Sbírka příkladů do IFJ

Petr Zemek

11. ledna 2012

Obsah

Předmluva

Tento text slouží jako pomocný materiál pro studenty předmětu *Formální jazyky a překladače* (IFJ) bakalářského studijního programu *Informační technologie* na *Fakultě informačních technologií VUT v Brně*. Jeho cílem je poskytnout studentům příklady, na kterých si mohou vyzkoušet své nabyté znalosti. Obsahuje řadu příkladů z každé probírané oblasti, včetně jejich řešení.

Pokud v textu narazíte na chybu, neváhejte mi poslat email na izemek@fit.vutbr.cz. Předmět emailu v ideálním případě začínejte prefixem `"IFJ sbírka: "`. V případě dotazů lze využít fórum předmětu ve WISu (preferovaná varianta, protože váš dotaz může zajímat více studentů). Nezapomeňte vždy zmínit, o kterou verzi textu (= datum na přední straně) se jedná, abychom předešli nesrovnalostem.

Doufám, že vám text přinese užitek.

Petr Zemek, 11. ledna 2012

Použití a struktura

Každá kapitola obsahuje příklady k jednomu probíranému tématu. Číslování kapitol by mělo sedět s rozvržením přednášek na [3], ale jelikož se pořadí přednášek může změnit, nelze to stoprocentně zaručit. V závěru dokumentu je obsaženo řešení příkladů. Při řešení příkladů lze postupovat od prvního po poslední. Příklady označené hvězdičkou (viz dále) lze při prvním řešení vynechat.

Použitá terminologie, konvence a notace vychází z přednášek předmětu IFJ [3] a knihy, na které je tento předmět založen [2]. Pokud byl příklad převzán, je u něj vždy uvedena reference na původní zdroj.

Použité konvence, které se mohou lišit od přednášek:

- místo několika hran z jednoho stavu do druhého je použita jediná hrana, kde jednotlivé symboly jsou odděleny čárkou.

Význam hvězd u příkladů:

- $\star$ označuje příklady, u kterých je třeba zapřemýšlet nad rámec předmětu IFJ, ale které by měli studenti, kteří probírané látce rozumí, v pohodě vyřešit;
- $\star\star$ označuje příklady, k jejichž vyřešení je třeba mít znalosti mimo předmět IFJ, bez nichž nemusí být příklad pro studenta IFJ řešitelný v konečném čase.

Poděkování

Na tomto místě bych chtěl poděkovat všem, kteří se svými postřehy, poznámkami a radami zasloužili o zkvalitnění tohoto textu. Dále bych chtěl poděkovat studentům, kteří na cvičeních pokládali zajímavé otázky, které tvořily motivaci u řady příkladů. V neposlední řadě bych chtěl poděkovat své přítelkyni, která mi byla oporou.

Tento dokument nebyl financován z žádného grantu ani výzkumného záměru a vznikl ve volném čase.

Historie revizí

Jelikož bude tento dokument často aktualizován, uvádím seznam proběhlých revizí.

Datum	**Popis revize**
11.1.2012	Zjednodušeno řešení příkladu 6.3. Za nápad děkuji Radce Škvařilové.
11.1.2012	Doplněno zadání příkladu 8.2 (doplněny chybějící závorky v množině terminálů). Za nahlášení chyby děkuji Michalu Starigazdovi.
10.1.2012	Opraveno řešení příkladu 12.1. Za nahlášení chyby děkuji Jiřímu Honovi.
10.1.2012	Opraveno řešení příkladu 8.1. Za nahlášení chyby děkuji Matúšovi Fedorkovi.
9.1.2012	Opraveno řešení příkladu 7.4 a upraveno zadání příkladu 7.5. Za nahlášení chyby děkuji Jakubu Jeřábkovi.
9.1.2012	Opraveno řešení příkladů 8.1 a 8.3. Za nahlášení chyb děkuji Fridolínu Pokornému.
5.12.2011	Doplněno zadání příkladu 8.2 (specifikace priority operátorů) a opraveno jeho řešení. Za nahlášení chyb děkuji Fridolínu Pokornému.
5.12.2011	Doplněno zadání příkladu 6.6. Za postřeh děkuji Fridolínu Pokornému.
24.10.2011	Opraveno číslo řešení u příkladu 1.9. Za nahlášení chyby děkuji Michalu Starigazdovi.
23.10.2011	Zjednodušeno řešení příkladu 3.4. Za postřeh děkuji Fridolínu Pokornému.
22.10.2011	Zjednodušeno řešení příkladu 3.3. Za postřeh děkuji Dávidu Antolíkovi.
22.10.2011	Opraveno řešení bodu (k) v příkladu 1.7. Za nahlášení chyby děkuji Fridolínu Pokornému.
21.10.2011	Opraveno zadání bodu (h) v příkladu 1.6. Za nahlášení chyby děkuji Lukáši Svatému.
20.10.2011	Opraveno řešení bodu (e) v příkladu 1.6. Za nahlášení chyby děkuji Ladislavu Szántovi a jednomu studentovi na cvičení, jehož jméno bohužel nevím.
4.10.2011	Sjednocen průměr uzlů v přechodových grafech konečných automatů (kosmetická změna).
20.9.2011	Zveřejněna první verze.

