоритъм

Следва примерна реализация на алгоритъма, който по дадено правило с ограничен брой маркери Ω намира Ω-представима функция, която е еквивалентна на функцията на заместване, породена от даденото правило.

Нека $\mathcal{R}$, $E \to T(E)\backslash L\_R$ е правило за заместване с ограничен брой маркери. Нека още са дадени константите e и r със свойството:

$$E \cup (\Sigma^*\Omega\Sigma^*)^{>e} = \varnothing$$
$$R \cup (\Sigma^*\Omega\Sigma^*)^{>r} = \varnothing.$$

Знаем, че $L \subset (\Sigma \cup \Omega)^*\Omega$, $R \subset \Omega(\Sigma \cup \Omega)^*$ и T е подпоследователен преобразувател.

Алгоритъмът следва конструкцията, описана в доказателството теоремата 13.

Започваме с конструкцията на Ω-крайната, дясно-инвариантна релация $\sim_2$. Тя се дефинира като:

$$\beta' \sim_2 \beta''$$

тогава и сама тогава, когато:

1. за всеки $0 \leq i \leq e$ за всяко $\gamma \in (\Sigma \cup \Omega)^*$

$$S_1(\beta')\gamma \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma \cup \Omega)^* \Leftrightarrow$$
$$S_1(\beta'')\gamma \in (E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma \cup \Omega)^*$$

и

$$S(\beta')\gamma \in (E \cap (\Sigma^*\Omega)^i\Sigma^+)R(\Sigma \cup \Omega)^* \Leftrightarrow$$
$$S_1(\beta'')\gamma \in (E \cap (\Sigma^*\Omega)^i\Sigma^+)R(\Sigma \cup \Omega)^*$$

2. За всеки $\omega' \sim_{(\Sigma\cup\Omega)^*L} \omega''$ е в сила, че $\omega' P_1(\beta') \sim_{(\Sigma\cup\Omega)^*L} \omega'' P_1(\beta'')$.

3. За всяко $q \in Q$:

$$\delta^*(q, P_1(\beta')) = \delta^*(q, P_1(\beta'')).$$

Ще смятаме, че са ни дадени минимални, крайни, детерминирани автомати:

$$A_i^+ =< \Sigma \cup \Omega, Q_i^+, s_i^+, \delta_i^+, F_i^+ > \text{ и }$$
$$A_i =< \Sigma \cup \Omega, Q_i, s_i, \delta_i, F_i >$$

за $0 \leq i \leq e$ за езиците:

$$(E \cap (\Sigma^*\Omega)^i\Sigma^+)R(\Sigma \cup \Omega)^* \text{ и съответно}$$
$$(E \cap (\Sigma^*\Omega)^i\Omega)R(\Sigma \cup \Omega)^*.$$

Ясно е, как те могат да се построят ако са ни дадени регулярните изрази E и R. Ще смятаме още, че ни е даден детерминираният автомат:

$$A_L =< \Sigma \cup \Omega, Q_L, s_L, \delta_L, F_L >$$

за езика $(\Sigma \cup \Omega)^*L$. Нека:

$$T =< (\Sigma \cup \Omega) \times \Xi^*, Q_T, s_T, \delta_T, \lambda_T, \phi_T >$$

е представянето на подпоследователния преобразувател T. При тези означения, първо ще построим краен детерминиран автомат, чиито състояния ще отразяват класовете на еквивалентност на $\sim_2$. Всяко състояние Q на нашия автомат ще съдържа следните компоненти:

1. $Q.flag$, който ще има стойност 0, тогава и само тогава, когато всяка дума от този клас е дума от Σ^*, 1, ако думите от този клас са подмножество на $\Sigma^*\Omega$ и 2 в противен случай.

