IB102 – úkol 1, příklad 1 – řešení **Odevzdání: 24. 9. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [**2 body**] Mějme abecedu $\Sigma = \{a, b\}$.

Pro každé z následujících slov a jazyků:
a) rozhodněte, zda se jedná o slovo nebo jazyk
b) pokud se jedná o slovo, napište jej jako posloupnost znaků abecedy a pokud o jazyk, napište jej jako množinu slov (tedy množinu posloupností znaků).

$a.(ab)^3$	slovo	$aababab$
$\{a\}.\{ab\}^3$	jazyk	$\{aababab\}$
$\{b\}.(\{aba\} \cup \{aa\}.\{\varepsilon\})^2.\{b\}^3$	jazyk	$\{babaababbb, babaaabbb, baaababbb, baaaabbb\}$
$\emptyset^*.\{\varepsilon\}$	jazyk	$\{\varepsilon\}$ (jazyk obsahující prázdné slovo)
$a.b.\varepsilon^6.a^2$	slovo	$abaa$
$\{a\}^* \setminus \{a^3, a^4\}^+$	jazyk	$\{\varepsilon, a, aa, aaaaa\}$ (resp. zkráceně $\{\varepsilon, a, a^2, a^5\}$)

IB102 – úkol 1, příklad 2 – řešení **Odevzdání: 24. 9. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [**2 body**] Nechť L je jazyk nad abecedou $\Sigma = \{a, b\}$ tvořený právě všemi slovy, která mají lichý počet znaků a zároveň se ve slovech nevyskytují 2 znaky a za sebou (tedy mezi každými dvěma výskyty znaku a je alespoň jeden znak b).
Zapište jazyk L pomocí jednoprvkových jazyků $\{a\}$ a $\{b\}$ s využitím konečného počtu operací sjednocení ($\cup$), průniku ($\cap$), rozdílu ($\setminus$), doplňku ($\mathsf{co}-$), zřetězení ($\cdot$), mocniny ($^2, ^3, \ldots$), iterace (*) a pozitivní iterace ($^+$).

Řešení: Pro zpřehlednění zápisu si nejprve definujeme následující jazyky. Jazyk L_1 bude obsahovat všechna slova s lichým počtem znaků, jazyk L_2 bude obsahovat všechna slova ve kterých se nevyskytují 2 znaky a za sebou (jakožto doplněk k jazyku, který obsahuje všechna slova, ve kterých se vyskytují alespoň 2 znaky a za sebou).

$$L_1 = (\{a\} \cup \{b\}) \cdot ((\{a\} \cup \{b\}) \cdot (\{a\} \cup \{b\}))^*$$
$$L_2 = \mathsf{co}-((\{a\} \cup \{b\})^* \cdot \{a\} \cdot \{a\} \cdot (\{a\} \cup \{b\})^*)$$

Řešení pak můžeme snadno zapsat jako průnik obou jazyků takto: $L = L_1 \cap L_2$, neboli

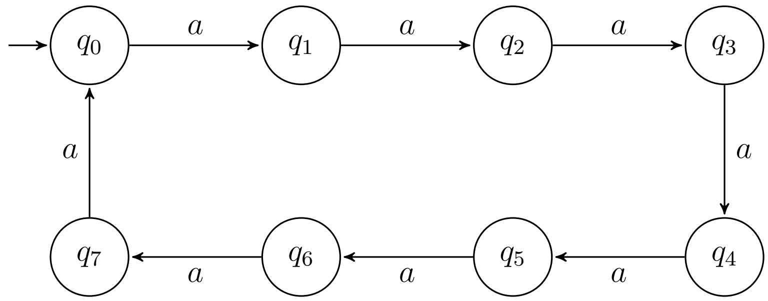
$$L = (\{a\} \cup \{b\}) \cdot ((\{a\} \cup \{b\}) \cdot (\{a\} \cup \{b\}))^* \ \cap \ \mathsf{co}-((\{a\} \cup \{b\})^* \cdot \{a\} \cdot \{a\} \cdot (\{a\} \cup \{b\})^*)$$

IB102 – úkol 1, příklad 1 – řešení **Odevzdání: 1. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [2 body]

a) [1 bod] Mějme následující deterministický konečný automat $\mathcal{A}$ nad abecedou $\{a, b\}$:



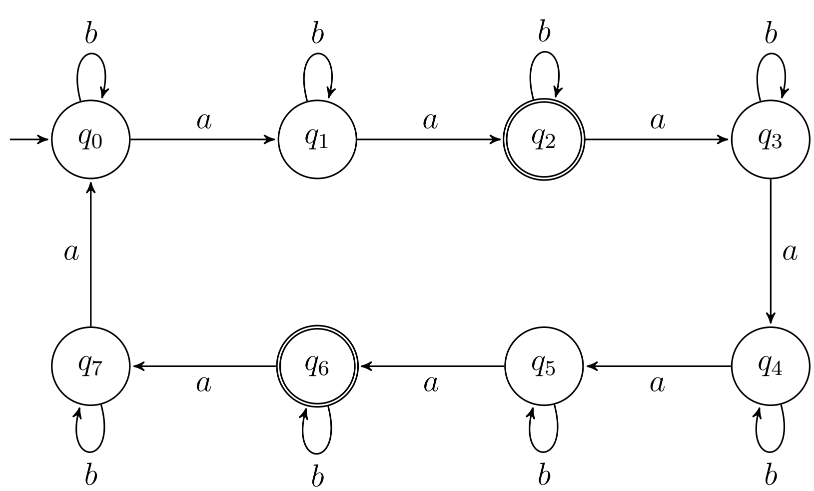
S použitím (libovolného, konečného počtu) níže uvedených *povolených úprav* (pouze a jen povolených úprav) změňte zadaný automat tak, aby akceptoval jazyk

$$L_a = \{w \in \{a, b\}^* \mid \#_a(w) \bmod 4 = 2\}.$$

Povolené úpravy jsou:

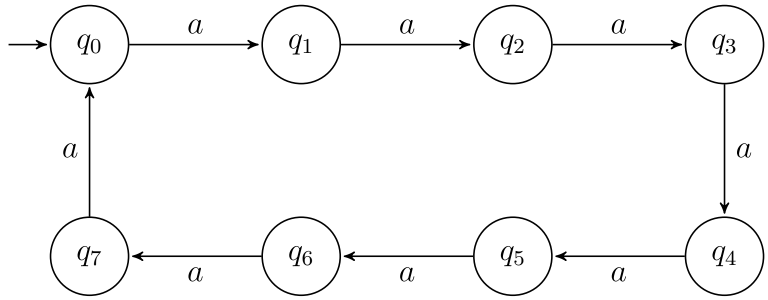
- přidávání libovolných přechodů pod b,
- označování akceptujících stavů,

Řešení: V automatu je potřeba označit stavy q_2 a q_6 jako akceptující, tak, aby automat akceptoval jazyk $\{w \in \{a\}^* \mid \#_a(w) \bmod 4 = 2\}$. Dále je potřeba zařídit, aby automat uměl akceptovat i slova obsahující znaky b. Přitom přítomnost nebo nepřítomnost znaků b ve slově neovlivňuje, jestli je slovo akceptováno nebo ne. Proto přechody pod b nemění stav automatu a jsou znázorněny jako smyčky.

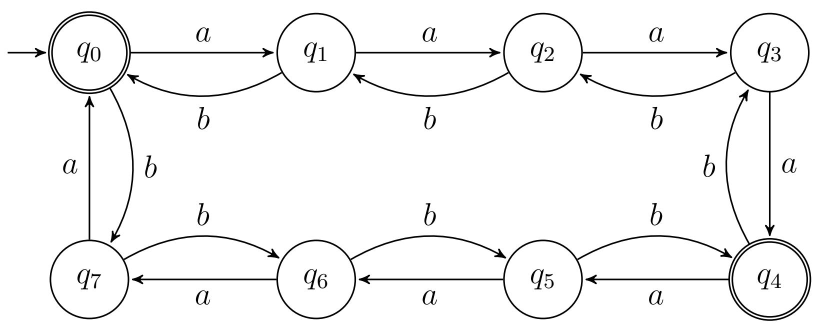


b) [1 bod] S použitím stejných *povolených úprav* jako u v zadání a) změňte následující automat nad abecedou $\{a, b\}$ tak, aby akceptoval jazyk

$$L_b = \{w \in \{a,b\}^* \mid (\#_a(w) + 7 \cdot \#_b(w)) \bmod 4 = 0\}.$$



Řešení: Zadání jazyka nám říká, že každé b se započítává stejně jako $7a$, a že zbytek takto váženého součtu po dělení 4 má být 0. Každý čtvrtý stav tedy musíme označit jako akceptující a přechody pod b z každého stavu vést tam, kam vede sekvence sedmi přechodů pod a.



IB102 – úkol 2, příklad 2 – řešení **Odevzdání: 1. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [**2 body**] Navrhněte regulární gramatiku, která generuje **všechna čísla v trojkové soustavě dělitelná pěti**. Pro jednoduchost předpokládejme, že korektně zapsané číslo je i to, které začíná nulou či nulami. Například čísla 12 nebo 101 mají být generována gramatikou, stejně jako čísla 012, 0012, 0101, atd. Naopak čísla $20, 21, 22$ ani 100 gramatikou být generována nemají. Nezapomeňte, že 0 je dělitelná pěti. Naopak prázdné slovo ϵ není číslem v trojkové soustavě.

TIP: Neterminály gramatiky si označte S_0, S_1, S_2, S_3, S_4.

Řešení: Pro řešení je potřeba si uvědomit, že pokud k zápisu čísla x v trojkové soustavě přiřetězíme symbol 0, znamená to vlastně vynásobit hodnotu čísla x třemi. Například pokud $x = 102$, tj. 11 v soustavě desítkové, pak 1020 odpovídá $3 \cdot 11 = 33$. To ale znamená, že třemi násobíme i zbytek po dělení pěti. Například, zbytek po dělení čísla $x = 102$ (tj. 11 v desítkové soustavě) je 1 a zbytek po dělení $x = 1020$ (tj. 33 v desítkové soustavě) je $3 \cdot 1 = 3$. Analogické úvahy vedou k tomuto řešení. Index neterminálu i vždy označuje zbytek po dělení 3 pro již vygenerovaný prefix terminálů α aktuální větné formy αS_i. Tedy například lze vygenerovat větnou formu $102S_1$ a v dalším kroce $1020S_3$, ale naopak nelze vygenerovat $102S_0$, ani $102S_2$.

$G = (\{S_0, S_1, S_2, S_3, S_4\}, \{0, 1, 2\}, P, S_0)$, kde

$$
\begin{aligned}
P = \{S_0 &\to 0S_0 \mid 1S_1 \mid 2S_2 \mid 0, \\
S_1 &\to 0S_3 \mid 1S_4 \mid 2S_0 \mid 2, \\
S_2 &\to 0S_1 \mid 1S_2 \mid 2S_3, \\
S_3 &\to 0S_4 \mid 1S_0 \mid 2S_1 \mid 1, \\
S_4 &\to 0S_2 \mid 1S_3 \mid 2S_4\}
\end{aligned}
$$

IB102 – úkol 3, příklad 1 – řešení **Odevzdání: 8. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [2 body] Uvažme jazyk

$L = \{w \in \{a, b\}^* \mid$ právě každý 2. symbol ve w je a anebo právě každý 3. symbol ve w je $a\}$.

(Tedy například slovo *babab* do tohoto jazyka patří, zatímco slovo *babaaa* nebo *babba* nikoliv.)

Rozhodněte, zda jazyk L je či není regulární a dokažte:

- Pokud L je regulární, uveďte regulární gramatiku generující anebo konečný deterministický automat akceptující daný jazyk. Gramatiku/automat zapište se všemi formálními náležitostmi.
- Pokud L není regulární, dokažte tuto skutečnost pomocí Lemmatu o vkládání (tzv. Pumping Lemma).

Řešení: Jazyk je regulární.

$$G = (\{S, A, B, C, D, E, F\}, \{a, b\}, P, S)$$

$$\begin{array}{rrcl} P = \{ & S & \to & b \mid bA \mid \varepsilon, \\ & A & \to & a \mid aB \mid b \mid bD, \\ & B & \to & b \mid bC, \\ & C & \to & a \mid aB, \\ & D & \to & a \mid aE, \\ & E & \to & b \mid bF, \\ & F & \to & b \mid bD\} \end{array}$$

IB102 – úkol 3, příklad 2 – řešení **Odevzdání: 8. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [2 body] Uvažme jazyk

$$L = \{w \in \{a,b\}^* \mid \text{právě každý } k\text{-tý symbol ve } w \text{ je } a, \\ \text{kde } k \text{ je libovolné, pevné, kladné přirozené číslo}\}$$

Tedy například slova $aaa, babab, bbabbabba, bbbba, bbbbbabbb$ do tohoto jazyka patří, zatímco slova $babba, bbbababbba$ nikoliv.