Kapitola 1

Abecedy, řetězce a jazyky

Pokud není řečeno jinak, uvažujte u všech příkladů abecedu $\Sigma = \{a, b, c\}$.

Příklad 1.1. Jakou délku mají řetězce *abcca*, *b* a ε?

Příklad 1.2. Jaké jsou reverzace řetězců *abca*, *aba* a ε?

Příklad 1.3. Určete všechny prefixy řetězce *aab*. Které z nich jsou vlastní?

Příklad 1.4. Určete všechny sufixy řetězce *bca*. Které z nich jsou vlastní?

Příklad 1.5. Určete všechny podřetězce řetězce *aabaaa*. Které z nich jsou vlastní?

Příklad 1.6. Určete:

(a) $(abc)^3 = ?$

(b) $\varepsilon^4 = ?$

(c) $\varepsilon\varepsilon = ?$

(d) $\varepsilon a = ?$

(e) $\{ab\}\{cc, ca\}\{a\} = ?$

(f) $\{a\}^*\emptyset = ?$

(g) $\{a\}^+ \cup \{\varepsilon\} = ?$

(h) $\Sigma^* - \{a, b, c\}^* = ?$

(i) $\{ab, bc, ac\} - \{cb\} = ?$

(j) $\{ab, cc\}^2 = ?$

(k) $\{a, b, c\}^3 = ?$

(l) $\{ab, bc, bb, aa\} \cap \{cb, ab\} = ?$

(m) $\overline{\Sigma^*} = ?$

Příklad 1.7. Určete:

(a) $\emptyset^* = ?$

(b) $\emptyset^+ = ?$

(c) $\emptyset^3 = ?$

(d) $\emptyset\emptyset = ?$

(e) $\emptyset \cup \emptyset = ?$

(f) $\{a\}\emptyset = ?$

(g) $\{a\} \cup \emptyset = ?$

(h) $\{\varepsilon\}^* = ?$

(i) $\{\varepsilon\}^+ = ?$

(j) $\{\varepsilon\} \cup \emptyset = ?$

(k) $\{\varepsilon\}\emptyset = ?$

(l) $\{\varepsilon\}^5 = ?$

Příklad 1.8 ($\star$)**.** Operace *symetrický rozdíl* ($\ominus$) dvou množin je definována jako množina, která obsahuje prvky, které jsou buď v první množině, nebo ve druhé množině, ale ne v obou zároveň. Nechť L_1 a L_2 jsou dva jazyky nad Σ.

(a) Definujte formálně symetrický rozdíl L_1 a L_2 (s využitím náležitosti do množiny $\in$).

(b) Definujte formálně symetrický rozdíl L_1 a L_2 pouze pomocí operací sjednocení a rozdílu dvou množin.

Příklad 1.9 ($\star\star$)**.** Mějme jazyk

$$L = \{n > 2 \mid \text{existují } a, b, c \geq 1 \text{ takové, že } a^n + b^n = c^n\}$$

Je L konečný?

Kapitola 2

Úvod do překladačů

Příklad 2.1. Vypište a seřaďte fáze překladu, jak jdou typicky za sebou na logické úrovni.

