БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ
ИНСТИТУТ ПО ИНФОРМАЦИОННИ И КОМУНИКАЦИОННИ ТЕХНОЛОГИИ

Стоян Милков Михов

Крайни автомати, преобразуватели и бимашини: алгоритмични конструкции и имплементации

АВТОРЕФЕРАТ

НА ДИСЕРТАЦИЯ
за присъждане на научната степен
„доктор на науките“
по професионално направление 4.6.
Информатика и компютърни науки

София, 2019

Дисертацията е обсъдена и допусната до защита на разширено заседание на секция Лингвистично моделиране и обработка на знания на ИИКТ-БАН, състояло се на 05.12.2019 г.

Дисертационният труд се състои от 226 страници, 8 глави, увод, заключение и библиография, съдържащи 38 фигури и 5 стр. литература, включваща 48 заглавия.

Защитата на дисертацията ще се състои на ..... г. от ..... часа в зала 218 на блок 25А на ИИКТ-БАН на открито заседание на научно жури в състав:

1. проф. д.м.н Галя Ангелова
2. чл. кор. д.м.н. Светозар Маргенов
3. проф. д.м.н. Иван Димов
4. акад. д.м.н. Веселин Дренски
5. проф. д.м.н Иван Ланджев
6. проф. д-р Тинко Тинчев
7. доц. д-р Христо Ганчев

Материалите за защитата са на разположение на интересуващите се в стая 215 на ИИКТ-БАН, ул. „Акад. Г. Бончев“, бл. 25А.
Автор: Стоян Милков Михов

Заглавие: КРАЙНИ АВТОМАТИ, ПРЕОБРАЗУВАТЕЛИ И БИМАШИНИ: АЛГОРИТМИЧНИ КОНСТРУКЦИИ И ИМПЛЕМЕНТАЦИИ

Обща характеристика на дисертационния труд

Актуалност на темата и обзор на основните резултати в областта

Теорията на крайните автомати предлага теоретично елегантни и изчислително ефективни решения на редица нетривиални и трудни задачи от областта на обработката на текстове и естествен език [Roche and Schabes, 1997b, Mohri, 1997, Karttunen et al., 1997a], обработа на реч [Mohri et al., 2008], търсене на шаблони [Navarro and Raffinot, 2002], представяне на знания [Angelova and Mihov, 2008] и много други. Поради важността ѝ за редица фундаментални приложения, теорията на крайните автомати и различните машини с краен брой състояния е интензивно изучавана и продължава да се развива.

Основите на теорията на крайните автомати е представена от изчислителна гледна точка в редица трудове като например [Hopcroft et al., 2006, Kozen, 1997, Lewis and Papadimitriou, 1998]. В тези книги се разглеждат предимно крайните автомати и регулярните езици над свободен моноид и основни свойства като: теоремата на Клини за еквивалентност на регулярните езици с езиците на крайни автомати, детерминизацията на крайните автомати, затвореност относно сечение и допълнение, релацията на Майхил-Нерод и минимизацията на крайните автомати, както и конструиране на крайни автомати от регулярни изрази.

От теоретико-алгебрична гледна точка крайните автомати са изучавани в монографиите [Eilenberg, 1974, Eilenberg, 1976, Berstel, 1979, Sakarovitch, 2009]. В тях, освен класическия случай на свободен моноид, се разглеждат и крайни автомати над произволни моноиди. В тези трудове се изследват и редица допълнителни алгебрични свойства, както и свойствата на крайните преобразуватели.

Изложение, фокусирано върху приложенията на крайните автомати за търсене и обработка на текстове и естествен език, е представено в трудовете [Kaplan and Kay, 1994, Mohri, 1996, Roche and Schabes, 1997b, Navarro and Raffinot, 2002, Beesley and Karttunen, 2003, Maurel and Guenthner, 2005]. Едни от най-използваните им приложения са приложенията на крайни преобразуватели за представяне и реализиране на презапис на текстове с помощта на правила за замяна. Тези техники са представени например в [Mohri and Sproat, 1996, Karttunen, 1997, Kaplan and Kay, 1994, Gerdemann and van Noord, 1999, Hulden, 2009].

Крайните преобразуватели и по-специално подпоследователните крайни преобразуватели с изходи над числови моноиди са в основата на съвременните приложения за разпознаване на реч. Те са описани например в [Mohri, 1997, Mohri et al., 2008]. Резултатите за подпоследователни преобразуватели са обобщени в случая на други изходни моноиди в [Gerdjikov and Mihov, 2017b, Gerdjikov and Mihov, 2017a].

Приближеното търсене е друга важна област на приложение на крайните автомати и преобразуватели. Приложения на техниките с краен брой състояния за корекция на текстове са описани в [Ringlstetter et al., 2007, Mitankin et al., 2014]. В [Schulz and Mihov, 2002, Mihov and Schulz, 2004, Mitankin et al., 2011] са представени ефективни методологии за приближено търсене в речници и конструкции за детерминирани крайни автомати на Левенщейн.

Вследствие на трудностите при конструирането им, теорията на бимашините е сравнително слабо развита. След тяхното въвеждане и изследване в [Schützenberger, 1961, Reutenauer and Schützenberger, 1991], те се предлагат за обработка на естествен език например в [Roche and Schabes, 1997b]. За да се предолеят тези трудности, в [Gerdjikov et al., 2017] представяме една нова конструкция за бимашини, която избягва построяването на междинен преобразувател, в който за всяка входна дума съществува най-много един успешен път. Показва се, че за определени класове от преобразуватели, новата конструкция води до експоненциално по-малък брой на състоянията на резултатната бимашина.

Съществуват множество реализации на софтуерни библиотеки за конструиране и приложение на крайни автомати и преобразуватели. Най-разпространените системи, които предлагат и конструкции за преобразуватели, са [Karttunen et al., 1997b, Schmid, 2006, Allauzen et al., 2007].

Цели и задачи на дисертационния труд

Целта на настоящия труд е да представи основите на теорията на крайните автомати, преобразуватели и бимашини, като следва комбиниран, едновременно математически и имплементационен подход. Въпреки че всички концепции са въведени формално и са дадени пълни и математически доказателства за коректност на всички представени конструкции, трудът не цели само теоретичното запознаване с областта. Целта на дисертацията е да представи както конструкциите за устройства с краен брой състояния заедно със съответни доказателства за коректност, така и работещи имплементации на всички представени конструкции заедно с документиран код. Това позволява едновременно със запознаването да

се имплементират сложни процедури базирани на техники с краен брой състояния за решаването на практически релевантни задачи.

Методология

Изложението в настоящия труд следва следните принципи:

1. *Теоретични обобщения целящи разширяване обхвата на приложимост.*
   Обхванатият спектър от абстрактни машини с краен брой състояния не се ограничава до класическите крайни автомати и "разпознаватели". Изучават се също така и устройства за вход-изход и превод като многолентови автомати, крайни преобразуватели и бимашини. Всички тези машини могат да се използват например за ефективно презаписване на текст, извличане на информация от текстови корпуси и морфологичен анализ.

2. *Практическа реализируемост на създаваните абстрактни конструкции.*
   След концептуално въвеждане се дават пълни имплементации – изпълними програми, включващи документация на програмния код – за всички представени конструкции. По този начин е възможно да се изследва и наблюдава конкретното поведението на алгоритмите за произволно избрани примери. Също така не е трудно да се разширят дадените програми чрез допълнителни функции за проследяване, което спомага за още по-обстойно изследване на конкретни детайли на представените алгоритми и програми.

3. *Приложимост към съществени практически проблеми.*
   Концептуалните описания и имплементациите по естествен начин постигат една междинна цел. В края на дисертацията показваме как да използваме въведената технология за решаване на някои основни практически проблеми като правописна корекция, фонетизация, аритметика с неограничено големи числа и други.

Апробация на резултатите

Резултати, включени в дисертационния труд, са представени на:

1. Конференция по математическа логика, СУ "Св. Климент Охридски", Гьолечица, 12 май 2018 г.

2. Конференция по математическа логика, СУ "Св. Климент Охридски", Гьолечица, 7 октомври 2016 г.

3. Gastvortrag, Centrum für Informations und Sprachverarbeitung (CIS), Ludwig-Maximilians Universität München, 23 февруари 2015 г.

4. Конференция по математическа логика, СУ "Св. Климент Охридски", Гьолечица, 19 септември 2014 г.

Част от резултатите са докладвани на следните международни конференции:

1. 11th International Conference on Language and Automata Theory and Applications, LATA 2017; Umea; Sweden; 6 March 2017

2. 22nd International Conference on Implementation and Application of Automata, CIAA 2017; Marne-la-Vallee; France; 27 June 2017

3. International Conference on Advanced Computing for Innovation, AComIn 2015; Sofia; Bulgaria; 10 November 2015

4. 1st International Conference on Digital Access to Textual Cultural Heritage, DATeCH 2014; Madrid; Spain; 19 May 2014

5. 11th IAPR International Workshop on Document Analysis Systems, DAS 2014; Tours; France; 7 April 2014

6. 12th International Conference on Document Analysis and Recognition, ICDAR 2013; Washington, DC; United States; 25 August 2013

7. 7th Conference on Computability in Europe, CiE 2011; Sofia; Bulgaria; 27 June 2011

Публикации по дисертационния труд

Представената дисертация е по същество обхваща глави 1-8 на монографията:

- Mihov, S. and Schulz, K. (2019). *Finite-State Techniques: Automata, Transducers and Bimachines.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

Резултати, представени в дисертацията, са публикувани в 11 статии и 1 глава на книга – 3 от статиите са публикувани в журнали с IMPACT фактор, а 7 в журнали и сборници с SJR фактор. До момента в SCOPUS са регистрирани 227 цитирания (без авто-цитирания) на тези статии.

1. Angelova, G. and Mihov, S. (2008). Finite state automata and simple conceptual graphs with binary conceptual relations. In *Supplementary Proceedings of the 16th International Conference on Conceptual Structures, ICCS 2008, Toulouse, France, July 7-11, 2008*, pages 139–148.

2. Daciuk, J., Mihov, S., Watson, B., and Watson, R. (2000). Incremental construction of minimal acyclic finite state automata. *Computational Linguistics*, 26(1):3–16.
   IMPACT factor: Q1, SCOPUS citations: 84

3. Ganchev, H., Mihov, S., and Schulz, K. U. (2008). One-letter automata: How to reduce k tapes to one. In Hamm, F and Kepser, S, editor, *Logics For Linguistic Structures*, volume 201 of *Trends in Linguistics-Studies and Monographs*, pages 35–55.

4. Gerdjikov, S. and Mihov, S. (2017a). Myhill-nerode relation for sequentiable structures. *CoRR*, abs/1706.02910.

5. Gerdjikov, S. and Mihov, S. (2017b). Over which monoids is the transducer determinization procedure applicable? In *Language and Automata Theory and Applications - 11th International Conference, LATA 2017, Umeå, Sweden, March 6-9, 2017, Proceedings*, volume 10168 LNCS, pages 380–392.

6. Gerdjikov, S., Mihov, S., and Schulz, K. U. (2017). A simple method for building bimachines from functional finite-state transducers. In Carayol, A. and Nicaud, C., editors, *Implementation and Application of Automata*, volume 10329 LNCS, pages 113–125. Springer International Publishing.

7. Mihov, S. and Maurel, D. (2001). Direct construction of minimal acyclic subsequential transducers. In *Proceedings of the Conference on Implementation and Application of Automata CIAA'2000*, volume 2088 of *LNCS*, pages 217–229. Springer.
   SCOPUS citations: 3

8. Mihov, S. and Schulz, K. U. (2004). Fast approximate search in large dictionaries. *Computational Linguistics*, 30(4):451–477.
   IMPACT factor: Q1, SCOPUS citations: 45

9. Mitankin, P., Gerdjikov, S., and Mihov, S. (2014). An approach to unsupervised historical text normalisation. In *Digital Access to Textual Cultural Heritage 2014, DATeCH 2014, Madrid, Spain, May 19-20, 2014*, pages 29–34.
SCOPUS citations: 3

10. Mitankin, P., Mihov, S., and Schulz, K. U. (2011). Deciding word neighborhood with universal neighborhood automata. *Theoretical Computer Science*, 412(22):2340–2355.
IMPACT factor: Q3, SCOPUS citations: 1

11. Ringlstetter, C., Schulz, K. U., and Mihov, S. (2007). Adaptive text correction with web-crawled domain-dependent dictionaries. *ACM Transactions on Speech and Language Processing*, 4(4).
SCOPUS citations: 10

12. Schulz, K. U. and Mihov, S. (2002). Fast String Correction with Levenshtein-Automata. *International Journal of Document Analysis and Recognition*, 5(1):67–85.
SCOPUS citations: 81

Съдържание на дисертационния труд

Дисертационният труд се състои от 226 страници, 8 глави, увод, заключение и библиография, съдържащи 38 фигури и 5 стр. литература, включваща 48 заглавия. Основното тяло на дисертацията покрива глави 1-8 на монографията [Mihov and Schulz, 2019].

1 Формално въведение

В първа глава се преследват две цели. Първо, да се въведат основните математически понятия, необходими за представянето на следващите глави. Второ, да се представи един чисто математически поглед върху централните теми на дисертацията: езици, релации и функции между низове, както и основните операции върху езици, релации и функции. Представяме и понятието моноид – клас от алгебрични структури, което дава обща абстракция за низовете, езиците и релациите.

1.1 Множества, функции и релации

В секцията се дават дефиниции и се въвеждат означенията за множества, n-торки, релации и функции. Дефинират се операциите композиция на релации и функции, проекция и обръщане на релации, образ на множество през бинарна релация, както и рефлексивно и транзитивно затваряне на бинарни релации (релационна звезда на Клини R^*). За бинарните релации се въвеждат понятията рефлексивност, симетричност, транзитивност и антисиметричност. По стандартен начин се дефинират и понятията релация и класове на еквивалентност, изфиняване на релация, както и инективност, сюрективност и биективност на функции.
Конвенция: Когато пишем функция ще имаме предвид частична функция, освен ако не сме казали друго. С израза $f(m)$ имаме предвид, че f е дефинирана за m. Изразът $!f(n)$ означава, че f е дефинирана за n.

Следните понятия са по-специфични:

Дефиниция 1.1.6 Една бинарна релация $R \subseteq M \times N$ е *безкрайно многозначна*, ако съществува $m \in M$, така че множеството $R(\{m\})$ е безкрайно.

Дефиниция 1.1.9 Нека $R \subseteq \prod_{i=1}^{n} M_i$, където $n \geq 2$. Релацията

$$R_{\times i} := \{\bar{m}_{\times i} \mid R(\bar{m})\}$$

се нарича *проекцията на R относно множеството координати* $\{1, \ldots, i-1, i+1, \ldots, n\}$. Нека $\emptyset \neq \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$. Тогаве

$$Proj(\langle i_1, \ldots, i_k\rangle, R) := \{\langle m_{i_1}, \ldots, m_{i_k}\rangle \mid \langle m_1, \ldots, m_n\rangle \in R\}$$

се нарича *обобщената проекция на R относно последователността от координати* $\langle i_1, \ldots, i_k\rangle$.

Дефиниция 1.1.16 Нека $R \subseteq \prod_{i=1}^{n} M_i$ и $n \geq 2$. Нека $I = \langle i_1, \ldots, i_k\rangle$ и $J = \langle j_1, \ldots, j_l\rangle$ са две последователности като $\{i_1, \ldots, i_k\}$ и $\{j_1, \ldots, j_l\}$ са две подпоследователности на индексното множество $\{1, \ldots, n\}$. Тогава с $Func(I, J, R)$ означаваме функцията, която изобразява всеки елемент $\langle m_{i_1}, \ldots, m_{i_k}\rangle$ от $Proj(I, R)$ в множеството

$$\{\langle m_{j_1}, \ldots, m_{j_l}\rangle \in Proj(J, R) \mid \exists\langle m_1, \ldots m_n\rangle \in R\}.$$

1.2 Повдигане на функции до множества и n-торки

В тази секция се въвежда понятието "повдигане" на функция, което се използва в следващите глави на дисертацията.

Дефиниция 1.2.1 Нека $f : M \to N$ е функция. "Повдигнатата до множества" версия на f е функцията $\hat{f} : 2^M \to 2^N$, дефинирана поточково:

$$\hat{f}(T) = \{f(t) \,|\, t \in T \cap dom(f)\}.$$

Функцията $\hat{f}$ е тотална – $\hat{f}(T)$ е дефинирана за всяко $T \subseteq M$.

По-нататък ще използваме просто символа f за да означим повдигнатата версия, ако това не води до объркване.

Твърдение 1.2.3 *Нека $f : M \to N$ е функция, и нека $T_i \subseteq M$ за всяко $i \in I$. Тогава*

$$f(\bigcup_{i \in I} T_i) = \bigcup_{i \in I} f(T_i).$$

Дефиниция 1.2.4 Нека $f : M^k \to N$ е k-арна функция за $n \geq 2$. "Повдигнатата до n-торки" версия на f е k-арната функция $\bar{f} : (M^n)^k \to N^n$, дефинирана покомпонентно:

$$\begin{array}{rl} & \bar{f}(\langle m_{1,1}, \ldots, m_{1,n}\rangle, \ldots, \langle m_{k,1}, \ldots, m_{k,n}\rangle) \\ = & \langle f(m_{1,1}, \ldots, m_{k,1}), \ldots, f(m_{1,n}, \ldots, m_{k,n})\rangle. \end{array}$$

1.3 Азбуки, думи и езици

Думите, езиците и операциите върху езици са формалната основа на класическите крайни автомати. В тази секция тези понятия се въвеждат по стандартния начин.

Азбука Σ е множество от символи. Ако не е указано друго, ще предполагаме, че азбуката е крайно и непразно множество.

Дефиниция 1.3.1 Нека Σ е азбука. *Дума* w над Σ е n-торка

$$w = \langle a_1, \ldots, a_n\rangle\,,$$

където $n \geq 0$ и $a_i \in \Sigma$ за $i = 1, \ldots, n$. Числото n наричаме *дължина* на w и означаваме с $|w|$. Празната n-торка $\langle\rangle = \varepsilon$, която е с дължина 0, наричаме *празната дума*. С Σ^* означаваме множеството от всички думи над Σ, и $\Sigma^\varepsilon := \Sigma \cup \{\varepsilon\}$. *Конкатенацията* на две думи $u = \langle a_1, \ldots, a_n\rangle$ и $v = \langle b_1, \ldots, b_m\rangle \in \Sigma^*$ е думата

$$u \cdot v := \langle a_1, \ldots, a_n, b_1, \ldots, b_m\rangle\,.$$

Дефиниция 1.3.3 Нека $t \in \Sigma^*$ и t може да се представи във формата $t = u \cdot v \cdot w$ за някои $u, v, w \in \Sigma^*$. Тогава $v \in \Sigma^*$ е *инфикс* на t. Ако $u = \varepsilon$, то v е *префикс* на t. Ако $w = \varepsilon$, то v е *суфикс* на t. Означението $v \leq t$ ($v < t$) изразява, че v е (същински) префикс на t.

