Съдържание

1 Правила за заместване

Дефиниция 1.1. *Двоична стрингова релация* $R \subseteq \Sigma_1^* \times \Sigma_2^*$ наричаме множество от двойки $\langle u, v\rangle$, където $u \in \Sigma_1^*$ и $v \in \Sigma_2^*$.

За двоичните стрингови релации можем да си мислим като множество от преводи, като на пример за двойката $\langle u, v\rangle \in R$ казваме, че думата u се превежда като v.

Дефиниция 1.2. Нека $R_1, R_2, \in \Sigma^* \times \Sigma^*$ са *двоични стрингови релации. Конкатенацията* на R_1 и R_2 дефинираме както следва

$$R_1 \cdot R_2 = \{\langle u_1 \cdot v_1, u_2 \cdot v_2\rangle | \langle u_1, u_2\rangle \in R_1, \langle v_1, v_2\rangle \in R_2\}$$

Дефиниция 1.3. *Двоична стрингова релация* $R \subseteq L_1 \times L_2$ е *регулярна*, ако L_1 и L_2 са регулярни езици.

Дефиниция 1.4. *Правило за заместване* представяме във вида

$$E \to \beta$$

където E е регулярен израз над крайна азбука Σ, а $\beta \in \Sigma^*$ е дума. Приложение на правилото върху дума $\alpha \in \Sigma^*$ представлява заместването на поднизовете на α, които са в езика $L(E)$ с думата β.

Правила за заместване се представят като *регулярни стрингови релации*, използвайки единствено алгебрата на регулярните множества и релации. Те могат да се реализират програмно чрез крайни преобразуватели. [1]

1.1 Разрешаване на конфликти при заместване

1.2 Лексически анализ чрез регулярни релации

Релация за лексически анализ за даден език дефинираме като комбинация на *правила за заместване*. Тези правила биват два вида - такива, които *маркират* дадена лексема, или *нормализиращи правила*, които служат за предварителна обработка входния текст, като на пример за премахване на излишни символи.

Пример 1.1. *Лексическа граматика на аритметичен израз от естествени числа.*

1. `WHITESPACE+` $\rightarrow \epsilon$
2. `[0-9]+` $\rightarrow \ldots$ `TB`
3. `+|-|*|\` $\rightarrow \ldots$ `TB`

Правила 2 и 3 маркират лексемите съответно за естествено число и аритметичен оператор като поставят символ за граница на лексема `TB`. Правило 1 е нормализиращо правило, което премахва интервалите от входната дума, тъй като те не носят значение за аритметичния израз.

Всяко от тези правила само по себе си представлява регулярна релация, като релацията за лексически анализ в конкретния случай получаваме като резултат от композицията на нормализиращото правило с обединението на лексическите правила. С други думи, ако всяко правило i представим като регулярна релация R_i, токенизиращата релация под стратегията най-ляво-най-дълго срещане можем да представим както следва

$$R = R^{LML}(R_1) \circ R^{LML}(R_2 \cup R_3)$$

Стратегията най-ляво-най-дълго срещане приложена над функционални релации, каквито са правилата в лексическата граматика, гарантира, че получената релация също е функционална. Това означава, че процесът на токенизация винаги ще има най-много един резултат.

$$R(\texttt{2+14}) = \texttt{2 TB + TB 14 TB}$$

$$R(\texttt{42 - 15 * 5}) = \texttt{42 TB - TB 15 TB * TB 5 TB}$$

2 Крайни Автомати

Дефиниция 2.1. *Краен автомат* дефинираме като петорка $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$, където

- Σ е *крайна азбука от символи*

- Q е *крайно множество от състояния*

- $I \subseteq Q$ е *множество от начални състояния*

- $F \subseteq Q$ е *множество от финални състояния*

- $\Delta \subseteq Q \times \Sigma \times Q$ е *релация на прехода*

Тройки от вида $\langle q_1, m, q_2 \rangle \in \Delta$ наричаме *преходи* и казваме, че започва състояние q_1, има етикет m и завършва в състояние q_2. Алтернативно, тези преходи обозначаваме като $q_1 \rightarrow^m q_2$.