Kapitola 3

Modely regulárních jazyků

Pokud není řečeno jinak, uvažujte u všech příkladů abecedu $\Sigma = \{a, b, c\}$.

Příklad 3.1. Rozhodněte, které z následujících výrazů jsou platné regulární výrazy (uvažujte i konvence zavedené na přednáškách). Pokud není výraz platným regulárním výrazem, zdůvodněte proč.

(a) a^*

(b) a^3

(c) aa

(d) $abcd$

(e) $ab + bc$

(f) ε

(g) $ab^* - b$

(h) ε^*

(i) ε^ε

(j) $a^+ + b^+ + c^+$

(k) Σ^*

Příklad 3.2. Určete, které jazyky značí následující regulární výrazy.

(a) $a + aa + aaa = ?$

(b) $a^+ + \varepsilon = ?$

(c) $(a + b + c)^* = ?$

(d) $abc = ?$

(e) $(\emptyset + \varepsilon)^* = ?$

(f) $(b + c)^* a = ?$

Příklad 3.3. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě řetězce obsahující lichý (nenulový) počet znaků 1.

Příklad 3.4. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě řetězce obsahující jako podřetězec 101.

Příklad 3.5. Vytvořte regulární výraz nad abecedou $\Sigma = \{a, b\}$, značící jazyk obsahující právě řetězce, které obsahují jako podřetězec $aabb$.

Příklad 3.6. Vytvořte regulární výraz nad abecedou $\Sigma = \{a, b\}$, značící jazyk obsahující právě řetězce, které neobsahují jako podřetězec aa.

Příklad 3.7. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě řetězce neobsahující podřetězec 11.

Příklad 3.8. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá prázdný jazyk ($\emptyset$).

Příklad 3.9. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě řetězce začínající prefixem 010.

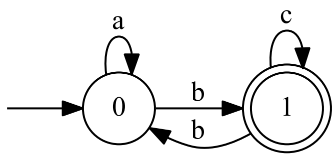
Příklad 3.10. Vytvořte regulární výraz nad abecedou $\Sigma = \{a, b\}$, značící jazyk obsahující právě řetězce končící sufixem aa.

Příklad 3.11. Převeďte regulární výraz $(aa + b)^*c$ na konečný automat (použijte algoritmus z přednášek).

Příklad 3.12. Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě neprázdné řetězce sudé délky neobsahující 1.

Příklad 3.13 ($\star$)**.** Vytvořte konečný automat nad abecedou $\Sigma = \{0, 1\}$, který přijímá právě řetězce obsahující stejný počet znaků 0 a 1.

Příklad 3.14 ($\star$)**.** Převeďte následující konečný automat nad abecedou $\Sigma = \{a, b, c\}$ na ekvivalentní regulární výraz.

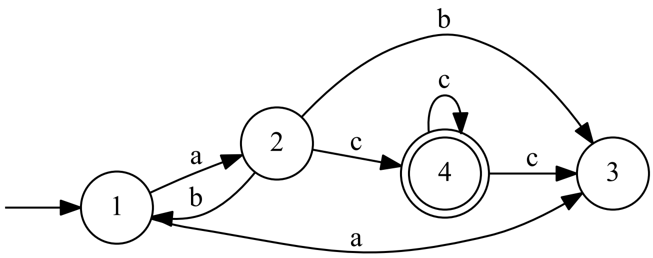


Kapitola 4

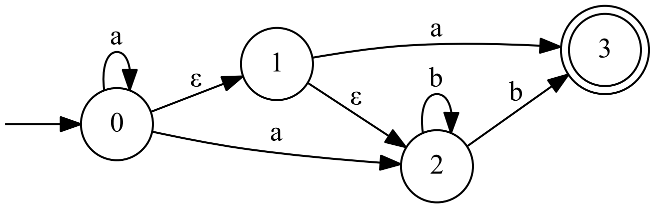
Speciální konečné automaty

Pokud není řečeno jinak, uvažujte u všech příkladů abecedu $\Sigma = \{a, b, c\}$.