Дефиниция 1.3.9 Нека Σ е азбука. Множество $L \subseteq \Sigma^*$ наричаме *език* над Σ.

Дефиниция 1.3.10 Нека L_1, L_2 са езици над азбука Σ. Тогава

$$L_1 \cdot L_2 := \{w_1 \cdot w_2 \mid w_1 \in L_1, w_2 \in L_2\}$$

наричаме *конкатенация* на L_1 и L_2.

Дефиниция 1.3.13 Нека L е език. Индуктивно дефинираме

1. $L^0 = \{\varepsilon\}$,
2. $L^{k+1} = L^k \cdot L$,

Езикът $L^* = \bigcup_{k=0}^{\infty} L^k$ наричаме *звезда на Клини* на L.

1.4 n-торки от думи, релации и функции върху думи

За понятията, въведени в предишната секция, съществуват естествени обобщения при повдигане към n-торки, които са релевантни при разглеждането на класически n-лентови крайни автомати в следващите глави.

Дефиниция 1.4.1 *Конкатенация на n-торки от думи* се дефинира като

$$\langle u_1, \ldots, u_n \rangle \bar{\cdot} \langle v_1, \ldots, v_n \rangle := \langle u_1 \cdot v_1, \ldots, u_n \cdot v_n \rangle.$$

Дефиниция 1.4.6 Нека $n \geq 1$. Нека Σ_i е азбука за всяко $i = 1, \ldots, n$. Тогава множество $R \subseteq \prod_{i=1}^{n} \Sigma_i^*$ наричаме *n-арна релация върху думи.*

Дефиниция 1.4.8 Нека R_1, R_2 са n-арни релации върху думи. *Конкатенацията* на R_1 и R_2 е

$$R_1 \bar{\cdot} R_2 := \{\bar{u} \bar{\cdot} \bar{v} \mid \bar{u} \in R_1, \bar{v} \in R_2\}.$$

Дефиниция 1.4.12 *(Конкатенационна) звезда на Клини* на n-арна релация върху думи R е дефинирана като $R^* = \bigcup_{k=0}^{\infty} R^k$, където

- $R^0 = \{\bar{\varepsilon}\}$,
- $R^{k+1} = R^k \bar{\cdot} R$,

1.5 Обща моноидна перспектива

В предните две секции въведохме алгебричните структури, които описват думите и n-торки от думи. Тези две структури са частни случаи на моноиди. В тази секция се въвеждат основните свойства на моноидите, които се използват в следващите глави при изследването на моноидни крайни автомати.

Дефиниция 1.5.1 *Моноид* $\mathcal{M}$ наричаме тройка $\langle M, \circ, e\rangle$, където

- M е непразно множество от елементи на моноида,
- $\circ : M \times M \to M$ е моноидната операция,
- $e \in M$ е моноидният единичен елемент

и са изпълнени следните условия:

- $\forall a, b, c \in M : a \circ (b \circ c) = (a \circ b) \circ c$ (асоциативност на "$\circ$"),
- $\forall a \in M : a \circ e = e \circ a = a$ (e е единичен елемент).

Дефиниция 1.5.4 Нека $\mathcal{M} = \langle M, \circ, e\rangle$ е моноид. *Моноиден език над* $\mathcal{M}$ наричаме всяко подмножество на M.

Дефиниция 1.5.5 Нека $\mathcal{M} = \langle M, \circ, e\rangle$ е моноид и $T_1, T_2 \subseteq M$. Тогава множеството

$$T_1 \circ T_2 := \{t_1 \circ t_2 \,|\, t_1 \in T_1, t_2 \in T_2\}.$$

наричаме *моноидно произведение* на моноидните езици T_1 и T_2.

Дефиниция 1.5.6 Нека $\mathcal{M} = \langle M, \circ, e\rangle$ е моноид и $T \subseteq M$. Индуктивно дефинираме

1. $T^0 = \{e\}$,
2. $T^{k+1} = T^k \circ T$,

Моноидният език $T^* = \bigcup_{k=0}^{\infty} T^k$ наричаме *итерация* или (моноидна) *звезда на Клини* на T.

Дефиниция 1.5.7 Подмножеството $T \subseteq M$ на моноида $\mathcal{M} = \langle M, \circ, e\rangle$ е *подмоноид* на $\mathcal{M}$, ако $e \in T$ и $T^2 \subseteq T$.

Твърдение 1.5.8 *За всяко подмножество $T \subseteq M$ на моноида $\mathcal{M}$, множеството T^* е най-малкият подмоноид на $\mathcal{M}$, съдържащ T.*

Дефиниция 1.5.9 Нека $\mathcal{M}_1 = \langle M_1, \circ, e_1 \rangle$ и $\mathcal{M}_2 = \langle M_2, \bullet, e_2 \rangle$ са моноиди. Тотална функция $h : M_1 \to M_2$ е *моноиден хомоморфизъм*, ако са изпълнени следните условия:

- $h(e_1) = e_2$,
- $\forall a, b \in M_1 : h(a \circ b) = h(a) \bullet h(b)$.

Твърдение 1.5.14 *Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид, Σ е азбука и $f : \Sigma \to M$ е тотална функция. Тогава* разширението h_f *на f над Σ^*, дефинирано индуктивно като*

1. $h_f(\varepsilon) = e$
2. $h_f(\alpha \cdot a) = h_f(\alpha) \circ f(a)$, *където* $\alpha \in \Sigma^*, a \in \Sigma$,

е хомоморфизъм между моноидите Σ^* и $\mathcal{M}$ и е единственият хомоморфизъм, разширяващ f.

Дефиниция 1.5.15 Нека $n \geq 1$, и $\mathcal{M}_i = \langle M_i, \circ_i, e_i \rangle$ е моноид за $1 \leq i \leq n$. Нека $\bar{e} := \langle e_1, \ldots, e_n \rangle$ и $\bar{\circ} : (\prod_{i=1}^n M_i) \times (\prod_{i=1}^n M_i) \to \prod_{i=1}^n M_i$ е функцията

$$\langle u_1, \ldots, u_n \rangle \,\bar{\circ}\, \langle v_1, \ldots, v_n \rangle := \langle u_1 \circ_1 v_1, \ldots, u_n \circ_n v_n \rangle .$$

Тогава тройката $\prod_{i=1}^n \mathcal{M}_i := \langle \prod_{i=1}^n M_i, \bar{\circ}, \bar{e} \rangle$ наричаме *декартово произведение* на моноидите $\mathcal{M}_i$.

2 Моноидни крайни автомати

В тази глава разглеждаме по-общата концепция за моноиден краен автомат, като фокусът са общите конструкции и резултати над произволни моноиди.

2.1 Основна конкцепция и примери

Обобщавайки концепцията за класически автомат с краен брой състояния, въвеждаме понятието за моноиден краен автомат.

Дефиниция 2.1.1 *Моноиден краен автомат* е петорка $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$, където

- $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид,

- Q е крайно *множество от състояния*,
- $I \subseteq Q$ е множеството от *начални състояния*,
- $F \subseteq Q$ е множеството от *финални състояния*, и
- $\Delta \subseteq Q \times M \times Q$ е крайно множество, наречено *релация на преходите*.

Тройките $\langle p, m, q\rangle \in \Delta$ наричаме *преходи*. Преходът $\langle p, m, q\rangle$ *е с начало* p, *край* q и има *етикет* m.

Дефиниция 2.1.3 *Класически краен автомат* е моноиден краен автомат върху свободен моноид над крайна азбука Σ, в който всички преходи от релацията на преходите са с етикети от $\Sigma^\varepsilon = \Sigma \cup \{\varepsilon\}$.

Дефиниция 2.1.7 Нека $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta\rangle$ е моноиден краен автомат. *Същински път* в $\mathcal{A}$ е крайна последователност от $k > 0$ прехода

$$\pi = \langle q_0, a_1, q_1\rangle \langle q_1, a_2, q_2\rangle \ldots \langle q_{k-1}, a_k, q_k\rangle ,$$

където $\langle q_{i-1}, a_i, q_i\rangle \in \Delta$ за $i = 1 \ldots k$. Числото k наричаме *дължина* на π. Казваме, че π е с начало q_0 и край q_k. Състоянията $q_0, \ldots, q_k$ са *състоянията по пътя* π. Моноидният елемент $w = a_1 \circ \ldots \circ a_k$ наричаме *етикет* на π. Ще означаваме пътя π като

$$\pi = q_0 \rightarrow^{a_1} q_1 \ldots \rightarrow^{a_k} q_k.$$

Нулевият път за $q \in Q$ е 0_q с начало и край q и етикет e. *Успешен път* е път с начало в начално състояние и край във финално състояние.

Дефиниция 2.1.10 Множеството от етикетите на всички успешни пътища на моноидния краен автомат $\mathcal{A}$ наричаме *моноидния език, разпознаван* от $\mathcal{A}$ и ще означаваме с $L(\mathcal{A})$.

Дефиниция 2.1.14 Нека $\mathcal{A}$ е моноиден краен автомат. *Обобщената релация на преходите* Δ^* дефинираме като най-малката релация в $Q \times M \times Q$, затворена относно:

- $\forall q \in Q : \langle q, e, q\rangle \in \Delta^*$.
- $\forall q_1, q_2, q_3 \in Q \forall w, a \in M : \langle q_1, w, q_2\rangle \in \Delta^* \ \& \ \langle q_2, a, q_3\rangle \in \Delta \rightarrow \langle q_1, w \circ a, q_3\rangle \in \Delta^*$.

Теорема 2.1.22 *Всеки моноиден краен автомат е хомоморфен образ на класически краен автомат. Всеки език на моноиден краен автомат е хомоморфен образ на език на класически краен автомат.*

2.2 Основни свойства на моноидните крайни автомати

В секцията показваме затвореност на моноидните езици относно основните моноидни операции.

Твърдение 2.2.1 *Класът на моноидните автоматни езици над даден моноид $\mathcal{M}$ е затворен относно моноидни хомоморфизми и операциите обединение, моноидно произведение и моноидна звезда на Клини.*

2.3 Моноидни регулярни езици и моноидни регулярни изрази

Дефиниция 2.3.1 Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид. Класът от *моноидните регулярни езици* над $\mathcal{M}$ дефинираме индуктивно:

1. $\emptyset$ е моноиден регулярен език над $\mathcal{M}$;
2. ако $m \in M$, то $\{m\}$ е моноиден регулярен език над $\mathcal{M}$;
3. ако $L_1, L_2 \subseteq M$ са моноидни регулярни езици над $\mathcal{M}$, то
   - $L_1 \cup L_2$ е моноиден регулярен език над $\mathcal{M}$,
   - $L_1 \circ L_2$ е моноиден регулярен език над $\mathcal{M}$,
   - L_1^* е моноиден регулярен език над $\mathcal{M}$.

Дефиниция 2.3.5 Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид. *Моноиден регулярен израз* над $\mathcal{M}$ за $M \cap \{(,), *, +, \cdot, \emptyset\} = \emptyset$ е дума над $M \cup \{(,), *, +, \cdot, \emptyset\}$.Множеството от моноидните регулярни изрази над $\mathcal{M}$ дефинираме индуктивно:

1. $\emptyset$ е моноиден регулярен израз над $\mathcal{M}$;
2. ако $m \in M$, то m е моноиден регулярен израз над $\mathcal{M}$;
3. ако E_1 и E_2 са моноидни регулярни изрази над $\mathcal{M}$, то
   - $(E_1 + E_2)$ е моноиден регулярен израз над $\mathcal{M}$,
   - $(E_1 \cdot E_2)$ е моноиден регулярен израз над $\mathcal{M}$,
   - (E_1^*) е моноиден регулярен израз над $\mathcal{M}$.

2.4 Еквивалентност между моноидни регулярни езици и моноидни автоматни езици

В тази секция се доказва обобщение на теоремата на Клини за моноидния случай.

Твърдение 2.4.1

1. *За моноидния краен автомат* $\mathcal{A}_\emptyset = \langle \mathcal{M}, \emptyset, \emptyset, \emptyset, \emptyset \rangle$ *е изпълнено* $L(\mathcal{A}_\emptyset) = \emptyset$.

2. *Нека* $m \in M$. *За моноидния краен автомат* $\mathcal{A}_m = \langle \mathcal{M}, \{q_0, q_1\}, \{q_0\}, \{q_1\}, \{\langle q_0, m, q_1 \rangle\} \rangle$ *е изпълнено* $L(\mathcal{A}_m) = \{m\}$.

Теорема 2.4.2 (Клини) *Един моноиден език е регулярен тогава и само тогава, когато е език на моноиден краен автомат.*

2.5 Опростяване на структурата на моноидните крайни автомати

В тази секция разглеждаме някои методи за опростяване на автоматната структура с цел да се опрости проверката за принадлежност към автоматния език.

Дефиниция 2.5.1 Моноидният краен автомат $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ е *тримован*, ако всяко състояние $q \in Q$ лежи върху успешен път в $\mathcal{A}$.

Ако е даден автомат $\mathcal{A}$, можем да премахнем всички състояния, които не лежат върху успешен път, заедно със съответните им входящи и изходящи преходи. Така получаваме еквивалентен автомат $\mathcal{A}'$, който е тримован.

Дефиниция 2.5.2 Нека $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ е моноиден краен автомат над $\mathcal{M} = \langle M, \circ, e \rangle$. $\mathcal{A}$ наричаме *е-свободен*, ако $\Delta \subseteq Q \times (M \setminus \{e\}) \times Q$.

Твърдение 2.5.4 *За всеки моноиден краен автомат* $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ *съществува еквивалентен е-свободен моноиден краен автомат* $\mathcal{A}$' *със същото множество от състояния.*

Твърдение 2.5.6 *За всеки моноиден краен автомат* $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ *съществува еквивалентен е-свободен моноиден краен автомат* $\mathcal{A}$' *със същото множество от състояния, такъв че за всяко състояние* $q \in Q$ *е изпълнено* $L_{\mathcal{A}}(q) = L_{\mathcal{A}'}(q)$.

3 Класически крайни автомати и регулярни езици

Класическите крайни автомати са най-често разглежданият клас моноидни крайни автомати. Тъй като базовият моноид е свободен, този клас автомати има някои интересни специфични характеристики. В тази глава се разглеждат основните свойства, свързани с детерминизация и минимизация на класически крайни автомати.

3.1 Детерминирани крайни автомати

Дефиниция 3.1.1 Нека $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ е класически краен автомат. $\mathcal{A}$ е *детерминиран*, ако са изпълнени следните условия:

- $\mathcal{A}$ има точно едно начално състояние q_0,
- всички етикети на преходи са в Σ и
- ако $\langle p, a, q \rangle \in \Delta$ и $\langle p, a, q' \rangle \in \Delta$, то $q = q'$.

В този случай релацията на преходите може да се представи като (частична) функция $\delta : Q \times \Sigma \to Q$.

Детерминираните крайни автомати често се представят във формата

$$\langle \Sigma, Q, q_0, F, \delta \rangle .$$

Дефиниция 3.1.4 Нека $D \subseteq \Sigma^*$ е крайно множество от думи над Σ. *Trie* на D е детерминираният краен автомат

$$\mathcal{A}_D = \langle \Sigma, Pref(D), \varepsilon, D, \delta \rangle ,$$

където

$$\delta = \{ \langle \alpha, \sigma, \alpha \cdot \sigma \rangle \mid \sigma \in \Sigma \ \& \ \alpha, \alpha \cdot \sigma \in Pref(D) \}.$$

Твърдение 3.1.5 *Нека $D \subseteq \Sigma^*$ е крайно множество от думи. Тогава*

1. $L(\mathcal{A}_D) = D$,
2. *Всяко състояние q от $\mathcal{A}_D$ е достижимо по единствен път от началното състояние ε. Етикетът на този път е думата q.*

3.2 Детерминизация на класически крайни автомати

Следващата ни цел е да покажем конструкция, с която от всеки моноиден краен автомат над свободен моноид може да бъде получен еквивалентен детерминиран класически краен автомат.

Твърдение 3.2.1 *За всеки моноиден краен автомат $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ над свободен моноид Σ^* съществува класически краен автомат $\mathcal{A}'$ без ε-преходи с множество от състояния Q', така че $Q \subseteq Q'$ и за всяко $q \in Q$ имаме $L_{\mathcal{A}}(q) = L_{\mathcal{A}'}(q)$.*

Теорема 3.2.2 (Детерминизация)
За всеки класически краен автомат $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ съществува еквивалентен детерминиран краен автомат $\mathcal{A}_D$ с тотална функция на преходите.

3.3 Допълнителни свойства на класическите крайни автомати

Твърдение 3.3.1 (Допълнение на детерминиран краен автомат) *Нека*

$$\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$$

е детерминиран краен автомат и δ е тотална функция. Нека

$$\mathcal{A}' = \langle \Sigma, Q, q_0, Q \setminus F, \delta \rangle .$$

Тогава $L(\mathcal{A}') = \Sigma^* \setminus L(\mathcal{A})$.

Твърдение 3.3.2 *Нека $\mathcal{A}_1 = \langle \Sigma, Q_1, I_1, F_1, \Delta_1 \rangle$ и $\mathcal{A}_2 = \langle \Sigma, Q_2, I_2, F_2, \Delta_2 \rangle$ са два класически ε-свободни крайни автомата. Тогава е изпълнено:*

1. *(Сечение на ε-свободни класически крайни автомати) За крайния автомат*

   $$\mathcal{A} := \langle \Sigma, Q_1 \times Q_2, I_1 \times I_2, F_1 \times F_2, \Delta' \rangle ,$$

   където $\Delta' := \{ \langle \langle q_1, q_2 \rangle, a, \langle r_1, r_2 \rangle \rangle \mid \langle q_1, a, r_1 \rangle \in \Delta_1 \ \& \ \langle q_2, a, r_2 \rangle \in \Delta_2 \}$, е изпълнено $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

2. *(Разлика на детерминирани класически крайни автомати) Ако $\mathcal{A}_2$ е детерминиран класически краен автомат и функцията на преходите на $\mathcal{A}_2$ е тотална, то за крайния автомат*

   $$\mathcal{A} := \langle \Sigma, Q_1 \times Q_2, I_1 \times I_2, F_1 \times (Q_2 \setminus F_2), \Delta' \rangle ,$$

   където $\Delta' := \{ \langle \langle q_1, q_2 \rangle, a, \langle r_1, r_2 \rangle \rangle \mid \langle q_1, a, r_1 \rangle \in \Delta_1 \ \& \ \langle q_2, a, r_2 \rangle \in \Delta_2 \}$, е изпълнено $L(\mathcal{A}) = L(\mathcal{A}_1) \setminus L(\mathcal{A}_2)$.