Rozhodněte, zda jazyk L je či není regulární a dokažte:

- Pokud L je regulární, uveďte regulární gramatiku generující anebo konečný deterministický automat akceptující daný jazyk. Gramatiku/automat zapište se všemi formálními náležitostmi.
- Pokud L není regulární, dokažte tuto skutečnost pomocí Lemmatu o vkládání (tzv. Pumping Lemma).

Řešení: L není regulární. Důkaz provedeme pomocí PL.

- Nechť n je libovolné přirozené číslo, dále pevné.
- Zvolíme slovo $w = b^n a b^n a$. Zřejmě $w \in L$ a $|w| \geq n$.
- Všechna možná rozdělení $w = xyz$, $|xy| \leq n$, $y \neq \varepsilon$ vypadají takto:

$$\begin{aligned} x &= b^k && k \geq 0 \\ y &= b^l && l > 0,\ k + l \leq n \\ z &= b^{n-k-l} a b^n a && \end{aligned}$$

- Zvolíme $i = 3$, slovo xy^iz pak vypadá takto:

$$xy^3z = b^k b^{3l} b^{n-k-l} a b^n a = b^{n+2l} a b^n a$$

  Zřejmě $b^{n+2l} a b^n a \notin L$, neboť $n + 2l \neq n$ (protože $l > 0$). Podle Lemmatu o vkládání tedy L není regulární.

IB102 – úkol 4, příklad 1 – řešení **Odevzdání: 15. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [3 body] Uvažte následující čtyři relace nad abecedou $\{a, b\}$:

$u\, R_1\, v \iff$ délka slova u i délka slova v je nejméně 10 (tj. $|u| \geq 10$ a $|v| \geq 10$)

$u\, R_2\, v \iff$ délka slova u a délka slova v se liší nejvýše o 1 (tj. $\big||u| - |v|\big| \leq 1$)

$u\, R_3\, v \iff \#_a(u) = \#_a(v)$

$u\, R_4\, v \iff \big(u = v\big) \vee \big(|u| = |v| = 2n$ pro nějaké $n \geq 1$ a zároveň n-tý znak slova u se shoduje s n-tým znakem slova $v\big)$.

a) Zjistěte, které z uvedených relací **nejsou reflexivní** a dokažte to o nich.

b) Zjistěte, které z uvedených relací **nejsou tranzitivní** a dokažte to o nich.

c) Zjistěte, které z uvedených relací **nejsou pravou kongruencí** a dokažte to o nich. U relací, které jsou pravou kongruencí určete jejich index.

Řešení

a) Z uvedených relací není reflexivní R_1. Uvažme například slovo $u = aa$. Neplatí, že $aa\ R_1\ aa$, neboť $|aa| < 10$. Neplatí tedy, že všechna slova nad danou abecedou jsou v relaci R_1 samy se sebou a relace R_1 tak není reflexivní.

b) Relace R_2 není tranzitivní. Jako protipříklad uvažme slova $u = aa$, $v = aaa$ a $w = aaaa$. Platí, že $u\ R_2\ v$ a $v\ R_2\ w$, neboť délka u a délka v se liší maximálně o 1, stejně jako délka v a délka w. Už však neplatí, že $u\ R_2\ w$, neboť délka u a w se liší o 2. Relace R_2 tedy není tranzitivní.

c) Relace R_1 není reflexivní, není tudíž ekvivalencí, a proto není ani pravou kongruencí.

Relace R_2 není tranzitivní, není tudíž ekvivalencí, a proto není ani pravou kongruencí.

Relace R_3 je pravou kongruencí s nekonečným indexem.

Relace R_4 je ekvivalencí, ale není pravou kongruencí. Uvažme například slova $u = aa$, $v = ab$ a slovo $w = a$. Platí, že $u\ R_4\ v$, neboť $|u| = |v| = 2$ a slova u a v se shodují na první pozici. Neplatí však, že $uw\ R_4\ vw$, neboť $uw \neq vw$ a zároveň $|uw| \neq 2n$ pro žádné n. Relace R_4 tedy není pravou kongruencí.

IB102 – úkol 4, příklad 2 – řešení **Odevzdání: 15. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [1 bod] Rozhodněte a zdůvodněte, zda platí toto tvrzení:

Existuje abeceda Σ a regulární jazyky L_1, L_2 nad Σ takové, že $\sim_{L_1} \neq \sim_{L_2}$ a zároveň $\sim_{L_1}$ je pravá kongruence taková, že L_2 je sjednocením některých tříd rozkladu Σ^* podle $\sim_{L_1}$.

(*Pozn.:* Pokud bude Vaše odpověď "ano, platí", uveďte zcela konkrétní příklad abecedy Σ, jazyků L_1, L_2 a prefixových ekvivalencí $\sim_{L_1}, \sim_{L_2}$. Pokud bude Vaše odpověď "ne, neplatí", pokuste se zdůvodnit, proč to neplatí pro žádnou abecedu Σ, jazyky L_1, L_2 a prefixové ekvivalence $\sim_{L_1}, \sim_{L_2}$).

Řešení Tvrzení platí.