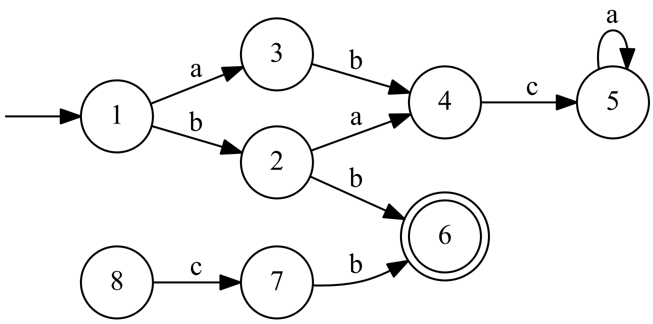
Příklad 4.1. Převeďte zadaný konečný automat na deterministický.



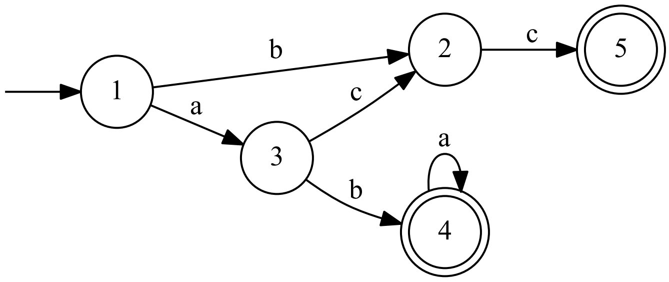
Příklad 4.2. Převeďte zadaný konečný automat na deterministický.



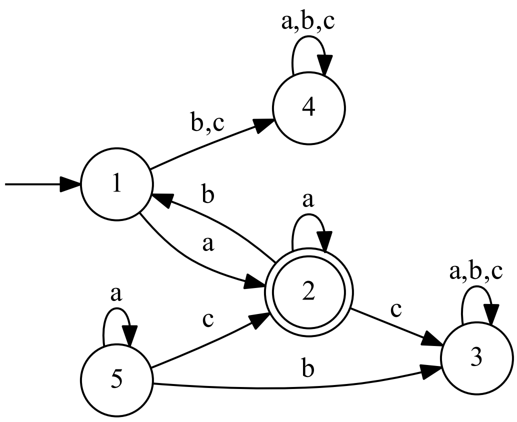
Příklad 4.3. Převeďte zadaný deterministický konečný automat na ekvivalentní deterministický konečný automat bez nedostupných a neukončujících stavů.



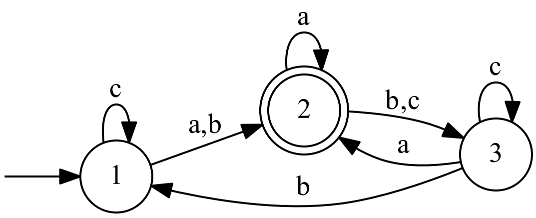
Příklad 4.4. Převeďte zadaný deterministický konečný automat na úplný konečný automat.



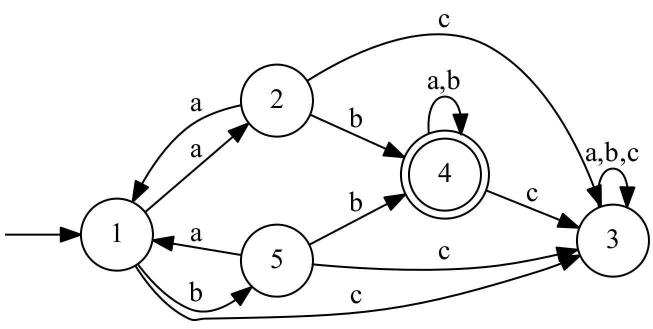
Příklad 4.5. Převeďte zadaný úplný konečný automat na dobře specifikovaný konečný automat.



Příklad 4.6. Převeďte zadaný úplný konečný automat na dobře specifikovaný konečný automat.



Příklad 4.7. Převeďte zadaný dobře specifikovaný konečný automat na minimální.



Kapitola 5

Lexikální analýza

Příklad 5.1. Jaký je rozdíl mezi tokenem a lexémou?