2. $Q.q_i$ за $0 \leq i \leq e$, които ще съответстват на факта, че за всяка дума β от класа на Q е в сила, че:

$$\delta_i^*(s_i, S_1(\beta)) = Q.q_i.$$

3. $Q.q_i^+$ за $0 \leq i \leq e$, които ще съответстват на факта, че за всяка дума β от класа на Q е в сила, че:

$$(\delta_i^+)^*(s_i^+, S_1(\beta)) = Q.q_i^+.$$

4. $Q.l_q$ за всяко състояние $q \in Q_L$, което ще съответства на:

$$\delta_L^*(q, P_1(\beta)) = Q.l_q$$

за всяко β от класа Q а релацията $\sim_2$.

5. $Q.t_q$ за всяко състояние $q \in Q_T$, което ще съответства на:

$$\delta_T(q, P_1(\beta)) = Q.t_q,$$

за всяко β от класа на Q.

Следващият алгоритъм намира състоянията, тоест непразните класове на еквивалентност на релацията $\sim_2$ и функцията на преходите между тях - Δ.

$Q[0] \leftarrow newstate()$ $Q[0].flag \leftarrow 0$ `for`
$i = 0$ `to` e `do`
    $Q[0].q_i \leftarrow s_i$
    $Q[0].q_i^+ \leftarrow s_i^+$
`done` `for` $\forall\ q \in Q_L$ `do`
    $Q[0].l_q \leftarrow q$
`done` `for` $\forall\ q \in Q_T$ `do`
    $Q[0].t_q \leftarrow q$
`done`

$Queue \leftarrow \varnothing$ $EnQueue(Queue, Q[0]$
$N \leftarrow 1$ `while` $Queue \neq \varnothing$ `do`
        $Q[k] \leftarrow DeQueue(Queue)$
        `for` $\forall\ a \in \Sigma \cup \Omega$ `do`
            $P \leftarrow newstate()$
            `if` $(Q[k].flag = 0)$
                `if` $(a \in \Sigma)$
                    $P.flag \leftarrow 0$
                `else`
                    $P.flag \leftarrow 1$
            `else`

```
            P.flag  ←  2
        if (P.flag  <  2)
            for i = 0 to e do
                P.q_i^+ ←  Q[k].q_i^+
                P.q_i  ←  Q[k].q_i
            done
            for ∀ q  ∈  Q_L do
                P.l_q  ←  δ_L(Q.l_q, a)
            done
            for ∀ q  ∈  Q_T
                P.t_q  ←  δ_T(Q[k].t_q, a)
            done
        else
            for i = 0 to e do
                st^+  ←  Q[k].q_i^+
                P.q_i^+ ← δ_i^+(st^+, a)
                st ←  Q[k].q_i
                P.q_i  ←  δ_i(st, a)
            done
            for ∀ q  ∈  Q_L do
                P.l_q  ←  Q.l_q
            done
            for ∀ q  ∈  Q_T
                P.t_q  ←  Q[k].t_q
            done
        if ∃ Q[i]  =  P
             Δ(Q[k], a)  ←  Q[i]
             free(P)
        else
             Q[N]  ←  P
             N  ←  N  +  1
             EnQueue(Queue, Q[N])
    done
done
```

Сега релацията $\sim_2$ се задава от автомата $A_2 = < \Sigma \cup \Omega, Q, Q[0], \Delta, Q >$.

Сега преминаваме към алгоритъма за релацията $\sim_1$. Да си припомним, че:

$$(\alpha', \beta') \sim_1 (\alpha'', \beta'')$$

точно когато:

1. $\alpha' \sim_{(\Sigma \cup \Omega)^* L} \alpha''$
2. $state(\alpha', \beta') = state(\alpha'', \beta'')$
3. $count(\alpha', \beta') = count(\alpha'', \beta'')$.

Така всяко състояние или клас на еквивалентност, понятия които ние утъждествяваме, P на релацията $\sim_1$ може да се опише със следните компоненти:

1. $P.l \in Q_L$, което ще съответства на

   $$\delta_L(s_L, \alpha) = P.l,$$

   за всяка двойка думи (α, β) от класа на P.

2. $P.t \in Q_T$, което ще съответства на:

   $$state(\alpha, \beta) = P.t,$$

   за всяка (α, β) от класа определен от P.

