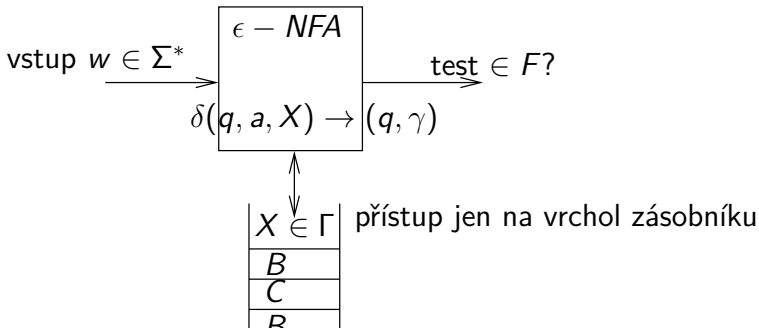


Zásobníkové automaty

- Zásobníkové automaty jsou rozšířením ϵ -NFA nedeterministických konečných automatů s ϵ přechody.
- Přidanou věcí je **zásobník**. Má vlastní abecedu Γ .
- V každém kroku vidíme horní písmeno zásobníku (zde X), můžeme dát navrch libovolný konečný počet znaků $\gamma \in \Gamma^*$.
- Může si pamatovat neomezené množství informace.
- Deterministické zásobníkové automaty přijímají jen vlastní podmnožinu bezkontextových jazyků.



Zásobníkový automat (PDA)

Definition 9.1 (Zásobníkový automat (PDA))

Zásobníkový automat (PDA) je $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde

Q konečná množina stavů

Σ neprázdná konečná množina vstupních symbolů

Γ neprázdná konečná zásobníková abeceda

δ přechodová funkce $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow P_{FIN}(Q \times \Gamma^*)$,
 $\delta(p, a, X) \ni (q, \gamma)$

kde q je nový stav a γ je řetězec zásobníkových symbolů, který nahradí X na vrcholu zásobníku

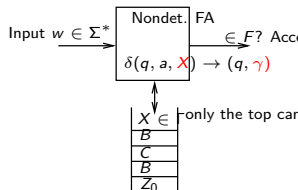
$q_0 \in Q$ počáteční stav

$Z_0 \in \Gamma$ Počáteční zásobníkový symbol. Víc na začátku na zásobníku není.

F Množina přijímajících (koncových) stavů; může být nedefinovaná.

V jednom časovém kroku zásobníkový automat:

- Přečte na vstupu žádný nebo jeden symbol. (ϵ přechody pro prázdný vstup.)
- Přejde do nového stavu.
- Nahradí symbol na vrchu zásobníku libovolným řetězcem (ϵ odpovídá samotnému pop, jinak následuje push jednoho nebo více symbolů).



Example 9.1

Zásobníkový automat pro jazyk: $L_{wwr} = \{ww^R \mid w \in (\mathbf{0} + \mathbf{1})^*\}$.

PDA přijímající L_{wwr} :

- Start q_0 reprezentuje odhad, že ještě nejsme uprostřed.
- V každém kroku nedeterministicky hádáme;
 - Zůstat q_0 (ještě nejsme uprostřed).
 - Přejít ϵ přechodem do q_1 (už jsme viděli střed).
- V q_0 , přečte vstupní symbol a dá (push) ho na zásobník
- V q_1 , srovná vstupní symbol s vrcholem zásobníku
 - pokud se shodují, přečte vstupní symbol a umaže (pop) vrchol zásobníku
- Když vyprázdníme zásobník, přijmeme vstup, který jsme doteď přečetli.

PDA pro L_{wwr} Example 9.2 (PDA pro L_{wwr})

PDA pro L_{wwr} můžeme popsat

$P_{da} = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$ kde δ je definovaná:

$$\delta(q_0, 0, Z_0) = \{(q_0, 0Z_0)\}$$

$$\delta(q_0, 1, Z_0) = \{(q_0, 1Z_0)\}$$

Ulož vstup na zásobník, startovní symbol tam nech

$$\delta(q_0, 0, 0) = \{(q_0, 00)\}$$

$$\delta(q_0, 0, 1) = \{(q_0, 01)\}$$

$$\delta(q_0, 1, 0) = \{(q_0, 10)\}$$

$$\delta(q_0, 1, 1) = \{(q_0, 11)\}$$

Zůstaň v q_0 , přečti vstup a dej ho na zásobník

$$\delta(q_0, \epsilon, Z_0) = \{(q_1, Z_0)\}$$

$$\delta(q_0, \epsilon, 0) = \{(q_1, 0)\}$$

$$\delta(q_0, \epsilon, 1) = \{(q_1, 1)\}$$

ϵ přechod q_1 bez změny zásobníku (a vstupu)

$$\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$$

$$\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$$

stav q_1 srovná vstupní symbol a vrchol zásobníku

$$\delta(q_1, \epsilon, Z_0) = \{(q_2, Z_0)\}$$

našli jsme ww^R a jdeme do přijímajícího stavu

Grafická notace PDA's

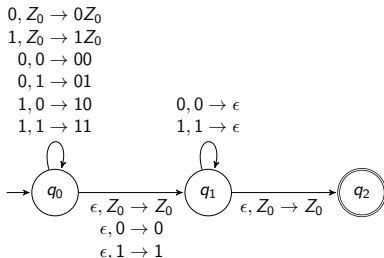
Definition 9.2 (Přechodový diagram pro zásobníkový automat)

Přechodový diagram pro zásobníkový automat obsahuje:

- Uzly, které odpovídají stavům PDA.
- Šipka 'odnikud' ukazuje počáteční stav, dvojité kruhy označují přijímající stavy.
- hrana odpovídá přechodu PDA. Hrana označená $a, X \rightarrow \alpha$ ze stavu p do q znamená $\delta(p, a, X) \ni (q, \alpha)$
- Konvence je, že počáteční symbol zásobníku značíme Z_0 .

Anotace hrany:

vstupní_znak, zásobníkový_znak \rightarrow push_řetězec



Notace zásobníkových automatů

Example 9.3 (Notace)

| | |
|--------------------------|-------------------------------|
| $a, b, c, *, +, 1, (,)$ | symboly vstupní abecedy |
| p, q, r | stavy |
| u, v, w, x, y, z | řetězce vstupní abecedy |
| X, Y, E, I, S | zásobníkové symboly |
| α, β, γ | řetězce zásobníkových symbolů |

- Narozdíl od gramatik může vstupní a zásobníková abeceda obsahovat stejné symboly.
- Vyhýbáme se stejným názvům stavů jako jsou písmena kterékoli z abeced.

Definition 9.3 (Konfigurace zásobníkového automatu)

Konfiguraci zásobníkového automatu reprezentujeme trojicí (q, w, γ) , kde

q je stav

w je zbývajícím vstup a

γ je obsah zásobníku (vrch zásobníku je vlevo).

