

Automaty a gramatiky (BI-AAG)

1. Základní pojmy

Jan Holub

Katedra teoretické informatiky
Fakulta informačních technologií
ČVUT v Praze



© Jan Holub, 2022

Motivace

Motivace

- Teorie jazyků a automatů je jedním ze stavebních kamenů Computer Science.
- Teorie nabízí efektivní řešení pro celou řadu základních problémů.
- Správné použití této teorie šetří čas a prostředky.
- Aplikace: konstrukce překladačů, model checking, verifikace komunikačních protokolů, vyhledávání, komprese dat, . . .

Cíle předmětu

Cíle předmětu

- Seznámit se s teorií jazyků a automatů.
- Umět ji efektivně používat.
- Rozeznat třídy jazyků.

Hodnocení předmětu

Cvičení

- domácí úkoly: jeden za 10 bodů, ostatní po 0 bodech (Studentova zodpovědnost.)
- 2 testy (7. týden: Marast online za 5 bodů; 12. týden: kontaktně za 25 bodů)
- až 3 bodů za aktivitu na cvičení
- celkem 40 bodů
- zápočet: minimálně 20 bodů.

Hodnocení předmětu

Zkouška

- „rozstřel“: max. 20 bodů
- zkouškový test: max. 40 bodů
- požadavky: rozstřel min. 12 bodů, test min. 18 bodů)
(úspěšně absolvovaný rozstřel se počítá na dalším termínu zkoušky)

Obsah předmětu

1. Základní pojmy, Chomského hierarchie, uzavřenosť bezkontextových jazyků.
2. Deterministické a nedeterministické konečné automaty (DKA a NKA), NKA s epsilon přechody.
3. Operace s automaty (odstranění epsilon přechodů, determinizace, minimalizace, průnik, sjednocení).
4. Reg. výrazy, převody mezi RV, KA a RG, Kleeneova věta.
5. Operace s regulárními gramatikami, převody na KA.
6. Vlastnosti reg. jazyků (pumping lemma, Nerodova věta).
7. Jazyky bezkontextové, zásobníkový automat.
8. Analýza bezkontextových jazyků (nedeterm. versus determ.).
9. Rozšíření o překlad, Mealey, Moore, převody.
10. Jazyky kontextové a neomezené, Turingův stroj.
11. Programová a obvodová realizace DKA a NKA.
12. KA jako lexikální analyzátor, lex/flex generátor.

Doporučená literatura

- Aho A. V., Motwani R., Ullman, J. D.: Introduction to Automata Theory, Languages, and Computation. (2nd Edition). Addison Wesley, 2001. ISBN 0-201-44124-1. (Aho A. V. and Ullman, J. D. won 2020 ACM A.M. Turing Award.)
- Kozen, D. C.: Automata and Computability. Springer, 1997. ISBN 0387949070.
- Melichar B.: Jazyky a překlady. Praha, Vydavatelství ČVUT 2007.
- Šestáková E.: Automaty a gramatiky. Sbírka řešených příkladů. Praha, Pražská technika – Nakladatelství ČVUT, 2017.

Upozornění

- Slajdy jsou pouze pomocným materiélem k přednáškám a nemohou být jediným zdrojem pro přípravu k testům a ke zkoušce.

BI-AAG v angličtině

Vyzkoušejte si studium v angličtině:

- přednášky a cvičení v angličtině,
- testy a zkouška v češtině,
- menší skupina studentů (individuálnější přístup).
- Berte to jako přípravu na studium v zahraničí.
- Zájemci kontaktujte prof. Holuba.

Základní pojmy

Abeceda – konečná množina symbolů (značíme Σ nebo T)

- binární $\{0,1\}$, ternární $\{\text{Yes, No, Maybe}\}$,

→ DNA $\{\text{A, C, G, T}\}$, klíčová slova $\{\text{while, do, begin, end, to, for, false, true, ...}\}$

- možnost → nejménší, (Několik množin specifikovaných kterou abecedu nejsou všechny)

Řetězec nad abecedou – konečná posloupnost symbolů abecedy. Např. „0110101“, „ACCCGT“, „while true do“

Prázdná posloupnost = prázdný řetězec = ε .