3. $-1 \leq P.count \leq 2e + 1$, със свойството:

   $$count(\alpha, \beta) = P.count$$

   за всяка двойка думи (α, β) от класа на еквивалентност, определен от P.

Преди да преминем към самата реализация да означим:

$$T(q) = < (\Sigma \cup \Omega) \times \Xi^*, Q_T, q, \delta_T, \lambda_T, \phi >,$$

а
$$T^{\varepsilon}(q) =< (\Sigma \cup \Omega) \times \Xi^*, Q_T, q, \delta_T, \lambda_T, \phi_\varepsilon >,$$
където:
$$\phi_\varepsilon(q) = \varepsilon.$$
Нека накрая:
$$T^{+}(q) =< (\Sigma \cup \Omega) \times \Xi^*, Q_T, q, \delta_{T,\Omega}, \lambda_{T,\Omega}, \phi_\varepsilon >,$$
където ϕ_ε се дефинира както по горе, а:
$$\delta_{T,\Omega}(p,a) = \begin{cases} \delta_T(p,a) \text{ ако } a \in \Sigma \\ p \text{ ако } a \in \Omega \text{ и } !\phi(p) \\ \neg! \text{ иначе.} \end{cases}$$
$$\lambda_{T,\Omega}(p,a) = \begin{cases} \lambda_T(p,a) \text{ ако } a \in \Sigma \\ \phi(p) \text{ ако } a \in \Omega \text{ и } !\phi(p) \\ \neg! \text{ иначе.} \end{cases}$$

Следващият алгоритъм определя състоянията на релацията $\sim_1$, определя релацията $Rel = (\sim_1 \times \sim_2)\times \sim_1$, която по зададени клас $[(\alpha,\beta)]$ от sim_1 и $[\beta']$ от sim_2 дава класа $[\alpha P_1(\beta'), S_1(\beta')]$ на релацията $\sim_1$. Допълнително алгоритъмът ще намира подпоследователните трансдюсери $T_{[\alpha][\beta]}$.

```
N ← 0 for ∀ q_l ∈ Q_L do
    for ∀ q_t ∈ Q_T do
        for i = −1 to 2e + 1 do
            P[N] ← newstate()
            P[N].l ← q_l
            P[N].t ← q_t
            P[N].count ← i
            N ← N + 1
        done
    done
done
```

```
for Q ∈ ~_2 do
    for k = 0 to N − 1 do
        R ← newstate()
         start ← P[k].l
        R.l ← Q.l_start
        if (P[k].count > 1)
            t_start ← P[k].t
            R.t ← Q.q_{t_start}
            R.count ← P[k].count − 2
        else
            if (R.l ∈ F_L)
                m+ ← max{max{i | Q.q_i^+ ∈ F_i^+}, −1}
                m ← max{max{i | Q.q_i ∈ F_i}, −1}
                if (m+ > m)
                        R.count ← 2m+ + 1
                else
                        R.count ← 2m
                if (R.count < −1 )
                        R.count ← −1
            else
                 R.count ← −1
            R.t ← s_T

        P[j] е състояние равно на R

        Rel[P[k]][Q] ← P[j]
        free(R)
        if (P[k].count > 1)
            T[P[k]][Q] ← T^ε(t_start)
        if (P[k].count = 1)
            T[P[k]][Q] ← T(t_start)
        if (P[k].count = 0)
            T[P[k]][Q] ← T+(t_start)
        if P[k].count = −1)
```

$$T[P[k]][Q] \leftarrow id$$

```
    done
done
```

7.6 Алгоритми за построяването на FIFO-трансдюсери за регулярни правила за заместване и тяхната композиция. Проверка на достатъчните условия

Накрая ще покажем примерна реализация на алгоритмите, свързани с функциите, породени от регулярни правила $\mathcal{R}$:

$$E \to T(E)\backslash L\_R,$$

нуждата, от които се появи в параграфи 6.2 и 4.5 на част 6.

