

1) $L_{\text{prime}} = \{w \in \Sigma^*: |w| \text{ je prvočíslo}\}$ nad abecedou $\Sigma = \{a, b\}$

Díkaz sporem:

Předpokládáme, že jazyk $L_{\text{prime}} \in \mathcal{L}_2$, pak dle Pumping lemma pro Bkj:

- Uvažme libovolné $k > 0$ a zvolme slovo $z = a^r$, kde $r > k$ a r je prvočíslo, tedy $|z|$ je prvočíslo a $r > 1$.
- Z P. lemma platí, že $z = uvwxy \wedge vx \neq \epsilon$.
- Pokud zvolíme $i = r+1$, tak platí $|uv^rw^x y| = r + |vx| \cdot k$, což je dělitelné r , ale nerovná se r , protože $|vx| > 0$.
- Z toho plyne, že se nejedná o prvočíslo, docházíme ke sporu, proto $L_{\text{prime}} \notin \mathcal{L}_2$

2)

Hypotéza: L_{BKG} není rekurzivně výčislitelný (není ani částečně rozhodnutelný),
což dokážeme redukcí z CO-HP

Dоказ: $CO-HP \leq L_{BKG}$, redukční funkce Ψ má signaturu:

$\Psi: \{0,1,\#\}^* \rightarrow \{0,1,\#\}^*$. Pokud vstup redukční funkce
nebude ve tvaru $\langle M \rangle \# \langle w \rangle$, pak funkce Ψ vrátí řetězec
 $\langle M' \rangle \# \langle G' \rangle$, kde $\langle M' \rangle$ je kód Turingova stroje (TS)
nad abecedou $\{0,1\}$, přičemž $L(M') = \emptyset$ a $\langle G' \rangle$
je kód BKG, kde $G' = (\{\$\}, \{0,1\}, \{S \rightarrow 01\}, S)$.
Z toho plyne, že $\langle M' \rangle \# \langle G' \rangle \notin L_{BKG}$, neboť stroj
 M' přijímá prázdny jazyk, ale G' negeneruje prázdny jazyk,
tedy $L(G) \neq L(M)$.

V případě, že vstup redukční funkce bude mít tvar $\langle M \rangle \# \langle w \rangle$,
pak jej funkce zobrazí na tento výstup:

$$\Psi(\langle M \rangle \# \langle w \rangle) = \langle M' \rangle \# \langle G' \rangle,$$

kde $\langle G' \rangle$ je kód BKG, pro kterou platí $L(G') = \Sigma^*$
 M' je kód TS, který pracuje následovně:

1. Zapomnaje si délku vstupu w !

2. Provede nejvýše $|w|$ kroků simulace M nad w ,
přičemž kódy $\langle M \rangle$ a $\langle w \rangle$ jsou zakódovány ve starověm řízení M'

3. Pokud simulace M nad w skončila v rámci $|w|$ kroků
nebo krok M' odmítne

4. Pokud simulace M nad w neskončila ani do $|w|$ kroků,
stroj M' akceptuje.

Zavedeme si konstantu c , která bude značit počet provedených kroků simulace M nad w , pakž je tento počet konečný.

Pro stroj M' musí platit:

$$\langle M \rangle \# \langle w \rangle \in \text{co-HP} \Rightarrow L(M') = \Sigma^* \Rightarrow \langle M' \rangle \# \langle G \rangle \in L_{BKG}$$

$$\langle M \rangle \# \langle w \rangle \notin \text{co-HP} \Rightarrow L(M') = \{w \in \Sigma^* \mid |w| \leq c\} \Rightarrow \langle M' \rangle \# \langle G \rangle \notin L_{BKG}$$

$$\langle M \rangle \# \langle w \rangle \in \text{co-HP} \iff \varphi(\langle M \rangle \# \langle w \rangle) \in L_{BKG}$$

Tato redukce je správná, protože φ je totální, rekursivně výpočitelná funkce zachovávající příslušnost. Víme, že co-HP není rekursivně výpočitelný jazyk, není ani L_{BKG} rekursivně výpočitelný jazyk. \square

4) Vstup: $G = (N, \Sigma, P, S)$, $a \in \Sigma$

Výstup: Množina aN na všech нетерминалов gramatiky G dle specifikace v zadání!