Σ^* – množina všech řetězců nad Σ (*Alé množinu nad abec. nelze létat*)

Σ^+ – množina všech neprázdných řetězců nad Σ

$$\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$$

$$\Sigma^* = \left| \Sigma^+ \cup \{\varepsilon\} \right|$$

dejte 1 drž

Σ, a, c, 0, 1, ε, ac, cc, gg, εc, ...

Základní pojmy

Operace zřetězení (značíme jako '.'): *nov danou abc*

- $\forall x, y \in \Sigma^*$, připojením řetězce y za řetězec x vznikne řetězec $x.y$ (zkracujeme na xy).
- je asociativní, t.j. $\forall x, y, z \in \Sigma^* : (xy)z = x(yz)$,
- není komutativní, t.j. $\exists x, y \in \Sigma^* : xy \neq yx$, *↳ $a \neq a$ vzhledem k operaci zřetězení*
- ε se chová vzhledem k operaci zřetězení jako neutrální prvek: $x\varepsilon = \varepsilon x = x$
- $a^0 = \varepsilon, a^1 = a, a^2 = aa, a^3 = aaa, \dots$
↗ O. opakování ...

Reverze řetězce (značíme jako x^R): $(abc)^R = cba$

- $x = a_1 a_2 a_3 \dots a_n, x^R = a_n a_{n-1} \dots a_1$
- $y = abcd, y^R = dcba$

Základní pojmy

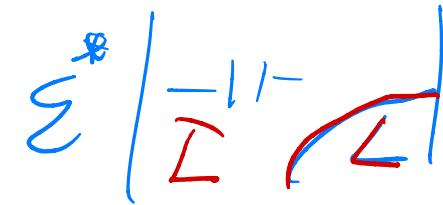
Délka řetězce x : *počet symbolů v daném řetězci.*

- značíme $|x|$
- $|x| \geq 0$,
- $|x| = 0 \Leftrightarrow x = \varepsilon$

Formální jazyk

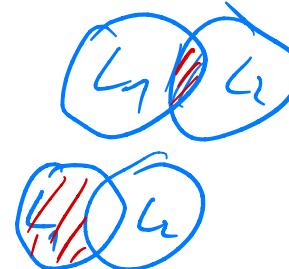
zjivo

Formální jazyk L nad Σ : $L \subseteq \Sigma^*$ (množina řetězců).



operace:

- množinové operace: sjednocení, průnik a rozdíl
- komplement (doplňek) jazyka L_1 : $\overline{L_1} = \Sigma^* \setminus L_1$
($\overline{L_1} \cup L_1 = \Sigma^*$, $L_1 \cap \overline{L_1} = \emptyset$).
- součin (zřetězení) jazyků: $L = L_1 \cdot L_2 = \{xy : x \in L_1, y \in L_2\}$ (L je definován nad abecedou $\Sigma = \Sigma_1 \cup \Sigma_2$)
- n -tá mocnina jazyka L : $L^n = L \cdot L^{n-1}$, $L^0 = \{\varepsilon\}$.



Iterace (Kleene star) L^* jazyka L : $L^* = \bigcup_{n=0}^{\infty} L^n$.

$L^* = L^+ \cup \{\varepsilon\}$,

$L^+ = L \cdot L^* = L^* \cdot L = \bigcup_{n=1}^{\infty} L^n$ (pozitivní iterace, Kleene plus)

$$L_1 = \{aa, bb, cc\}$$

$$L_2 = \{gg, hh\}$$

$$L^n = L \cdot L^{n-1}; L^0 = \{\varepsilon\}$$



$L_1 \cdot L_2 =$ zřetězením slov z 1. jazyka a 2. jazyka
Sacast, ll gg
cccccc, oahh...