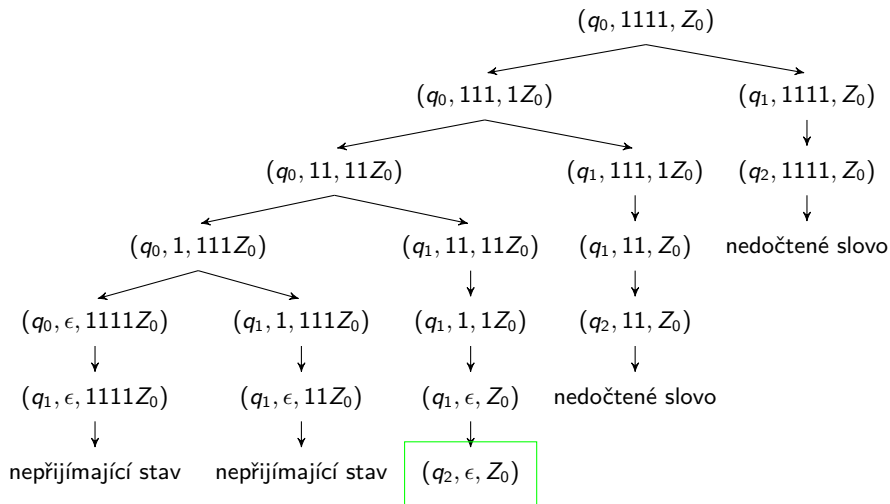
konfiguraci značíme zkratkou **(ID)** z anglického **instantaneous description (ID)**.

Definition 9.4 (\vdash, \vdash^* posloupnosti konfigurací)

Mějme PDA $P_{da} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$. Mějme stavy $p, q \in Q$, $a \in (\Sigma \cup \{\epsilon\})$, $X \in \Gamma$, $\alpha \in \Gamma^*$ a instrukci $\delta(p, a, X) \ni (q, \alpha)$. Pak říkáme, že

- konfigurace $(p, aw, X\beta)$ **bezprostředně vede** na konfiguraci $(q, w, \alpha\beta)$,
 - Značíme $(p, aw, X\beta) \vdash (q, w, \alpha\beta)$.
- **konfigurace I vede na konfiguraci J** $I \vdash_p^* J$ a $I \vdash^* J$ používáme na označení nuly a více kroků zásobníkového automatu, t.j.
 - $I \vdash^* I$ pro každou konfiguraci I
 - $I \vdash^* J$ pokud existuje konfigurace K tak že $I \vdash K$ a $K \vdash^* J$.

konfigurace zásobníkového automatu na vstup 1111



Jazyky zásobníkových automatů

Definition 9.5 (Jazyk přijímaný koncovým stavem, prázdným zásobníkem)

Mějme zásobníkový automat $P_{da} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$. Pak $L(P_{da})$, **jazyk přijímaný (akceptovaný) koncovým stavem** je

$$L(P_{da}) = \{w \mid (q_0, w, Z_0) \vdash_{P_{da}}^* (q, \epsilon, \alpha) \text{ pro nějaké } q \in F \text{ a libovolný řetězec } \alpha \in \Gamma^*; w \in \Sigma^*\}.$$

jazyk přijímaný prázdným zásobníkem $N(P_{da})$ definujeme

$$N(P_{da}) = \{w \mid (q_0, w, Z_0) \vdash_{P_{da}}^* (q, \epsilon, \epsilon) \text{ pro libovolné } q \in Q; w \in \Sigma^*\}.$$

- Protože je množina přijímajících stavů F nerelevantní, může se vynechat a PDA je šestice $P_{da} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$.

Example 9.4

Zásobníkový automat z předchozího příkladu přijímá L_{wwr} koncovým stavem.

Example 9.5

$P'_{da} \equiv P_{da}$ z předchozího příkladu, jen změníme instrukci, aby umazala poslední symbol $\delta(q_1, \epsilon, Z_0) = \{(q_2, Z_0)\}$ nahradíme $\delta(q_1, \epsilon, Z_0) = \{(q_2, \epsilon)\}$
Nyní $L(P'_{da}) = N(P'_{da}) = L_{wwr}$.

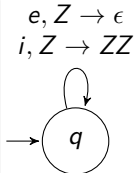
Příklad If-Else

Example 9.6 (If-else přijímané prázdným zásobníkem)

Následující zásobníkový automat zastaví při první chybě na if (*i*) a else (*e*), máme-li více else než if.

$P_N = (\{q\}, \{i, e\}, \{Z\}, \delta_N, q, Z)$ kde

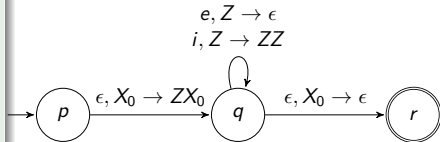
- $\delta_N(q, i, Z) = \{(q, ZZ)\}$ push
- $\delta_N(q, e, Z) = \{(q, \epsilon)\}$ pop



Example 9.7 (Přijímání koncovým stavem)

$P_F =$
 $(\{p, q, r\}, \{i, e\}, \{Z, X_0\}, \delta_F, p, X_0, \{r\})$
 kde

- $\delta_F(p, \epsilon, X_0) = \{(q, ZX_0)\}$ start
- $\delta_F(q, i, Z) = \{(q, ZZ)\}$ push
- $\delta_F(q, e, Z) = \{(q, \epsilon)\}$ pop
- $\delta_F(q, \epsilon, X_0) = \{(r, \epsilon)\}$ přijmi



Nečtený vstup a dno zásobníku P neovlivní výpočet

Lemma 9.1

Mějme PDA $P_{da} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ a $(p, u, \alpha) \vdash_{P_{da}}^* (q, v, \beta)$. Potom pro libovolné slovo $w \in \Sigma^*$ and $\gamma \in \Gamma^*$ platí: $(p, uw, \alpha\gamma) \vdash_{P_{da}}^* (q, vw, \beta\gamma)$.
Speciálně pro $\gamma = \epsilon$ a/nebo $w = \epsilon$.

Proof.

Indukcí podle počtu konfigurací mezi $(p, uw, \alpha\gamma)$ a $(q, vw, \beta\gamma)$. Každý krok $(p, u, \alpha) \vdash_{P_{da}}^* (q, v, \beta)$ je určen bez w a/nebo γ . Proto je možný i se symboly na konci vstupu / dně zásobníku. □

Lemma 9.2

Pro PDA $P_{da} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ a $(p, uw, \alpha) \vdash_{P_{da}}^* (q, vw, \beta)$ platí $(p, u, \alpha) \vdash_{P_{da}}^* (q, v, \beta)$.

Remark Pro zásobník ale obdoba neplatí. PDA může zásobníkové symboly γ použít a zase je tam naskládat (push). $L = \{0^i 1^i 0^j 1^j\}$, konfigurace $(p, 0^{i-j} 1^i 0^j 1^j, 0^j Z_0) \vdash^* (q, 1^j, 0^j Z_0)$, mezitím vyčistíme zásobník k Z_0 .

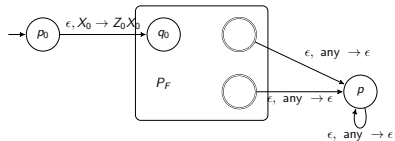
Od přijímajícího stavu k prázdnému zásobníku

Lemma 9.3 (Od přijímajícího stavu k prázdnému zásobníku)

Mějme $L = L(P_F)$ pro nějaký PDA

$P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$.

Pak existuje PDA P_N takový, že $L = N(P_N)$.