Започваме с алгоритъма, който по дадени регулярни езици M и N пределя езика:

$$fin(M, N) = \{\alpha \in M \mid \exists k \forall \beta \in N \; \beta \text{ съдъпжа неповече от } k \text{ срещания на } \alpha\}.$$

Ще предполагаме, че разполагаме с крайни датерминирани амтомати за тези езици.

fin(A_M, A_N)
    begin
        for $\forall\ q\ \in\ Q_N$
            $F(q) \leftarrow\ \varnothing$
            for $\forall\ p\ \in\ Q_N$
                if$\exists$ път от p до q в A_N
                    $F(q)\ \leftarrow\ F(q)\ \cup\ \{p\}$
            done
            $PrefC(q)\ \leftarrow\ <\Sigma, Q_N, q, \delta_N, F(q)>$
        done

        $B \leftarrow$ е автомат еквивалентен на $\overline{\cup_q PrefC(q)}$

        return $A_M\ \cap\ B$
    end

Сега ще представим и алгоритъм, който по дадени крайни детерминирани автомати за регулярните езици M и N, определя дали съществува крайно подмножество $S_0 \subset Suf(M)$ от наставки на M, за който:

1. $M \subset \Sigma^* S_0$ и
2. $S_0 \subset fin(Suf(M), N)$.

В случай, че съществува връща автомат за едно такова подмножество.

```
SuffixCover(A_M, A_N)
    begin
        S  ← < Σ, Q_M, Q_M, δ_M, F_M >
        SF  ←  fin(S, A_N)
        if  (A_M  ∪  ‾Σ*SF‾  ≠  ∅)
            return NULL - няма такова множество

        D  ← автомат за Suf(SF) \ Σ* SF
        if ∃ q  ∈  D
            q е достижимо от началното състояние D
            съществува път от q  до финално състояние на D
            съществува цикъл в D, който съдържа q.

            return NULL - няма такова множество
        else
            n  ← максималната дължина на път от
                              началното до крайно състояние на D
            return SF  ∩  Σˆn + 1
    end
```

Аналогично можем да решим дуалния проблем:

```
PrefixCover(A_M, A_N)
    begin
        P  ← < Σ, Q_M, s_M, δ_M, Q_M >
```

```
        PF ← fin(S, A_N)
        if (A_M ∪ ‾‾‾‾PFΣ*‾‾ ≠ ∅)
            return NULL - няма такова множество

        D ← автомат за Pref(PF) \ PF Σ*
        if ∃ q ∈ D
            q е достижимо от началното състояние D
            съществува път от q до финално състояние на D
            съществува цикъл в D, който съдържа q.

            return NULL - няма такова множество
        else
            n ← максималната дължина на път от
                              началното до крайно състояние на D
            return PF ∩ Σˆn+1
    end
```

Сега можем да опишем и алгоритъма, който проверява дали са налице достатъчните условия за това да можем да построим FIFO-трансдюсер за правилото $\mathcal{R}$ и построява един такъв ако това е така.

```
RuleRecognition(A_L, A_E, A_R, T)
    begin
        N ← автомат за ER
        S_0 ← SuffixCover(A_L, N)
        SE ← SuffixCover(A_E, N)
        PE ← PrefixCover(A_E, N)
        P_0 ← PrefixCover(A_R, N)
        if (S_0 = NULL and PE = NULL)
            return NULL
        if (SE = NULL and P_0 = NULL)
            return NULL
        P ← автомат за PE ∪ P_0
        S ← автомат за SE ∪ S_0
        k ← max{|u| | u ∈ P ∪ S}
        k ← k|Q_N|(|P| + |S|)
```

```
    UL  ← автомат за  𝒰(L){ω}
    UE  ← автомат за 𝒰(E) ∪ (Σ*{ω}Σ*)^≤ k
    UR  ← автомат за {ω}𝒰(R) ∪ (Σ*{ω}Σ*)^≤ k
    UT  ← h ∘ T

    g ← {ω}-представима функция за
     правилото UE → UT(UE)\ UL _ UR

    FT ←  - FIFO-трансдюсер за функцията породена от g
    PS ←  - подпоследователен преобразувател за f_{P,S}
    RuleFT ← (h ∘ (FT ∘ PS))

    return RuleFT
end
```