Důkaz. Nechť

$$\Sigma = \{a\}, L_1 = \big\{w \;\big|\; |w| = 1 \bmod 4\big\}, L_2 = \big\{w \;\big|\; |w| = 1 \bmod 2\big\}$$

pak

$$\begin{aligned} u \sim_{L_1} v &\overset{def}{\iff} |u| \bmod 4 = |v| \bmod 4 \\ u \sim_{L_2} v &\overset{def}{\iff} |u| \bmod 2 = |v| \bmod 2 \end{aligned}$$

Třídy rozkladu Σ^* podle $\sim_{L_1}$ jsou:

$$\begin{aligned} X_0 &= \big\{w \;\big|\; |w| = 0 \bmod 4\big\} \\ X_1 &= \big\{w \;\big|\; |w| = 1 \bmod 4\big\} = L_1 \\ X_2 &= \big\{w \;\big|\; |w| = 2 \bmod 4\big\} \\ X_3 &= \big\{w \;\big|\; |w| = 3 \bmod 4\big\} \end{aligned}$$

Třídy rozkladu Σ^* podle $\sim_{L_2}$ jsou:

$$\begin{aligned} Y_0 &= \big\{w \;\big|\; |w| = 0 \bmod 2\big\} = X_0 \cup X_2 \\ Y_1 &= \big\{w \;\big|\; |w| = 1 \bmod 2\big\} = X_1 \cup X_3 = L_2 \end{aligned}$$

Tedy skutečně L_2 je sjednocením některých (X_1, X_3) tříd rozkladu Σ^* podle $\sim_{L_1}$. □

Bonus: [+1 bod] Uvažte zároveň, že platí následující dodatečná podmínka: jazyky L_1 a L_2 jsou nesrovnatelné (tedy $L_1 \nsubseteq L_2 \wedge L_2 \nsubseteq L_1$). Rozhodněte a zdůvodněte i tento případ.

Řešení Tvrzení platí. (Nelze však použít jazyky z první části, protože tam $L_1 \subset L_2$)

Důkaz. Nechť

$$\Sigma = \{a\}, L_1 = \left\{w \;\middle|\; |w| = 1 \bmod 4\right\}, L_2 = \left\{w \;\middle|\; |w| = 0 \bmod 2\right\}$$

Pak $\sim_{L_1}, \sim_{L_2}$ jsou definované stejně jako v předchozím příkladě a platí:

$$\begin{aligned} X_1 &= L_1 \\ Y_0 &= X_0 \cup X_2 = L_2 \end{aligned}$$

Tedy skutečně L_2 je sjednocením některých (X_0, X_2) tříd rozkladu Σ^* podle $\sim_{L_1}$ a zároveň $L_1 \nsubseteq L_2 \wedge L_2 \nsubseteq L_1$. $\square$

IB102 – úkol 5 **Odevzdání: 22. 10. 2011**

Vypracoval(a): UČO:
Skupina:

Bonus [5 bodů]
Uvažme abecedu Σ a slovo $u \in \Sigma^*$. Definujme množinu slov $\mathsf{nafoukni}(u)$ následovně:

$$\mathsf{nafoukni}(u) \stackrel{\text{def}}{=} \{u\} \qquad \text{pokud } u = \epsilon$$

$$\mathsf{nafoukni}(u) \stackrel{\text{def}}{=} \{\sigma \cdot \sigma' \mid \sigma' \in \Sigma\} \cdot \mathsf{nafoukni}(v) \qquad \text{pokud } u = \sigma \cdot v, \text{ kde } \sigma \in \Sigma \text{ a } v \in \Sigma^*.$$

Neformálně řečeno, množina $\mathsf{nafoukni}(u)$ obsahuje slova, která jsou dvakrát delší než slovo u a mají k-tý znak ze slova u vždy na $(2k-1)$-té pozici. Například pro $u = aab$ platí

$$\mathsf{nafoukni}(u) = \{aaaaba, aaaabb, aaabba, aaabbb, abaaba, abaabb, ababba, ababbb\}.$$

Uvažme jazyk $L \subseteq \Sigma^*$. Definujeme jazyk $\ddot{L}$ jakožto sjednocení všech množin $\mathsf{nafoukni}(u)$, kde $u \in L$.

$$\ddot{L} \stackrel{\text{def}}{=} \bigcup_{u \in L} \mathsf{nafoukni}(u)$$

Mějme zadaný konečný, deterministický, automat $\mathcal{A}$ s totální přechodovou funkcí, který akceptuje jazyk $L(\mathcal{A}) = L$. Najděte algoritmus, který zkonstruuje konečný, deterministický, totální, automat $\ddot{\mathcal{A}}$, který akceptuje jazyk $L(\ddot{\mathcal{A}}) = \ddot{L}$. Algoritmus naprogramujte. Na výběr máte jazyky **Python, C, C++, Java, Perl, Pascal a Haskell**.

Vstupem pro váš program je textový soubor obsahující několik automatů oddělených od sebe prázdným řádkem. Každý automat je zadán následujícím způsobem. Na prvním řádku je n, počet stavů automatu. Množina stavů automatu je $\{1, \ldots, n\}$. Na druhém řádku je m, počet znaků abecedy. Abeceda je dána prvními m znaky anglické abecedy, například pro $m = 4$ je abeceda $\{a, b, c, d\}$. Na třetím řádku je číslo iniciálního stavu. Na čtvrtém jsou čísla koncových stavů oddělená mezerou. Řádky 5 až $n + 4$ jsou řádky tabulkového zápisu automatu. Každý řádek tudíž obsahuje přesně m čísel z rozmezí 1 až n, která jsou od sebe oddělena mezerou.

Příklad vstupu:

```
5
3
1
5 1
2 1 3
3 2 4
4 3 5
5 4 1
1 5 2
```

Odpovídající automat:

	a	b	c
$\leftrightarrow 1$	2	1	3
2	3	2	4
3	4	3	5
4	5	4	1
$\leftarrow 5$	1	5	2

Výstupní formát je stejný jako vstupní. Odevzdávejte zdrojové kódy vašich programů. Hodnocení bude probíhat na základě výsledků testu. Je možné dostat částečné body za algoritmus i v případě, že kód neprojde testem – ale to jen pokud svůj kód vybavíte komentáři, ze kterých bude jasná myšlenka algoritmu.

V učebních materiálech naleznete také ukázkový vstupní soubor.

```
5
3
1
5 1
2 1 3
3 2 4
4 3 5
5 4 1
1 5 2

4
2
1
1 2
2 1
3 2
4 3
1 4

3
2
1
3
2 1
2 3
3 3
```

IB102 – úkol 6, příklad 1 – řešení **Odevzdání: 29. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [**2 body**] Nechť K je libovolný konečný jazyk nad abecedou $\Sigma = \{\text{a}\}$, L je libovolný regulární jazyk nad abecedou $\Sigma = \{\text{a}\}$ a N je libovolný neregulární jazyk nad abecedou $\Sigma = \{\text{a}\}$. Rozhodněte a zdůvodněte, zda následující tvrzení platí:

(a) K je neregulární $\Rightarrow$ $\mathsf{co}\text{-}(K)$ je regulární.