$$L = \{ab, cd\}$$

$$L^0 = \{\epsilon\}$$

$$L^1 = L$$

$$L^2 = \{abab, abcd, cdab, cd\,cd\}$$

Nejdále zednictví

$$L : L^* = \bigcup_{n=0}^{\infty} L^n$$

$$\Sigma = \{a, c, g, \epsilon\}$$

(můžou mít i více různých významů)

$$L^1 = \emptyset, L^* = \{\epsilon\}$$

$$L^1 = L \cdot \{\epsilon\}$$

↳ hledáme řetezec s ϵ , ale nějaké pravidlo

$$L = \{a, c, g, \epsilon\}$$

$$L^* = \{\epsilon, a, c, g, \epsilon, \text{řet. dle } 2, 3, \dots\} \rightarrow \text{opakování}$$

↳ opakování některého znaku Σ^*
dále pak

Gramatika

Definice

Gramatika je čtveřice $G = (N, \Sigma, P, S)$, kde

- N je konečná množina neterminálních symbolů,
- Σ je konečná množina terminálních symbolů ($\Sigma \cap N = \emptyset$, značíme též T),
- P je množina (přepisovacích) pravidel. Je to konečná podmnožina množiny $(N \cup \Sigma)^*.N.(N \cup \Sigma)^* \times (N \cup \Sigma)^*$, (element (α, β) z P zapíšeme $\alpha \rightarrow \beta$ a nazveme pravidlo), *lze. některé*
- $S \in N$ je počáteční symbol gramatiky (též větný nebo startovací symbol).

X reprezentuje y

možné/dovolené, Doseňá

právní dovolené

→ množina (některé..)

*na co
přepisujeme*

cílem ho zavěříme

Gramatika

Příklad

vs výz.

Gramatika $G_1 = (\{A, S\}, \{0, 1\}, P, S)$, kde P :

- $S \rightarrow 0A$
 - $A \rightarrow 1A$
 - $A \rightarrow 0.$ ↗ *přepisovací pravidlo*
- slan*
- Nej → A → 1A | 0*

Poznámka:

$\alpha \rightarrow \beta_1, \alpha \rightarrow \beta_2, \dots, \alpha \rightarrow \beta_n$, lze zkrátit na:

$\alpha \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_n.$

Gramatika

Další možnosti zápisu gramatiky:

- Backus-Naurova forma (BNF),
- rozšířená Backus-Naurova forma (EBNF).

Příklad

Gramatika G generuje jazyk celých čísel bez znaménka. V této gramatice je použita BNF. *dva neterminy* *číslo*

$$G = (\{\langle \text{celé číslo} \rangle, \langle \text{číslice} \rangle\}, \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, P, \langle \text{celé číslo} \rangle)$$

Množina P obsahuje pravidla:

$$\langle \text{celé číslo} \rangle ::= \langle \text{číslice} \rangle \langle \text{celé číslo} \rangle \mid \langle \text{číslice} \rangle$$

$$\langle \text{číslice} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9.$$

Gramatika

Definice

$\xrightarrow{\text{def}} \gamma \beta \gamma$ if $\beta \xrightarrow{P} \gamma$ "přímo derivuje"

$G = (N, \Sigma, P, S)$, $x, y \in (N \cup \Sigma)^*$. Říkáme, že x přímo derivuje y ($x \Rightarrow y$), jestliže existuje $(\alpha \rightarrow \beta) \in P$ a $\gamma, \delta \in (N \cup \Sigma)^*$ takové, že $x = \gamma \alpha \delta$, $y = \gamma \beta \delta$ (t.j. $\gamma \alpha \delta \Rightarrow \gamma \beta \delta$).

Příklad

$G_1 = (\{A, S\}, \{0, 1\}, P, S)$, $P = \{S \rightarrow 0A00, A \rightarrow 1A, A \rightarrow 0\}$.