Proof:

Nechť $P_N = (Q \cup \{p_0, p\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$, kde

- $\delta_N(p_0, \epsilon, X_0) = \{(q_0, Z_0 X_0)\}$ start
- $\forall (q \in Q, a \in \Sigma \cup \{\epsilon\}, Y \in \Gamma)$
 $\delta_N(q, a, Y) = \delta_F(q, a, Y)$
 simulujeme
- $\forall (q \in F, Y \in \Gamma \cup \{X_0\})$,
 $\delta_N(q, \epsilon, Y) \ni (p, \epsilon)$ přijmout
 pokud P_F přijímá,
- $\forall (Y \in \Gamma \cup \{X_0\})$,
 $\delta_N(p, \epsilon, Y) = \{(p, \epsilon)\}$ vyprázdnit
 zásobník.

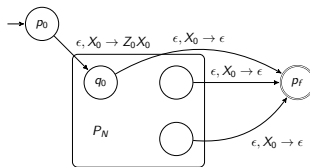
Pak $w \in N(P_N)$ iff $w \in L(P_F)$. □

Od prázdného zásobníku ke koncovému stavu

Lemma 9.4 (Od prázdného zásobníku ke koncovému stavu)

Pokud $L = N(P_N)$ pro nějaký PDA

$P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$,
pak existuje PDA P_F
takový, že $L = L(P_F)$.



Proof:

$$P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$$

kde δ_F je

- $\delta_F(p_0, \epsilon, X_0) = \{(q_0, Z_0 X_0)\}$ (start).
- $\forall (q \in Q, a \in \Sigma \cup \{\epsilon\}, Y \in \Gamma),$
 $\delta_F(q, a, Y) = \delta_N(q, a, Y).$
- Navíc, $\delta_F(q, \epsilon, X_0) \ni (p_f, \epsilon)$ pro každý $q \in Q.$

Chceme ukázat $w \in N(P_N)$ iff $w \in L(P_F)$.

- (If) P_F přijímá následovně:
 $(p_0, w, X_0) \vdash_{P_F} (q_0, w, Z_0 X_0) \vdash_{P_F=N_F}^* (q, \epsilon, X_0) \vdash_{P_F} (p_f, \epsilon, \epsilon).$
- (Only if) Do p_f nelze dojít jinak než předchozím bodem. □

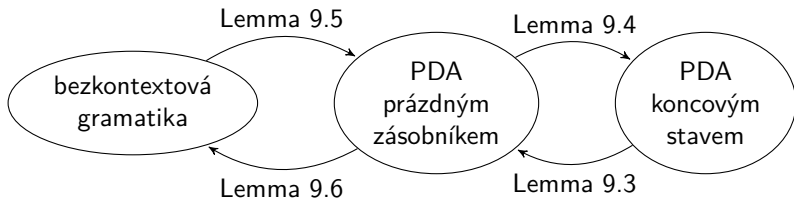
Ekvivalence jazyků rozpoznávaných zásobníkovými automaty a bezkontextových jazyků

Theorem 9.1 ($L(\text{CFG}), L(\text{PDA}), N(\text{PDA})$)

Následující tvrzení jsou ekvivalentní:

- Jazyk L je bezkontextový, tj. generovaný bezkontextovou gramatikou.
- Jazyk L je přijímaný nějakým zásobníkovým automatem koncovým stavem.
- Jazyk L je přijímaný nějakým zásobníkovým automatem prázdným zásobníkem.

Důkaz bude veden směry dle následujícího obrázku.



Od bezkontextové gramatiky k zásobníkovému automatu

Algorithm: Konstrukce PDA z CFG G

Mějme CFG gramatiku $G = (V, T, P, S)$.

Konstruuje PDA $P = (\{q\}, T, V \cup T, \delta, q, S)$.

- (1) Pro neterminály $A \in V$, $\delta(q, \epsilon, A) = \{(q, \beta) \mid A \rightarrow \beta \text{ je pravidlo } G\}$.
- (2) pro každý terminál $a \in T$, $\delta(q, a, a) = \{(q, \epsilon)\}$.

Example 9.8

Konvertujme gramatiku: $G = (\{E, I\}, \{a, b, 0, 1, (,), +, *\}, \{I \rightarrow a|b|la|lb|l0|l1, E \rightarrow I|E * E|E + E|(E)\}, E)$.

Množina vstupních symbolů PDA je $\Sigma = \{a, b, 0, 1, (,), +, *\}$, $\Gamma = \Sigma \cup \{I, E\}$,
přechodová funkce δ :

- $\delta(q, \epsilon, I) = \{(q, a), (q, b), (q, la), (q, lb), (q, l0), (q, l1)\}$.
- $\delta(q, \epsilon, E) = \{(q, I), (q, E * E), (q, E + E), (q, (E))\}$.
- $\forall s \in \Sigma$ je $\delta(q, s, s) = \{(q, \epsilon)\}$, např. $\delta(q, +, +) = \{(q, \epsilon)\}$.

CFG a PDA

Lemma 9.5 (Přijímání prázdným zásobníkem ze CFG)

Pro PDA P_{da} konstruovaný z CFG G algoritmem výše je $N(P) = L(G)$.

- (1) Pro neterminály $A \in V$, $\delta(q, \epsilon, A) = \{(q, \beta) \mid A \rightarrow \beta \text{ je pravidlo } G\}$.
- (2) pro každý terminál $a \in T$, $\delta(q, a, a) = \{(q, \epsilon)\}$.

- Levá derivace: $E \Rightarrow E * E \Rightarrow I * E \Rightarrow a * E \Rightarrow a * I \Rightarrow a * b$
- Posloupnost konfigurací:
 $(q, a * b, E) \vdash (q, a * b, E * E) \vdash (q, a * b, I * E) \vdash (q, a * b, a * E)$
 $\vdash (q, *b, *E) \vdash (q, b, E) \vdash (q, b, I) \vdash (q, b, b) \vdash (q, \epsilon, \epsilon)$

Pozorování:

- Kroky derivace simuluje PDA ϵ přepisy zásobníku
- odmazávaný vstup u PDA v derivaci zůstává až do konce
- až PDA vymaže terminály, pokračuje v přepisech.

CFG a PDA

$$w \in N(P_{da}) \Leftarrow w \in L(G).$$

Nechť $w \in L(G)$, w má levou derivaci $S = \gamma_1 \xRightarrow{lm} \gamma_2 \xRightarrow{lm} \dots \xRightarrow{lm} \gamma_n = w$.

Indukcí podle i dokážeme $(q, w, S) \vdash_P^* (q, v_i, \alpha_i)$, kde $\gamma_i = u_i \alpha_i$ je levá sentenciální forma a $u_i v_i = w$.

- Pokud γ_i obsahuje pouze terminály, $\gamma_i = u_i \alpha_i = w = u_i v_i$, tedy $\alpha_i = v_i$ a pravidly typu (2) vyprázdníme vstup i zásobník.
- Každá nekoncová sentenciální forma γ_i může být zapsaná $u_i A \alpha_i$,
A nejlevější neterminál, u_i řetězec terminálů
- indukční předpoklad nás dovedl do konfigurace $(q, v_i, A \alpha_i)$, $w = u_i v_i$
- Pro $\gamma_i \xRightarrow{lm} \gamma_{i+1}$ bylo použito pravidlo $(A \rightarrow \beta) \in P$
- PDA nahradí A na zásobníku β , přejde na konfiguraci $(q, v_i, \beta \alpha_i)$.
- odstraňme všechny terminály $v \in \Sigma^*$ zleva $\beta \alpha$ porovnáváním se vstupem
 - $v_i = v v_{i+1}$ a zároveň $\beta \alpha = v \alpha_{i+1}$
- přešli jsme do nové konfigurace $(q, v_{i+1}, \alpha_{i+1})$ a iterujeme.



$$w \in N(P_{da}) \Rightarrow w \in L(G).$$

Dokazujeme: Pokud $(q, u, A) \vdash_P^* (q, \epsilon, \epsilon)$, tak $A \xrightarrow{G}^* u$.