Накрая ще дадем и примерна реализация на алгоритъма, който при дадени две правила $\mathcal{R}_1$ и $\mathcal{R}_2$ и фиксирани конфигурации $\kappa_1 \subset \{L_1, E_1, R_1\}$ и $\kappa_2 \subset \{L_2, E_2, R_2\}$ проверява дали са изпълнени достатъчните условия за композиция на двете правила и в случай, че това е така връща FIFO-трансдюсер, който я представя.

```
RuleComposition(A_L1, A_E1, A_R1, T1, κ_1,
                A_L2, A_E2, A_R2, T2, κ_2)
  begin
    A1  ←  автомат за ‾Σ*(∪κ_1)Σ*‾
    A2  ←  автомат за ‾Σ*(∪κ_2)Σ*‾
    if (Range(T1) ∩ A2 = ∅)
        return NULL
    N1  ←  автомат за E_1R_1 ∪ A2
    N2  ←  автомат за E_2R_2 ∪ A1 ∪ Range(T1)

    P1 ← ∅
    S1 ← ∅
    left_flag ← false
    right_flag ← false
```

```
if L₁ ∈ κ₁
    PL ← PrefixCover(A_L1, N1)
    SL ← SufixCover(A_L1, N1)
    if (PL = NULL) and (SL = NULL)
        return NULL
    if (SL ≠ NULL)
        right_flag ← true
    P1 ← автомат за P1 ∪ PL
    S1 ← автомат за S1 ∪ SL
if E₁ ∈ κ₁
    PE ← PrefixCover(A_E1, N1)
    SE ← SufixCover(A_E1, N1)
    if (PE = NULL) and (SE = NULL)
        return NULL
    if (PE ≠ NULL)
        left_flag ← true
    if (SE ≠ NULL)
        right_flag ← true
    P1 ← автомат за P1 ∪ PE
    S1 ← автомат за S1 ∪ SE
if R₁ ∈ κ₁
    PR ← PrefixCover(A_R1, N1)
    SR ← SufixCover(A_R1, N1)
    if (PR = NULL) and (SR = NULL)
        return NULL
    if (PR ≠ NULL)
        left_flag ← true
    P1 ← автомат за P1 ∪ PR
    S1 ← автомат за S1 ∪ SR

if (left_flag = false)
    return NULL
if (right_flag = false)
    return NULL
l1 ← дължината на най-дългата дума в P1 ∪ S1
```

$k1 \leftarrow l1\ (|P1| + |S1|)|Q_{N1}|$

$P2 \leftarrow \varnothing$
$S2 \leftarrow \varnothing$
$left\_flag \leftarrow false$
$right\_flag \leftarrow false$
if$L_2 \in \kappa_2$
    $PL \leftarrow PrefixCover(A_{L2}, N2)$
    $SL \leftarrow SufixCover(A_{L2}, N2)$
    if $(PL = NULL)$ and $(SL = NULL)$
        return NULL
    if $(SL \neq NULL)$
        $right\_flag \leftarrow true$
    $P2 \leftarrow$ автомат за $P2 \cup PL$
    $S2 \leftarrow$ автомат за $S2 \cup SL$
if$E_2 \in \kappa_2$
    $PE \leftarrow PrefixCover(A_{E2}, N2)$
    $SE \leftarrow SufixCover(A_{E2}, N2)$
    if $(PE = NULL)$ and $(SE = NULL)$
        return NULL
    if $(PE \neq NULL)$
        $left\_flag \leftarrow true$
    if $(SE \neq NULL)$
        $right\_flag \leftarrow true$
    $P2 \leftarrow$ автомат за $P2 \cup PE$
    $S2 \leftarrow$ автомат за $S2 \cup SE$
if$R_2 \in \kappa_2$
    $PR \leftarrow PrefixCover(A_{R2}, N2)$
    $SR \leftarrow SufixCover(A_{R2}, N2)$
    if $(PR = NULL)$ and $(SR = NULL)$
        return NULL
    if $(PR \neq NULL)$
        $left\_flag \leftarrow true$
    $P2 \leftarrow$ автомат за $P2 \cup PR$
    $S2 \leftarrow$ автомат за $S2 \cup SR$

```
if (left_flag = false)
    return NULL
if (right_flag = false)
    return NULL
```
$l2 \leftarrow$ дължината на най-дългата дума в $P2 \cup S2$
$k2 \leftarrow l2\ (|P2| + |S2|)|Q_{N2}|$