$01A00 \Rightarrow 011A00$ ($\gamma = 01, \alpha = A, \beta = 1A, \delta = 00$)

Definice

$\alpha \xrightarrow{k} \beta$, jestliže existuje posloupnost $\alpha_0, \alpha_1, \dots, \alpha_k$, pro $k \geq 0, k+1$ řetězců takových, že $\alpha = \alpha_0, \alpha_{i-1} \Rightarrow \alpha_i$ pro $1 \leq i \leq k$, a $\alpha_k = \beta$. Tuto posloupnost nazveme derivací řetězce β z řetězce α , která má délku k v gramatice G .

Příklad

$G_1: 01A00 \xrightarrow{4} 011111A00$

$A \Rightarrow 1A$
ořešit

$01A00 \xrightarrow{1.} 011A00 \xrightarrow{2.} 01111A00 \xrightarrow{3.} 011111A00 \xrightarrow{4.} 011111A00$

Gramatika

2

Definice

Tranzitivní uzávěr relace \Rightarrow : $\alpha \Rightarrow^+ \beta$, když $\alpha \Rightarrow^i \beta$ pro nějaké $i \geq 1$.
(\Rightarrow^+ čteme jako „derivuje po nenulovém počtu kroků“.)

Definice

Tranzitivní a reflexivní uzávěr relace \Rightarrow : $\alpha \Rightarrow^* \beta$, když $\alpha \Rightarrow^i \beta$ pro nějaké $i \geq 0$.
(\Rightarrow^* čteme jako „derivuje po libovolném počtu kroků“.)

Gramatika

Definice

$G = (N, \Sigma, P, S)$. Řetězec α nazveme větnou formou v gramatice G , jestliže $S \Rightarrow^* \alpha, \alpha \in (N \cup \Sigma)^*$.

Příklad

$$G_1: S \Rightarrow^* 01A00$$

$$G_1: S \Rightarrow^* 011000$$

"Větná forma" → může se mít až několik už (ještě něco nekonečného)

i větu je vět. f

jen a terminální → "věta" gen G →

nekončí derivace

Definice

Větná forma v $G = (N, \Sigma, P, S)$, která neobsahuje žádné neterminální symboly, se nazývá věta generovaná gramatikou G.

Příklad

$$G_1: S \Rightarrow^* 011000$$

Gramatika

Definice

množina všech slov nad slovníkem, kteréž derivace
 $L(G) = \{w : w \in \Sigma^*, \exists S \Rightarrow^* w\}$ je jazyk generovaný gramatikou
 $G = (N, \Sigma, P, S)$.

(Jazyk generovaný gramatikou G je množina všech vět generovaných gramatikou G .)

Poznámka

Všimněte si, že ε je věta generovaná gramatikou G , pokud $\varepsilon \in L(G)$.

Příklad

$G_1 = (\{A, S\}, \{0, 1\}, \{S \rightarrow 0A, A \rightarrow 1A, A \rightarrow 0\}, S)$ generuje jazyk
 $L(G_1) = \{01^n 0 : n \geq 0\}$.

V gramatice G_1 existují například derivace:

$$S \Rightarrow 0A \Rightarrow 00$$

$$S \Rightarrow 0A \Rightarrow 01A \Rightarrow 010$$

$$S \Rightarrow 0A \Rightarrow 01A \Rightarrow 011A \Rightarrow 0110$$

$J \rightarrow OA$

$A \rightarrow 1A10$

$J \rightarrow OA \Rightarrow OO \Rightarrow$

$\Rightarrow O1A \Rightarrow O10$

$\Rightarrow O11A \Rightarrow O110$

\Downarrow

$\Rightarrow O111A$

jazek generowym jest zbiór $L(6_1) = \{01^n0 : n \geq 0\}$

Gramatika

Definice

Gramatiky G_1 a G_2 jsou ekvivalentní, když generují stejný jazyk. To znamená, že $L(G_1) = L(G_2)$.

||

ale zohledněte, že generují stejný jazyk

Klasifikace gramatik

Noam Chomsky (*7.12.1928 Philadelphia)

- práce na poli gramatik jak formálních, tak i přirozených jazyků

Definice

$G = (N, \Sigma, P, S)$. Říkáme, že G je:

Níže $(N \cup \Sigma)^* N (\Delta \cup \varepsilon)^* \times (N \cup \varepsilon)^*$

0. Neomezená (typu 0), jestliže odpovídá obecné definici gramatiky.
1. Kontextová (typu 1), jestliže každé pravidlo z P má tvar $\gamma A \delta \rightarrow \gamma \alpha \delta$, kde $\gamma, \delta \in (N \cup \Sigma)^*$, $\alpha \in (N \cup \Sigma)^+$, $A \in N$, nebo tvar $S \rightarrow \varepsilon$ v případě, že S se nevyskytuje na pravé straně žádného pravidla.
2. Bezkontextová (typu 2), jestliže každé pravidlo má tvar $A \rightarrow \alpha$, kde $A \in N, \alpha \in (N \cup \Sigma)^*$.
3. Regulární (typu 3), jestliže každé pravidlo má tvar $A \rightarrow aB$ nebo $A \rightarrow a$, kde $A, B \in N, a \in \Sigma$, nebo tvar $S \rightarrow \varepsilon$ v případě, že S se nevyskytuje na pravé straně žádného pravidla.

$$1. \Delta A \beta \rightarrow \alpha \gamma \beta \quad |\beta| \geq 1$$

$$aAb \rightarrow aCb$$

$$aAc \rightarrow aBc$$

$$2. A \rightarrow \alpha \quad \left\{ \begin{array}{l} 3. A \rightarrow \lambda \quad \text{nebn. termi nek.} \\ A \rightarrow A \end{array} \right.$$

