BI-AAG cvičení 1 - Seznam pojmů

Bc. Eliška Šestáková

25.9.2014

Abeceda (značíme Σ) – neprázdná (konečná) množina prvků, které nazýváme symboly abecedy.

Příklady abeced:

- anglická abeceda (26 základních symbolů (písmen) bez diakritiky)
- binární abeceda reprezentovaná množinou {0,1} resp. {true, false}
- abecedy programovacích jazyků

Řetězec nad abecedou (také slovo nebo věta) – každá konečná posloupnost symbolů abecedy

- ϵ – prázdný řetězec, prázdná posloupnost symbolů
- Σ^* – množina všech řetězců (včetně prázdného) nad abecedou Σ
- Σ^+ – množina všech neprázdných řetězců nad abecedou Σ
- $\Sigma^* = \Sigma^+ \cup \{\epsilon\}$

Příklady řetězců:

Je-li Σ = {a,b,+} abeceda, pak: ϵ, „a“, „b“, „+“, „aa“, „a+b“, „+ab“ jsou některé řetězce nad abecedou Σ.

Zřetězení (také konkatenace) – Nechť x a y jsou řetězce nad abecedou. Zřetězením řetězce x s řetězcem y vznikne řetězec xy připojením řetězce y za řetězec x.

- asociativní $x(yz) = (xy)z$
- není komutativní $xy \neq yx$
- $x\epsilon = \epsilon x = x$

Příklady zřetězení:

Je-li x = ab, y = ba, pak xy = abba, yx = baab, xϵ = ϵx = ab.

Délka řetězce (značíme symbolicky $|x|$) – nezáporné celé číslo udávající počet symbolů řetězce. Délka prázdného řetězce je nulová, tj. $|\epsilon| = 0$.

Řetězec, který sestává právě z k výskytů symbolu a budeme symbolicky značit a^k. Např. $a^3 = aaa$, $b^2 = bb$, $a^0 = \epsilon$.

Formální jazyk (značíme L) – libovolná podmnožina všech možných řetězců nad danou abecedou ($L \subseteq \Sigma^*$)

Příklady jazyků:

Mějme binární abecedu $\Sigma = \{0, 1\}$. Množinou Σ^* jsou všechny možné binární řetězce (včetně prázdného), tedy $\Sigma^* = \{\epsilon, 0, 1, 00, 01, \cdots 0000, 0001, \cdots\}$. Poté lze zavést jazyky:

- $L_1 = \{\epsilon\}$ (Pozor, tento jazyk není prázdný, neboť obsahuje jeden prvek a tím je prázdný řetěz!)
- $L_2 = \{1\}$
- $L_3 = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \cdots\}$
- $L_4 = \{w : w \in \{0, 1\}^*, w$ je binární zápis sudého čísla$\}$ (tj. množina všech řetězců (vět) w nad abecedou $\{0, 1\}$ taková, že w je binární zápis sudého čísla.

Operace nad jazyky – lze rozdělit do dvou skupin:

1. Množinové operace – plynoucí ze skutečnosti, že jazyk je množina
   - **Sjednocení:**
     $$L_1 \cup L_2 = \{x; x \in L_1 \vee x \in L_2\}$$
   - **Průnik:**
     $$L_1 \cap L_2 = \{x; x \in L_1 \wedge x \in L_2\}$$
   - **Rozdíl:**
     $$L_1 \setminus L_2 = \{x; x \in L_1 \wedge x \notin L_2\}$$
   - **Komplement (doplněk) v universu** Σ^*:
     $$\bar{L_1} = \{x \in \Sigma^*; x \notin L_1\}$$
2. Řetězcové operace – respektuje specifikum množin tvořících jazyky, tj. skutečnost, že jejich prvky jsou řetězce
   - **Součin:** Nechť L_1 je jazyk nad abecedou Σ_1, L_2 jazyk nad abecedou Σ_2. Součinem (konkatenací) jazyku L_1 a L_2 je jazyk $L_1 \cdot L_2$ nad abecedou $\Sigma_1 \cup \Sigma_2$, jenž je definován takto:
     $$L_1 \cdot L_2 = \{xy \mid x \in L_1, y \in L_2\}$$
     Operace součin jazyku je definována prostřednictvím konkatenace řetězců a má stejné vlastnosti – je asociativní a nekomutativní.
   - **Iterace:** Nechť L je jazyk nad abecedou Σ. Iteraci L^* jazyka L a pozitivní iteraci L^+ jazyka L definujeme takto:
     (a) $L^0 = \epsilon$
     (b) $L^n = L \cdot L^{n-1}$ pro $n \geq 1$
     (c) $L^* = \bigcup_{n \geq 0} L^n$
     (d) $L^+ = \bigcup_{n \geq 1} L^n$

     Příklad iterace jazyků:
     Je-li $L_1 = \{(p\}, L_2 = \{, p\}, L_3 = \{)\}$ potom jazyk $L_1 \cdot L_2^* \cdot L_3$ obsahuje řetězce: (p) (p, p) (p, p, p) $(p, p, p, p) \cdots$

Gramatika – prostředek pro reprezentaci jazyků (konečných i nekonečných). Definována jako (uspořádaná) čtveřice:

$$G = (N, T, P, S)$$

- **N** (neterminální symboly, neterminály), konečná množina, role pomocných proměnných označujících určité syntaktické celky
- **T** (terminální symboly, terminály), konečná množina, identická s abecedou, nad níž je definován jazyk
- **P** (přepisovací pravidla) ve tvaru:

$$(N \cup T)^* \cdot N \cdot (N \cup T)^* \to (N \cup T)^*$$

- **S** $\in$ N (počáteční, výchozí, startovací symbol gramatiky)