Indukcí podle počtu kroků P_{da} .

- $n = 1$ kroků:
 - $a \in \Sigma$, přechod $\delta(q, a, a) \ni (q, \epsilon)$, v derivaci žádný krok,
 - $A \in \Gamma$, přechod $\delta(q, \epsilon, A) \ni (q, \epsilon)$ pro pravidlo gramatiky $(A \rightarrow \epsilon) \in G$.
- $n > 1$ kroků:
 - První krok typu (2) – terminály, nerozšiřujeme derivaci.
 - První krok typu (1), A nahrazeno $Y_1 Y_2 \dots Y_k$ z pravidla $A \rightarrow Y_1 Y_2 \dots Y_k$.
Rozdělíme $u = u_1 u_2 \dots u_k$:
 - čtením symbolu Y_i skončilo slovo u_{i-1} a začíná u_i .



Použijeme indukční hypotézu na každé

$i = 1, \dots, k$:

$(q, u_i u_{i+1} \dots u_k, Y_i) \vdash^* (q, u_{i+1} \dots u_k, \epsilon)$ a
dostaneme $Y_i \xrightarrow{*} u_i$.

Příklad: Od zásobníkového automatu ke gramatice

Example 9.9

Převědme PDA $P_N = (\{q\}, \{i, e\}, \{Z\}, \delta_N, q, Z)$ na obrázku na gramatiku.

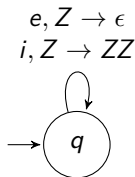
- Neterminály gramatiky budou $V = \{S, [qZq]\}$ nový start a jediná trojice P_N .
- Pravidla:
 - $S \rightarrow [qZq]$.
 - $[qZq] \rightarrow e$
 - $[qZq] \rightarrow i[qZq][qZq]$.

Můžeme nahradit trojici $[qZq]$ symbolem A a dostaneme:

$$S \rightarrow A$$

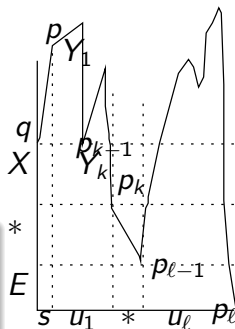
$$A \rightarrow iAA|e.$$

Protože A a S odvozují přesně stejné řetězce, můžeme je ztotožnit: $G = (\{S\}, \{i, e\}, \{S \rightarrow iSS|e\}, S)$.



Od zásobníkového automatu ke gramatice CFG

- Zásobní automat bere **jeden** symbol ze zásobníku. Stav před a po kroku může být různý.
- Neterminály gramatiky budou složené symboly $[qXp]$, PDA vyšel z q , vzal X a přešel do p ;
a zavedeme nový počáteční symbol S .



Lemma 9.6 (Gramatika pro PDA)

Mějme PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$. Pak existuje bezkontextová gramatika G taková, že $L(G) = N(P)$.

Pravidla definujeme:

- $\forall p \in Q: S \rightarrow [q_0 Z_0 p]$, tj. uhodni koncový stav a spusť PDA na $(q_0, w, Z_0) \vdash^* (p, \epsilon, \epsilon)$.
- Pro všechny dvojice $(p, Y_1 Y_2 \dots Y_k) \in \delta(q, s, X)$, $s \in \Sigma \cup \{\epsilon\}$, $\forall p_1, \dots, p_{k-1} \in Q$ vytvoř pravidlo

$$[qXp_k] \rightarrow s[pY_1p_1][p_1Y_2p_2] \dots [p_{k-1}Y_kp_k]$$

- spec. pro $(p, \epsilon) \in \delta(q, a, A)$ vytvoř $[qAp] \rightarrow a$.

Proof.

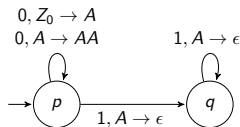
Pro $w \in \Sigma^*$ dokazujeme

$[qXp] \Rightarrow^* w$ právě když $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$

indukcí v obou směrech (počet kroků PDA, počet kroků derivace.) □

Example 9.10 ($\{0^n 1^n; n > 0\}$)

| δ | Pravidla | |
|-------------------------------------|--------------------------------------|-----|
| | $S \rightarrow [pZ_0p] \mid [pZ_0q]$ | (1) |
| $\delta(p, 0, Z_0) \ni (p, A)$ | $[pZ_0p] \rightarrow 0[pAp]$ | (2) |
| | $[pZ_0q] \rightarrow 0[pAq]$ | (3) |
| $\delta(p, 0, A) \ni (p, AA)$ | $[pAp] \rightarrow 0[pAp][pAp]$ | (4) |
| | $[pAp] \rightarrow 0[pAq][qAp]$ | (5) |
| | $[pAq] \rightarrow 0[pAp][pAq]$ | (6) |
| | $[pAq] \rightarrow 0[pAq][qAq]$ | (7) |
| $\delta(p, 1, A) \ni (q, \epsilon)$ | $[pAq] \rightarrow 1$ | (8) |
| $\delta(q, 1, A) \ni (q, \epsilon)$ | $[qAq] \rightarrow 1$ | (9) |



Derivace 0011

$S \Rightarrow^{(1)} [pZ_0q] \Rightarrow^{(3)} 0[pAq] \Rightarrow^{(7)} 00[pAq][qAq] \Rightarrow^{(8)} 001[qAq] \Rightarrow^{(9)} 0011$

Shrnutí

- Zásobníkový automat PDA je ϵ -NFA automat rozšířený o zásobník, potenciálně nekonečnou paměť
 - a zásobníkovou abecedu, počáteční zásobníkový symbol, přechodová funkce čte a píše na zásobník, píše i řetězec
- Přijímání koncovým stavem a prázdným zásobníkem, pro nedeterministické PDA přijímají stejnou třídu jazyků
- a to bezkontextové jazyky, generované bezkontextovými gramatikami.

Deterministický zásobníkový automat (DPDA)

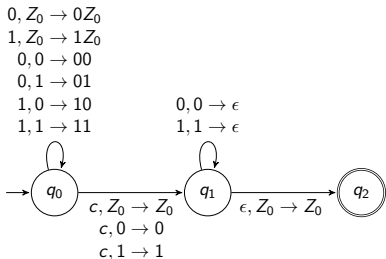
Definition 9.6 (Deterministický zásobníkový automat (DPDA))

Zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je **deterministický** PDA právě když platí zároveň:

- $\delta(q, a, X)$ je nejvýše jednoprvková $\forall (q, a, X) \in Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$.
- Je-li $\delta(q, a, X)$ neprázdná pro nějaké $a \in \Sigma$, pak $\delta(q, \epsilon, X)$ musí být prázdná.