Kapitola 6

Modely bezkontextových jazyků

Příklad 6.1. Vytvořte bezkontextovou gramatiku generující jazyk

$$\left\{a^n b^n \mid n \geq 1\right\}$$

Příklad 6.2. Vytvořte bezkontextovou gramatiku generující jazyk

$$\left\{w \operatorname{reversal}(w) \mid w \in \{a,b\}^*\right\}$$

Příklad 6.3. Vytvořte zásobníkový automat přijímající jazyk

$$\left\{w \mid w \in \{0,1\}^+, w \text{ obsahuje stejný počet 0 a 1}\right\}$$

Typ přijímání si zvolte.

Příklad 6.4. Vytvořte bezkontextovou gramatiku generující jazyk

$$\left\{a^m b^m c^n \mid m,n \geq 0\right\} \cup \left\{a^m b^n c^n \mid m,n \geq 0\right\}$$

Příklad 6.5. Vytvořte zásobníkový automat přijímající jazyk

$$\left\{w \operatorname{reversal}(w) \mid w \in \{a,b\}^*\right\}$$

Typ přijímání si zvolte.

Příklad 6.6. Ze zadané gramatiky G vytvořte obecný syntaktický analyzátor shora dolů.

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

kde P obsahuje následující pravidla:

- $S \to AaBbCc$
- $A \to BC$
- $B \to \varepsilon$
- $C \to abD$
- $D \to c$

Ukažte přijetí řetězce *abcababcc*.

Příklad 6.7. Z gramatiky G z příkladu 6.6 vytvořte obecný syntaktický analyzátor zdola nahoru. Ukažte přijetí řetězce *abcababcc*.

Příklad 6.8 ($\star$)**.** Vytvořte bezkontextovou gramatiku generující jazyk

$$\left\{a^i b^j c^k \mid i,j,k \;\geq 0, i \neq j \text{ nebo } j \neq k \text{ nebo } i \neq k\right\}$$

Příklad 6.9 ($\star$)**.** Vytvořte deterministický zásobníkový automat přijímající jazyk

$$\big\{w\,\mathrm{reversal}(w) \mid w \in \{a,b\}^*\big\}$$

Typ přijímání si zvolte.

Příklad 6.10 ($\star$)**.** Vytvořte zásobníkový automat přijímající jazyk

$$\big\{ww \mid w \in \{a,b\}^*\big\}$$

Typ přijímání si zvolte.

Kapitola 7

Syntaktická analýza shora dolů

Příklad 7.1. Uvažujte gramatiku pro aritmetické výrazy G_{expr3} z přednášek [3] a z ní vytvořenou LL tabulku. Proveďte prediktivní syntaktickou analýzu řetězce $i + i$. Jaké dvě informace jsou výsledkem prediktivní syntaktické analýzy? Jak je tomu v tomto příkladu?

Příklad 7.2. Uvažujte gramatiku

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

kde P obsahuje následujících šest pravidel:

$1\colon S \to aABC$	$3\colon A \to Bc$	$5\colon B \to c$
$2\colon A \to a$	$4\colon B \to bB$	$6\colon C \to cc$

Vytvořte z G LL tabulku. Je G LL gramatika?

Příklad 7.3. Uvažujte gramatiku G a vytvořenou tabulku z příkladu 7.2. Určete levý rozbor pro řetězec $aabccc$.

Příklad 7.4. Uvažujte gramatiku

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

kde P obsahuje následujících šest pravidel:

$1\colon S \to AcB$	$3\colon A \to \varepsilon$	$5\colon B \to \varepsilon$
$2\colon A \to BC$	$4\colon B \to bBa$	$6\colon C \to \varepsilon$

Vytvořte z G LL tabulku. Je G LL gramatika?

Příklad 7.5. Uvažujte gramatiku G z příkladu 7.4. Určete levý rozbor pro řetězec $bacba$.

Příklad 7.6. Uvažujte gramatiku

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

kde P obsahuje následující čtyři pravidla:

$X \to abY$ $\quad$ $Y \to cZ$

$X \to a$ $\quad$ $Z \to bb$

Použitím faktorizace (vytýkání) transformujte G na ekvivalentní LL gramatiku H.

Příklad 7.7. Uvažujte gramatiku

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

kde P obsahuje následující čtyři pravidla:

$X \to XabZ$ $\quad$ $Z \to YaaY$

$X \to c$ $\quad$ $Y \to \varepsilon$

Transformujte G na ekvivalentní gramatiku H bez levé rekurze.