Твърдение 3.3.3 (Посимволно обръщане на краен автомат) *Нека* $\mathcal{A} := \langle \Sigma^*, Q, F, I, \Delta \rangle$ *е моноиден краен автомат над свободния моноид* Σ^**. Тогава за моноидния краен автомат*

$$\mathcal{A}' := \langle \Sigma^*, Q, F, I, \Delta' \rangle ,$$

където $\Delta' = \{\langle q_2, \rho(a), q_1 \rangle \mid \langle q_1, a, q_2 \rangle \in \Delta\}$*, е изпълнено* $L(\mathcal{A}') = \rho(L(\mathcal{A}))$*.*

Следствие 3.3.4 *Класът на езиците на класически крайни автомати е затворен относно допълнение, сечение, разлика и обръщане.*

3.4 Минимални детерминирани крайни автомати и релация на Майхил-Нерод

За да намерим минималния детерминиран автомат за даден език, е необходимо да приложим алгебричен подход.

Дефиниция 3.4.1 Релацията на еквивалентност $R \subseteq \Sigma^* \times \Sigma^*$ наричаме *дясно инвариантна*, ако

$$\forall u, v \in \Sigma^* : u \; R \; v \rightarrow (\forall w \in \Sigma^* : u \cdot w \; R \; v \cdot w).$$

Дефиниция 3.4.2 Нека $L \subseteq \Sigma^*$ е език и $R \subseteq \Sigma^* \times \Sigma^*$ е релация на еквивалентност. L и R наричаме *съвместими*, ако

$$\forall u, v \in \Sigma^* : (u \in L \;\&\; u \; R \; v) \rightarrow v \in L.$$

Твърдение 3.4.4 *Нека* $R \subseteq \Sigma^* \times \Sigma^*$ *е дясно инвариантна релация на еквивалентност и индексът на* R *е краен, нека* $L \subseteq \Sigma^*$ *е език над* Σ *съвместим с* R*. Тогава за детерминирания краен автомат*

$$\mathcal{A}_{R,L} = \langle \Sigma, \{[s]_R \mid s \in \Sigma^*\}, [\varepsilon]_R, \{[s]_R \mid s \in L\}, \delta_R \rangle$$

с функция на преходите $\delta_R = \{\langle [u]_R, a, [u \cdot a]_R \rangle \mid u \in \Sigma^*, a \in \Sigma\}$ *е изпълнено* $L(\mathcal{A}_{R,L}) = L$*.*

Твърдение 3.4.7 *Нека* $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ *е детерминиран краен автомат. Тогава*

1. $R_\mathcal{A} := \{\langle u, v \rangle \in \Sigma^* \times \Sigma^* \mid \delta^*(q_0, u) = \delta^*(q_0, v)\}$ *е дясно инвариантна релация на еквивалентност и* $L(\mathcal{A})$ *е съвместим с* $R_\mathcal{A}$*,*
2. *автоматът* $\mathcal{A}_{R_\mathcal{A}, L(\mathcal{A})}$ *е изоморфен на* $\mathcal{A}$ *чрез функцията за преименуване на състояния* $h : \{[s]_{R_\mathcal{A}} \mid s \in \Sigma^*\} \rightarrow Q$*, дефинирана като* $h([w]_{R_\mathcal{A}}) = \delta^*(q_0, w)$*.*

Дефиниция 3.4.8 Нека $L \subseteq \Sigma^*$ е език над Σ. Тогава релацията

$$R_L = \{\langle u, v\rangle \in \Sigma^* \times \Sigma^* \,|\, \forall w \in \Sigma^* : u \cdot w \in L \leftrightarrow v \cdot w \in L\}$$

наричаме *релация на Майхил-Нерод* за езика L.

Твърдение 3.4.9 *Нека $L \subseteq \Sigma^*$ е език над Σ. Тогава релацията на Майхил-Нерод R_L е дясно инвариантна релация на еквивалентност и R_L е съвместима с L.*

Дефиниция 3.4.10 Нека R_L е с краен индекс. Тогава детерминирания краен автомат $\mathcal{A}_{R_L,L}$ за R_L и L наричаме *автомат на Майхил-Нерод* за езика L.

Твърдение 3.4.12 *Нека $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta\rangle$ е класически детерминиран краен автомат. Тогава $R_\mathcal{A} \subseteq R_{L(\mathcal{A})}$.*

Теорема 3.4.13 *За всеки класически краен автомат съществува единствен (с точност до преименуване на състояния) еквивалентен детерминиран краен автомат, който е минимален относно броя на състоянията.*

Теорема 3.4.14 *Нека $L \subseteq \Sigma^*$ е класически език. Тогава L е автоматен език тогава и само тогава, когато индексът на R_L е краен.*

Твърдение 3.4.17 *Детерминираният краен автомат $\mathcal{A} = (\Sigma, Q, q_0, F, \delta)$ е минимален тогава и само тогава, когато няма различни еквивалентни състояния в $\mathcal{A}$.*

3.5 Минимизация на детерминирани крайни автомати

Нека е даден детерминиран краен автомат $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta\rangle$ за езика $L(\mathcal{A}) = L$. Ще покажем конструкция за построяване на еквивалентния минимален краен автомат $\mathcal{A}_L$, като идентифицираме и слеем еквивалентните състояния.

Дефиниция 3.5.1 Релациите $R_i \subseteq Q \times Q$ $(i \geq 0)$ дефинираме като

$$q \; R_i \; p \;\; :\Leftrightarrow \;\; \forall \alpha \in \Sigma^* : |\alpha| \leq i \rightarrow (\delta^*(q, \alpha) \in F \leftrightarrow \delta^*(p, \alpha) \in F).$$

Лема 3.5.2 *За всички състояния $q, p \in Q$ следните условия са еквивалентни:*

1. $\forall \alpha \in \Sigma^* : |\alpha| \leq i+1 \rightarrow (\delta^*(q,\alpha) \in F \leftrightarrow \delta^*(p,\alpha) \in F)$,

2. (а) $\forall \alpha \in \Sigma^* : |\alpha| \leq i \rightarrow (\delta^*(q,\alpha) \in F \leftrightarrow \delta^*(p,\alpha) \in F)$, *и*
   (б) $\forall \sigma \in \Sigma, \forall \alpha \in \Sigma^* : |\alpha| \leq i \rightarrow (\delta^*(\delta(q,\sigma),\alpha) \in F \leftrightarrow \delta^*(\delta(p,\sigma),\alpha) \in F)$.

Твърдение 3.5.3

$$q \; R_{i+1} \; p \;\Leftrightarrow\; q \; R_i \; p \;\&\; \forall a \in \Sigma : \delta(q,a) \; R_i \; \delta(p,a)$$

Следствие 3.5.4 *Нека $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ е детерминиран краен автомат, δ е тотална и всички състояния са достижими. Тогава релацията $R = \bigcap_{i=0}^{\infty} R_i$, където*

$$\begin{aligned} q \; R_0 \; p &\;\Leftrightarrow\; (q \in F \leftrightarrow p \in F) \\ q \; R_{i+1} \; p &\;\Leftrightarrow\; q \; R_i \; p \;\&\; \forall a \in \Sigma : \delta(q,a) \; R_i \; \delta(p,a). \end{aligned}$$

съвпада с релацията на еквивалентност на състояния $\equiv$ на $\mathcal{A}$. Автоматът

$$\mathcal{A}' = \langle \Sigma, \{[q]_R \mid q \in Q\}, [q_0]_R, \{[f]_R \mid f \in F\}, \delta' \rangle \,,$$

където $\delta'([q]_R), \sigma) := [\delta(q,\sigma)]_R$, е минималният автомат, еквивалентен на $\mathcal{A}$.

Дефиниция 3.5.5 Нека $g : Q \rightarrow X$. Тогава релацията на еквивалентност $ker_Q(g) \subseteq Q \times Q$, дефинирана като

$$\langle p, q \rangle \in ker_Q(g) \;:\Leftrightarrow\; g(p) = g(q)$$

наричаме *ядро* на g над Q.

Твърдение 3.5.6 *Нека дефинираме $f : Q \rightarrow \{0,1\}$*

$$f(q) := \begin{cases} 1 & \text{ако } q \in F \\ 0 & \text{в противен случай} \end{cases} \,,$$

и за всяко $a \in \Sigma$ и $i \in \mathbb{N}$ дефинираме функцията $f_a^{(i)} : Q \rightarrow Q/R_i$ като

$$f_a^{(i)}(q) := [\delta(q,a)]_{R_i}.$$

Тогава

1. $R_0 = ker_Q(f)$

2. $R_{i+1} = \bigcap_{a \in \Sigma} ker_Q(f_a^{(i)}) \cap R_i$

Забележка 3.5.9 За крайни езици съществуват директни методи за конструиране на минимален детерминиран краен автомат, които имат по-добра ефективност [Daciuk et al., 2000].

3.6 Оцветени детерминирани крайни автомати

В тази секция въвеждаме едно директно обобщение на понятието детерминиран краен автомат наречено оцветен детерминиран краен автомат.

Дефиниция 3.6.1 Нека C е крайно множество наречено *множество от цветове*. *C-оцветен детерминиран краен автомат* наричаме детерминиран краен автомат $\langle \Sigma, Q, q_0, F, \delta \rangle$ с тотална функция на преходите δ заедно със сюрективна тотална функция $col : F \to C$. C-оцветен детерминиран краен автомат ще означаваме с $\mathcal{A} = \langle \Sigma, C, Q, q_0, F, \delta, col \rangle$. Цветът $col(q)$ ще наричаме цвят на състоянието $q \in F$.

Дефиниция 3.6.2 Нека $\mathcal{A} = \langle \Sigma, C, Q, q_0, F, \delta, col \rangle$ е C-оцветен детерминиран краен автомат. *Оцветяването на думите, разпознавани от $\mathcal{A}$*, наричаме функцията $col_{\mathcal{A}} : L(\mathcal{A}) \to C, w \mapsto col(\delta^*(q_0, w))$.

Твърдение 3.6.3 *Нека $L \subseteq \Sigma^*$ е език и $c : L \to C$ е оцветяване на думите от L. Разглеждаме модифицирания език*

$$L_c := \{\alpha \cdot c(\alpha) \mid \alpha \in L\} \subseteq \Sigma^* \cdot C.$$

Нека $\mathcal{A} = \langle \Sigma, C, Q, q_0, F, \delta, col \rangle$ е C-оцветен детерминиран краен автомат, такъв че $L(\mathcal{A}) = L$ и $col_{\mathcal{A}} = c$. Нека

$$\mathcal{A}_c := \langle \Sigma \cup C, Q \cup \{f\}, q_0, \{f\}, \delta \cup \{\langle q, col(q), f \rangle \mid q \in F\} \rangle,$$

където $f \notin Q$ е ново състояние. Тогава $L(\mathcal{A}_c) = L_c$.

Използвайки съответствието от Твърдение 3.6.3 ще обобщим резултатите за класически детерминирани крайни автомати за случая на оцветени детерминирани крайни автомати.

Теорема 3.6.12 *За всеки оцветен детерминиран краен автомат съществува единствен (с точност до преименуване на състояния) еквивалентен детерминиран краен автомат, който е минимален относно броя на състоянията.*

Твърдение 3.6.14 *Оцветеният детерминиран краен автомат $\mathcal{A}$ е минимален тогава и само тогава, когато няма различни еквивалентни състояния в $\mathcal{A}$.*

Дефиниция 3.6.15 Дефинираме релациите на еквивалентност R_i ($i \geq 0$) върху Q индуктивно:

$$\begin{aligned} q\ R_0\ p &\leftrightarrow (q \in F \leftrightarrow p \in F)\ \&\ (q \in F \to col(q) = col(p)) \\ q\ R_{i+1}\ p &\leftrightarrow q\ R_i\ p\ \&\ \forall a \in \Sigma : \delta(q, a)\ R_i\ \delta(p, a). \end{aligned}$$

Следствие 3.6.16 *Нека $\mathcal{A} = \langle \Sigma, C, Q, q_0, F, \delta, col \rangle$ е оцветен детерминиран краен автомат, δ е тотална и всяко състояние е достижимо. Тогава $R = \bigcap_{i=0}^{\infty} R_i$ съвпада с релацията еквивалентност на състояния $\equiv$ върху $\mathcal{A}$. Оцветеният детерминиран краен автомат*

$$\mathcal{A}' = \langle \Sigma, C, \{[q]_R \mid q \in Q\}, [q_0]_R, \{[f]_R \mid f \in F\}, \delta', col' \rangle ,$$

където $\delta'([q]_R), \sigma) := [\delta(q, \sigma)]_R$ и $col'([q]_R) := col(q)$, е минималният оцветен детерминиран краен автомат, еквивалентен на $\mathcal{A}$.

3.7 Псевдо-детерминизация и псевдо-минимизация на моноидни крайни автомати

В тази секция първо показваме, че дефинирането на детерминиран моноиден краен автомат за произволен моноид не е естествено. Затова въвеждаме едно по-слабо понятие, използвайки факта, че всеки моноиден краен автомат е хомоморфен образ на класически краен автомат (Теорема 2.1.22).

Дефиниция 3.7.1 Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид. Моноидният автомат $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ наричаме *псевдо-детерминиран*, ако в I има единствено състояние и за всяко състоние $q \in Q$ и всяко $m \in M$ съществува най-много едно състояние q', така че $\langle q, m, q' \rangle \in \Delta$.

Твърдение 3.7.2 *За всеки моноиден краен автомат $\mathcal{A}$ можем да конструираме псевдо-детерминиран моноиден краен автомат $\mathcal{A}'$, еквивалентен на $\mathcal{A}$.*

Дефиниция 3.7.3 Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид. Псевдо-детерминираният моноиден краен автомат $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ наричаме *псевдо-минимален* ако е хомоморфен образ на минимален детерминиран краен автомат.

Твърдение 3.7.4 *За всеки моноиден краен автомат $\mathcal{A}$ можем да конструираме псевдо-минимален моноиден краен автомат $\mathcal{A}'$, еквивалентен на $\mathcal{A}$.*

4 Моноидни многолентови крайни автомати и крайни преобразуватели

Едно важно обобщение на класическите крайни автомати са многолентовите крайни автомати, които представят релации. Релациите по естествен начин способстват формализирането на трансформации и функции,

което оправдава интереса към изследването на многолентовите крайни автомати. Следвайки перспективата от предишните глави, многолентовите крайни автомати въвеждаме като специален случай на моноидните крайни автомати. Един естествен техен подклас са моноидните преобразуватели. В тази глава представяме основните свойства на моноидните многолентови крайни автомати и в частност на моноидните крайни преобразуватели.

4.1 Моноидни многолентови крайни автомати

Първо ще въведем концепцията на моноидните многолентови крайни автомати.

Дефиниция 4.1.1 *Моноиден n-лентов краен автомат* е моноиден краен автомат $\mathcal{A} = \langle \prod_{i=1}^{n} \mathcal{M}_i, Q, I, F, \Delta \rangle$ над моноид $\mathcal{M} = \mathcal{M}_1 \times \mathcal{M}_2 \ldots \times \mathcal{M}_n$, който е декартово произведение на n моноида. Ако $n \geq 2$, наричаме автомата също и *многолентов краен автомат*.

За да наблегнем на факта, че езиците, разпознавани от многолентови крайни автомати, са релации, ще въведем следната терминология.

Дефиниция 4.1.5 *Моноидна n-лентова автоматна релация* е моноиден език разпознаван от моноиден n-лентов краен автомат.

4.2 Допълнителни свойства на моноидните многолентови крайни автомати

Ще покажем, че класът на релациите, разпознавани от многолентови крайни автомати, е затворен относно някои допълнителни релационни операции. За следващите конструкции дефинираме

$$E(\Delta) := \Delta \cup \{\langle q, e, q \rangle \mid q \in Q\}.$$

Твърдение 4.2.1

1. *(Декартово произведение) Нека $\mathcal{A}_1 = \langle \mathcal{M}_1, Q_1, I_1, F_1, \Delta_1 \rangle$ и $\mathcal{A}_2 = \langle \mathcal{M}_2, Q_2, I_2, F_2, \Delta_2 \rangle$ са два моноидни крайни автомата и нека*

   $$\Delta := \{\langle \langle q_1, q_2 \rangle, \langle u_1, u_2 \rangle, \langle q_1', q_2' \rangle \rangle \mid \langle q_1, u_1, q_1' \rangle \in E(\Delta_1) \ \& \ \langle q_2, u_2, q_2' \rangle \in E(\Delta_2)\}.$$

   Тогава за моноидния 2-лентов краен автомат

   $$\mathcal{A} := \langle \mathcal{M}_1 \times \mathcal{M}_2, Q_1 \times Q_2, I_1 \times I_2, F_1 \times F_2, \Delta \rangle$$

   е изпълнено $L(\mathcal{A}) = L(\mathcal{A}_1) \times L(\mathcal{A}_2)$.

2. *(Проекция) Нека* $\mathcal{A} = \langle \mathcal{M}_1 \times \mathcal{M}_2 \times \ldots \times \mathcal{M}_n, Q, I, F, \Delta \rangle$ *е моноиден* n*-лентов краен автомат и* $n \geq 2$*. Нека* $\Delta_{\times i} := \{\langle q, \bar{u}_{\times i}, q' \rangle \mid \langle q, \bar{u}, q' \rangle \in \Delta\}$.
Тогава за моноидния $(n-1)$*-лентов краен автомат*

$$\mathcal{A}' := \langle \mathcal{M}_1 \times \ldots \times \mathcal{M}_{i-1} \times \mathcal{M}_{i+1} \times \ldots \times \mathcal{M}_n, Q, F, I, \Delta_{\times i} \rangle$$

е изпълнено $L(\mathcal{A}') = L(\mathcal{A})_{\times i}$.

3. *(Обратна релация) Нека* $\mathcal{A} = \langle \mathcal{M}_1 \times \mathcal{M}_2, Q, I, F, \Delta \rangle$ *е моноиден* 2*-лентов краен автомат и*

$$\Delta' := \{\langle q_1, \langle v, u \rangle, q_2 \rangle \mid \langle q_1, \langle u, v \rangle, q_2 \rangle \in \Delta\}.$$

Тогава за моноидния 2*-лентов краен автомат*

$$\mathcal{A}' = \langle \mathcal{M}_2 \times \mathcal{M}_1, Q, I, F, \Delta' \rangle$$

е изпълнено $L(\mathcal{A}') = L(\mathcal{A})^{-1}$.

4. *(Идентитет) Нека* $\mathcal{A} = \langle \mathcal{M}, Q, I, F, \Delta \rangle$ *е моноиден краен автомат и*

$$\Delta' := \{\langle q_1, \langle u, u \rangle, q_2 \rangle \mid \langle q_1, u, q_2 \rangle \in \Delta\}.$$

Тогава за моноидния 2*-лентов краен автомат*

$$\mathcal{A}' = \langle \mathcal{M} \times \mathcal{M}, Q, I, F, \Delta' \rangle$$

е изпълнено $L(\mathcal{A}') = Id_{L(\mathcal{A})}$.