Example 9.11 (Det. PDA přijímající L_{wcvr})

- Jazyk L_{wcvr} palindromů je bezkontextový, ale nemá přijímající deterministický zásobníkový automat.
- Druhá podmínka zaručuje, že nebude volba mezi ϵ přechodem a čtením vstupního symbolu.
- Vložení středové značky c do $L_{wcvr} = \{wcw^R \mid w \in (\mathbf{0} + \mathbf{1})^*\}$ dostaneme jazyk rozpoznatelný DPDA.



Regulární jazyky, DPDA's

$$RL \subsetneq L_{DPDA} \subsetneq L_{PDA} = CFL = N_{PDA} \supsetneq N_{DPDA}.$$

Theorem 9.2

Nechť L je regulární jazyk, pak $L = L(P)$ pro nějaký DPDA P .

Proof.

DPDA může simulovat deterministický konečný automat a ignorovat zásobník. (nechat tam Z_0). □

Lemma

Jazyk $L_{w_cw^r}$ je přijímaný DPDA ale není regulární.

Důkaz neregularity z pumping lemmatu na slovo $0^n c 0^n$.

Example 9.12

Jazyk $L_{abb} = \{a^i b^i \mid i \in \mathbb{N}\} \cup \{a^i b^{2i} \mid i \in \mathbb{N}\}$ je bezkontextový, ale není přijímaný žádným deterministickým zásobníkovým automatem.

Proof.

- SPOREM: Předokládejme, že existuje deterministický PDA M přijímající jazyk L_{abb} .
- Vytvořme dvě kopie, M_1 a M_2 , odpovídající si uzly budeme nazývat sourozenci.
- Zkonstruujeme nový automat:
 - Počátečním stavem bude počáteční stav M_1
 - koncovými stavy budou koncové stavy M_2
 - přechody z koncových stavů M_1 $\delta(p, b, X) = (q, X)$
 - přesměrujeme do sourozenců q v M_2 a přejmenujeme b na c
 - v automatu M_2 hrany označené b přeznačíme na c .
- Výsledný automat přijímá $\{a^i b^i c^i \mid i \in \mathbb{N}\}$ protože
 - M je deterministický, nemá jinou cestu, tj. i ve slově $a^i b^{2i}$ musel jít začátek stejně a pak číst b^i , nyní c^i ,
- o $\{a^i b^i c^i \mid i \in \mathbb{N}\}$ víme, že není bezkontextový, tj. deterministický M nemůže existovat.

Bezprefixové jazyky

Definition 9.7 (bezprefixové jazyky)

Říkáme, že jazyk $L \subset \Sigma^*$ je **bezprefixový** pokud neexistují slova $u, v \in L$ a $z \in \Sigma^+$ tak, že $u = vz$. Tj. pro žádná slova jazyka u, v není vlastní **v prefix u** .

Example 9.13

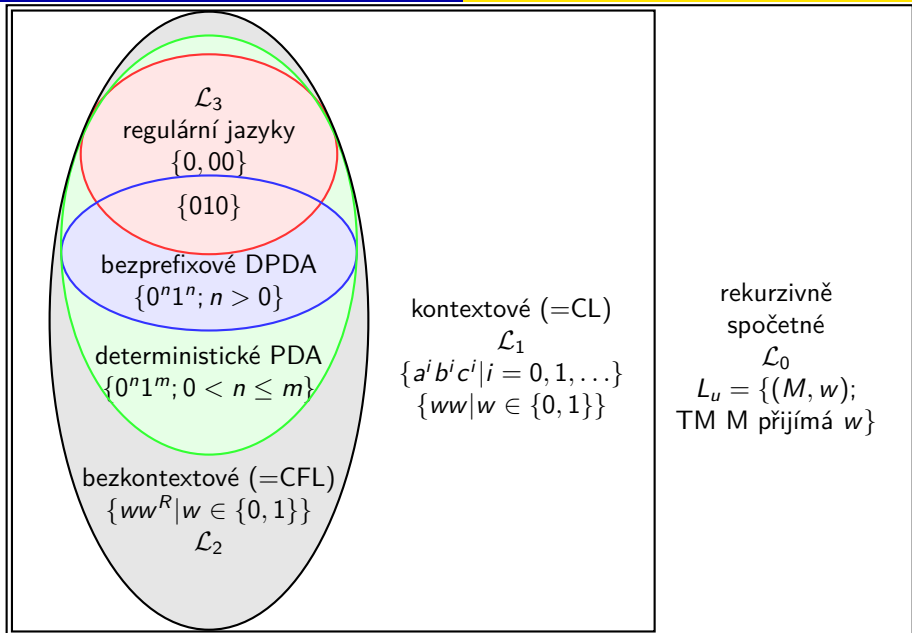
- Jazyk L_{wcwr} je bezprefixový.
- Jazyk $L = \{0\}^*$ není bezprefixový.

Theorem 9.3 ($L \in N(P_{DPDA})$ právě když L bezprefixový a $L \in L(P'_{DPDA})$)

Jazyk L je $N(P)$ pro nějaký DPDA P právě když L je bezprefixový a L je $L(P')$ pro nějaký DPDA P' .

Proof.

- \Rightarrow Prefix přijmeme prázdným zásobníkem, pro prázdný zásobník neexistuje instrukce, tj. žádné prodloužení není v $N(P)$.
- \Leftarrow Převod P^I na P nepřidá nedeterminismus (první koncový \rightarrow smaž hrany z něj, přijmi).



$L_d = \{w; \text{TM s kódem } w \text{ nepřijímá vstup } w\}$

Uzávěrové vlastnosti

Theorem 10.1 (CFL uzavřené na sjednocení, konkatenaci, iteraci, reverzi)

CFL jsou uzavřené na sjednocení, konkatenaci, iteraci (*), pozitivní iteraci (+), zrcadlový obraz w^R .

Proof:

- Sjednocení:
 - pokud $V_1 \cap V_2 \neq \emptyset$ přejmenujeme neterminály,
 - přidáme nový symbol S_{new} a pravidlo $S_{new} \rightarrow S_1 | S_2$
- zřetězení (=konkatenace) $L_1.L_2$

$$S_{new} \rightarrow S_1 S_2 \text{ (pro } V_1 \cap V_2 = \emptyset, \text{ jinak přejmenujeme)}$$
- iterace $L^* = \bigcup_{i \geq 0} L^i$

$$S_{new} \rightarrow S S_{new} | \epsilon$$
- pozitivní iterace $L^+ = \bigcup_{i \geq 1} L^i$

$$S_{new} \rightarrow S S_{new} | S$$
- zrcadlový obraz $L^R = \{w^R | w \in L\}$

$$X \rightarrow \omega^R \text{ obrátíme pravou stranu pravidel.}$$

□

Průnik bezkontextových jazyků

Example 10.1 (!CFL nejsou uzavřené na průnik)

- Jazyk $L = \{0^n 1^n 2^n \mid n \geq 1\} = \{0^n 1^n 2^i \mid n, i \geq 1\} \cap \{0^i 1^n 2^n \mid n, i \geq 1\}$ není CFL, i když oba členy průniku jsou bezkontextové, dokonce deterministické bezkontextové.