Kapitola 8

Syntaktická analýza zdola nahoru

Příklad 8.1. Podle gramatiky pro výrazy a precedenční tabulky z přednášek [3] proveďte syntaktickou analýzu řetězce $i * (i + i)$. Jaké dvě informace jsou výsledkem precedenční syntaktické analýzy? Jak je tomu v tomto příkladu?

Příklad 8.2. Mějme bezkontextovou gramatiku

$$G = \big(\{S\}, \{\diamond, \bullet, \sqcap, i, (,)\}, S, P\big)$$

kde P obsahuje následujících pět pravidel:

$1\colon S \to S \diamond S$	$3\colon S \to S \sqcap S$	$5\colon S \to i$
$2\colon S \to S \bullet S$	$4\colon S \to (S)$	

Operátor $\sqcap$ má vyšší prioritu než operátory $\diamond$ a $\bullet$. Operátor $\bullet$ má vyšší prioritu než $\diamond$. Operátory $\bullet$ a $\diamond$ jsou levě asociativní, operátor $\sqcap$ je pravě asociativní. Vytvořte pro tuto gramatiku precedenční tabulku.

Příklad 8.3. Uvažujte gramatiku G a precedenční tabulku z příkladu 8.2. Určete pravý rozbor pro řetězec $i \sqcap i \bullet (i \diamond i)$.

Příklad 8.4. Uvažujte gramatiku pro aritmetické výrazy G_{expr1} a LR tabulku z přednášek [3]. Proveďte podle ní LR syntaktickou analýzu řetězce $i + i$. Jaký je jeho pravý rozbor?

Kapitola 9

Syntaxí řízený překlad

Příklad 9.1. Vysvětlete hlavní myšlenku syntaxí řízeného překladu.

Příklad 9.2. Vyjmenujte tři základní metody generování tříadresného kódu (3AK).

Kapitola 10

Optimalizace, generování cílového kódu

Příklad 10.1. Rozdělte následující kód na základní bloky.

```
    int a = 0;
L1:
    a += 1;
    printf("%d", a);
L2:
    if (a < 5) goto L1;
    a = rand();
L3:
    if (a > 10) goto L2;
L4:
    printf("%d", a);
```

Příklad 10.2. Uvažujte následující kód.

```
switch (a) {
    case 1:  b = a * b * c; break;
    case 2:  b = a * b + c; break;
    case 3:  b = c - a * b; break;
    case 4:  b = c / a * b; break;
    default: b = 2 * a * b; break;
}
```

Na tento kód byla aplikována optimalizace snížení velikosti programu (došlo k nahrazení výrazu `a * b` za konstantu), jejíž výsledkem je následující kód.

```
const int AB = a * b;
switch (a) {
    case 1:  b = AB * c; break;
    case 2:  b = AB + c; break;
    case 3:  b = c - AB; break;
    case 4:  b = c / AB; break;
    default: b = 2 * AB; break;
}
```

Je tato optimalizace v tomto případě korektní? Zdůvodněte.

Příklad 10.3. Mějme následující posloupnost instrukcí.

```
1: v = a / c
2: w = v - b
3: u = w * c
4: d = u + w
```

Proměnné a, b, c a d jsou programátorské, zbylé proměnné jsou pomocné. Vytvořte a vyplňte pro tuto posloupnost tabulku základního bloku.

Kapitola 11

Vlastnosti regulárních jazyků

Příklad 11.1. Napište znění pumping lemma pro regulární jazyky.

Příklad 11.2. Vysvětlete, proč pomocí pumping lemma pro regulární jazyky nelze dokázat, že daný jazyk je regulární.

Příklad 11.3 ($\star$)**.** Pomocí pumping lemma dokažte, že jazyk

$$L = \big\{ww \mid w \in \{a,b\}^*\big\}$$

není regulární.

Příklad 11.4 ($\star$)**.** Konstrukčně dokažte, že pro každé dva konečné automaty M_1 a M_2 platí, že

$$K = L(M_1) \cup L(M_2)$$

je regulární. Konstrukčně znamená, že sestrojíte konečný automat, který přijímá K.