Следствие 4.2.2 *Класът на моноидните многолентови крайни релации е затворен относно декартово произведение, проекции и обратна релация.*

4.3 Класически многолентови крайни автомати и едносимволни крайни автомати

Дефиниция 4.3.1 *Класически* n*-лентов краен автомат* наричаме моноиден n-лентов краен автомат над моноид, който е декартово произведение на n свободни моноида.

Дефиниция 4.3.2 n*-лентов посимволен краен автомат* наричаме класически n-лентов краен автомат $\mathcal{A} = \langle \Sigma_1 \times \Sigma_2 \times \ldots \times \Sigma_n, Q, I, F, \Delta \rangle$, такъв че $\Delta \subseteq Q \times (\Sigma_1^{\varepsilon} \times \ldots \times \Sigma_n^{\varepsilon}) \times Q$.

Твърдение 4.3.3 *Нека $\mathcal{A}$ е класически n-лентов краен автомат. Тогава за $\mathcal{A}$ можем да конструираме еквивалентен n-лентов посимволен краен автомат.*

Твърдение 4.3.4 *(Релационна композиция) Нека*

$$\begin{aligned}\mathcal{A}_1 &= \langle \Sigma_1 \times \Sigma, Q_1, I_1, F_1, \Delta_1 \rangle \\ \mathcal{A}_2 &= \langle \Sigma \times \Sigma_2, Q_2, I_2, F_2, \Delta_2 \rangle\end{aligned}$$

са 2-лентови посимволни крайни автомати. Нека Δ е множеството от тройки

$$\langle \langle q_1, q_2 \rangle, \langle u, w \rangle, \langle q_1', q_2' \rangle \rangle,$$

за които съществува $v \in \Sigma \cup \{\varepsilon\}$, така че $\langle q_1, \langle u, v \rangle, q_1' \rangle \in E(\Delta_1)$ и $\langle q_2, \langle v, w \rangle, q_2' \rangle \in E(\Delta_2)$. Тогава за 2-лентовия посимволен краен автомат

$$\mathcal{A} := \langle \Sigma_1 \times \Sigma_2, Q_1 \times Q_2, I_1 \times I_2, F_1 \times F_2, \Delta \rangle$$

е изпълнено $L(\mathcal{A}) = L(\mathcal{A}_1) \circ L(\mathcal{A}_2)$.

Забележка 4.3.7 В [Ganchev et al., 2008] се въвеждат n-лентови едносимволни крайни автомати, които са n-лентови посимволни крайни автомати, на които етикетите имат ε на всички ленти без една. Всички свойства от тази секция се прехвърлят върху n-лентовите едносимволни крайни автомати.

4.4 Моноидни крайни преобразуватели

Дефиниция 4.4.1 *Моноиден краен преобразувател* наричаме моноиден 2-лентов краен автомат $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ над моноид, който е във формата $\Sigma^* \times \mathcal{M}$ за някоя азбука Σ. Ако $\Delta \subseteq Q \times (\Sigma^\varepsilon \times \mathcal{M}) \times Q$, тогава $\mathcal{T}$ наричаме моноиден *посимволен краен преобразувател.*

Дефиниция 4.4.4 Моноидният краен преобразувател $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ е *функционален*, ако езикът $L(\mathcal{T})$ представя частична функция $\Sigma^* \to \mathcal{M}$.
В този случай казваме, че $\mathcal{T}$ *представя* функцията $L(\mathcal{T})$.

Дефиниция 4.4.5 Моноидният краен преобразувател $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ наричаме *безкрайно многозначен*, ако релацията $L(\mathcal{T})$ е безкрайно многозначна.

Дефиниция 4.4.6 Моноидният краен преобразувател $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ наричаме *реално-временен*, ако $\Delta \subseteq Q \times (\Sigma \times \mathcal{M}) \times Q$.

Твърдение 4.4.8 *Нека $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ е моноиден посимволен краен преобразувател. Нека множеството от всички етикети на пътища от вида $\langle \varepsilon, m \rangle$ в $\mathcal{T}$ е крайно. Тогава съществува реално-временен краен преобразувател $\mathcal{T}'$, еквивалентен на $\mathcal{T}$ с точност до ε.*

4.5 Класически крайни преобразуватели

Дефиниция 4.5.1 *Класически краен преобразувател* е моноиден краен преобразувател

$$\mathcal{T} = \langle \Sigma_1^* \times \Sigma_2^*, Q, I, F, \Delta \rangle ,$$

където и втората лента е върху свободен моноид Σ_2^*.

Твърдение 4.5.7 *Тримован класически краен преобразувател $\mathcal{T}$ е безкрайно многозначен тогава и само тогава, когато съществува цикъл в $\mathcal{T}$ с етикет $\langle \varepsilon, u \rangle$, където $u \neq \varepsilon$.*

4.6 Разрешимост на функционалността на класическите крайни преобразуватели

Дефиниция 4.6.1 Нека Σ е крайна азбука. *Функцията на напредък* $\omega : (\Sigma^* \times \Sigma^*) \times (\Sigma^* \times \Sigma^*) \to \Sigma^* \times \Sigma^*$ дефинираме като:

$$\omega(\langle x, y \rangle , \langle \alpha, \beta \rangle) = \left\langle c^{-1}x\alpha, c^{-1}y\beta \right\rangle , \text{ където } c = x\alpha \wedge y\beta.$$

Итерирана функция на напредък $\omega^* : (\Sigma^* \times \Sigma^*) \times (\Sigma^* \times \Sigma^*)^* \to (\Sigma^* \times \Sigma^*)$ дефинираме индуктивно:

- $\omega^*(\langle x, y \rangle , \varepsilon) = \langle x, y \rangle$,
- $\omega^*(\langle x, y \rangle , U \langle \alpha, \beta \rangle) = \omega(\omega^*(\langle x, y \rangle , U), \langle \alpha, \beta \rangle)$.

Напредъкът $\omega(\langle x, y \rangle , \langle \alpha, \beta \rangle) = \langle u, v \rangle$ на двойката думи $\langle x, y \rangle \in \Sigma^* \times \Sigma^*$ с $\langle \alpha, \beta \rangle \in \Sigma^* \times \Sigma^*$ наричаме *балансируем*, ако $u = \varepsilon$ или $v = \varepsilon$.

Дефиниция 4.6.4 Нека $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ е класически реалновременен краен преобразувател ($\Delta \subseteq Q \times (\Sigma_I \times \Sigma^*) \times Q$). *Квадратирания изходен автомат* на $\mathcal{T}$ наричаме класическия 2-лентов краен автомат

$$\mathcal{S}_\mathcal{T} = \langle \Sigma^* \times \Sigma^*, Q \times Q, I \times I, F \times F, \Delta' \rangle ,$$

където Δ' е множеството на всички преходи от вида

$$\langle \langle q_1', q_2' \rangle , \langle \alpha_1, \alpha_2 \rangle , \langle q_1'', q_2'' \rangle \rangle ,$$

където съществува $a \in \Sigma$, така че $\langle q_1', \langle a, \alpha_1 \rangle , q_1'' \rangle \in \Delta$ и $\langle q_2', \langle a, \alpha_2 \rangle , q_2'' \rangle \in \Delta$.

Твърдение 4.6.5 *Нека $\mathcal{T}$ е класически реално-временен краен преобразувател. Тогава $\mathcal{T}$ е функционален тогава и само тогава, когато за всеки успешен път с етикет $\langle \alpha, \beta \rangle$ в квадратирания изходен автомат $\mathcal{S}_\mathcal{T}$ е изпълнено $\alpha = \beta$.*

Дефиниция 4.6.6 Нека $\mathcal{T} = \langle \Sigma^* \times \Sigma^*, Q, I, F, \Delta \rangle$ е класически краен преобразувател. Двойката $\langle u, v \rangle \in \Sigma^* \times \Sigma^*$ наричаме *допустим напредък* на състоянието $q \in Q$, ако съществува път

$$\pi : q_0 \rightarrow^{\langle \alpha_1, \beta_1 \rangle} q_1 \rightarrow^{\langle \alpha_2, \beta_2 \rangle} q_2 \ldots \rightarrow^{\langle \alpha_n, \beta_n \rangle} q_n = q$$

с начало начално състояние $q_0 \in I$, така че

$$\langle u, v \rangle = \omega^*(\langle \varepsilon, \varepsilon \rangle, \langle \alpha_1, \beta_1 \rangle \ldots \langle \alpha_n, \beta_n \rangle).$$

С $Adm(q)$ ще означаваме *множеството от всички допустими напредъци* на състоянието $q \in Q$.

Следствие 4.6.7 *Нека $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ е класически реално-временен преобразувател. Нека $\mathcal{S}_\mathcal{T}$ е квадратираният изходен автомат на $\mathcal{T}$ с множество от финални състояния $F' = F \times F$. Тогава $\mathcal{T}$ е функционален тогава и само тогава, когато за всяко $p \in F'$ имаме $Adm(p) \subseteq \{\langle \varepsilon, \varepsilon \rangle\}$.*

Твърдение 4.6.8 *Нека $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ е класически реално-временен краен преобразувател и $\mathcal{S}_\mathcal{T}$ е квадратираният изходен автомат на $\mathcal{T}$. Нека p е състояние от успешен път в $\mathcal{S}_\mathcal{T}$. Ако допуснем, че*

1. *съществува $\langle u, v \rangle \in Adm(p)$, така че $\langle u, v \rangle$ не е балансируем (т.е., $u \neq \varepsilon$ и $v \neq \varepsilon$) или*
2. *$|Adm(p)| > 1$,*

то $\mathcal{T}$ не е функционален.

Твърдение 4.6.10 *Нека $\mathcal{T}$ е класически реално-временен краен преобразувател и $\mathcal{S}_\mathcal{T} = \langle \Sigma^* \times \Sigma^*, Q \times Q, I \times I, F \times F, \Delta' \rangle$ е квадратираният изходен автомат на $\mathcal{T}$. Тогава двойката $\langle u, v \rangle \in \Sigma^* \times \Sigma^*$ е допустим напредък на $q \in Q \times Q$ тогава и само тогава, когато*

- *$q \in I \times I$ и $\langle u, v \rangle = \langle \varepsilon, \varepsilon \rangle$ или*
- *съществува състояние $q' \in Q \times Q$, допустим напредък $\langle u', v' \rangle$ на q' и преход $\langle q', \langle \alpha, \beta \rangle, q \rangle \in \Delta'$, така че $\langle u, v \rangle = \omega(\langle u, v \rangle, \langle \alpha, \beta \rangle)$.*

Следствие 4.6.11 *Нека $\mathcal{T}$ е класически реално-временен краен преобразувател и $\mathcal{S}_{\mathcal{T}} = \langle \Sigma^* \times \Sigma^*, Q \times Q, I \times I, F \times F, \Delta' \rangle$ е квадратираният изходен автомат на $\mathcal{T}$. Нека функцията $Adm^{(k)} : Q \times Q \to 2^{\Sigma^* \times \Sigma^*}$ е дефинирана индуктивно:*

1. $Adm^{(0)}(q) := \begin{cases} \{\langle \varepsilon, \varepsilon \rangle\} & \text{ако } q \in I \times I \\ \emptyset & \text{в противен случай.} \end{cases}$

2. $Adm^{(k+1)}(q) := Adm^{(k)}(q) \cup$
$\{\{\omega(\langle u', v' \rangle, \langle \alpha, \beta \rangle)\} \mid \langle q', \langle \alpha, \beta \rangle, q \rangle \in \Delta', \langle u', v' \rangle \in Adm^{(k)}(q')\}.$

Тогава $Adm = \bigcup_{k=0}^{\infty} Adm^{(k)}$.

Следствие 4.6.12 *Нека $\mathcal{T}$ е класически краен преобразувател. T е функционален тогава и само тогава, когато*

1. $|(\{\varepsilon\} \times \Sigma^*) \cap L(\mathcal{T})| \leq 1$;

2. *$\mathcal{T}$ не е безкрайно многозначен;*

3. *$\mathcal{T}'$ е функционален, където $\mathcal{T}'$ е класически реално-временен краен преобразувател, еквивалентен на $\mathcal{T}$ с точност до ε.*

Твърденията в тази секция ни предоставят процедура за разрешаване на функционалността на даден класически краен преобразувател, която е реализирана в Програма 8.2.12.

5 Детерминирани крайни преобразуватели

В тази глава изследваме детерминираните крайни преобразуватели. Ясно е,че има смисъл да разглеждаме детерминираност само ако се ограничим върху функционалните преобразуватели. В главата ще се фокусираме върху преобразуватели, които са детерминирани по входната лента (наречени последователни и подпоследователни преобразуватели). Подпоследователните преобразуватели са широко използвани за обработка на текстове [Mohri, 1996, Roche and Schabes, 1997b] и разпознаване на реч [Mohri et al., 2008].

5.1 Детерминирани крайни преобразуватели и подпоследователни крайни преобразуватели

Дефиниция 5.1.1 Моноидният краен преобразувател $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ е *детерминиран*, ако са изпълнени следните условия:

1. $|I| = 1$, т.е. има точно едно начално състояние;

2. $\delta := \{\langle q_1, a, q_2\rangle \mid \exists m \in M : \langle q_1, \langle a, m\rangle, q_2\rangle \in \Delta\}$ е (частична) функция $Q \times \Sigma \to Q$;

3. $\lambda := \{\langle q_1, a, m\rangle \mid \exists q_2 \in Q : \langle q_1, \langle a, m\rangle, q_2\rangle \in \Delta\}$ е (частична) функция $Q \times \Sigma \to M$;

Моноиден детерминиран краен преобразувател ще означаваме с

$$\mathcal{T} = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda\rangle,$$

където $I = \{q_0\}$ и $\Delta = \{\langle q, \langle a, \lambda(q,a)\rangle, \delta(q,a)\rangle \mid \langle q, a\rangle \in dom(\delta)\}$.

Класически детерминиран краен преобразувател се дефинира по съответния начин, като се изисква моноидът $\mathcal{M}$ да е свободен.

Дефиниция 5.1.4 *Моноиден подпоследователен преобразувател* е осморка

$$\mathcal{T} = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda, \Psi\rangle,$$

където $\langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda\rangle$ е детерминиран моноиден краен преобразувател и $\Psi : F \to \mathcal{M}$ е *функцията на изходите на състоянията* с домейн F. *Подлежащият автомат* на $\mathcal{T}$ е детерминираният краен автомат $\mathcal{A}_\mathcal{T} = \langle \Sigma, Q, q_0, F, \delta\rangle$ и *входният език* на $\mathcal{T}$ е множеството

$$L(\mathcal{T})_{\times 2} = L(\mathcal{A}_\mathcal{T}) = \{t \in \Sigma^* \mid \delta^*(q_0, t) \in F\}.$$

Класически подпоследователен преобразувател наричаме моноиден подпоследователен преобразувател, в който моноидът $\mathcal{M}$ е свободен моноид Σ'^* над крайна *изходна азбука* Σ'.

Дефиниция 5.1.5 Нека $\mathcal{T} = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda, \Psi\rangle$ е моноиден подпоследователен преобразувател. *Изходната функция* $O_\mathcal{T} : L(\mathcal{T})_{\times 2} \to \mathcal{M}$ на $\mathcal{T}$ е дефинирана като:

$$\forall t \in L(\mathcal{T})_{\times 2} : \qquad O_\mathcal{T}(t) := \lambda^*(q_0, t) \cdot \Psi(\delta^*(q_0, t)).$$

Дефиниция 5.1.7 Нека $\mathcal{T} = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda, \Psi\rangle$ е моноиден подпоследователен преобразувател. *Изходната функция за* $q \in Q$ е

$$O^q_\mathcal{T} : \Sigma^* \to \mathcal{M};\ \alpha \mapsto \lambda^*(q, \alpha) \cdot \Psi(\delta^*(q, \alpha)).$$

Твърдение 5.1.10 *Нека* $\mathcal{T}_1 = \langle \Sigma, \Sigma', Q_1, q_{01}, F_1, \delta_1, \lambda_1, \Psi_1 \rangle$ *е класически подпоследователен преобразувател и* $\mathcal{T}_2 = \langle \Sigma', \mathcal{M}, Q_2, q_{02}, F_2, \delta_2, \lambda_2, \Psi_2 \rangle$ *е моноиден подпоследователен преобразувател. Тогава за монодния подпоследователен преобразувател*

$$\mathcal{T} = \langle \Sigma, \mathcal{M}, Q_1 \times Q_2, \langle q_{01}, q_{02} \rangle, F, \delta, \lambda, \Psi \rangle ,$$

където

- $\delta = \{\langle \langle q_1, q_2 \rangle, \sigma, \langle \delta_1(q_1, \sigma), \delta_2^*(q_2, \lambda_1(q_1, \sigma)) \rangle \rangle \mid q_1 \in Q_1, q_2 \in Q_2, \sigma \in \Sigma\}$,
- $\lambda = \{\langle \langle q_1, q_2 \rangle, \sigma, \lambda_2^*(q_2, \lambda_1(q_1, \sigma)) \rangle \mid q_1 \in Q_1, q_2 \in Q_2, \sigma \in \Sigma\}$,
- $F = \{\langle q_1, q_2 \rangle \mid q_1 \in F_1, \delta_2^*(q_2, \Psi_1(q_1)) \in F_2\}$,
- $\Psi = \{\langle \langle q_1, q_2 \rangle, \lambda_2^*(q_2, \Psi_1(q_1)) \odot \Psi_2(\delta_2^*(q_2, \Psi_1(q_1))) \rangle \mid \langle q_1, q_2 \rangle \in F\}$;

е изпълнено $O_{\mathcal{T}} = O_{\mathcal{T}_1} \circ O_{\mathcal{T}_2}$.

Дефиниция 5.1.12 *Секвенциално разстояние* между $u \in \Sigma^*$ и $v \in \Sigma^*$ се дефинира като $d_S(u, v) = |u| + |v| - 2|u \wedge v|$.

Дефиниция 5.1.13 Функция между думи $f : \Sigma^* \to \Sigma'^*$ е с *ограничена вариация*, ако за всяко $k \geq 0$ съществува $K \geq 0$, така че за всеки $u, v \in dom(f)$, ако $d_S(u, v) \leq k$, то $d_S(f(u), f(v)) \leq K$.

Дефиниция 5.1.14 Функционалният класически краен преобразувател $\mathcal{T}$ е с *ограничена вариация*, ако функцията, която се представя от $\mathcal{T}$, е с ограничена вариация.