(b) $(\mathsf{co}\text{-}(K \cap N) \setminus L) \cap N$ není regulární.

Řešení:

(a) **Tvrzení platí**.

Důkaz: Dle zadání je jazyk K konečný. Každý konečný jazyk je zároveň regulární. Předpoklad implikace ("K je neregulární") je tedy nepravdivý, celá implikace je tím pádem platí.

(b) **Tvrzení neplatí**.

Důkaz: Pro důkaz najdeme protipříklad, tj. jazyky K, L, N, pro něž tvrzení neplatí. Uvažme následující jazyky:

$$K = \emptyset$$
$$L = \{a\}^*$$
$$N = \{a^i \mid i = 2^n, n \in \mathbb{N}\}.$$

Pak $K \cap N = \emptyset$, tedy $\mathsf{co}\text{-}(K \cap N)) = \{a\}^*$, a tudíž $\mathsf{co}\text{-}(K \cap N)) \setminus L = \emptyset$ a konečně $(\mathsf{co}\text{-}(K \cap N) \setminus L) \cap N = \emptyset$. Prázdný jazyk $\emptyset$ je ovšem regulární, nalezli jsme tedy protipříklad a tvrzení v obecnosti neplatí.

IB102 – úkol 6, příklad 2 – řešení **Odevzdání: 29. 10. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [**2 body**] Uvažme operaci *double*, která je pro jazyk L nad abecedou Σ definována následovně:

$$double(L) = \{ww \mid w \in L\}.$$

Dále uvažme operaci *codoco*, která je pro jazyk L nad abecedou Σ definována takto:

$$codoco(L) = co{-}(double(co{-}L)).$$

a) **Rozhodněte a zdůvodněte**, zda třída všech **co−konečných jazyků** (*co*−konečné jazyky jsou ty, jejichž komplement je konečný) je uzavřená na operaci *codoco*.

b) **Rozhodněte a zdůvodněte**, zda třída všech **regulárních** jazyků je uzavřená na operaci *codoco*.

Řešení

a) **Tvrzení platí**, třída *co*−konečných jazyků je uzavřená na operaci *codoco*.

Důkaz: Nechť L je libovolný *co*−konečný jazyk. Potom jeho doplňek ($co{-}L$) musí být konečný. Operace *double* nemění počet slov v jazyce, pouze zřetězuje stejná slova z jazyka za sebe, tedy výsledkem $double(co{-}L)$ je také konečný jazyk. Doplněk tohoto jazyka je tedy co-konečný jazyk $co{-}(double(co{-}L))$, který je výsledkem $codoco(L)$. Z toho plyne, že třída *co*−konečných jazyků je uzavřená na operaci *codoco*.

b) **Tvrzení neplatí**, třída regulárních jazyků není uzavřená na operaci *codoco*.
Důkaz: K tomu, abychom dokázali, že tvrzení neplatí, stačí najít libovolný regulární jazyk L, takový, že $codoco(L)$ není regulární.

Příkladem takového jazyka je:

$$L = co{-}\{a^n b \mid n \geq 0\}$$

Komplementem k tomuto jazyku je jazyk:

$$co{-}L = \{a^n b \mid n \geq 0\}$$

Po aplikování operace *double* dostáváme:

$$double(co{-}L) = \{a^n b a^n b \mid n \geq 0\}$$

Tento jazyk je ovšem neregulární (lze snadno dokázat pomocí Pumping lemmatu, resp. Myhill-Nerodovy věty). Komplement k neregulárnímu jazyku je opět neregulární (jak bylo dokázáno na cvičeních), tedy výsledek $codoco(L)$ je neregulární, z čehož plyne, že třída regulárních jazyků není uzavřená na operaci *codoco*.

IB102 – úkol 8, příklad 1 – řešení **Odevzdání: 12. 11. 2012**

Vypracoval: James Bond UČO: 007

Skupina: MI6

1. [**2 body**] Navrhněte algoritmus, který pro zadanou regulární gramatiku $\mathcal{G}$ rozhodne, zda tato gramatika generuje alespoň jedno slovo sudé délky (0 je sudé číslo).

Řešení může vypadat například takto:
Na vstupu máme regulární gramatiku $\mathcal{G} = (N, \Sigma, P, S)$, na výstupu odpověď na zadanou otázku ve formě $True$ nebo $False$.

Algorithm 1 Test na přítomnost slova sudé délky v regulární gramatice

if existuje pravidlo $S \to \varepsilon$ **then**
    return $True$ ▷ *0 je* sudé *číslo*
end if
$i \leftarrow 1$ ▷ *inicializace 1. počítadla*
$ii \leftarrow 1$ ▷ *inicializace 2. počítadla*
▷ *neterminály, které se dají přepsat na slovo* sudé *délky rovné nejvýše i*
$R_s^i \leftarrow \{X \in N \mid X \to \epsilon \in P\}$
▷ *neterminály, které se dají přepsat na slovo* liché *délky rovné nejvýše i*
$R_l^i \leftarrow \{X \in N \mid X \to a \in P$ pro nějaké $a \in \Sigma\}$
repeat
    $i \leftarrow ii$
    $ii \leftarrow i + 2$
    ▷ *neterminály, které se dají přepsat na slovo délky nejvýše i se jistě dají přepsat na slovo délky nejvýše i + 2*
    $R_s^{ii} = R_s^i$
    $R_l^{ii} = R_l^i$
    for all $X \in R_l^i$ **do** ▷ *pokud lze X přepsat na slovo* liché *délky rovné nejvýše i*
        for all $Z \to aX \in P$, kde $a \in \Sigma$ a $Z \in N$ **do** ▷ *pak neterminál Z*
            $R_s^{ii} \leftarrow R_s^{ii} \cup \{Z\}$ ▷ *lze přepsat na slovo* sudé *délky rovné nejvýše i*
        end for
    end for
    for all $X \in R_0^i$ **do** ▷ *pokud lze X přepsat na slovo* sudé *délky rovné nejvýše i*
        for all $Z \to aX \in P$, kde $a \in \Sigma$ a $Z \in N$ **do** ▷ *pak neterminál Z*
            $R_l^{ii} \leftarrow R_l^{ii} \cup \{Z\}$ ▷ *lze přepsat na slovo* liché *délky rovné nejvýše i*
        end for
    end for
    if $S \in R_s^{ii}$ **then**
        return $True$
    end if
until $R_s^{ii} = R_s^i \wedge R_l^{ii} = R_l^i$ ▷ *opakujeme, dokud nám přibývají neterminály v počítaných množinách oproti minulé iteraci cyklu*
return $False$

IB102 – úkol 8, příklad 2 – řešení **Odevzdání: 12. 11. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [2 body] Uvažme bezkontextovou gramatiku $\mathcal{G} = (\{S\}, \{a, b\}, P, S)$, kde

$$P = \{S \to aaSb \mid aab\}.$$

Navrhněte bezkontextovou gramatiku $\mathcal{G}'$, která generuje všechny prefixy všech slov generovaných gramatikou $\mathcal{G}$, tj. takovou, že $L(\mathcal{G}') = \{u \in \{a,b\}^* \mid \exists v \in \{a,b\}^*,\ \text{kde } uv \in L(\mathcal{G})\}$.