Klasifikace jazyků

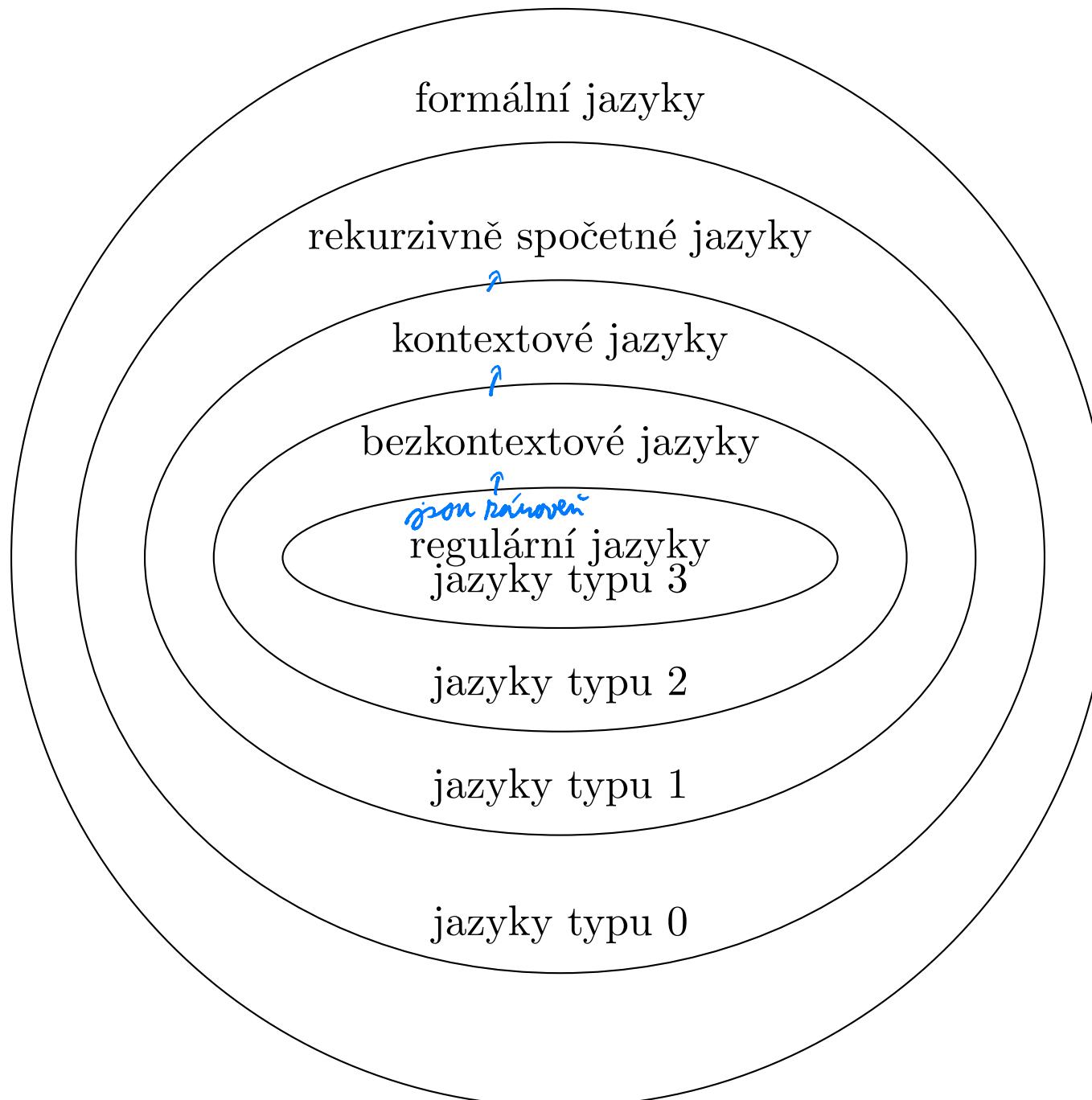
Definice

Řekneme že, jazyk

0. je *rekurzivně spočetný* (typu 0), pokud \exists neomezená gramatika, která ho generuje.
vždy lineárně omezen délka vstupu *(vždy neomezen)*
- rozpoznatelné Turingovým strojem *jsem neomezen*
1. je *kontextový* (typu 1), pokud \exists kontextová gramatika, která ho generuje.
- rozpoznatelné lineárně omezeným Turingovým strojem
2. je *bezkontextový* (typu 2), pokud \exists bezkontextová gramatika, která ho generuje.
- rozpoznatelné zásobníkovým automatem
3. je *regulární* (typu 3), pokud \exists regulární gramatika, která ho generuje.
- rozpoznatelné konečným automatem

nejméně reikován

Klasifikace jazyků



Klasifikace jazyků

Příklad

$G_1 = (\{S\}, \{0, 1\}, \{S \rightarrow 0S, S \rightarrow 1S, S \rightarrow 1, S \rightarrow 0\}, S)$ je regulární gramatika a generuje jazyk $L(G_1) = \{0, 1\}^+$.

*neobsahuje
není → bezkon.
→ regulární zjednodušení neobsahuje regulární.*

$G_2 = (\{S\}, \{0, 1\}, \{S \rightarrow 0S1, S \rightarrow 01\}, S)$ je bezkontextová a generuje jazyk $L(G_2) = \{0^n 1^n : n \geq 1\}$.

bezkontextová

$G_3 = (\{S, A\}, \{0, 1\}, \{S \rightarrow 0A1, S \rightarrow 01, 0A \rightarrow 00A1, A \rightarrow 01\}, S)$ je kontextová a generuje jazyk $L(G_3) = \{0^n 1^n : n \geq 1\}$.

bezkontextová

$G_4 = (\{S\}, \{0, 1\}, \{S \rightarrow 0S1, S \rightarrow 01, 0S1 \rightarrow S\}, S)$ je neomezená a generuje jazyk $L(G_4) = \{0^n 1^n : n \geq 1\}$.

*neomezená
máj kontext
nereadnorám*

Gramatiky G_2, G_3 a G_4 jsou ekvivalentní, protože

$$L(G_2) = L(G_3) = L(G_4) = \{0^n 1^n : n \geq 1\}.$$

$L(G_1)$ je regulární, $L(G_2)$ není, přesto $L(G_2) \subseteq L(G_1)$.