| | |
|------------------------------------|--|
| $\{0^n 1^n 2^i \mid n, i \geq 1\}$ | $\{S \rightarrow AC, A \rightarrow 0A1 \mid 01, C \rightarrow 2C \mid 2\}$ |
| $\{0^i 1^n 2^n \mid n, i \geq 1\}$ | $\{S \rightarrow AB, A \rightarrow 0A \mid 0, B \rightarrow 1B2 \mid 12\}$ |
- průnik není CFL z pumping lemmatu.

paralelní běh dvou zásobníkových automatů

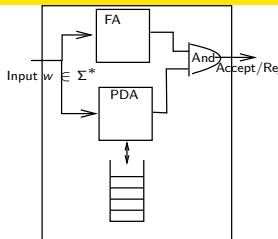
- řídící jednotky umíme spojit (viz konečné automaty)
- čtení umíme spojit (jeden automat může čekat)
- bohužel dva zásobníky nelze obecně spojit do jednoho

dva neomezené zásobníky = Turingův stroj
 = rekurzivně spočetné jazyky \mathcal{L}_0

Průnik bezkontextového a regulárního jazyka

Theorem 10.2 (CFL i DCFL jsou uzavřené na průnik s regulárním jazykem)

- Mějme L bezkontextový jazyk a R regulární jazyk. Pak $L \cap R$ je bezkontextový jazyk.
- Mějme L deterministický CFL a R regulární jazyk. Pak $L \cap R$ je deterministický CFL.



Proof:

- zásobníkový a konečný automat můžeme spojit
 - FA $A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$
 - PDA přijímání stavem $M_1 = (Q_2, \Sigma, \Gamma, \delta_2, q_2, Z_0, F_2)$
- nový automat $M = (Q_1 \times Q_2, \Sigma, \Gamma, \delta, (q_1, q_2), Z_0, F_1 \times F_2)$
 - $((r, s), \alpha) \in \delta((p, q), a, Z)$ právě když

| | | | | |
|--------------------|--|------|-------------------------------------|-------------------------|
| $a \neq \epsilon:$ | $r = \delta_1(p, a)$ | $\&$ | $(s, \alpha) \in \delta_2(q, a, Z)$ | ... automaty čtou vstup |
| $a = \epsilon:$ | $(s, \alpha) \in \delta_2(q, \epsilon, Z)$ | | | PDA mění zásobník |
| | $r = p$ | | | FA stojí |
- zřejmě $L(M) = L(A_1) \cap L(M_2)$

Substituce a homomorfismus

- Opakování definice:

Definition ((5.1,5.2)substituce, homomorfismus, inverzní homomorfismus)

Mějme jazyk L nad abecedou Σ .

Substituce σ ; $\forall a \in \Sigma : \sigma(a) = L_a$ jazyk abecedy Σ_a , tj. $\sigma(a) \subseteq \Sigma_a^*$
převádí slova na jazyky:

- $\sigma(\epsilon) = \{\epsilon\}$,
- $\sigma(a_1 \dots a_n) = \sigma(a_1) \dots \sigma(a_n)$ (konkatenace), tj. $\sigma : \Sigma^* \rightarrow P((\bigcup_{a \in \Sigma} \Sigma_a)^*)$
- $\sigma(L) = \bigcup_{w \in L} \sigma(w)$.

homomorfismus h , $\forall a \in \Sigma : h(a) \in \Sigma_a^*$

převádí slova na slova

- $h(\epsilon) = \epsilon$,
- $h(a_1 \dots a_n) = h(a_1) \dots h(a_n)$ (konkatenace) tj. $h : \Sigma^* \rightarrow (\bigcup_{a \in \Sigma} \Sigma_a)^*$
- $h(L) = \{h(w) \mid w \in L\}$.

Inverzní homomorfismus převádí slova zpět

- $h^{-1}(L) = \{w \mid h(w) \in L\}$.

Příklad: Substitute

Example 10.2

Mějme gramatiku $G = (\{E\}, \{a, +, (,)\}, \{E \rightarrow E + E | (E) | a\}, E)$. Mějme substituci:

- $\sigma(a) = L(G_a)$, $G_a = (\{I\}, \{a, b, 0, 1\}, \{I \rightarrow I0 | I1 | Ia | Ib | a | b\}, I)$,
- $\sigma(+)$ = $\{-, *, :, \text{div}, \text{mod}\}$,
- $\sigma(($ = $\{($,
- $\sigma())$ = $\{)\}$.

- $(a + a) + a \in L(G)$
- v $\sigma(+)$ chybí $+$ pro ukázkou, že $(a + a) + a \notin \sigma(L(G))$.
- $(a001 - bba) * b1 \in \sigma((a + a) + a) \subset \sigma(L(G))$

Co se stane, když změníme definici:

- $\sigma(($ = $\{(, [$,
- $\sigma())$ = $\{),]\}$?

Příklad: Homomorfismus

Example 10.3

Mějme gramatiku

$G = (\{E\}, \{a, +, (,)\}, \{E \rightarrow E + E | (E) | a\}, E)$. Mějme homomorfismus:

- $h(a) = \epsilon$
 - $h(+)$ = ϵ ,
 - $h(($) = *left*,
 - $h())$ = *right*.
-
- $h((a + a) + a) = \textit{leftright}$,
 - $h^{-1}(\textit{leftright}) \ni (a + +)a$.

Example 10.4

Mějme gramatiku

$G = (\{E\}, \{a, +, (,)\}, \{E \rightarrow a + E | (E) | a\}, E)$. Mějme homomorfismus:

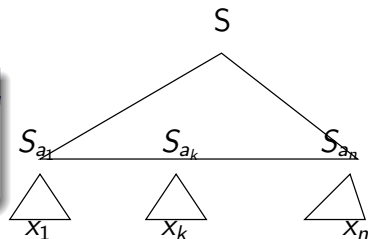
- $h_2(a) = a$
- $h_2(+)$ = $+$,
- $h_2(($) = ϵ ,
- $h_2())$ = ϵ .

- 1 Je jazyk $L(G)$ regulární?
- 2 Je jazyk $h(L(G))$ regulární?
- 3 Je jazyk $h^{-1}(h(L(G)))$ regulární?
- 4 Je $h^{-1}(h(L(G))) = L(G)$?

Uzávěrové vlastnosti bezkontextových jazyků

Theorem 10.3 (CFL jsou uzavřené na substituci)

Mějme CFL jazyk L nad Σ a substituci σ na Σ takovou, že $\sigma(a)$ je CFL pro každé $a \in \Sigma$. Pak je $\sigma(L)$ bezkontextový (CFL).



Proof:

- Idea: listy v derivačním stromu generují další stromy.
- Přejmenujeme neterminály na jednoznačné všude v $G = (V, \Sigma, P, S)$, $G_a = (V_a, T_a, P_a, S_a)$, $a \in \Sigma$.
- Vytvoříme novou gramatiku $G = (V', T', P', S)$ pro $\sigma(L)$:
 - $V' = V \cup \bigcup_{a \in \Sigma} V_a$
 - $T' = \bigcup_{a \in \Sigma} T_a$
 - $P' = \bigcup_{a \in \Sigma} P_a \cup \{p \in P \text{ kde všechna } a \in \Sigma \text{ nahradíme } S_a\}$.