5.2 Детерминизиране на функционални крайни преобразуватели с ограничена вариация

Конструкция
Нека е даден тримован реално-временен краен преобразувател

$$\mathcal{T} = \langle \Sigma^* \times \Sigma'^*, Q, I, F, \Delta \rangle ,$$

т.е. $\Delta \subseteq Q \times (\Sigma \times \Sigma'^*) \times Q$.
Подпоследователният преобразувател

$$\mathcal{T}' = \langle \Sigma, \Sigma', Q', q_0', F', \delta', \lambda', \Psi' \rangle ,$$

еквивалентен на $\mathcal{T}$, ще строим индуктивно.

Базата на индукцията е:

$$\mathcal{T}'^{(0)} = \langle \Sigma, \Sigma', \{q'_0\}, q'_0, \emptyset, \emptyset, \emptyset, \emptyset \rangle \text{ , където } q'_0 = I \times \{\varepsilon\}.$$

Да допуснем, че сме построили

$$\mathcal{T}'^{(n)} = \left\langle \Sigma, \Sigma', Q'^{(n)}, q'_0, F'^{(n)}, \delta'^{(n)}, \lambda'^{(n)}, \Psi'^{(n)} \right\rangle .$$

Дефинираме $\mathcal{T}'^{(n+1)} = \langle \Sigma, \Sigma', Q'^{(n+1)}, q'_0, F'^{(n+1)}, \delta'^{(n+1)}, \lambda'^{(n+1)}, \Psi'^{(n+1)} \rangle$ със следните компоненти:

- новата функция на изходите на преходите $\lambda'^{(n+1)}$ продължава $\lambda'^{(n)}$ с всички тройки от вида $\langle S, \sigma, w \rangle$, за които $\{\langle q, \langle \sigma, v \rangle, q' \rangle \in \Delta \mid \langle q, u \rangle \in S\} \neq \emptyset$, където $S \in Q'^{(n)}$, $\sigma \in \Sigma$ и

$$w = \bigwedge_{\langle q,u \rangle \in S} \bigwedge_{\langle q,\langle \sigma,v \rangle,q' \rangle \in \Delta} u \cdot v.$$

- Новата функция на преходите $\delta'^{(n+1)}$ продължава $\delta'^{(n)}$ с всички тройки от вида $\langle S, \sigma, S' \rangle$, където $\langle S, \sigma, w \rangle$ е тройка в $\lambda'^{(n+1)}$ и

$$S' = \bigcup_{\langle q,u \rangle \in S} \bigcup_{\langle q,\langle \sigma,v \rangle,q' \rangle \in \Delta} \{\langle q', w^{-1}(u \cdot v) \rangle\}$$

- $Q'^{(n+1)} = Q'^{(n)} \cup codom(\delta'^{(n+1)})$,
- $F'^{(n+1)} = \{S \in Q'^{(n+1)} \mid \exists \langle q, \beta \rangle \in S \ : \ q \in F\}$,
- $\Psi'^{(n+1)} = \{\langle S, \beta \rangle \mid S \in F'^{(n+1)}, \exists \langle q, \beta \rangle \in S \ : \ q \in F\}$.

Лема 5.2.2 *Нека $\mathcal{T}'^{(n)} = \langle \Sigma, \Sigma', Q', q'_0, F', \delta', \lambda', \Psi' \rangle$ е конструиран с горната конструкция след n стъпки. Тогава за всяка дума $w \in \Sigma^*$, такава че $\lambda'^*(q'_0, w)$ и $\delta'^*(q'_0, w)$ са в сила, са изпълнени следните свойства:*

$$\begin{aligned} \lambda'^*(q'_0, w) &= \bigwedge_{q_0 \in I, \langle q_0, \langle w,u \rangle, q \rangle \in \Delta^*} u \\ \delta'^*(q'_0, w) &= \{\langle q, \gamma \rangle \mid \exists q_0 \in I \ \exists \langle q_0, \langle w, u \rangle, q \rangle \in \Delta^* \ : \ \gamma = \lambda'^*(q'_0, w)^{-1} u\}. \end{aligned}$$

Лема 5.2.3 *Нека $\mathcal{T}'^{(n)} = \langle \Sigma, \Sigma', Q', q'_0, F', \delta', \lambda', \Psi' \rangle$ е конструиран с горната конструкция след n стъпки от преобразователя $\mathcal{T} = \langle \Sigma^* \times \Sigma'^*, Q, \{q_0\}, F, \Delta \rangle$. Тогава за всяко състояние $S \in Q'$ и $q \in Q$ е изпълнено $|\{v \in \Sigma'^* \mid \exists \langle q, v \rangle \in S\}| \leq 1$.*

Лема 5.2.4 *Нека* $\mathcal{T}'^{(n)} = \langle \Sigma, \Sigma', Q', q'_0, F', \delta', \lambda', \Psi' \rangle$ *е конструиран с горната конструкция след* n *стъпки от преобразователя* $\mathcal{T} = \langle \Sigma^* \times \Sigma'^*, Q, \{q_0\}, F, \Delta \rangle$. *Тогава за всяко състояние* $S \in Q'$ *и* $\langle q_1, v_1 \rangle, \langle q_2, v_2 \rangle \in S$ *е изпълнено*

$$q_1 \in F \ \& \ q_2 \in F \rightarrow v_1 = v_2.$$

Твърдение 5.2.5 *Нека* $\mathcal{T} = \langle \Sigma^* \times \Sigma'^*, Q, I, F, \Delta \rangle$ *е класически реално-временен краен преобразувател, такъв че индуктивната конструкция на* $\mathcal{T}'$, *представена отгоре приключва, в смисъл, че съществува* $k \in \mathbb{N}$, *така че* $\mathcal{T}'^{(k)} = \mathcal{T}'^{(k+1)} = \mathcal{T}'^{(k+2)} = \ldots = \mathcal{T}'$. *Нека* $\mathcal{T}' = \langle \Sigma, \Sigma', Q', q'_0, F', \delta', \lambda', \Psi' \rangle$. *Тогава* $O_{\mathcal{T}'} = L(\mathcal{T})$.

Теорема 5.2.6 *Нека* $\mathcal{T} = \langle \Sigma^* \times \Sigma'^*, Q, I, F, \Delta \rangle$ *е класически реално-временен краен преобразувател с ограничена вариация. Тогава индуктивната конструкция на* $\mathcal{T}'$, *представена отгоре приключва, в смисъл, че съществува* $k \in \mathbb{N}$, *така че* $\mathcal{T}'^{(k)} = \mathcal{T}'^{(k+1)} = \mathcal{T}'^{(k+2)} = \ldots = \mathcal{T}'$.

Следствие 5.2.7 *Една регулярна функция между думи* $f : \Sigma^* \rightarrow \Sigma'^*$ *се представя с класически подпоследователен преобразувател тогава и само тогава, когато* f *е с ограничена вариация.*

5.3 Разрешимост на въпроса за ограничена вариация на класически краен преобразувател

Лема 5.3.3 *Нека* $\mathcal{T}$ *е класически реално-временен краен преобразувател и нека* q *е състояние от квадратирания изходен автомат* $\mathcal{S}_{\mathcal{T}}$ *на* $\mathcal{T}$.

1. *Ако* $\langle u, v \rangle \in \mathrm{Adm}(q)$ *и съответният път в* $\mathcal{S}_{\mathcal{T}}$ *е* $\pi = q_0 \rightarrow \ldots \overset{\langle \alpha, \beta \rangle}{\rightarrow} q$, *то* $d_S(\alpha, \beta) = |u| + |v|$.

2. *Ако* $\mathrm{Adm}(q)$ *е крайно и съществува* $\langle u, v \rangle \in \mathrm{Adm}(q)$, *такова че* $\langle u, v \rangle$ *не е балансируемо, то всеки цикъл* $\langle q, \langle \alpha, \beta \rangle, q \rangle \in \Delta'^*$ *е с етикет* $\langle \alpha, \beta \rangle = \langle \varepsilon, \varepsilon \rangle$.

Теорема 5.3.4 *Нека* $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ *е тримован класически реално-временен краен преобразувател и нека* $\mathcal{S}_{\mathcal{T}}$ *е квадратираният изходен автомат на* $\mathcal{T}$. *Тогава* $\mathcal{T}$ *е с ограничена вариация тогава и само тогава, когато за всяко състояние* q *от* $\mathcal{S}_{\mathcal{T}}$ *множеството от допустими напредъци* $\mathrm{Adm}(q)$ *е крайно.*

Лема 5.3.5 *Нека* $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ *е тримован класически реално-временен краен преобразувател и нека* $C := \max_{\langle q', \langle \sigma, \alpha \rangle, q'' \rangle \in \Delta} |\alpha|$. *Нека* $\langle u, v \rangle$ *е допустим напредък на състоянието* $\langle p, q \rangle \in Q \times Q$ *на квадратирания изходен автомат* $\mathcal{S}_\mathcal{T}$. *Ако* $\mathcal{T}$ *е с ограничена вариация, то* $|u| < C|Q|^2$ *и* $|v| < C|Q|^2$.

Горните три твърдения ни предоставят процедура за разрешаване на въпроса за ограничена вариация на даден класически реално-временен краен преобразувател, която е реализирана в Програма 8.3.2.

Следствие 5.3.7 *Нека* $\mathcal{T} = \langle \Sigma_I^* \times \Sigma^*, Q, I, F, \Delta \rangle$ *е тримован класически реално-временен краен преобразувател и нека* $\mathcal{S}_\mathcal{T} = \langle \Sigma^* \times \Sigma^*, Q \times Q, I \times I, F \times F, \Delta' \rangle$ *е квадратираният изходен автомат на* $\mathcal{T}$. *Нека* $\mathcal{T}'^{(n)} = \langle \Sigma_I, \Sigma^*, Q', q_0', F', \delta', \lambda', \Psi' \rangle$ *е конструиран след* n *с индуктивната конструкция от предната секция. Нека* $S \in Q'$ *е състояние в* $\mathcal{T}'^{(n)}$. *Тогава за всяка двойка* $\langle p, u \rangle \in S$ *съществува двойка* $\langle q, v \rangle \in S$, *така че* $\langle u, v \rangle$ *е допустим напредък на състоянието* $\langle p, q \rangle$ *(т.е.* $\langle u, v \rangle \in Adm(\langle p, q \rangle)$*).*

Теорема 5.3.8 *Нека* $\mathcal{T} = \langle \Sigma^* \times \Sigma^*, Q, I, F, \Delta \rangle$ *е тримован класически реално-временен краен преобразувател и нека* $C := \max_{\langle q', \langle \sigma, \alpha \rangle, q'' \rangle \in \Delta} |\alpha|$. *Тогава индуктивната конструкция от предната секция приключва тогава и само тогава, когато при всяка стъпка* n *от конструкцията, за съответния подпоследователен преобразувател*

$$\mathcal{T}'^{(n)} = \langle \Sigma_I, \Sigma^*, Q', q_0', F', \delta', \lambda', \Psi' \rangle$$

за всяко състояние S *в* Q' *и всяка двойка* $\langle p, u \rangle \in S$ *е изпълнено* $|u| < C|Q|^2$.

Горната теорема ни дава ефективно условие за проверка на липса на ограничена вариация, което може да се интегрира в индуктивната конструкция от предната секция.

5.4 Минимални подпоследователни крайни преобразуватели и релация на Майхил-Нерод за подпоследователни крайни преобразуватели

Тук ще покажем, че в случая на подпоследователни преобразуватели можем да дефинираме вид релация на Майхил-Нерод, с която да получим минимален подпоследователен преобразувател.

Дефиниция 5.4.1 Нека $f : \Sigma^* \to \Sigma'^*$ е (частична) функция. Тогава

$$\begin{array}{lll} R_f = \{\langle u, v\rangle \in \Sigma^* \times \Sigma^* \mid & \exists u' \in \Sigma'^* \ \exists v' \in \Sigma'^* & \forall w \in \Sigma^* : \\ & (u \cdot w \in dom(f) \leftrightarrow & v \cdot w \in dom(f)) \ \& \\ & (u \cdot w \in dom(f) \rightarrow & u' \in Pref(f(u \cdot w)) \ \& \ v' \in Pref(f(v \cdot w)) \ \& \\ & & u'^{-1} f(u \cdot w) = v'^{-1} f(v \cdot w))\} \end{array}$$

наричаме *релацията на Майхил-Нерод* за f.

Твърдение 5.4.2 *Нека $f : \Sigma^* \to \Sigma'^*$ е функция. Тогава релацията на Майхил-Нерод за f е дясно инвариантна релация на еквивалентност.*

Дефиниция 5.4.3 Нека $f : \Sigma^* \to \Sigma'^*$ е функция. *Най-дългият общ изход* на f е функцията $lco_f : \Sigma^* \to \Sigma'^*$, дефинирана като:

$$lco_f(u) = \begin{cases} \bigwedge\limits_{w \in \Sigma^* \ \&\ u \cdot w \in dom(f)} f(u \cdot w) & \text{ако } u \in Pref(dom(f)) \\ \varepsilon & \text{в противен случай.} \end{cases}$$

Дефиниция 5.4.4 *Подпоследователен краен преобразувател с начален изход* над моноид $\mathcal{M} = \langle M, \circ, e\rangle$ е деветорка $\mathcal{T} = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda, \iota, \Psi\rangle$, така че $\langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda, \Psi\rangle$ е подпоследователен краен преобразувател и $\iota \in M$. Изходната функция на подпоследователен краен преобразувател с начален изход $\mathcal{T}$ дефинираме като $O_{\mathcal{T}}(\alpha) = \iota \circ \lambda^*(q_0, \alpha) \circ \Psi(\delta^*(q_0, \alpha))$.

Твърдение 5.4.6 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi\rangle$ е класически подпоследователен краен преобразувател с начален изход, представящ $f : \Sigma^* \to \Sigma'^*$, с тотална функция на преходите δ. Нека*

$$R_{\mathcal{T}} := \{\langle u, v\rangle \in \Sigma^* \times \Sigma^* \mid \delta^*(q_0, u) = \delta^*(q_0, v)\}.$$

Тогава $R_{\mathcal{T}}$ е дясно инвариантна релация на еквивалентност и $R_{\mathcal{T}}$ е изфиняване на релацията на Майхил-Нерод R_f.

Следствие 5.4.7 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi\rangle$ е класически подпоследователен краен преобразувател с начален изход с тотална функция на преходите δ, представящ $f : \Sigma^* \to \Sigma'^*$. Тогава $|\Sigma^*/R_f| \leq |\Sigma^*/R_{\mathcal{T}}| = |Q|$.*

Твърдение 5.4.8 *Нека $f : \Sigma^* \to \Sigma'^*$ е функция, такава че индексът на релацията на Майхил-Нерод R_f е краен. Нека*

$$\mathcal{T}_f = \left\langle \Sigma, \Sigma', \Sigma^*/R_f, [\varepsilon]_{R_f}, \{[s]_{R_f} \mid s \in dom(f)\}, \delta, \lambda, \iota, \Psi \right\rangle,$$

където:

- $\delta = \{\left\langle [u]_{R_f}, a, [u \cdot a]_{R_f} \right\rangle \mid u \in \Sigma^*, a \in \Sigma\}$,
- $\lambda = \{\left\langle [u]_{R_f}, a, lco_f(u)^{-1} lco_f(u \cdot a) \right\rangle \mid u \in \Sigma^*, a \in \Sigma, u{\cdot}a \in Pref(dom(f))\} \cup$ $\{\left\langle [u]_{R_f}, a, \varepsilon \right\rangle \mid u \in \Sigma^*, a \in \Sigma, u \cdot a \notin Pref(dom(f))\}$,
- $\iota = lco_f(\varepsilon)$,
- $\Psi = \{\left\langle [u]_{R_f}, lco_f(u)^{-1} f(u) \right\rangle \mid u \in dom(f)\}$.

Тогава $\mathcal{T}_f$ е класически подпоследователен краен преобразувател с начален изход и е изпълнено, че $O_{\mathcal{T}_f} = f$ и функцията на преходите δ е тотална.

Теорема 5.4.10 *Нека $f : \Sigma^* \to \Sigma'^*$ е функция. Тогава f се представя от подпоследователен краен преобразувател тогава и само тогава, когато индексът на R_f е краен.*

Теорема 5.4.11 *Нека $f : \Sigma^* \to \Sigma'^*$ е функция, такава че индексът на R_f е краен. Тогава подпоследователният преобразувател $\mathcal{T}_f$ за f е минимален относно броя на състоянията измежду всички класически подпоследователни крайни преобразуватели с начален изход и тотална функция на преходите, представящи f.*

5.5 Минимизация на подпоследователни крайни преобразуватели

В тази секция представяме конструкция за минимизация на даден подпоследователен краен преобразувател, следвайки поддхода от [Mohri, 2000]. Тук представяме процедурата в случай на класически подпоследователен преобразувател. Но представената техника се обобщава за по-общи изходни моноиди, които удовлетворяват някои допълнителни свойства [Gerdjikov and Mihov, 2017a].

Дефиниция 5.5.1 Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ е тримован класически подпоследователен краен преобразувател с начален изход. *Максималният изход на състояние* $mso_{\mathcal{T}}(q)$ за състояние $q \in Q$ се дефинира като

$$mso_{\mathcal{T}}(q) := \bigwedge_{w \in \Sigma^* \ \& \ \delta^*(q,w) \in F} O^q_{\mathcal{T}}(w).$$

Дефиниция 5.5.2 *Каноничната форма* на $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ е $\mathcal{T}' := \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda', \iota', \Psi' \rangle$, където

- $\iota' := \iota \cdot mso_{\mathcal{T}}(q_0)$,
- $\Psi'(q) := mso_{\mathcal{T}}(q)^{-1}\Psi(q)$ за всяко $q \in F$,
- $\lambda'(q, \sigma) := mso_{\mathcal{T}}(q)^{-1}(\lambda(q, \sigma) \cdot mso_{\mathcal{T}}(\delta(q, \sigma)))$, за всяко $q \in Q, \sigma \in \Sigma$, такова че $\delta(q, \sigma)$ е дефинирано.

Твърдение 5.5.8 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ е тримован класически подпоследователен краен преобразувател в канонична форма, представящ функцията f. Нека $q_1, q_2 \in Q$ са състояния. Тогава q_1 и q_2 са еквивалентни тогава и само тогава, когато*

1. *$q_1 \in F$ т.с.т.к. $q_2 \in F$,*
2. *ако $q_1 \in F$, то $\Psi(q_1) = \Psi(q_2)$,*
3. *за всяко $\sigma \in \Sigma$: $\delta(q_1, \sigma)$ е дефинирано т.с.т.к. $\delta(q_2, \sigma)$ е дефинирано. Ако и двете са дефинирани, то $\delta(q_1, \sigma)$ и $\delta(q_2, \sigma)$ са еквивалентни и $\lambda(q_1, \sigma) = \lambda(q_2, \sigma)$.*

Лема 5.5.9 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ е тримован класически подпоследователен краен преобразувател в канонична форма, представящ функцията $f : \Sigma^* \to \Sigma'^*$. Нека $u, v \in \Sigma^*$, $\delta^*(q_0, u) = q_1$ и $\delta^*(q_0, v) = q_2$ са дефинирани. Тогава и R_f v тогава и само тогава, когато q_1 е еквивалентно на q_2.*

Теорема 5.5.10 *Класическият подпоследователен краен преобразувател $\mathcal{T}$ в канонична форма е минимален тогава и само тогава, когато $\mathcal{T}$ е тримован и няма различни еквивалентни състояния в $\mathcal{T}$.*

Преобразуване в канонична форма – намиране на максималните изходи на състоянията

Дефиниция 5.5.11 Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ е класически подпоследователен краен преобразувател с начален изход. Тогава моноидният краен автомат

$$\mathcal{A}_{\mathcal{T}} := \langle \Sigma'^*, Q \cup \{f\}, \{q_0\}, \{f\}, \Delta \rangle ,$$

където $f \notin Q$ е ново състояние и

$$\Delta = \{\langle q, \Psi(q), f \rangle \mid q \in F\} \cup \{\langle q', \lambda(q', \sigma), q'' \rangle \mid \langle q', \sigma, q'' \rangle \in \delta\},$$

наричаме *автомат на изходите* на $\mathcal{T}$.