Дефиниция 2.2. Нека $\mathcal{A}$ е краен автомат. *Разширена релация на прехода* $\Delta^* \subseteq Q \times \Sigma^* \times Q$ дефинираме индуктивно:

- $\langle q, \epsilon, q \rangle \in \Delta^*$ за всяко $q \in Q$

- $\langle q_1, wa, q_2 \rangle \in \Delta^*$ за всяко $q_1, q_2, q \in Q$, $a \in \Sigma, w \in \Sigma^*$, ако $\langle q_1, w, q \rangle \in \Delta^*$ и $\langle q, a, q_2 \rangle \in \Delta$

Дефиниция 2.3. Нека $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ е краен автомат. *Път* в $\mathcal{A}$ наричаме крайна редица от преходи с дължина $k > 0$

$$\pi = q_0 \rightarrow^{a_1} q_1 \rightarrow^{a_2} \ldots \rightarrow^{a_k} q_k$$

където $\langle q_{i-1}, a_i, q_i \rangle \in \Delta$ за $i = 1 \ldots k$. Казваме, че *пътят* започва от състояние q_0 и завършва в състояние q_k. Елементите $q_0, q_1, \ldots, q_k$ наричаме *състояния на пътя*, а думата $w = a_1 a_2 \ldots a_k$ наричаме *етикет на пътя.*
Успешен път в автомата е *път*, който започва от начално състояние и завършва във финално състояние.

Дефиниция 2.4. Нека $\mathcal{A}$ е краен автомат. Множеството от етикети на всички успешни пътища в $\mathcal{A}$ наричаме *език на* $\mathcal{A}$ и обозначаваме като $L(\mathcal{A})$.

$$L(\mathcal{A}) = \{w \in \Sigma^* \mid \exists i \in I, f \in F : \langle i, w, f \rangle \in \Delta^*\}$$

Дефиниция 2.5. Нека $\mathcal{A}_1$ и $\mathcal{A}_2$ са крайни автомати. Казваме, че $\mathcal{A}_1$ е еквивалентен на $\mathcal{A}_2$ ($\mathcal{A}_1 \equiv \mathcal{A}_2$), ако езиците им съвпадат ($L(\mathcal{A}_1) = L(\mathcal{A}_2)$)

3 Детерминирани Крайни Автомати

Дефиниция 3.1. Краен автомат $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$ е *детерминиран*, ако:

- $\mathcal{A}$ има единствено начално състояние $I = \{q_0\}$.
- За всяко $q_1 \in Q$ и символ $a \in \Sigma$, съществува не повече от едно $q_2 \in Q$, такова че $\langle q_1, a, q_2 \rangle \in \Delta$.

Иначе казано, релацията на прехода може да се представи като частична функция $\delta : Q \times \Sigma \to Q$ и *детерминираните автомати* можем преставим в следния вид

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

Предимството на *детерминираните автомати* се изразява в това, че могат да разпознават дали дума w принадлежи на езика на автомата $L(\mathcal{A}_D)$ за линейно време спрямо дължината ѝ - $O(|w|)$.

Дефиниция 3.2. Нека $\mathcal{A}_D = \langle \Sigma, Q, q_0, F, \delta \rangle$ е *детерминиран краен автомат. Разширена функция на прехода* $\delta^* : Q \times \Sigma^* \to Q$ дефинираме индуктивно:

- $\delta^*(q, \epsilon) = q$
- $\delta^*(q, aw) = \delta^*(\delta(q, a), w)$, където $a \in \Sigma, w \in \Sigma^*$

Теорема 3.1. За всеки краен автомат $\mathcal{A} = \langle \Sigma, Q, I, F, \Delta \rangle$, съществува еквивалентен на него, детерминиран краен автомат $\mathcal{A}_D$.