Uzavřenost bezkontextových jazyků

Uzavřenost:

Jestliže určité jazyky jsou bezkontextové a jazyk L z nich vznikl nějakou operací, pak L je také bezkontextový.

Věta

Třída bezkontextových jazyků je uzavřena na operacích *sjednocení, součin a iterace*.

Gramatiky a operace nad jazyky

Algoritmus Gramatika pro *sjednocení* jazyků.

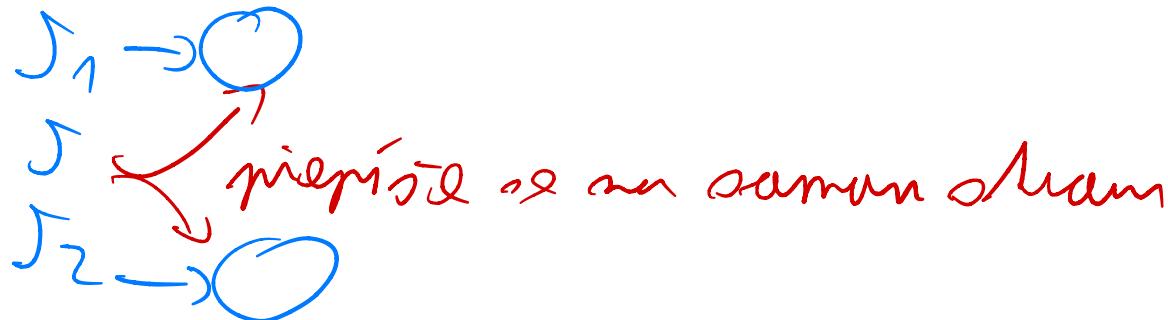
Vstup: Bezkontextové gramatiky $G_1 = (N_1, \Sigma, P_1, S_1)$ a $G_2 = (N_2, \Sigma, P_2, S_2)$ generující jazyky L_1 a L_2 , $N_1 \cap N_2 = \emptyset$.

Výstup: Bezkontextová gramatika G , $L(G) = L_1 \cup L_2$.

1: $G \leftarrow (N_1 \cup N_2 \cup \{S\}, \Sigma, P_1 \cup P_2 \cup \{S \rightarrow S_1 \mid S_2\}, S)$, kde $S \notin N_1 \cup N_2$.

Předpokládáme, že množiny terminálních symbolů jsou stejné.

Máme $G_1 \Rightarrow L_1$; $G_2 \Rightarrow L_2$ (bez důk.)



Gramatiky a operace nad jazyky

Algoritmus Gramatika pro *součin* jazyků.

Vstup: Bezkontextové gramatiky $G_1 = (N_1, \Sigma, P_1, S_1)$ a $G_2 = (N_2, \Sigma, P_2, S_2)$ generující jazyky L_1 a L_2 , $N_1 \cap N_2 = \emptyset$.

Výstup: Bezkontextová gramatika G , že $L(G) = L_1 \cdot L_2$.

1: $G \leftarrow (N_1 \cup N_2 \cup \{S\}, \Sigma, P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}, S)$, kde $S \notin N_1 \cup N_2$.

$\Sigma \rightarrow \Sigma_1 \cdot \Sigma_2$
↓

Gramatiky a operace nad jazyky

Algoritmus Gramatiky pro *iteraci* jazyka.

Vstup: Bezkontextová gramatika $G = (N, \Sigma, P, S)$ generující jazyk L .

Výstup: Bezkontextová gramatika G' , že $L(G') = L^*$.

1: $G' \leftarrow (N \cup \{S'\}, \Sigma, P \cup \{S' \rightarrow SS', S' \rightarrow \varepsilon\}, S')$, kde $S' \notin N$.

$$S \rightarrow S_1 S \mid \varepsilon$$

↓
lze tak generovat rieboľ S_1 v ďalšej \bigcirc

$$\varepsilon, S_1, S_1, S_1$$

Konečné jazyky

Definice

Jazyk $L \subset \Sigma^*$ je nazván konečný, pokud $\exists n \in \mathbb{N}: |L| < n$.