Ецикът на $\mathcal{A}_\mathcal{T}$ е равен на $\iota^{-1}codom(O_\mathcal{T})$ и за всяко състояние $q \in Q$ е изпълнено

$$L_{\mathcal{A}_\mathcal{T}}(q) = \bigcup_{w\in\Sigma^* \ \&\ \delta^*(q,w)\in F} O^q_\mathcal{T}(w).$$

Следвайки твърдение 3.2.1 конструираме класическия краен автомат $\mathcal{A}'_\mathcal{T}$ с разширено множество на състоянията Q' без ε-преходи, такъв че за всяко $q \in Q$ е изпълнено $L_{\mathcal{A}_\mathcal{T}}(q) = L_{\mathcal{A}'_\mathcal{T}}(q)$. $\mathcal{A}'_\mathcal{T}$ наричаме *експандирания автомат на изходите* на $\mathcal{T}$.

Твърдение 5.5.13 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi\rangle$ е класически подпоследователен краен преобразувател с начален изход и*

$$\mathcal{A}'_\mathcal{T} = \langle \Sigma', Q \cup Q'' \cup \{f\}, q_0, F'', \Delta''\rangle$$

е експандираният автомат на изходите на $\mathcal{T}$. Нека $q \in Q$. Тогава

$$L_{\mathcal{A}'_\mathcal{T}}(q) = \bigcup_{w\in\Sigma^*, \delta^*(q,w)\in F} O^q_\mathcal{T}(w)$$

и

$$mso_\mathcal{T}(q) = \bigwedge L_{\mathcal{A}'_\mathcal{T}}(q).$$

Твърдение 5.5.14 *Нека $\mathcal{A} = \langle \Sigma', Q, q_0, F, \delta\rangle$ е тримован детерминиран краен автомат. Нека*

$$\pi = q_0 \to^{a_1} q_1 \to^{a_2} q_2 \ldots q_{k-1} \to^{a_k} q_k$$

е път с начало q_0. Тогава $w = \bigwedge L(\mathcal{A})$ за $w = a_1a_2\ldots a_k$ тогава и само тогава, когато са изпълнени следните условия:

- *q_k е финално или има повече от един изходящ преход от q_k. Т.е.*

$$q_k \in F \vee |\{\sigma \in \Sigma' \mid !\delta(q_k, \sigma)\}| > 1.$$

- *За всяко i в $0, \ldots, k-1$ състоянието q_i не е финално и съществува единствен преход от q_i. Т.е.*

$$\forall i \in \{0, \ldots, k-1\} : q_i \notin F \ \&\ |\{\sigma \in \Sigma' \mid !\delta(q_i, \sigma)\}| = 1.$$

Следствие 5.5.15 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi\rangle$ е тримован класически подпоследователен краен преобразувател с начален изход и*

$$\mathcal{A}'_\mathcal{T} = \langle \Sigma', Q \cup Q'' \cup \{f\}, q_0, F'', \Delta''\rangle$$

е експандираният автомат на изходите на $\mathcal{T}$. *Нека* $\mathcal{D}^q = \langle \Sigma', Q_q, \{q\}, F_q, \delta_q \rangle$ *е детерминиран краен автомат, получен при детерминизация на* ${\mathcal{A}'_{\mathcal{T}}}^q = \langle \Sigma', Q \cup Q'' \cup \{f\}, q, F'', \Delta'' \rangle$. *Тогава*

$$mso_{\mathcal{T}}(q) = \bigwedge L(\mathcal{D}^q) = w_q,$$

където $w_q \in \Sigma'^*$ *е етикетът на единствения максимален път в* $\mathcal{D}^q$.

Забележка 5.5.16 За да намерим най-дългия общ префикс на езика на ${\mathcal{A}'_{\mathcal{T}}}^q$ (Следствие 5.5.15), можем да подходим като детерминираме само началната част на автомата, докато достигнем до състояние q_k, за което условията от Твърдение 5.5.14 са изпълнени. Така само малка част от автомата се детерминизира. Сложността на намирането на $mso_{\mathcal{T}}(q)$ е $O(|w_q||Q \cup Q''|^2)$ (Програма 8.3.5).

Псевдо-минимизация - конструиране на минимален подпоследователен преобразувател

Твърдение 5.5.18 *Нека* $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \iota, \Psi \rangle$ *е каноничен класически подпоследователен краен преобразувател с начален изход. Разглеждаме азбуката*

$$\Gamma := \{\langle c, \lambda(q,c) \rangle \mid c \in \Sigma, q \in Q \ \&\ !\delta(q,c)\},$$

и множеството от преходи $\delta_{\mathcal{A}} := \{\langle q, \langle c, \lambda(q,c)\rangle, q' \rangle \mid \langle q, c, q' \rangle \in \delta\}$. *Тогава*

$$\mathcal{A}_{\mathcal{T}} := \langle \Gamma, codom(\Psi), Q, q_0, F, \delta_{\mathcal{A}}, \Psi \rangle$$

е $codom(\Psi)$*-оцветен детерминиран краен автомат. Нека*

$$\mathcal{A}'_{\mathcal{T}} = \langle \Gamma, codom(\Psi), Q', q'_0, F', \delta'_{\mathcal{A}}, \Psi' \rangle$$

е минималният $codom(\Psi)$*-оцветен детерминиран краен автомат, еквивалентен на* $\mathcal{A}_{\mathcal{T}}$. *Тогава подпоследователният краен преобразувател*

$$\mathcal{T}' := \langle \Sigma, \Sigma', Q', q'_0, F', \delta', \lambda', \iota, \Psi' \rangle,$$

където

$$\begin{aligned} \delta' &:= \{\langle q, c, q' \rangle \mid \langle q, \langle c, \alpha \rangle, q' \rangle \in \delta'_{\mathcal{A}}\} \\ \lambda' &:= \{\langle q, c, \alpha \rangle \mid \langle q, \langle c, \alpha \rangle, q' \rangle \in \delta'_{\mathcal{A}}\}, \end{aligned}$$

е минималният каноничен подпоследователен краен преобразувател, еквивалентен на $\mathcal{T}$.

Забележка 5.5.20 За крайни функции между думи съществуват директни методи за конструиране на минмалния краен подпоследователен преобразувател, които имат по-добра ефективност [Mihov and Maurel, 2001].

5.6 Числови подпоследователни крайни преобразуватели

В тази секция показваме как резултатите, получени в предходните секции, се прехвърлят за случая на изходен моноид на естествените числа относно събиране.

Дефиниция 5.6.1 Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \Psi \rangle$ е класически подпоследователен преобразувател. Нека $\phi : \Sigma'^* \to \mathcal{M}$ е хомоморфизъм между моноидите Σ'^* и $\mathcal{M} = \langle M, \bullet, e \rangle$. Тогава моноидният подпоследователен преобразувател $\mathcal{T}_\phi = \langle \Sigma, \mathcal{M}, Q, q_0, F, \delta, \lambda_\phi, \Psi_\phi \rangle$, където

- $\lambda_\phi := \lambda \circ \phi$ (т.е. $\lambda_\phi(q, \sigma) := \phi(\lambda(q, \sigma))$),
- $\Psi_\phi := \Psi \circ \phi$ (т.е. $\Psi_\phi(q) := \phi(\Psi(q))$),

наричаме *образ на $\mathcal{T}$ през ϕ*.

Твърдение 5.6.2 *Нека $\mathcal{T}_\phi$ е образ на подпоследователния преобразувател $\mathcal{T}$ през ϕ, където $\phi : \Sigma'^* \to \mathcal{M}$ е хомоморфизъм между моноидите Σ'^* и $\mathcal{M}$. Тогава $O_{\mathcal{T}_\phi} = O_{\mathcal{T}} \circ \phi$.*

Твърдение 5.6.3 *Нека $\Sigma' = \{1\}$ и $\mathcal{N} = \langle \mathbb{N}, +, 0 \rangle$. Тогава функцията $\varphi : \Sigma'^* \to \mathbb{N}$, дефинирана като*

$$\varphi(1^n) := n, \text{ за всяко } n \in \mathbb{N},$$

е моноиден хомоморфизъм между моноидите Σ'^* и $\mathcal{N}$.

Твърдение 5.6.4 *Нека $\mathcal{T} = \langle \Sigma, \Sigma', Q, q_0, F, \delta, \lambda, \Psi \rangle$ е класически подпоследователен преобразувател и $\Sigma' = \{1\}$. Тогава $\mathcal{T}$ може да бъде получен като образ на $\mathcal{T}_\varphi$ през φ^{-1} т.е. $\mathcal{T} = \mathcal{T}_{\varphi_{\varphi^{-1}}}$.*

Следствие 5.6.5 *Всички резултати за детерминизация на преобразуватели (Секция 5.2), разрешаване на функционалност (Секция 4.6), разрешавана на ограничена вариация (Секция 5.3) и минимизация на преобразуватели (Секция 5.4), които получихме за класически подпоследователни преобразуватели, директно се пренасят за моноидни подпоследователни преобразуватели над моноида $\mathcal{N}$.*

6 Бимашини

В тази глава ще изследваме бимашините. Те представляват специфичен клас детерминирани машини с краен брой състояния, които представят класа на регулярните функции между думи.

6.1 Основни дефиниции

Дефиниция 6.1.1 *Моноидна бимашина* наричаме четворка $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$, където

- $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид,
- $\mathcal{A}_L = \langle \Sigma, L, s_L, L, \delta_L \rangle$ and $\mathcal{A}_R = \langle \Sigma, R, s_R, R, \delta_R \rangle$ са детерминирани крайни автомати, които наричаме *ляв* и *десен автомат* на бимашината;
- $\psi : (L \times \Sigma \times R) \to M$ е частична функция, наречена *функция на изходите.*

Ако $\mathcal{M}$ е свободен моноид, то $\mathcal{B}$ наричаме *класическа бимашина.* В този случай ще предполагаме, че изходната азбука съвпада с азбуката на левия и десния автомат Σ, и ще означаваме $\mathcal{B} = \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$.

Дефиниция 6.1.2 Нека $\mathcal{M} = \langle M, \circ, e \rangle$ е моноид и $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ е моноидна бимашина. Нека Σ е азбуката на $\mathcal{A}_L$ и $\mathcal{A}_R$. Разглеждаме входната дума $t = \sigma_1\sigma_2 \cdots \sigma_n \in \Sigma^*$ $(n \geq 0)$ със символи σ_i $(i = 1, \ldots, n)$. Ако са дефинирани $\delta_L^*(\sigma_1\sigma_2 \cdots \sigma_i)$ и $\delta_R^*(\sigma_n\sigma_{n-1} \cdots \sigma_i)$ за $(1 \leq i \leq n)$, то получаваме двойка пътища

$$\begin{array}{lccccccc} \pi_L: & l_0 \to^{\sigma_1} l_1 & \to \ldots & l_{i-1} \to^{\sigma_i} l_i & \to \ldots & l_{n-1} \to^{\sigma_n} l_n \\ \pi_R: & r_0 \leftarrow^{\sigma_1} r_1 & \leftarrow \ldots & r_{i-1} \leftarrow^{\sigma_i} r_i & \leftarrow \ldots & r_{n-1} \leftarrow^{\sigma_n} r_n, \end{array}$$

където π_L е път в левия автомат $\mathcal{A}_L$ с начало $l_0 := s_L$ и π_R е път в десния автомат $\mathcal{A}_R$ с начало $r_n := s_R$. Ако всички изходи $\psi(l_{i-1}, \sigma_i, r_i)$ са дефинирани $(1 \leq i \leq n)$, то (π_L, π_R) наричаме *успешна двойка пътища* на $\mathcal{B}$ с етикет $\sigma_1\sigma_2 \ldots \sigma_n$ и изход

$$O_{\mathcal{B}}(t) := \psi(l_0, \sigma_1, r_1) \circ \psi(l_1, \sigma_2, r_2) \circ \ldots \circ \psi(l_{i-1}, \sigma_i, r_i) \circ \ldots \circ \psi(l_{n-1}, \sigma_n, r_n).$$

В частния случай $t = \varepsilon$ имаме $O_{\mathcal{B}}(t) = e$. Частичната функция $O_{\mathcal{B}}$ наричаме *изходна функция* на бимашината или *функция, представена от бимашината.*

Дефиниция 6.1.4 Нека $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ е моноидна бимашина. *Обобщената функция на изходите* ψ^* на $\mathcal{B}$ дефинираме индуктивно:

- $\psi^*(l, \varepsilon, r) = e$ за всяко $l \in L, r \in R$;
- $\psi^*(l, t\sigma, r) = \psi^*(l, t, \delta_R(r, \sigma)) \circ \psi(\delta_L^*(l, t), \sigma, r)$ за $l \in L, r \in R, t \in \Sigma^*, \sigma \in \Sigma$.

6.2 Еквивалентност на регулярните функции върху думи и класическите бимашини

Тук следваме подхода представен в [Gerdjikov et al., 2017].

Твърдение 6.2.1 *За всяка моноидна бимашина $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ съществува моноиден краен преобразувател $\mathcal{A} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$, така че $O_\mathcal{B} = L(\mathcal{A})$.*

Доказателство. [Конструкция] Конструираме реално-временния краен преобразувател

$$\mathcal{A} := \langle \Sigma^* \times \mathcal{M}, L \times R, \{s_L\} \times R, L \times \{s_R\}, \Delta \rangle ,$$

където Δ съдържа всички преходи $\langle l, r \rangle \to^\sigma_m \langle l', r' \rangle$, такива, че $\delta_L(l, \sigma) = l', \delta_R(r', \sigma) = r, \psi(l, \sigma, r') = m$.

Твърдение 6.2.5 *Нека $\mathcal{T} = \langle \Sigma^* \times \mathcal{M}, Q, I, F, \Delta \rangle$ е тримован функционален реално-временен краен преобразувател с изход в моноида $\mathcal{M} = \langle M, \circ, e \rangle$, такъв че $\langle \varepsilon, e \rangle \in L(\mathcal{T})$. Тогава съществува моноидна бимашина $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$, така че $L(\mathcal{T}) = O_\mathcal{B}$.*

Доказателство. [Конструкция] *Десният автомат* на бимашината $\mathcal{A}_R = \langle \Sigma, Q_R, s_R, Q_R, \delta_R \rangle$ се дефинира като резултат от детерминизацията на обърнатия подлежащ автомат на $\mathcal{T}$. Това означава, че $Q_R \subseteq 2^Q$, $s_R = F$ и

$$\delta_R(R, a) := \{q \in Q \mid \exists q' \in R, m \in M : \langle q, \langle a, m \rangle , q' \rangle \in \Delta\}.$$

Селекторна функция ще наричаме функция $\phi : Q_R \to Q$, която избира от всяко непразно множество $P \in Q_R$ елемент $p = \phi(P) \in P$. Състояние на *левия автомат* $\mathcal{A}_L = \langle \Sigma, Q_L, s_L, Q_L, \delta_L \rangle$ е двойка от подмножество на Q и селекторна функция ϕ. Т.е. $Q_L \subseteq 2^Q \times 2^{Q_R \times Q}$, поради което Q_L е крайно. Състоянията и преходите на левия автомат дефинираме индуктивно:

- $s_L := \langle I, \phi_0 \rangle$, където $\phi_0(R) := \begin{cases} \text{произволен елемент на } R \cap I & \text{ако } R \cap I \neq \emptyset \\ \text{не е дефинирано} & \text{в противен случай.} \end{cases}$
- За $\langle L, \phi \rangle \in Q_L$ и $a \in \Sigma$ дефинираме $\delta_L(\langle L, \phi \rangle , a) := \langle L', \phi' \rangle$, където
  - $L' := \{q' \mid \exists q \in L, m \in M : \langle q, \langle a, m \rangle , q' \rangle \in \Delta\}$.
  - $\phi'(R') := \begin{cases} \text{произволен елемент на } \{q' \in R' \mid \exists m \in M : \langle q, \langle a, m \rangle , q' \rangle \in \Delta\} \\ \qquad \text{ако } q = \phi(\delta_R(R', a)) \text{ е дефинирано} \\ \text{не е дефинирано} \qquad \text{в противен случай.} \end{cases}$

Нека са дадени двойка състояния $\langle L, \phi\rangle$ и R' на левия и десния автомат и $a \in \Sigma$. Нека $\langle L', \phi'\rangle := \delta_L(\langle L, \phi\rangle, a)$ и $R := \delta_R(R', a)$. Тогава

$$\psi(\langle L, \phi\rangle, a, R') := \begin{cases} \text{произволен елемент на} \\ \{m \mid \langle \phi(R), \langle a, m\rangle, \phi'(R')\rangle \in \Delta\} & \text{ако } \phi(R) \text{ е дефинирано} \\ \text{не е дефинирано} & \text{в противен случай.} \end{cases}$$

6.3 Псевдо-минимизация на моноидните бимашини

За да минимизираме бимашина, не е достатъчно просто да минимизираме двата ѝ автомата. Процедурата за минимизация трябва да вземе под внимание и изходната функция на бимашината.

Дефиниция 6.3.1 Нека $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi\rangle$ е моноидна бимашина с автомати $\mathcal{A}_L = \langle \Sigma, L, s_L, L, \delta_L\rangle$ и $\mathcal{A}_R = \langle \Sigma, R, s_R, R, \delta_R\rangle$. *Лявата профилна функция* $\psi_L : L \to 2^{\Sigma \times R \times \Sigma^*}$ се дефинира като

$$\psi_L(q_L) := \{\langle \sigma, q_R, \psi(q_L, \sigma, q_R)\rangle \mid \sigma \in \Sigma, q_R \in R\}.$$

Множеството от леви профили на $\mathcal{B}$ е $\Gamma_L := \{\psi_L(q_L) \mid q_L \in L\}$.