1. Zkonstrujeme množinu $N_t \in \{ A \in N \mid \exists w \in \Sigma^*: A \Rightarrow^* w \}$ pomocí algoritmu prvního bodu takto:

$$1. N_t^0 = \emptyset$$

$$2. N_t^i = \{ A \in N \mid (A \rightarrow \alpha) \in P \text{ a } \alpha \in (N_{t-1} \cup \Sigma)^* \}$$

3. Je-li $N_t^i = N_t^{i-1}$, pak $N_t = N_t^{i-1}$ a algoritmus skončí, jinak opakuj krok 2.

2. Zkonstrujeme množinu $aN = \{ A \in N \mid \exists w \in \Sigma^*: A \Rightarrow^* aw \}$ pomocí algoritmu prvního bodu následovně:

$$1. aN^0 = \emptyset$$

$$2. aN^i = \{ A \in N \mid \exists (\alpha \rightarrow \beta) \in P, \text{ přičemž } \alpha \in N_{t-1}^*, \beta \in (\Sigma \cup N_t)^* \}$$

$$3. aN^{i+1} = \{ A \in N \mid \exists (\alpha \rightarrow \beta) \in P, \text{ kde } \alpha \in (aN^i), \beta \in (\Sigma \cup N_t)^* \}$$

4. Pokud $aN^{i+1} = aN^i$, pak $aN = aN^i$ a algoritmus končí, jinak opakuj krok 3

3. Zkonstrujeme množinu $Na = \{ A \in N \mid \exists w \in \Sigma^*: A \Rightarrow^* wa \}$ pomocí algoritmumu prvního bodu následovně:

$$1. Na^0 = \emptyset$$

$$2. Na^i = \{ A \in N \mid \exists (\alpha \rightarrow \beta) \in P, \text{ přičemž } \beta \in N_{t-1}^*, \alpha \in (\Sigma \cup N_t)^* \}$$

$$3. Na^{i+1} = \{ A \in N \mid \exists (\alpha \rightarrow \beta) \in P, \text{ kde } \alpha \in (Na^i), \beta \in (\Sigma \cup N_t)^* \}$$

4. Pokud $Na^{i+1} = Na^i$, pak $Na = Na^i$ a algoritmus končí, jinak opakuj krok 3.

4. zavedeme relaci ξ následovně: $A \xi B \Leftrightarrow A, B \in N \wedge \exists (A \rightarrow \alpha B \beta) \in P : \alpha, \beta \in (\Sigma \cup N_t)^*$
5. zavedeme relaci ξ_{an} následovně: $A \xi_{an} B \Leftrightarrow A, B \in N \wedge \exists (A \rightarrow \alpha B) \in P :$
 $(\alpha \in (aN \cup \{\alpha\}) (\Sigma \cup N_t)^*$
6. zavedeme relaci ξ_{Na} následovně: $A \xi_{Na} B \Leftrightarrow A, B \in N \wedge \exists (A \rightarrow B \alpha) \in P :$
 $\alpha \in (\Sigma \cup N_t)^* (Na \cup \{\alpha\})$
7. spojene reflexivní transitivity uzávěr ξ^* relace ξ .
8. zkonstruujeme množinu $aNa = \sum_{\epsilon N_t} \{ A \in N \mid \exists B, C, D : A \xi_{an} B \wedge B \xi^* C \wedge C \xi_{Na} D \}$
9. vrátíme aNa jako výstup.

Použití algoritmu:

$$aNa = \{ S, W \}$$