Řešení: Gramatika $\mathcal{G}$ generuje jazyk $L(\mathcal{G}) = \{a^{2n}b^n \mid n \geq 1\}$. Naším cílem je tedy najít gramatiku $\mathcal{G}'$, která generuje jazyk $L(\mathcal{G}') = \{a^{2n}b^m \mid n \geq 1 \wedge m \leq n\} \cup \{a^n \mid n \geq 0\}$. Řešením je gramatika $\mathcal{G}' = (\{S', A, B\}, \{a, b\}, P', S')$, kde

$$\begin{aligned} P' = \{S' &\to A \mid B, \\ A &\to aA \mid \epsilon, \\ B &\to aaBb \mid aaB \mid \epsilon\}. \end{aligned}$$

Pozn.: Využili jsme zavedené konvence, že bezkontextové gramatiky mohou obsahovat ϵ-pravidla.

IB102 – úkol 10, příklad 1 **Odevzdání: 3. 12. 2012**

Vypracoval(a): UČO:
Skupina:

1. [**2 body**] Z následující vlastní bezkontextové gramatiky odstraňte levou rekurzi.

$$G = (\{S, A, B\}, \{a, b\}, P, S)$$
$$\begin{aligned} P = \{ \quad S &\rightarrow SAb \mid SaS \mid aB, \\ A &\rightarrow AAa \mid SS \mid a, \\ B &\rightarrow Aab \mid SaS \mid b\} \end{aligned}$$

Řešení: Zvolíme uspořádání $A < B < S$.

- U neterminálu A nemáme nepřímou levou rekurzi, máme zde však přímou. Tu odstraníme.
$$\begin{aligned} A &\rightarrow SS \mid a \mid SSA' \mid aA', \\ A' &\rightarrow AaA' \mid Aa \end{aligned}$$
Nový neterminál A' zařadíme do našeho uspořádání: $A' < A < B < S$.

- U neterminálu B nejprve odstraníme nepřímou levou rekurzi a dostáváme:
$$B \rightarrow SSab \mid aab \mid SSA'ab \mid aA'ab \mid SaS \mid b,$$
Nemáme zde žádnou přímou levou rekurzi a tudíž jsme hotovi.

- U neterminálu S nemáme nepřímou levou rekurzi a odstraníme tedy rovnou přímou levou rekurzi:
$$\begin{aligned} S &\rightarrow aB \mid aBS', \\ S' &\rightarrow AbS' \mid Ab \mid aSS' \mid aS \end{aligned}$$
Nový neterminál S' zařadíme do našeho uspořádání: $S' < A' < A < B < S$.

Celkově vypadá gramatika po odstranění levé rekurze takto:

$$G = (\{S, A, B, A', S'\}, \{a, b\}, P, S)$$
$$\begin{aligned} P = \{ \quad S &\rightarrow aB \mid aBS', \\ S' &\rightarrow AbS' \mid Ab \mid aSS' \mid aS, \\ A &\rightarrow SS \mid a \mid SSA' \mid aA', \\ A' &\rightarrow AaA' \mid Aa, \\ B &\rightarrow SSab \mid aab \mid SSA'ab \mid aA'ab \mid SaS \mid b\} \end{aligned}$$

IB102 – úkol 10, příklad 2 – řešení **Odevzdání: 3. 12. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [**1 bod**] Převeďte gramatiku $\mathcal{G}$, která je ve tvaru **bez levé rekurze**, do Greibachové normální formy. Použijte algoritmus z přednášky (nebo dokažte, že je vaše gramatika v GNF ekvivalentní $\mathcal{G}$).

$$\mathcal{G} = (\{W, X, Y, Z\}, \{a, b, c\}, P, W)$$

$$\begin{array}{llllllll} P = \{ & W & \to & bX & \mid & YW, & & \\ & X & \to & aXaX & \mid & ZZ, & & \\ & Y & \to & a & \mid & cYa, & & \\ & Z & \to & bZ & \mid & YcY & \mid & WW \quad \} \end{array}$$

Schematický nákres pro zjednodušení volby uspořádání:

$$\begin{tikzcd} X \arrow[r] & Z \arrow[r] \arrow[rr, bend right] & W \arrow[r] & Y \end{tikzcd}$$

Vhodné uspořádání pro algoritmus převodu do GNF je tedy $X \prec Z \prec W \prec Y$.

- $i = 3, A_i = W$
  - $j = 4, A_j = Y$: $W \to bX \mid aW \mid cYaW$
- $i = 2, A_i = Z$
  - $j = 4, A_j = Y$: $Z \to bZ \mid acY \mid cYacY \mid WW$
  - $j = 3, A_j = W$: $Z \to bZ \mid acY \mid cYacY \mid bXW \mid aWW \mid cYaWW$
- $i = 1, A_i = X$
  - $j = 4, A_j = Y$: $X \to aXaX \mid ZZ$
  - $j = 3, A_j = W$: $X \to aXaX \mid ZZ$
  - $j = 2, A_j = Z$: $X \to aXaX \mid bZZ \mid acYZ \mid cYacYZ \mid bXWZ \mid aWWZ \mid cYaWWZ$

Nyní zbývá jen nahradit terminály za neterminály. Výsledná gramatika $\mathcal{G}'$ v GNF:

$$\mathcal{G}' = (\{W, X, Y, Z, A, C\}, \{a, b, c\}, P, W)$$

$$\begin{array}{llll} P = \{ & W & \to & bX \mid aW \mid cYAW, \\ & X & \to & aXAX \mid bZZ \mid aCYZ \mid cYACYZ \mid bXWZ \mid aWWZ \mid cYAWWZ, \\ & Y & \to & a \mid cYA, \\ & Z & \to & bZ \mid aCY \mid cYACY \mid bXW \mid aWW \mid cYAWW, \\ & A & \to & a, \\ & C & \to & c \quad \} \end{array}$$

IB102 – úkol 11, příklad 1 – řešení **Odevzdání: 10. 12. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [**2 body**] Informatik se živí párky. Když sní malý párek, získá energii na hodinu paření. Když sní velký párek, získá energii na dvě hodiny paření. Předpokládejme, že do informatika se vleze libovolné množství párků. Informatik nikdy nesmí hrát více hodin, než na kolik se dopředu najedl. Povolené sekvence akcí informatika jsou tedy jen ty, ve kterých nikdy nepaří déle než na kolik má energie z párků.