Конструкция на псевдо-минимизирана бимашина Можем да разглеждаме $\psi_L(q_L)$ като цвят на q_L. Да разгледаме оцветения краен детерминиран автомат $\mathcal{A}^c_L = \langle \Sigma, \Gamma_L, L, s_L, L, \delta_L, \psi_L\rangle$ с цветове от множеството на левите профили. Чрез минимизация на този оцветен детерминиран краен автомат получаваме

$$\mathcal{A}^c_L{}' = \langle \Sigma, \Gamma_L, \{[q]_{\equiv_L} \mid q \in L\}, [s_L]_{\equiv_L}, \{[q]_{\equiv_L} \mid q \in L\}, \delta'_L, col'\rangle,$$

където $col'([q]_{\equiv_L}) := col(q)$. Тогава минимизираният ляв автомат е

$$\mathcal{A}_L{}' := \langle \Sigma, \{[q]_{\equiv_L} \mid q \in L\}, [s_L]_{\equiv_L}, \{[q]_{\equiv_L} \mid q \in L\}, \delta'_L\rangle.$$

Десният автомат на бимашината минимизираме по съответния дуален начин и получаваме:

$$\mathcal{A}_R{}' := \langle \Sigma, \{[q]_{\equiv_R} \mid q \in R\}, [s_R]_{\equiv_R}, \{[q]_{\equiv_R} \mid q \in R\}, \delta'_R\rangle.$$

Новата функция на изходите дефинираме като

$$\psi'([l]_{\equiv_L}, \sigma, [r]_{\equiv_R}) := \psi(l, \sigma, r).$$

Така получаваме бимашина $\mathcal{B}' = \langle \mathcal{M}, \mathcal{A}'_L, \mathcal{A}'_R, \psi'\rangle$.

Дефиниция 6.3.3 Нека $\mathcal{B} = \langle \mathcal{M}, \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ е моноидна бимашина. Тогава моноидната бимашина $\mathcal{B}' = \langle \mathcal{M}, \mathcal{A}'_L, \mathcal{A}'_R, \psi' \rangle$, конструирана чрез минимизация на $\mathcal{A}_L$ и $\mathcal{A}_R$ (разглеждани като оцветени автомати, както е показано по-горе като $\psi'([l], \sigma, [r]) := \psi(l, \sigma, r)$), наричаме *псевдоминимална бимашина*, еквивалентна на $\mathcal{B}$.

6.4 Директна композиция на класически бимашини

Формална конструкция. Нека $\mathcal{B}' = \langle \mathcal{A}'_L, \mathcal{A}'_R, \psi' \rangle$ и $\mathcal{B}'' = \langle \mathcal{A}''_L, \mathcal{A}''_R, \psi'' \rangle$ са две бимашини над азбуката Σ, и

$$\begin{array}{ll} \mathcal{A}'_L = \langle \Sigma, L', s'_L, L', \delta'_L \rangle & \mathcal{A}'_R = \langle \Sigma, R', s'_R, R', \delta'_R \rangle \\ \mathcal{A}''_L = \langle \Sigma, L'', s''_L, L'', \delta''_L \rangle & \mathcal{A}''_R = \langle \Sigma, R'', s''_R, R'', \delta''_R \rangle \, . \end{array}$$

Нека

$$\begin{array}{rcl} \mathcal{A}^N_L & := & \langle \Sigma, L' \times R' \times L'', \{s'_L\} \times R' \times \{s''_L\}, L' \times R' \times L'', \Delta_L \rangle \\ \mathcal{A}^N_R & := & \langle \Sigma, L' \times R' \times R'', L' \times \{s'_R\} \times \{s''_R\}, L' \times R' \times R'', \Delta_R \rangle \, , \end{array}$$

където

1. Δ_L съдържа всички преходи от вида $\langle \langle l'_1, r'_1, l'''_1 \rangle, a, \langle l'_2, r'_2, l''_2 \rangle \rangle$, такива че $\delta'_L(l'_1, a) = l'_2$, $\delta'_R(r'_2, a) = r'_1$, $\delta''^*_L(l''_1, \psi'(l'_1, a, r'_2)) = l''_2$,
2. Δ_R съдържа всички преходи от вида $\langle \langle l'_2, r'_2, r''_2 \rangle, a, \langle l'_1, r'_1, r''_1 \rangle \rangle$, такива че $\delta'_L(l'_1, a) = l'_2$, $\delta'_R(r'_2, a) = r'_1$, $\delta''^*_R(r''_2, \rho(\psi'(l'_1, a, r'_2))) = r''_1$.

Нека $\mathcal{A}_L$ и $\mathcal{A}_R$ са детерминираните крайни автомати получени от $\mathcal{A}^N_L$ and $\mathcal{A}^N_R$ чрез подмножествената конструкция, дадена в Теорема 3.2.2. Т.е.

$$\begin{array}{rcl} \mathcal{A}_L & = & \langle \Sigma, Q_L, s_L, Q_L, \delta_L \rangle \\ \mathcal{A}_R & = & \langle \Sigma, Q_R, s_R, Q_R, \delta_R \rangle, \end{array}$$

където

$$\begin{array}{rcl} s_L & = & \{s'_L\} \times R' \times \{s''_L\} \\ s_R & = & L' \times \{s'_R\} \times \{s''_R\} \\ \hat{\delta}_L(A, a) & = & \{\langle l'_2, r'_2, l''_2 \rangle \mid \exists \langle l'_1, r'_1, l'''_1 \rangle \in A : \langle \langle l'_1, r'_1, l'''_1 \rangle, a, \langle l'_2, r'_2, l''_2 \rangle \rangle \in \Delta_L\} \\ \hat{\delta}_R(A, a) & = & \{\langle l'_1, r'_1, r''_1 \rangle \mid \exists \langle l'_2, r'_2, r''_2 \rangle \in A : \langle \langle l'_2, r'_2, r''_2 \rangle, a, \langle l'_1, r'_1, r''_1 \rangle \rangle \in \Delta_R\} \\ Q_L & = & \{A \in 2^{L' \times R' \times L''} \mid \exists v \in \Sigma^* : A = \hat{\delta}^*_L(s_L, v)\} \\ Q_R & = & \{A \in 2^{L' \times R' \times R''} \mid \exists v \in \Sigma^* : A = \hat{\delta}^*_R(s_R, v)\} \\ \delta_L & = & \hat{\delta}_L|_{Q_L \times \Sigma} \\ \delta_R & = & \hat{\delta}_R|_{Q_R \times \Sigma} \end{array}$$

За състоянията A и B на $\mathcal{A}_L$ и $\mathcal{A}_R$ и $a \in \Sigma$ дефинираме

$$\psi(A, a, B) := w \text{ тогава и само тогава, когато}$$

съществуват тройки $\langle l'_1, r'_1, l''_1 \rangle \in A$ и $\langle l'_2, r'_2, r''_2 \rangle \in B$, така че съществуват $l''_2 \in L''$ и $r''_1 \in R''$, такива че

$$\begin{aligned} \langle \langle l'_1, r'_1, l''_1 \rangle, a, \langle l'_2, r'_2, l''_2 \rangle \rangle &\in \Delta_L, \\ \langle \langle l'_2, r'_2, r''_2 \rangle, a, \langle l'_1, r'_1, r''_1 \rangle \rangle &\in \Delta_R, \end{aligned}$$

и $w = \psi''^*(l''_1, \psi'(l'_1, a, r'_2), r''_2)$.

Твърдение 6.4.1 *Нека $\mathcal{B}'$, $\mathcal{B}''$ са две класически бимашини над азбуката Σ. Нека класическата бимашина $\mathcal{B}$ и нейната изходна функция ψ са конструирани чрез горната конструкция. Тогава ψ е коректно дефинирана и $\mathcal{B} := \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ представя композицията на бимашините $\mathcal{B}'$ и $\mathcal{B}''$, т.е. $O_{\mathcal{B}} = O_{\mathcal{B}'} \circ O_{\mathcal{B}''}$.*

7 Програмният език *C(M)*

В тази глава представяме езика за програмиране *C(M)*. Този език се използва в останалата част от настоящия труд за реализирането на представените алгоритми.

Изразите на *C(M)* наподобяват обозначенията, използвани за представяне на формални конструкции в теоретико-множествен стил. Обичайните обекти като множества, функции, релации, n-торки и т.н. са естествено интегрирани в езика. За разлика от императивните езици като C или Java, *C(M)* е функционален декларативен език за програмиране. *C(M)* има някои прилики с Haskell [Hutton, 2007], но използва стандартните математически обозначения като в SETL [Schwartz et al., 1986]. За да реализираме решението на дадена задача, ние просто формално описваме вида на математическия обект, който искаме да получим. Това ни позволява да се съсредоточим върху математическите стъпки на абстрактно ниво, вместо да описваме детайлите за изпълнение на ниско ниво. Компилаторът *C(M)* компилира съставената програма на *C(M)* до ефективен C код, който може да бъде изпълнен след компилация. Тъй като е лесно да се четат *C(M)* програми, описанието на псевдо-код става излишно.

Всички представените програми са тествани и напълно функционални и могат да се компилират без модификации и да се стартират на компютър. Компилаторът за езика *C(M)* е свободно достъпен на адрес `http://lml.bas.bg/~stoyan/lmd/C(M).html`.

7.1 Основи на езика и прости примери

Вероятно най-доброто начално представяне на езика *C(M)* е с няколко примера.

Пример 7.1.1 Композицията на релации R_1 и R_2 се дефинира като $R_1 \circ R_2 := \{\langle a, c\rangle \mid \exists b : \langle a, b\rangle \in R_1, \langle b, c\rangle \in R_2\}$. В *C(M)* са на разположение средствата за описване на множества – можем да дефинираме композицията директно като

$$\text{compose}(R_1, R_2) := \{(a, c) \mid (a, b) \in R_1, (b, c) \in R_2\};$$

Ако са дадени две релации R_1 и R_2, горната функция връща множеството от двойки (a, c), където (a, b) пробягва R_1 (b е произволно) и (b, c) пробягва R_2.
Математически *n-тата композиция* $R^{\langle n\rangle}$ на бинарна релация R се дефинира индуктивно:

- $R^1 := R$,
- $R^{i+1} := R^i \circ R$

Експлицитните индуктивни дефиниции са основно средство в *C(M)*. Тези дефиниции започват с базова стъпка, като например "step 1", в която се конструира базовата форма на елемента. След това стъпка "step i+1" описва как да се получи $i + 1$-ият вариант на обекта от i-ия вариант. Клаузата "until" описва условие за край на индукцията. Следвайки тази схема, n-тата композиция може да се опише на *C(M)* като обекта "$\text{compose}_\text{n}(R, n)$" по следния начин:

$\text{compose}_\text{n}(R, n) := R'$, **where**
  $R' :=$ **induction**
    step 1 :
      $R'^{(1)} := R$;
    step $i + 1$:
      $R'^{(i+1)} := \text{compose}(R'^{(i)}, R)$;
    until $i = n$
    ;
  ;

Математически транзитивното затваряне на бинарна релация R се де-

финира като релацията

$$C_R := \bigcup_{i=1}^{\infty} R^i.$$

За конструкцията на C_R можем да подходим с индукция, дефинирайки $C_R^{(n)} = \bigcup_{i=1}^{n} R^i$. За крайна релация индуктивната стъпка трябва да се повтаря, докато се изпълни $R^{\langle n+1 \rangle} \subseteq C_R^{(n)}$. В *C(M)* можем да възпроизведем същата конструкция и да конструираме C_R чрез следната индукция:

$C_R :=$ **induction**
  step 1 :
    ${C_R}^{(1)} := R$;
  step $n+1$:
    ${C_R}^{(n+1)} := {C_R}^{(n)} \cup \mathrm{compose_n}(R, n+1)$;
  until $\mathrm{compose_n}(R, n+1) \subseteq {C_R}^{(n)}$
  ;

Програма 7.1.2 За да завършим тази *C(M)* програма, трябва да специфицираме типа на всеки обект. Резултатната програма има следния вид:

```
1  REL is 2^(N×N);
2  compose : REL × REL → REL;
3  compose(R1, R2) := {(a, c) | (a, b) ∈ R1, (b, c) ∈ R2};
4  compose_n : REL × N → REL;
5  compose_n(R, n) := R', where
6     R' := induction
7        step 1 :
8           R'^(1) := R;
9        step i + 1 :
10          R'^(i+1) := compose(R'^(i), R);
11       until i = n
12       ;
13    ;
14 transitiveClosure : REL → REL;
15 transitiveClosure(R) := C_R, where
16    C_R := induction
17       step 1 :
18          C_R^(1) := R;
19       step n + 1 :
20          C_R^(n+1) := C_R^(n) ∪ compose_n(R, n + 1);
```

21 **until** $\text{compose}_\text{n}(R, n+1) \subseteq {C_R}^{(n)}$
22 ;
23 ;

Математически даден обект може да бъде описан по различен начин и няма голямо значение как точно обектът е специфициран. От изчислителна гледна точка начинът на описание има голямо значение за сложноста относно време за извършване на изчислението.

Програма 7.1.3 Следващата подобрена конструкция избягва неефективностите за изчисление на транзитивното затваряне. Първо, построяваме функция F_R, чрез която композицията се изчислява ефективно. Второ, в индукцията на всяка стъпка се разглежда само по една двойка от C_R. По този начин, стартирайки от ${C_R}^{(0)} := R$, в стъпка $n+1$ генерираме композицията на $n+1$-ия елемент в $C_R^{(n)}$ с релацията R, докато се изчерпи множеството $C_R^{(n)}$.
Програмата използва две нови средства на *C(M)*, използва анализ на случаи в дефиниция (ред 9), използва оператор за "функционализиране" $\mathcal{F}$ в ред 3, виж Дефиниция 1.1.16.

1 $\text{transitiveClosure} : 2^{\mathbb{N}\times\mathbb{N}} \to 2^{\mathbb{N}\times\mathbb{N}}$;
2 $\text{transitiveClosure}(R) := C_R$, **where**
3 $F_R := \mathcal{F}_{1\to 2}(R)$;
4 $C_R :=$ **induction**
5 **step** 0 :
6 ${C_R}^{(0)} := R$;
7 **step** $n+1$:
8 $(a, b) := ({C_R}^{(n)}$ **as** $(\mathbb{N}\times\mathbb{N})^*)_{n+1}$;
9 ${C_R}^{(n+1)} := \begin{cases} {C_R}^{(n)} \cup \{(a, c) \mid c \in \text{F}_\text{R}(b)\} & \textbf{if } !\,\text{F}_\text{R}(b) \\ {C_R}^{(n)} & \textbf{otherwise;} \end{cases}$
10 **until** $n = \left|{C_R}^{(n)}\right|$
11 ;
12 ;

7.2 Типове, термове и изкази в *C(M)*

В тази секцията на представения труд са описани допустимите типове, термове и изкази в *C(M)*.

8 Имплементация на крайни автомати, преобразуватели и бимашини с *C(M)*

В тази глава са представени реализации на *C(M)* на автоматните конструкции от предходните глави. В настоящия автореферат са дадени само описания на програмите, като съответните им пълни имплементации са дадени в представения труд.

8.1 Имплементации на алгоритми за крайни автомати на *C(M)*

Програма 8.1.1 Тук представяме основните типове за крайни автомати. В нашата формализация крайният автоматът е с *множество* от начални състояния, а етикетите на преходите са произволни *думи* над входната азбука.

Програма 8.1.2 Следващите конструкции представят регулярните операции **обединение**, **конкатениция** и **звезда на Клини** за крайни автомати.

Програма 8.1.3 Следващите "допълнителни конструкции" са дадени за удобство. Представяме **положителна итерация на Клини** за даден автомат, **опционалност** за автомат, автомат, разпознаващ **множество от символи** и автомата, разпознаващ **всички думи** над дадена азбука.

Програма 8.1.4 Дадената *C(M)* програма премахва **ε-преходи**, като следва Твърдение 2.5.4.

Програма 8.1.5 Дадената *C(M)* програма премахва **ε-преходи, запазвайки езиците на състоянията**, като следва Твърдение 2.5.6.

Програма 8.1.6 Следващата програма за **тримоване** на автомат изтрива всички състояния, които не са достижими от начално, и всички състояния, от които не е достижимо финално (Дефиниция 2.5.1).

Програма 8.1.7 При даден краен автомат нашата следваща програма конструира **класически** краен автомат, т.е. автомат, в който всеки етикет на преход е с дължина ≤ 1.

Програма 8.1.8 Също като в общия случай (Програма 8.1.6), **тримоване** на детерминиран краен автомат означава премахване на всички състояния, които не са достижими от начално, и всички състояния, от които не е достижимо финално (Дефиниция 2.5.1).

Програма 8.1.9 Тук представяме ефективна конструкция за детерминизация (Теорема 3.2.2), като строим детерминиран автомат, на който всички състояния са достижими от началното.

Програма 8.1.10 Следващият алгоритъм (product_Δ) конструира функцията на преходите на автомат, който е декартово произведение на два детерминирани входни автомата.

Горната функция е съществена част от програмите за сечение и допълнение дадени по-долу.

Програма 8.1.11 Следващата *C(M)* програма реализира **сечение** на крайни автомати, както е описано в част 1 от Твърдение 3.3.2.

Програма 8.1.12 Следващата *C(M)* програма реализира **допълнение** на крайни автомати, както е описано в част 2 от Твърдение 3.3.2.

Програма 8.1.13 Следващата *C(M)* програма реализира **посимволно обръщане** на крайни автомати, както е описано в Твърдение 3.3.3.

Програма 8.1.14 Алгоритъмът реализира процедура за **минимизация** на детерминирани крайни автомати, базирана на индуктивната конструкция представена в Следствие 3.5.4, използвайки функциите дефинирани в Твърдение 3.5.6.

Програма 8.1.15 Тук реализираме две основни функции върху детерминирани крайни автомати. Първата функция 'C_δ' пресмята транзитивното затваряне на функцията на преходите δ^*. Втората функция 'stateseq' връща последователността от състояния в автомата, стартирайки от дадено състояние с дадена дума.

Пример 8.1.16 За да илюстрираме използването на представените алгоритми, представяме програма за конструирането на детерминиран краен автомат, разпознаващ всички валидни дати от грегорианския календар във формата

```
AUGUST 11, 1996
```

Следваме приблизително метода от [Karttunen et al., 1997a]. Дати като "`FEBRUARY 30, 2015`" или "`APRIL 31, 1921`" лесно се описват като некоректни. По-интересни са датите през висококосните години. "`FEBRUARY 29, 2000`" и "`FEBRUARY 29, 2016`" са валидни, но "`FEBRUARY 29, 2017`" и "`FEBRUARY 29, 1900`" не съществуват.

8.2 Имплементации на алгоритми за класически крайни преобразуватели на $C(M)$

В тази секция показваме имплементации на $C(M)$ на основните алгоритми за n-лентови автомати както, са описани в Секция 4. Тук ще се ограничим до 2-лентови автомати над свободен моноид, т.е. класически крайни преобразуватели.