G' generuje jazyk $\sigma(L)$. □

Substitute bezkontextových jazyků

Example 10.5 (substitute)

$$\begin{aligned}
 L &= \{a^i b^j \mid 0 \leq i \leq j\} & S &\rightarrow aSb \mid Sb \mid \epsilon \\
 \sigma(a) &= L_1 = \{c^i d^i \mid i \geq 0\} & S_1 &\rightarrow cS_1 d \mid \epsilon \\
 \sigma(b) &= L_2 = \{c^i \mid i \geq 0\} & S_2 &\rightarrow cS_2 \mid \epsilon \\
 \sigma(L) & & S &\rightarrow S_1 S_2 \mid S S_2 \mid \epsilon, S_1 \rightarrow cS_1 d \mid \epsilon, S_2 \rightarrow cS_2 \mid \epsilon
 \end{aligned}$$

Theorem 10.4 (homomorfismus)

Bezkontextové jazyky jsou uzavřeny na homomorfismus.

Proof:

- Příímý důsledek předchozí věty.
- Terminál a v derivačním stromě nahradím slovem $h(a)$. □

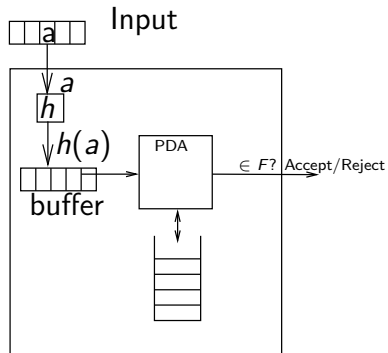
CFL jsou uzavřené na inverzní homomorfismus

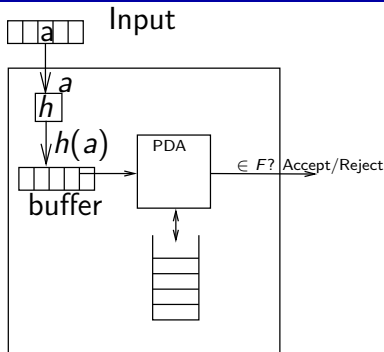
Theorem 10.5 (CFL jsou uzavřené na inverzní homomorfismus)

Mějme CFL jazyk L a homomorfismus h . Pak $h^{-1}(L)$ je bezkontextový jazyk. Je-li L deterministický CFL, je i $h^{-1}(L)$ deterministický CFL.

Idea

- přečteme písmeno a a do vnitřního bufferu dáme $h(a)$
- simulujeme výpočet M , kdy vstup bereme z bufferu
- po vyprázdnění bufferu načteme další písmeno ze vstupu
- slovo je přijato, když je buffer prázdný a M je v koncovém stavu
- ! buffer je konečný, můžeme ho tedy modelovat ve stavu





Proof:

- pro L máme PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ (koncovým stavem)
- $h : T \rightarrow \Sigma^*$
- definujeme PDA $M' = (Q', T, \Gamma, \delta', [q_0, \epsilon], Z_0, F \times \{\epsilon\})$ kde

$$Q' = \{[q, u] \mid q \in Q, u \in \Sigma^*, \exists(a \in T) \exists(v \in \Sigma^*) h(a) = vu\} \quad u \text{ je buffer}$$

$$\delta'([q, \epsilon], a, Z) = \{([q, h(a)], Z)\} \quad \text{naplňuje buffer}$$

$$\delta'([q, u], \epsilon, Z) = \{([p, u], \gamma) \mid (p, \gamma) \in \delta(q, \epsilon, Z)\}$$

$$\cup \{([p, v], \gamma) \mid (p, \gamma) \in \delta(q, x, Z), u = xv\} \quad \text{čte buffer, } x \in \Sigma$$

Pro deterministický PDA M je i M' deterministický. □

Použití uzavřenosti průniku CFL a RL

Example 10.6

Jazyk $L = \{0^i 1^j 2^k 3^l \mid i = 0 \vee j = k = l\}$ není bezkontextový.

Proof: Důkaz sporem:

- Necht L je bezkontextový jazyk
- $L_1 = \{01^j 2^k 3^l \mid j, k, l \geq 0\}$ je regulární jazyk
 - $\{S \rightarrow 0B, B \rightarrow 1B \mid C, C \rightarrow 2C \mid D, D \rightarrow 3D \mid \epsilon\}$
- $L \cap L_1 = \{01^i 2^i 3^i \mid i \geq 0\}$ není bezkontextový \Rightarrow SPOR s uzavřeností na průnik s regulárním jazykem. □

L je kontextový jazyk

$$S \rightarrow \epsilon \mid 0 \mid 0A \mid B_1 \mid C_1 \mid D_1$$

$$B_1 \rightarrow 1 \mid 1B_1 \mid C_1, C_1 \rightarrow 2 \mid 2C_1 \mid D_1, D_1 \rightarrow 3 \mid 3D_1$$

$$A \rightarrow 0 \mid 0A \mid P$$

$$P \rightarrow 1PCD \mid 1CD$$

$$DC \rightarrow CD \text{ přepíšeme } \{DC \rightarrow XC, XC \rightarrow XY, XY \rightarrow CY, CY \rightarrow CD\}$$

$$1C \rightarrow 12, 2C \rightarrow 22, 2D \rightarrow 23, 3D \rightarrow 33.$$

Rozdíl a doplněk

Theorem 10.6 (Odečtení regulárního jazyka)

Mějme bezkontextový jazyk L a regulární jazyk R . Pak:

- $L - R$ je CFL.

Proof.

$L - R = L \cap \overline{R}$, \overline{R} je regulární. □

Theorem 10.7 (CFL **nej**sou uzavřené na doplněk ani rozdíl)

Třída bezkontextových jazyků není uzavřená na doplněk ani na rozdíl.

CFL **nej**sou uzavřené na doplněk ani rozdíl.

Mějme bezkontextové jazyky L, L_1, L_2 , regulární jazyk R . Pak:

- \overline{L} nemusí být CFL. $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$.
- $L_1 - L_2$ nemusí být CFL. $\Sigma^* - L$ není vždy CFL.

□

Uzávěrové vlastnosti deterministických CFL

- Rozumné programovací jazyky jsou deterministické CFL.
- Deterministické bezkontextové jazyky
 - nejsou uzavřené na průnik
 - jsou uzavřené na průnik s regulárním jazykem
 - jsou uzavřené na inverzní homomorfismus.

Lemma

Doplněk deterministického CFL je opět deterministický CFL.

Proof:

- idea: prohodíme koncové a nekoncové stavy
- nedefinované kroky ošetříme 'podložkou' na zásobníku
- cyklus odhalíme pomocí čítače
- až po přečtení slova prochází koncové a nekoncové stavy stačí si pamatovat, zda prošel koncovým stavem.



Neuzavřenost deterministických CFL

Example 10.7 (DCFL nejsou uzavřené na sjednocení)

Jazyk $L = \{a^i b^j c^k \mid i \neq j \vee j \neq k \vee i \neq k\}$ je CFL, ale není DCFL.

Proof.

Vzhledem k uzavřenosti DCFL na doplněk by byl DCFL i

$\bar{L} \cap a^* b^* c^* = \{a^i b^j c^k \mid i = j = k\}$, o kterém víme, že není CFL (pumping lemma) □

Example 10.8 (DCFL nejsou uzavřené na homomorfismus)

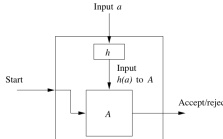
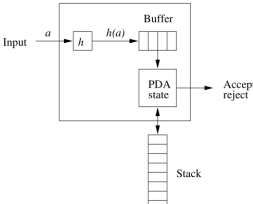
Jazyky $L_1 = \{a^i b^j c^k \mid i \neq j\}$, $L_2 = \{a^i b^j c^k \mid j \neq k\}$, $L_3 = \{a^i b^j c^k \mid i \neq k\}$ jsou deterministické bezkontextové.