Přímá derivace – Nechť $G = (N, T, P, S)$ a $x, y \in (N \cup T)^*$. Říkáme, že x přímo derivuje y:

$$x \Rightarrow y$$

jestliže existuje pravidlo $\alpha \to \beta$ a $\gamma, \delta \in (N \cup T)^*$ takové, že

$$x = \gamma\alpha\delta$$
$$y = \gamma\beta\delta$$

Derivace – Nechť $G = (N, T, P, S)$ je gramatika a $x, y \in (N \cup T)^*$. Mezi řetězci x a y platí relace $\Rightarrow^+$ nazývaná derivace, jestliže existuje posloupnost přímých derivací

$$\lambda_{i-1} \Rightarrow \lambda_i$$

kde $i = 1, \cdots n$ a $n \geq 1$ taková že platí:

$$x = \lambda_0 \Rightarrow \lambda_1 \Rightarrow \lambda_2 \cdots \Rightarrow \lambda_{n-1} \Rightarrow \lambda_n = y$$

Tuto posloupnost nazýváme derivací délky n. Platí-li $x \Rightarrow^+ y$, pak říkáme, že řetězec y lze generovat z řetězce x v gramatice G (v $n \geq 1$ krocích) (tranzitivní uzávěr). Pro $x \Rightarrow^* y$ říkáme, že řetězec y lze generovat z řetězce x v gramatice G v $n \geq 0$ krocích (tranzitivní a reflexivní uzávěr).

Větná forma – Nechť $G = (N, T, P, S)$. Řetězec $\alpha \in (N \cup T)^*$ nazveme větnou formou v gramatice G, jestliže platí

$$S \Rightarrow^* \alpha$$

tj. řetězec α je generovatelný z výchozího symbolu S a může obsahovat terminální i neterminální symboly.

Věta – větná forma, která obsahuje pouze terminální symboly

Jazyk L generovaný gramatikou G tj. L(G) – množina všech vět generovaných gramatikou G (tj. všechny řetězce nad vstupní abecedou takové, že existuje derivace ze startovního symbolu S do těchto řetězců):

$$L(G) = \{w : w \in T^* \wedge S \Rightarrow^* w\}$$

Ekvivalentní gramatiky – gramatiky jsou ekvivalentní, když generují stejný jazyk.

Chomského klasifikace gramatik – vymezuje 4 typy gramatik, podle tvaru přepisovacích pravidel

1. **(typ 0) Neomezená gramatika** – pravidla v nejobecnějším tvaru, shodným s definicí gramatiky:

   $$(N \cup T)^* \cdot N \cdot (N \cup T)^* \rightarrow (N \cup T)^*$$

   Neomezené gramatiky generují rekurzivně spočetné jazyky, jenž jsou rozpoznatelné Turingovým strojem.

2. **(typ 1) Kontextová gramatika** – pravidla ve tvaru:

   $$\gamma \cdot N \cdot \delta \rightarrow \gamma \cdot (N \cup T)^+ \cdot \delta$$

   kde $\gamma, \delta \in (N \cup T)^*$. Případně $S \rightarrow \epsilon$, pokud se S neobjevuje na pravé straně žádného pravidla.

   Kontextové gramatiky nepřipouštějí, aby byl neterminál nahrazen prázdným řetězcem (Jedinou přípustnou výjimkou je pravidlo $S \rightarrow \epsilon$ a to jen za předpokladu, že se S neobjevuje na pravé straně žádného pravidla. Případné použití pravidla $S \rightarrow \epsilon$ umožňuje popsat příslušnost prázdného řetězce k jazyku, který je danou gramatikou generován).

   V důsledku této vlastnosti nemůže při generování vety dojít ke zkrácení generovaných řetězců.

   Kontextové gramatiky generují kontextové jazyky, jenž jsou rozpoznatelné lineárně omezených Turingovým strojem.

3. **(typ 2) Bezkontextová gramatika** – pravidla ve tvaru:

   $$N \rightarrow (N \cup T)^*$$

   Bezkontextové gramatiky generují bezkontextové jazyky, jenž jsou rozpoznatelné zásobníkovým automatem.

4. **(typ 3) Regulární gramatika** – pravidla ve tvaru:

   $$N \rightarrow T \,|\, TN$$

   Případně $S \rightarrow \epsilon$, pokud se S neobjevuje na pravé straně žádného pravidla.

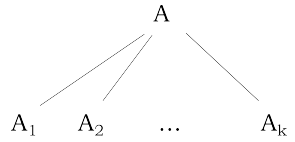
   Regulární gramatiky generují regulární jazyky, jenž jsou rozpoznatelné konečným automatem.

Konečný jazyk – Jazyk L je konečný právě tehdy, když L je konečná množina (obsahuje konečný počet vět). Platí, že každý konečný jazyk je regulární (ne naopak).

Derivační strom – grafické vyjádření syntaktické struktury věty. Orientovaný strom.

Jestliže máme dánu gramatiku $G = (N, T, P, S)$, pak derivační strom v této gramatice musí mít tyto vlastnosti:

- Uzly derivačního stromu jsou ohodnoceny terminálními a neterminálními symboly.
- Kořen stromu je ohodnocen počátečním (neterminálním) symbolem.
- Jestliže má uzel alespoň jednoho následovníka, je ohodnocen neterminálním symbolem.
- Pokud uzel ohodnocený symbolem A má bezprostřední následovníky zleva doprava ohodnoceny symboly $A_1, A_2 \cdots A_k$ (viz obrázek) poté $A \to A_1 A_2 \cdots A_k$ je pravidlo v P.



- Koncové uzly derivačního stromu tvoří zleva doprava větnou formu nebo větu v gramatice G, která je výsledkem derivačního stromu.