Доказателство. □

4 Бимашини

Дефиниция 4.1. *Бимашина* дефинираме като тройка $\mathcal{B} = \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$, където

- $\mathcal{A}_L = \langle \Sigma, Q_L, s_L, Q_L, \delta_L \rangle$ и $\mathcal{A}_R = \langle \Sigma, Q_R, s_R, Q_R, \delta_R \rangle$ са *детерминирани крайни автомати* и ги наричаме съответно *ляв и десен автомат* на бимашината. Всички състояния на тези автомати са финални.
- $\psi : (Q_L \times \Sigma \times Q_R) \to \Sigma^*$ е частична функция, която наричаме *изходна функция*.

Дефиниция 4.2. Нека $\mathcal{B} = \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ е класическа бимашина, Σ е азбуката на на автоматите $\mathcal{A}_L$ и $\mathcal{A}_R$ и $w = a_1 a_2 \dots a_k \in \Sigma^*$ $(k \geq 0)$, дума и $a_i \in \Sigma$ $(1 \leq i \leq k)$ са букви. Ако $\delta_L^*(a_1 a_2 \dots a_k)$ и $\delta_R^*(a_k a_{k-1} \dots a_1)$ са дефинирани, то можем да получим двата пътя:

$$\pi_L = l_0 \to^{a_1} l_1 \to^{a_2} \dots l_{k-1} \to^{a_k} l_k$$

$$\pi_R = r_0 \leftarrow^{a_1} r_1 \leftarrow^{a_2} \ldots r_{k-1} \leftarrow^{a_k} r_k$$

Където π_L и π_R са пътища в съответно левия и десния автомат и думата w се разпознава от $\mathcal{A}_L$ в посока от ляво на дясно, а от $\mathcal{A}_R$, съответно от дясно на ляво. Ако за всички тройки $\langle l_{i-1}, a_i, r_i \rangle$, изходната функция $\psi(l_{i-1}, a_i, r_i)$ е дефинирана, то двойката пътища $\langle \pi_L, \pi_R \rangle$ наричаме *успешно изпълнение* на $\mathcal{B}$ с етикет $w = a_1 a_2 \ldots a_k$ и *изход*

$$\mathcal{O}_{\mathcal{B}}(w) = \psi(l_0, a_1, r_1) \cdot \psi(l_1, a_2, r_2) \cdot \ldots \cdot \psi(l_{k-1}, a_k, r_k)$$

$\mathcal{O}_{\mathcal{B}}$ наричаме *изходна функция на бимашината* и казваме, че бимашината *превежда* w в m, ако $\mathcal{O}_{\mathcal{B}}(w) = m$, където m е резултат от конкатенация на всички $\psi(l_{i-1}, a_i, r_i)$ $1 \leq i \leq k$.

Бимашината чете входната дума и за всеки символ извежда дума над азбуката си. На всяка стъпка изведената от изходната функция ψ дума зависи от входния символ и двете състояния в които биха преминали левият и десният автомат, четейки входа съответно от ляво на дястно и от дясно на ляво. Крайният резултат е конкатенацията на всички така изведени думи.

Пример 4.1. *(Бимашина и изпълнение) ...*

Дефиниция 4.3. Нека $\mathcal{B} = \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ е бимашина. *Разширената изходна функция* ψ^* дефинираме индуктивно:

- $\psi^*(l, \epsilon, r) = \epsilon$ за всяко $l \in Q_L, r \in Q_R$
- $\psi^*(l, wa, r) = \psi^*(l, w, \delta_R(r, a)) \cdot \psi(\delta_L^*(l, w), a, r)$, за $l \in Q_L, r \in Q_R, w \in \Sigma^*, a \in \Sigma$

Дефиниция 4.4. Бимашина $\mathcal{B} = \langle \mathcal{A}_L, \mathcal{A}_R, \psi \rangle$ наричаме *тотална*, ако функциите на прехода $\delta_L : Q_L \times \Sigma \to Q_L$ и $\delta_R : Q_R \times \Sigma \to Q_R$ на левия и десния автомат съответно, както и функцията на изхода $\psi : (Q_L \times \Sigma \times Q_R) \to \Sigma^*$ са *тотални* .

Литература

[1] Kaplan, Ronald and Kay, Martin. 1994. *Regular models of phonological rule systems.* Computational Linguistics 20(3):331-378

[2] Karttunen, Lauri. 1996. *Directed Replacement.*