$UL1 \leftarrow$ автомат за $\mathcal{U}(L_1)\{\omega\}$
$UE1 \leftarrow$ автомат за $\mathcal{U}(E_1) \cup (\Sigma^*\{\omega\}\Sigma^*)^{\leq k1}$
$UR1 \leftarrow$ автомат за $\{\omega\}\mathcal{U}(R_1) \cup (\Sigma^*\{\omega\}\Sigma^*)^{\leq k1}$
$UT1 \leftarrow h \circ T1$
g_1 $\{\omega\}$-представима функция за
правилото $UE1 \rightarrow UT1(UE1) \backslash UL1$ _ $UR1$

$PS2 \leftarrow$ каноничен преобразувател за $f_{P2,S2} \circ h$
$\widetilde{g}_1 \leftarrow PS2 \circ g_1$
композиция на подпоследователен преобразувател и
$\{\omega\}$-представима функция.

$UL2 \leftarrow$ автомат за $\mathcal{U}(L_2)\{\omega\}$
$UE2 \leftarrow$ автомат за $\mathcal{U}(E_2) \cup (\Sigma^*\{\omega\}\Sigma^*)^{\leq k2}$
$UR2 \leftarrow$ автомат за $\{\omega\}\mathcal{U}(R_2) \cup (\Sigma^*\{\omega\}\Sigma^*)^{\leq k2}$
$UT2 \leftarrow h \circ T2$
g_2 $\{\omega\}$-представима функция за
правилото $UE2 \rightarrow UT2(UE2) \backslash UL2$ _ $UR2$

$m \leftarrow 2k2 + 4l2$
$M \leftarrow 2k1 + 2l1 + l2(k1 + 2l2)$

$g \leftarrow g_2^* \circ \widetilde{g}_1$ -
$\{\omega\}$-представима функция, получена при
композицията на g_2 и $\widetilde{g}_1$ с параметри
M и m.
$FT \leftarrow$ FIFO-трансдюсер за функцията $g(\varepsilon, \alpha)$

```
    PS1 ← преобразувател за f_{P1,S1}
    ComposeFT ← (h ∘ (FT ∘ PS1)) -
    FIFO-трансдюсер за композицията
    return ComposeFT
end
```

8 Заключение

В тази дипломна работа дефинирахме и изследвахме част от свойствата на детерминираното, последователно устройство FIFO-трансдюсер. Очаквано, защото разполагат с неограничена памет, FIFO-трансдюсерите разпознават и нерационални функции. В част 5 ние доказахме и обратното, именно, че съществуват рационални функции, които не се разпознават от FIFO-трансдюсери. Видяхме още, че FIFO-трансдюсерите дефинират клас от функции, който не е затворен относно композиция в общия случай. Изследвахме в пълнота композицията с подпоследователни преобразуватели и показахме, че тя не ни извежда от класа на FIFO-трансдюсерите. В част 4 отделихме общо подмножество от функции за класа на рационалните функции и тези, които се задават с FIFO-трансдюсер. В част 6 дадохме достатъчни условия едно правило да се разпознава от FIFO-трансдюсер и също да можем да композираме две такива правила. Показахме още как това може да стане ефективно, като описахме съответната конструкция. Използвайки идеи от [SM06] разработихме алгоритъм, който ефективно строи подпоследователен преобразувател, който маркира началните и крайните позиции в даден текст на всички думи от даден речник. Използвайки идеята за tree за крайно множество от думи, което предоставя ефективното представяне на всички наставки в този речник и идеи от [Bre96] разработихме и алгоритъм за ефективното изчисляване на функциите от вида на $walk_P$ и $walk_Q$, които съществено участваха в конструкциите за композиция на подпоследовтален преобразувател с FIFO-трансдюсер.