Jazyk L nad abecedou $\{m, p, v\}$ je množina všech povolených sekvencí akcí informatika, kde m znamená akci "informatik snědl malý párek", v znamená akci "informatik snědl velký párek"a p znamená akci "informatik hodinu pařil". Například slovo ϵ, *mmmvvvmmm*, *mmp*, *mvppmpp*, nebo *vpmp* patří do jazyka L, zatímco slova *pmm*, *mpp*, *vmpmppppm* nikoliv.

Sestrojte zásobníkový automat akceptující jazyk L. Jasně uveďte, jakým způsobem Váš automat akceptuje (koncovým stavem, prázdným zásobníkem).

Řešení: Hledaný automat je vlastně automat rozpoznávající jazyk

$$L = \{w \in \{m, v, p\}^* \mid \#_m(u) + \#_v(u) \geq \#_p(u), \text{ pro každý prefix } u \text{ slova } w\}$$

a můžeme zkonstruovat například následovně. Idea konstrukce bude taková, že použijeme zásobník jakožto počitadlo energie z párků, kterou lze použít na paření. Na zásobníku udržujeme přesně tolik znaků E, kolik energie informatik zrovna má. Má-li informatik 0 energie, na zásobníku nám zůstane jen symbol Z značící dno. Akceptujeme kdykoliv, když vidíme na vrcholu E nebo Z. Vidíme-li Z, informatik může jen jíst, vidíme-li E, informatik může buď jíst nebo pařit.

$$A = (\{q, q_f\}, \{p, v, m\}, \{E, Z\}, \delta, q, Z, \{q_f\}), \text{ kde}$$

$$\begin{aligned} \delta(q, \varepsilon, Z) &= \{(q_f, \varepsilon)\} & \delta(q, \varepsilon, E) &= \{(q_f, \varepsilon\} \\ \delta(q, m, Z) &= \{(q, EZ)\} & \delta(q, m, E) &= \{(q, EE)\} \\ \delta(q, v, Z) &= \{(q, EEZ\} & \delta(q, v, E) &= \{(q, EEE)\} \\ & & \delta(q, p, E) &= \{(q, \varepsilon)\} \end{aligned}$$

Automat akceptuje koncovým stavem.

IB102 – úkol 11, příklad 2 – řešení **Odevzdání: 10. 12. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [**2 bod**] Rozhodněte pomocí CYK algoritmu, zda gramatika $\mathcal{G}$ generuje slovo *daabb* ($daabb \in L(\mathcal{G})$).

$$\mathcal{G} = (\{S, A, B, C, D\}, \{a, b, c, d\}, P, S)$$

$$\begin{array}{llllllll} P = \{ & S & \rightarrow & AA, & & & & \\ & A & \rightarrow & CD & \mid & a & \mid & AA, \\ & B & \rightarrow & SS & \mid & b, & & \\ & C & \rightarrow & DA, & & & & \\ & D & \rightarrow & AB & \mid & d & \} & \end{array}$$

Řešení:

D				
A	$-$			
C	D	$-$		
C	S, A	D	$-$	
D	A	A	B	B
d	a	a	b	b

Slovo *daabb* lze generovat jedině z neterminálu D a tedy $daabb \notin L(\mathcal{G})$.

IB102 – úkol 12, příklad 1 – řešení **Odevzdání: 17. 12. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

1. [**2 body**] Uvažme bezkontextovou gramatiku $\mathcal{G} = (\{S, Á, B\}, \{a, l, n, ý, z\}, P, S)$, kde

$$\begin{aligned} P = \{S &\to aSa \mid nSl \mid ÁB, \\ Á &\to aS \mid Áa \mid z \mid \epsilon. \\ B &\to lB \mid ýÁ \mid \epsilon\}. \end{aligned}$$

Sestrojte analyzátor *shora dolů* a analyzujte slovo „*analýza*“.

Řešení

Analyzátor shora dolů pro jazyk $L(\mathcal{G})$ sestrojený podle algoritmu je zásobníkový automat $\mathcal{M} = (\{q\}, \Sigma = \{a, l, n, ý, z\}, \{S, Á, B\} \cup \Sigma, \delta, q, S, \emptyset)$, kde

$$\begin{aligned} & \delta(q, \varepsilon, S) = \{(q, aSa), (q, nSl), (q, ÁB)\} \\ & \delta(q, \varepsilon, Á) = \{(q, aS), (q, Áa), (q, z), (q, \varepsilon)\} \\ & \delta(q, \varepsilon, B) = \{(q, lB), (q, ýÁ), (q, \varepsilon)\} \\ \forall x \in \Sigma \quad & \delta(q, x, x) = \{(q, \varepsilon)\} \end{aligned}$$

Automat akceptuje prázdným zásobníkem.