Програма 8.2.1 Като начало дефинираме типовете необходими за представянето на краен преобразувател над свободен моноид, функцията за **преименуване** на състояния на класически краен преобразувател и преобразувател, който преобразува дадена дума в друга.

Програма 8.2.2 Следващата програма описва **обединение**, **конкатенация**, **звезда на Клини**, **положителна итерация на Клини**, **опционалност** и $\langle\varepsilon,\varepsilon\rangle$**-елиминация** за крайни преобразуватели.

Програма 8.2.3 Програмата взима два 1-лентови автомата за вход и конструира преобразувател представящ **декартовото произведение** на два автомата, както е показано в Твърдение 4.2.1, част 1.

Програма 8.2.4 Следвайки Твърдение 4.2.1, програмата представя конструкции за **проекции** на 2-лентов автомат по първата и втората лента, **обръщане на релацията** и **идентитет** за даден автоматен език.

Програма 8.2.5 Следващата програма по даден краен преобразувател строи еквивалентен класически 2-лентов **посимволен** автомат, т.е. всички етикети на преходи са в $\Sigma^{\varepsilon} \times \Sigma^{\varepsilon}$, следвайки Забележка 4.3.3.

Програма 8.2.6 Програмата конструира краен преобразувател, който е **композиция** на два крайни преобразувателя, в съответствие на Твърдение 4.3.4.

Програма 8.2.7 Следващият алгоритъм конструира краен преобразувател, който представя **посимволно обръщане** на краен преобразувател.

Преобразуване до реално-временни и псевдо-детерминирани преобразуватели

Програма 8.2.8 Следващият алгоритъм реализира премахване на преходи с етикет ε по горната лента, както това е показано в Твърдение 4.4.8, и преобразуването до **реално-временен преобразувател**.

Програма 8.2.9 Алгоритъмът конструира **псевдо-детерминиран преобразувател** еквивалентен на даден входен преобразувател, използвайки метода, описан в Твърдение 3.7.2.

Програма 8.2.10 Програмата конструира **псевдо-минимален преобразувател**, следвайки Твърдение 3.7.4.

Разрешаване функционалност на преобразуватели

Програма 8.2.11 Следващата програма конструира **квадратирания изходен автомат** за реално-временен преобразувател в съответствие с Дефиниция 4.6.4.

Програма 8.2.12 Тази програма разрешава **функционалността** на даден преобразувател, следвайки Следствие 4.6.7, Твърдение 4.6.8 и Следствие 4.6.11.

Пример 8.2.13 Следващата програма реализира пълната функционалност необходима за правописна корекция (spell checker). Подобни имплементации са използвани например в [Ringlstetter et al., 2007, Mitankin et al., 2014]. Програмата имплементира функция за тестване принадлежността на входна дума към даден речник, както и функция, връщаща множеството от всички речникови думи, които са "близки" до дадената. Това множество представлява кандидатите за корекция за дадената дума. Близостта е дефинирана според Левенщейн разстоянието. Припомняме, че Левенщейн разстоянието между две думи е минималният брой символни субституции, изтривания и вмъквания, които са необходими, за да се получи от първата дума втората. Представеният код може да се приложи върху всеки подходящ списък от речникови думи.

Забележка 8.2.14 Горната функция може да се приложи за големи речници и предоставя практически приложимо решение за правописна корекция и други изчисления, базирани на търсене на близост. Въпреки това някои от стъпките могат съществено да се оптимизират.

1. Минималният детерминиран краен автомат може да бъде построен по много по-ефективен начин [Daciuk et al., 2000].

2. Детерминираният Левенщейн автомат може да бъде построен директно, чрез метода от [Schulz and Mihov, 2002]. Усъвършенстван метод, представен в [Mihov and Schulz, 2004], построява *универсалния* автомат на Левенщейн, който не зависи от входната дума. Задълбочено изследване на универсалните крайни автомати и преобразуватели на Левенщейн е дадено в [Mitankin et al., 2011].

3. Думите от сечението на речниковия автомат с автомата на Левенщейн могат да се получат по-ефективно чрез тяхното паралелно обхождане, избягвайки експлицитното конструиране на автомата за сечението.

8.3 Имплементации на алгоритми за детерминирани крайни преобразуватели на $C(M)$

В секцията представяме алгоритмите за конструиране и минимизиране на детерминирани преобразуватели.

Програма 8.3.1 Следващата програма конструира **подпоследователен краен преобразувател** от даден краен преобразувател с ограничена вариация, следвайки индуктивната конструкция от Секция 5.2.

Програма 8.3.2 Програмата тества свойството **ограничена вариация** за преобразувател в съответствие с Теорема 5.3.4 и Лема 5.3.5.

Минимизация на подпоследователни преобразуватели

Програма 8.3.3 Програмата дефинира типовете за **подпоследователни преобразуватели с начален изход** и процедурата за конвертиране от обикновен подпоследователен преобразувател в съответствие с Дефиниция 5.4.4.

Програма 8.3.4 Този алгоритъм връща **експандирания изходен автомат** за даден подпоследователен преобразувател, както е дефинирано в Дефиниция 5.5.11.

Програма 8.3.5 Функцията изчислява **функцията за максимален изход на състояние** *mso* за даден преобразувател $\mathcal{T}$ според Следствие 5.5.15.

Програма 8.3.6 Следната програма конвертира подпоследователен преобразувател до **канонична форма**, както е дефинирано в Дефиниция 5.5.2.

Програма 8.3.7 Програмата представя процедури за **псевдо-минимизация** и **минимизация** за подпоследователни преобразуватели.

Програма 8.3.8 Функцията C_λ реализира обобщената функция на изходите λ^* за даден подпоследователен преобразувател при зададена функция на преходите δ и функция на изходите на преходите λ.

Програма 8.3.9 Следващата програма конструира подпоследователен преобразувател, представящ **композицията** на два подпоследователни преобразувателя, следвайки Твърдение 5.1.10.

Приложение за фонетизация на числа За да демонстрираме използването на горните конструкции, представяме програма за конструиране на минимален подпоследователен преобразувател, който преобразува число, записано като последователност от цифри в неговата фонетизация на английски език. Подобна функционалност се използва например за синтез на реч.

Пример 8.3.10 В нашата имплементация използваме фонетичната система от речника за произнасяне, предоставен от Университета Карнеги-Мелон[1]. В примера се ограничаваме до числа между 1 и 999999, но този интервал може лесно да бъде разширен.

Забележка 8.3.11 Минималният подпоследователен преобразувател, представящ крайна функция, може да се построи много по-ефективно чрез приилагане на алгоритъма представен в [Mihov and Maurel, 2001].

8.4 Имплементации на алгоритми за бимашини на $C(M)$

В тази секция представяме алгоритмите за конструиране, композиция и псевдо-минимизиране на бимашини.

Програма 8.4.1 Следващата програма конвертира **функционален краен преобразувател в бимашина**, следвайки конструкцията представена в Твърдение 6.2.5.

Програма 8.4.2 Програмата конструира **псевдо-минимална бимашина** в съответствие с Дефиниция 6.3.3 чрез минимизиране на левия и десния автомат на бимашината, разглеждани като детерминирани крайни автомати оцветени с профилите на състоянията.

Програма 8.4.3 Следният алгоритъм представя **композицията на две бимашини**, следвайки конструкцията от Секция 6.4.

[1] `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`

Приложение за аритметика с неограничени числа с бимашини Завършваме секцията с пример за използване на бимашини за реализиране на някои основни аритметични операции върху неограничени естествени числа.

Пример 8.4.4 Всяко число (например 5389) може да се представи като низ от символи за цифри като например `['5','3','8','9']`. Програмата взима за вход произволно естествено число $K \in \mathbb{N}$. За дадено K конструира "адитивна" бимашина, която чете на входа естествено число x представено като низ от цифри. Изходът на адитивната бимашина е представянето на числото $x + K$ като низ от цифри. По подобен начин за дадено естествено число K се конструират бимашини за аритметичните операции $x \mapsto x - K$ (изваждане, дефинирано за $x \geq K$), $x \mapsto x \cdot K$ (умножение), $x \mapsto x/K$ (целочислено деление) и остатък при деление на K. Във всеки един от случаите входното и изходното число са представени като низ от цифри.

Авторска справка

Основните научни приноси на настоящия дисертационен труд са:

1. Представено е пълно и съгласувано изложение на теорията на крайните автомати, преобразуватели и бимашини заедно с подробни доказателства на основните свойства и коректност на конструкциите, което комбинира абстрактната алгебрична гледна точка с ефективни от изчислителна гледна точка конструкции.

2. Разработен е метод за тестване за ограничена вариация на краен преобразувател, който се интегрира в конструкцията за секвенциализация. В предишните подходи (виж например [Roche and Schabes, 1997a]), за да се избегне безкрайната работа на алгоритъма за секвенциализация, предварително се тества ограничената вариация със специален алгоритъм. С представения метод този проблем се решава елегантно с добавянето на една допълнителна проверка в рамките на конструкцията (виж Теорема 5.3.8).

3. Представена е нова конструкция с полиномиална сложност за канонизация на подпоследователен преобразувател (виж Следствие 5.5.15 и Забележка 5.5.16). Предимството на новата конструкция е добрата ефективност и използването на изцяло автоматен подход.

4. Разработена е нова конструкция за построяване на бимашина от краен преобразувател (виж Твърдение 6.2.5). Предимството на новата конструкция е избягването на предварителната конструкция за получаване на еднозначен краен преобразувател. За определени класове от преобразуватели, вариант на новата конструкция води до експоненциално по-малък брой на състоянията на резултатната бимашина [Gerdjikov et al., 2017].

5. Получена е конструкция заедно с доказателство за коректност за директно композиране на бимашини (Секция 6.4). За разлика от стандартния подход, при който е необходимо конвертиране на бимашините към посимволни крайни преобразуватели и обратно, в новата конструкция резултатната бимашина се конструира директно.

Основните научно-приложни приноси на настоящия дисертационен труд са:

1. Разработен е новият език за програмиране $C(M)$, който позволява при описанието на алгоритмите да се съсредоточим върху математическите стъпки на абстрактно ниво, вместо да описваме детайлите за изпълнение на ниско ниво.

2. Представени са работещи имплементации на $C(M)$ на всички основни конструкции за крайни автомати, преобразуватели и бимашини.

3. Създадени са реализации на реално работещи програми, базирани на крайни автомати, преобразуватели и бимашини, за редица практически интересни задачи като правописна корекция, фонетизация, аритметика с неограничени числа и други.

Литература

[Allauzen et al., 2007] Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., and Mohri, M. (2007). Openfst: A general and efficient weighted finite-state transducer library. In Holub, J. and Žďárek, J., editors, *Implementation and Application of Automata*, pages 11–23, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Angelova and Mihov, 2008] Angelova, G. and Mihov, S. (2008). Finite state automata and simple conceptual graphs with binary conceptual relations.

In *Supplementary Proceedings of the 16th International Conference on Conceptual Structures, ICCS 2008, Toulouse, France, July 7-11, 2008*, pages 139–148.

[Beesley and Karttunen, 2003] Beesley, K. and Karttunen, L. (2003). *Finite State Morphology*. CSLI studies in computational linguistics: Center for the Study of Language and Information. CSLI Publications.

[Berstel, 1979] Berstel, J. (1979). *Transductions and context-free languages*. Leitfäden der angewandten Mathematik und Mechanik. Teubner.

[Daciuk et al., 2000] Daciuk, J., Mihov, S., Watson, B., and Watson, R. (2000). Incremental construction of minimal acyclic finite state automata. *Computational Linguistics*, 26(1):3–16.

[Eilenberg, 1974] Eilenberg, S. (1974). *Automata, Languages, and Machines - Volume A*, volume volume 59 of Pure and Applied Mathematics. Academic Press.

[Eilenberg, 1976] Eilenberg, S. (1976). *Automata, Languages, and Machines - Volume B*. Academic Press.

[Ganchev et al., 2008] Ganchev, H., Mihov, S., and Schulz, K. U. (2008). One-letter automata: How to reduce k tapes to one. In Hamm, F and Kepser, S, editor, *LOGICS FOR LINGUISTIC STRUCTURES*, volume 201 of *Trends in Linguistics-Studies and Monographs*, pages 35–55. WALTER DE GRUYTER GMBH. Conference in Honor of Uwe Monnich on his 70th Birthday, Freudenstadt, GERMANY, NOV, 2004.

[Gerdemann and van Noord, 1999] Gerdemann, D. and van Noord, G. (1999). Transducers from rewrite rules with backreferences. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL 99)*, pages 126–133.

[Gerdjikov and Mihov, 2017a] Gerdjikov, S. and Mihov, S. (2017a). Myhill-nerode relation for sequentiable structures. *CoRR*, abs/1706.02910.

[Gerdjikov and Mihov, 2017b] Gerdjikov, S. and Mihov, S. (2017b). Over which monoids is the transducer determinization procedure applicable? In *Language and Automata Theory and Applications - 11th International Conference, LATA 2017, Umeå, Sweden, March 6-9, 2017, Proceedings*, volume 10168 LNCS, pages 380–392.

[Gerdjikov et al., 2017] Gerdjikov, S., Mihov, S., and Schulz, K. U. (2017). A simple method for building bimachines from functional finite-state transducers. In Carayol, A. and Nicaud, C., editors, *Implementation and Application of Automata*, volume 10329 LNCS, pages 113–125. Springer International Publishing.

[Hopcroft et al., 2006] Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2006). *Introduction to Automata Theory, Languages, and Computation.* Pearson. 3nd edition.

[Hulden, 2009] Hulden, M. (2009). *Finite-State Machine Construction Methods and Algorithms for Phonology and Morphology.* PhD thesis, University of Arizona.

[Hutton, 2007] Hutton, G. (2007). *Programming in Haskell.* Cambridge University Press.

[Kaplan and Kay, 1994] Kaplan, R. and Kay, M. (1994). Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–279.

[Karttunen, 1997] Karttunen, L. (1997). The replace operator. In Roche, E. and Schabes, Y., editors, *Finite-State Language Processing*, pages 117–147. MIT Press.

[Karttunen et al., 1997a] Karttunen, L., Chanod, J.-P., Grefenstette, G., and Schiller, A. (1997a). Regular expressions for language engineering. *Journal of Natural Language Engineering*, 2(4):307–330.

[Karttunen et al., 1997b] Karttunen, L., Gaál, T., and Kempe, A. (1997b). Xerox finite-state tool. Technical report, Xerox Corporation.

[Kozen, 1997] Kozen, D. C. (1997). *Automata and Computability.* Springer, New York, Berlin.

[Lewis and Papadimitriou, 1998] Lewis, H. R. and Papadimitriou, C. H. (1998). *Elements of the Theory of Computation.* Prentice-Hall, Upper Saddle River, New Jersey. 2nd edition.

[Maurel and Guenthner, 2005] Maurel, D. and Guenthner, F. (2005). *Automata and Dictionaries.* Texts in Computer Science. College Publications.

[Mihov and Maurel, 2001] Mihov, S. and Maurel, D. (2001). Direct construction of minimal acyclic subsequential transducers. In *Proceedings*

of the Conference on Implementation and Application of Automata CIAA'2000, volume 2088 of *LNCS*, pages 217–229. Springer.

[Mihov and Schulz, 2019] Mihov, S. and Schulz, K. (2019). *Finite-State Techniques: Automata, Transducers and Bimachines*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

[Mihov and Schulz, 2004] Mihov, S. and Schulz, K. U. (2004). Fast approximate search in large dictionaries. *Computational Linguistics*, 30(4):451–477.

[Mitankin et al., 2014] Mitankin, P., Gerdjikov, S., and Mihov, S. (2014). An approach to unsupervised historical text normalisation. In *Digital Access to Textual Cultural Heritage 2014, DATeCH 2014, Madrid, Spain, May 19-20, 2014*, pages 29–34.

[Mitankin et al., 2011] Mitankin, P., Mihov, S., and Schulz, K. U. (2011). Deciding word neighborhood with universal neighborhood automata. *Theoretical Computer Science*, 412(22):2340–2355.

[Mohri, 1996] Mohri, M. (1996). On some applications of finite-state automata theory to natural language processing. *Journal of Natural Language Engineering*, 2:1–20.

[Mohri, 1997] Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

[Mohri, 2000] Mohri, M. (2000). Minimization algorithms for sequential transducers. *Theoretical Computer Science*, 234:177–201.

[Mohri et al., 2008] Mohri, M., Pereira, F., and Riley, M. (2008). Speech recognition with weighted finite-state transducers. In *Springer Handbook of Speech Processing*, pages 559–584. Springer.

[Mohri and Sproat, 1996] Mohri, M. and Sproat, R. (1996). An efficient compiler for weighted rewrite rules. In *Proceedings of the 34th Meeting of the Association for Computational Linguistics (ACL'96)*, pages 231–238, Santa Cruz, CA.

[Navarro and Raffinot, 2002] Navarro, G. and Raffinot, M. (2002). *Flexible Pattern Matching in Strings*. Cambridge University Press, Cambridge, UK.

[Reutenauer and Schützenberger, 1991] Reutenauer, C. and Schützenberger, M. P. (1991). Minimization of rational word functions. *SIAM J. Computing*, 20(4):669–685.

[Ringlstetter et al., 2007] Ringlstetter, C., Schulz, K. U., and Mihov, S. (2007). Adaptive text correction with web-crawled domain-dependent dictionaries. *ACM Transactions on Speech and Language Processing*, 4(4).

[Roche and Schabes, 1997a] Roche, E. and Schabes, Y. (1997a). Deterministic part-of-speech tagging with finite-state transducers. In Roche, E. and Schabes, Y., editors, *Finite-State Language Processing*, Language, Speech, and Communication, pages 205–240. The MIT Press.

[Roche and Schabes, 1997b] Roche, E. and Schabes, Y. (1997b). Introduction. In Roche, E. and Schabes, Y., editors, *Finite-State Language Processing*, pages 1–66. MIT Press.

[Sakarovitch, 2009] Sakarovitch, J. (2009). *Elements of Automata Theory*. Cambridge University Press, New York, NY, USA.

[Schmid, 2006] Schmid, H. (2006). A programming language for finite state transducers. In Yli-Jyrä, A., Karttunen, L., and Karhumäki, J., editors, *Finite-State Methods and Natural Language Processing*, pages 308–309, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Schulz and Mihov, 2002] Schulz, K. U. and Mihov, S. (2002). Fast String Correction with Levenshtein-Automata. *International Journal of Document Analysis and Recognition*, 5(1):67–85.

[Schützenberger, 1961] Schützenberger, M.-P. (1961). A remark on finite transducers. *Information and Control*, 4:185–196.

[Schwartz et al., 1986] Schwartz, J. T., Dewar, R. B., Schonberg, E., and Dubinsky, E. (1986). *Programming with Sets; an Introduction to SETL*. Springer-Verlag, Berlin, Heidelberg.