В заключение ще отбележим някои любопитни въпроси, на които дадената дипломна работа не дава отговор.

Първият е доколко представимите функции, описани в част 4, затворени с композиция отдясно с подпоследователен преобразувател описват сечението на рационалните функции и функциите, представими с FIFO-трансдюсери. Могат ли да се усилят условията, достатъчни за композиция на две правила, така че отново да има FIFO-трансдюсер, който да ги разпознава. Би било интересно ако има и други, алгоритмично-проверими условия, при които такава композиция е възможна. При какви "разумни"условия можем да композираме два FIFO-трансдюсера?

Това са въпроси свързани със семантиката на FIFO-трансдюсерите. Интерес, според нас, представляват и някои синтактически въпроси, свързани най-вече с езика и азбуката на опашката на FIFO-трансдюсерите. Например, вярно ли е че винаги можем да пишем в опашката символът, който четем от входната лента? От теорема 3 следва, че винаги можем да пишем един единствен символ, който обаче зависи и от състоянието, в което се намираме. Ако разгледаме случая, когато левият език на всяко основно състояние е регулярен - тоест думите над входната азбука, с които от началното състояние стигаме до това състояние, образуват регулярно множество - то сравнително лесно може да се покаже, че това твърдение е в сила. Това е свързано и с изучаването на рационалните функции, които се разпознават от FIFO-трансдюсери. Именно съществува ли рационална функция, разпознавана от FIFO-трансдюсер, но за която няма FIFO-трансдюсер, за който всяко основно състояние дефинира ляв регулярен език?

9 Благодарности

Искам да благодаря на дипломния ми ръководител, ст.н.с. Стоян Михов, за възложената от него тема, за ползотворните дискусии в процеса на работа и за критичните забележки. Специално благодаря на доц.д-р Тинко Тинчев за вниманието, с което посрещаше моите идеи и ценните съвети, които ми даде. Искам да благодаря и на всички членове на катедрата за тяхната отзивчивост при съвместната ни работа.

Литература

[Bre96] D. Breslauer. The suffix tree of a tree and minimizing sequential transducers. In D. S. Hirschberg and E. W. Myers, editors, *Proceedings of the7th Annual Symposium on Combinatorial*

Pattern Matching, number 1075, pages 116–129, Laguna Beach, CA, 1996. Springer-Verlag, Berlin.

[Eil74] Samuel Eilenberg. *Automata, Languages, and Machines.* Academic Press, Inc., Orlando, FL, USA, 1974.

[Hop71] John E. Hopcroft. An n log n algorithm for minimizing states in a finite automaton. Technical report, Stanford, CA, USA, 1971.

[KK94] Ronald M. Kaplan and Martin Kay. Regular models of phonological rule systems. *Comput. Linguist.*, 20(3):331–378, 1994.

[Moh94] M. Mohri. Minimization of sequential transducers. In M. Crochemore and D. Gusfield, editors, *Proceedings of the 5th Annual Symposium on Combinatorial Pattern Matching*, number 807, pages 151–163, Asilomar, CA, 1994. Springer-Verlag, Berlin.

[Roc97] E. Roche. Parsing with finite state transducers, 1997.

[Sch65] M.P. Schützenberger. On finite monoids having only trivial subgroups. *Inf. Control*, 8:190–194, 1965.

[SM06] K.U.Schulz S. Mihov. Efficient dictionary-based text rewriting using subsequential transducers. *Natural Language Egineering*, 13(4):353–381, 2006.