Analýza slova „*analýza*“:

$$\begin{aligned} (q, analýza, S) &\overset{\varepsilon}{\vdash} (q, analýza, ÁB) \overset{\varepsilon}{\vdash} (q, analýza, aSB) \overset{a}{\vdash} (q, nalýza, SB) \\ &\overset{\varepsilon}{\vdash} (q, nalýza, nSlB) \overset{n}{\vdash} (q, alýza, SlB) \overset{\varepsilon}{\vdash} (q, alýza, ÁBlB) \\ &\overset{\varepsilon}{\vdash} (q, alýza, ÁaBlB) \overset{\varepsilon}{\vdash} (q, alýza, aBlB) \overset{a}{\vdash} (q, lýza, BlB) \\ &\overset{\varepsilon}{\vdash} (q, lýza, lB) \overset{l}{\vdash} (q, ýza, B) \overset{\varepsilon}{\vdash} (q, ýza, ýÁ) \overset{y}{\vdash} (q, za, Á) \\ &\overset{\varepsilon}{\vdash} (q, za, Áa) \overset{\varepsilon}{\vdash} (q, za, za) \overset{z}{\vdash} (q, a, a) \overset{a}{\vdash} (q, \varepsilon, \varepsilon) \end{aligned}$$

IB102 – úkol 12, příklad 1 – řešení **Odevzdání: 17. 12. 2012**

Vypracoval: James Bond UČO: 007
Skupina: MI6

2. [**2 body**] Uvažme bezkontextovou gramatiku $\mathcal{G} = (\{S, Á, B\}, \{a, l, n, ý, z\}, P, S)$, kde

$$\begin{aligned} P = \{ & S \to aSa \mid nSl \mid ÁB, \\ & Á \to aS \mid Áa \mid z \mid \epsilon. \\ & B \to lB \mid ýÁ \mid \epsilon\}. \end{aligned}$$

Sestrojte analyzátor *zdola nahoru* a analyzujte slovo „*analýza*“.

Řešení

Analyzátor zdola nahoru pro jazyk $L(\mathcal{G})$ sestrojený podle algoritmu je zásobníkový automat $\mathcal{M} = (\{q, r\}, \Sigma = \{a, l, n, ý, z\}, \{S, Á, B, \bot, a, l, n, ý, z\}, \delta, q, \bot, \{r\})$, kde

$$\begin{array}{rll} & \delta(q, \varepsilon, aSa) & = \{(q, S)\}, \\ & \delta(q, \varepsilon, nSl) & = \{(q, S)\}, \\ & \delta(q, \varepsilon, ÁB) & = \{(q, S)\}, \\ & \delta(q, \varepsilon, aS) & = \{(q, Á)\} \\ & \delta(q, \varepsilon, Áa) & = \{(q, Á)\}, \\ & \delta(q, \varepsilon, z) & = \{(q, Á)\}, \\ & \delta(q, \varepsilon, \varepsilon) & = \{(q, Á), (q, B)\}, \\ & \delta(q, \varepsilon, lB) & = \{(q, B)\}, \\ & \delta(q, \varepsilon, ýÁ) & = \{(q, B)\}, \\ \forall x \in \Sigma & \delta(q, x, \varepsilon) & = \{(q, x)\}, \\ & \delta(q, \varepsilon, \bot S) & = \{(r, \varepsilon)\}. \end{array}$$

Automat akceptuje koncovým stavem.

Analýza slova „*analýza*“:

$$\begin{aligned} (q, analýza, \bot) & \overset{a}{\vdash} (q, nalýza, \bot a) \overset{n}{\vdash} (q, alýza, \bot an) \overset{\varepsilon}{\vdash} (q, alýza, \bot anÁ) \\ & \overset{a}{\vdash} (q, lýza, \bot anÁa) \overset{\varepsilon}{\vdash} (q, lýza, \bot anÁ) \overset{\varepsilon}{\vdash} (q, lýza, \bot anÁB) \\ & \overset{\varepsilon}{\vdash} (q, lýza, \bot anS) \overset{l}{\vdash} (q, ýza, \bot anSl) \overset{\varepsilon}{\vdash} (q, ýza, \bot aS) \\ & \overset{\varepsilon}{\vdash} (q, ýza, \bot Á) \overset{ý}{\vdash} (q, za, \bot Áý) \overset{z}{\vdash} (q, a, \bot Áýz) \overset{\varepsilon}{\vdash} (q, a, \bot ÁýÁ) \\ & \overset{a}{\vdash} (q, \epsilon, \bot ÁýÁa) \overset{\varepsilon}{\vdash} (q, \epsilon, \bot ÁýÁ) \overset{\varepsilon}{\vdash} (q, \varepsilon, \bot ÁB) \overset{\varepsilon}{\vdash} (q, \varepsilon, \bot S) \overset{\varepsilon}{\vdash} (r, \varepsilon, \varepsilon) \end{aligned}$$

IB102 – úkol 12, příklad 3 – bonus **Odevzdání: 17. 12. 2012**

Vypracoval(a): UČO:
Skupina:

3. [**3 body**] Sestrojte gramatiku typu 0, která generuje jazyk

$$L = \{a^1ba^2b \dots a^nb \mid n \geq 1\}.$$

Řešení: Gramatika $\mathcal{G} = (\{S, A, K, Q, Z_1, Z_2, Z_3\}, \{a, b\}, P, S)$, kde

$$\begin{aligned} P = \{ & S \to Z_1AZ_2K, \\ & AZ_2 \to Z_2aA, \\ & Aa \to aA, \\ & Z_1Z_2a \to aZ_1Z_2, \\ & Z_1Z_2A \to bZ_1Z_3A \mid bQ, \\ & Z_3A \to AZ_3, \\ & Z_3K \to AZ_2K, \\ & QA \to Q, \\ & QK \to \epsilon\}. \end{aligned}$$

Intuitivně, Z_1, Z_2, Z_3 fungují jako zarážky, neterminál K značí konec větné formy. Cokoliv je před zarážkou Z_1 už zůstává tak, jak je, mezi zarážkou Z_1 a neterminálem K probíhá generování áček, která se pak přesunou před zarážku Z_1. Například slovo ab odvodíme v gramatice $\mathcal{G}$ takto:

$$S \Rightarrow Z_1AZ_2K \Rightarrow Z_1Z_2aAK \Rightarrow aZ_1Z_2AK \Rightarrow abQK \Rightarrow ab$$

Slovo $abaab$ odvodíme takto:

$S \Rightarrow Z_1AZ_2K \Rightarrow Z_1Z_2aAK \Rightarrow aZ_1Z_2AK \Rightarrow abZ_1Z_3AK \Rightarrow abZ_1AAZ_2K \Rightarrow abZ_1AZ_2aAK \Rightarrow abZ_1Z_2aAaAK \Rightarrow abZ_1Z_2aaAAK \Rightarrow abaZ_1Z_2aAAK \Rightarrow abaaZ_1Z_2AAK \Rightarrow abaabQAK \Rightarrow abaabQK \Rightarrow abaab$