Věta

Nejmenší třída jazyků, která obsahuje všechny konečné jazyky a jazyky vzniklé z konečných jazyků pomocí operací

- a) sjednocení,
- b) součinu,
- c) iterace,
- d) doplněk,

je množina všech regulárních jazyků.

Derivační strom pro bezkontext. jazyky

Derivační strom je grafickým vyjádřením syntaktické struktury větné formy.

Derivační strom pro bezkontext. jazyky

Definice

Mějme gramatiku $G = (N, \Sigma, P, S)$. *Derivační strom* je strom, který má následující vlastnosti:

1. Uzly derivačního stromu jsou ohodnoceny terminálními a neterminálními symboly a symbolem ε (pak je to jediný syn svého otcovského uzlu).
2. Kořen stromu je ohodnocen počátečním symbolem S .
3. Jestliže uzel má alespoň jednoho následovníka, je ohodnocen neterminálním symbolem.
4. Jestliže n_1, n_2, \dots, n_k jsou bezprostřední následovníci uzlu n , který je ohodnocen symbolem A , a tyto uzly jsou zleva doprava ohodnoceny symboly A_1, A_2, \dots, A_k , pak $A \rightarrow A_1 A_2 \dots A_k$ je pravidlo v P .
5. Koncové uzly derivačního stromu tvoří zleva doprava větnou formu nebo větu v gramatice G , která je *výsledkem* derivačního stromu.

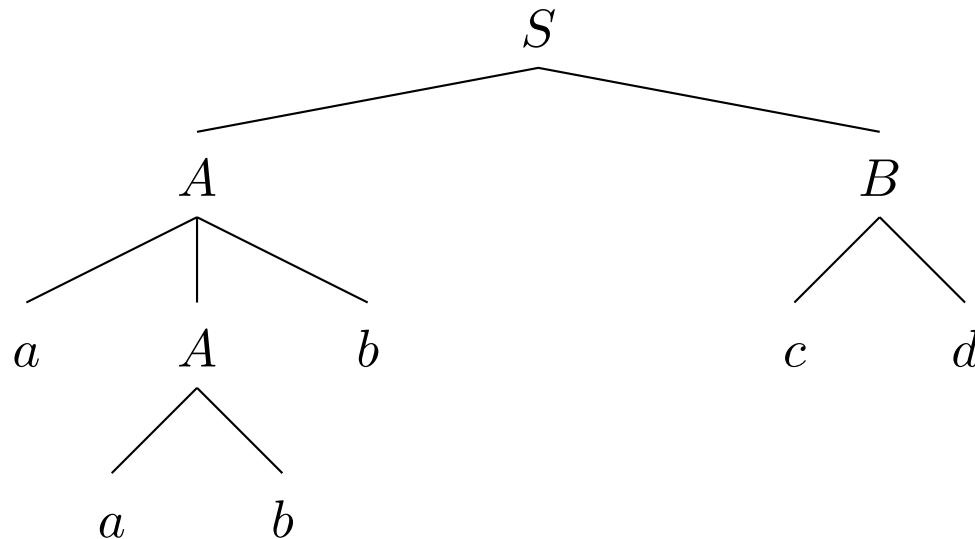
Derivační strom pro bezkontext. jazyky

Příklad

$G = (\{S, A, B\}, \{a, b, c, d\}, P, S)$, kde P :

- (1) $S \rightarrow AB$
- (2) $A \rightarrow aAb$
- (3) $A \rightarrow ab$
- (4) $B \rightarrow cBd$
- (5) $B \rightarrow cd$

$S \Rightarrow AB \Rightarrow aAbB \Rightarrow aabbB \Rightarrow aabbcd.$



$S \Rightarrow AB \Rightarrow Acd \Rightarrow aAbcd \Rightarrow aabbcd$

$S \Rightarrow AB \Rightarrow aAbB \Rightarrow aAbcd \Rightarrow aabbcd$