- Jazyk $0L_1 \cup 1L_2 \cup 2L_3$ je deterministický bezkontextový
- Jazyk $L_1 \cup L_2 \cup L_3$ není deterministický bezkontextový
 položme $h(0) = \epsilon$, $h(1) = \epsilon$, $h(2) = \epsilon$
 $h(x) = x$ pro ostatní symboly
 - $h(0L_1 \cup 1L_2 \cup 2L_3) = L_1 \cup L_2 \cup L_3$,
 - doplněk $L_1 \cup L_2 \cup L_3 \cap a^* b^* c^* = \{a^i b^j c^k \mid i = j = k\}$.

Uzávěrové vlastnosti v kostce

| jazyk | regulární (RL) | bezkontextové | deterministické CFL |
|---------------|----------------|---------------|---------------------|
| sjednocení | ANO | ANO | NE |
| průnik | ANO | NE | NE |
| \cap s RL | ANO | ANO | ANO |
| doplňěk | ANO | NE | ANO |
| homomorfizmus | ANO | ANO | NE |
| inverzní hom. | ANO | ANO | ANO |

Uzávěrové vlastnosti v kostce

| jazyk | regulární (RL) | bezkontextové | deterministické CFL |
|----------------|---|--|---|
| sjednocení | $F = F_1 \times Q_2 \cup Q_1 \times F_2$ | $S \rightarrow S_1 S_2$ | $A \cap B = \overline{\overline{A} \cup \overline{B}}$ |
| průnik | $F = F_1 \times F_2$ | $L = \{0^n 1^n 2^n n \geq 1\} = \left\{ \begin{array}{l} \{0^n 1^n 2^i n, i \geq 1\} \\ \cap \{0^i 1^n 2^n n, i \geq 1\} \end{array} \right\}$ | |
| \cap s RL | $F = F_1 \times F_2$ | $F = F_1 \times F_2$ | $F = F_1 \times F_2$ |
| doplňěk homom. | $F = Q_1 - F_1, \delta \text{ tot.}$ Kleene + RegExp + uz. | $A \cap B = \overline{\overline{A} \cup \overline{B}}$ $a \text{ nahrad' } S_a$ | $F = Q_1 - F_1, Z_0, \text{ cykly, tot.}$ $h(0) = h(1) = 0 \text{ cca. } \cup$ |
| inverzní hom. |  |  | |

Dyckovy jazyky

Definition 10.1 (Dyckův jazyk)

Dyckův jazyk D_n je definován nad abecedou $Z_n = \{a_1, a_1^|, \dots, a_n, a_n^|\}$ následující gramatikou: $S \rightarrow \epsilon | SS | a_1 Sa_1^| \dots | a_n Sa_n^|$.

Úvodní pozorování:

- jedná se zřejmě o jazyk bezkontextový
- Dyckův jazyk D_n popisuje správně uzávorkované výrazy s n druhy závorek
- tímto jazykem lze popisovat výpočty libovolného zásobníkového automatu
- pomocí Dyckova jazyka lze popsat libovolný bezkontextový jazyk.

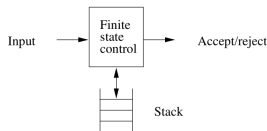
$$L = h(D \cap R)$$

homomorfismus;
čistí pomocné sym-
boly

Dyckův jazyk; vy-
bírá pouze korektní
výpočty

regulární jazyk;
popisuje všechny
stroky výpočtu

Jak charakterizovat bezkontextové jazyky?



- Pokud do zásobníku pouze přidáváme
potom si stačí pamatovat poslední symbol
- stačí konečná paměť → konečný automat.
- potřebujeme ze zásobníku také odebírat (čtení symbolu)
takový proces nelze zaznamenat v konečné struktuře
- přidávání a odebírání není zcela libovolné
jedná se o zásobník, tj. LIFO (last in, first out) strukturu
- roztáhněme si výpočet se zásobníkem do lineární struktury
 X symbol přidán do zásobníku
 X^{-1} symbol odebrán do zásobníku
- přidávaný a odebíraný symbol tvoří pár $\underbrace{ZZ^{-1}} \underbrace{BAA^{-1}CC^{-1}B^{-1}}$
který se v celé posloupnosti chová jako závorka

Theorem 10.8 (Dyckovy jazyky)

Pro každý bezkontextový jazyk L existuje regulární jazyk R tak, že $L = h(D \cap R)$ pro vhodný Dyckův jazyk D a homomorfismus h .

Proof:

- máme PDA přijímající L prázdným zásobníkem
- převedeme na instrukce tvaru $\delta(q, a, Z) \in (p, w), |w| \leq 2$
 - delší psaní na zásobník rozdělíme zavedením nových stavů
- necht' R^l obsahuje všechny výrazy
 - $q^{-1}aa^{-1}Z^{-1}BAp$ pro instrukci $\delta(q, a, Z) \ni (p, AB)$
 - podobně pro instrukce $\delta(q, a, Z) \in (p, A), \delta(q, a, Z) \in (p, \epsilon)$
 - je-li $a = \epsilon$, potom dvojici aa^{-1} nezařazujeme
- definujeme R takto: $Z_0q_0(R^l)^*Q^{-1}$
- Dyckův jazyk je definován nad abecedou $\Sigma \cup \Sigma^{-1} \cup Q \cup Q^{-1} \cup \Gamma \cup \Gamma^{-1}$
- $D \cap Z_0q_0(R^l)^*Q^{-1}$ popisuje korektní výpočty

$$\underbrace{Z_0 \overbrace{q_0q_0^{-1}} \overbrace{aa^{-1}} Z_0^{-1}} \underbrace{B \overbrace{A \overbrace{pp^{-1}} \overbrace{bb^{-1}} A^{-1}} \overbrace{qq^{-1}} \overbrace{cc^{-1}} B^{-1}} \overbrace{rr^{-1}}$$

- homomorfismus h vydělí přečtené slovo, tj.

$h(a) = a$ pro vstupní (čtené) symboly

$h(y) = \epsilon$ pro ostatní.

Přehled kapitol

- 1 Úvod, Iterační lemma pro reg. jazyky
- 2 Redukovaný DFA a ekvivalence automatů, stavů
- 3 Nedeterministické ϵ -NFA, Operace zachovávající regularitu
- 4 Regulární výrazy, Kleeneova věta, Substitute, Homomorfismus
- 5 Dvousměrné FA, Mealy a Moore stroje
- 6 Gramatiky, Chomského hierarchie, víceznačnost
- 7 Chomského NF, Pumping Lemma pro CFL, CYK – náležení do CFL
- 8 Zásobníkové automaty, Deterministické PDA
- 9 Uzávěrové vlastnosti, Dykovy jazyky
- 10 Turingův stroj, rozšíření
- 11 Lineárně omezené automaty, Univerzální TM, Diagonální jazyk
- 12 Nerozhodnutelné problémy, Postův korespondenční p.
- 13 Časová složitost