Příklad 11.5 ($\star$)**.** Konstrukčně dokažte, že pro každé dva konečné automaty M_1 a M_2 platí, že

$$K = L(M_1)L(M_2)$$

je regulární. Konstrukčně znamená, že sestrojíte konečný automat, který přijímá K.

Příklad 11.6 ($\star$)**.** Konstrukčně dokažte, že pro každé dva konečné automaty M_1 a M_2 platí, že

$$K = L(M_1) \cap L(M_2)$$

je regulární. Konstrukčně znamená, že sestrojíte konečný automat, který přijímá K.

Příklad 11.7 ($\star$)**.** Dokažte, že třída regulárních jazyků je uzavřena vůči reverzi. Nápověda: Ukažte, že pro každý konečný automat M lze sestrojit takový konečný automat, který přijímá

$$\mathrm{reversal}\big(L(M)\big)$$

Kapitola 12

Vlastnosti bezkontextových gramatik

Příklad 12.1 ($\star$)**.** Převeďte následující gramatiku na ekvivalentní gramatiku v Chomského normální formě.

$$G = \big(\{S, B, C\}, \{a, b, c\}, P, S\big)$$

kde P obsahuje pravidla

- $S \to CaB$
- $B \to CCCC \mid b$
- $C \to c$

Příklad 12.2 ($\star$)**.** Převeďte následující gramatiku na ekvivalentní gramatiku v Greibachové normální formě.

$$G = \big(\{S, A, B\}, \{a, b\}, P, S\big)$$

kde P obsahuje pravidla

- $S \to aaBA$
- $A \to bBbb$
- $B \to ab$

Příklad 12.3. Napište znění pumping lemma pro bezkontextové jazyky.

Příklad 12.4 ($\star$)**.** Vysvětlete, proč pomocí pumping lemma pro bezkontextové jazyky nelze dokázat, že daný jazyk je bezkontextový.

Příklad 12.5 ($\star$)**.** Dokažte, že třída bezkontextových jazyků je uzavřena vůči reverzi. Nápověda: Ukažte, že pro každou bezkontextovou gramatiku G lze sestrojit takovou bezkontextovou gramatiku, která generuje

$$\mathrm{reversal}\big(L(G)\big)$$

Kapitola 13

Turingovy stroje a Chomského hierarchie

Příklad 13.1 ($\star$)**.** Neformálně popište Turingův stroj, který přijímá jazyk

$$\{a^n b^n c^n \mid n \geq 0\}$$

(Stačí popsat princip, na jakém stroj přijímající tento jazyk funguje.)

Příklad 13.2 ($\star$)**.** Vytvořte neomezenou gramatiku, která generuje jazyk

$$\{ww \mid w \in \{a, b\}^*\}$$

Příklad 13.3 ($\star$)**.** Vytvořte neomezenou gramatiku, která generuje jazyk

$$\{a^n b^n \mid n \geq 1\}$$

Příklad 13.4 ($\star$)**.** Vytvořte pravou lineární gramatiku, která generuje jazyk

$$\{aa\}\{abc\}^*\{ac, ca\}$$

Příklad 13.5 ($\star$)**.** Vymyslete příklad jazyka, který

(a) patří do třídy jazyků typu 3;

(b) patří do třídy jazyků typu 2, ale nepatří do třídy jazyků typu 3;

(c) patří do třídy jazyků typu 1, ale nepatří do třídy jazyků typu 2.

Příklad 13.6 ($\star\star$)**.** Vytvořte kontextovou gramatiku, která generuje nenulová Fibonacciho čísla [6] v unárním zakódování (tzn. $1 = 0$, $2 = 00$, $3 = 000$, $5 = 00000$ atd.).

Řešení příkladů

Kapitola 1

Řešení příkladu 1.1. $|abcca| = 5$, $|b| = 1$ a $|\varepsilon| = 0$

Řešení příkladu 1.2. $\text{reversal}(abca) = acba$, $\text{reversal}(aba) = aba$ a $\text{reversal}(\varepsilon) = \varepsilon$

Řešení příkladu 1.3. Prefixy jsou ε, a, aa, aab. Vlastní prefixy jsou a a aa.

Řešení příkladu 1.4. Sufixy jsou bca, ca, a, ε. Vlastní sufixy jsou ca a a.

Řešení příkladu 1.5. Podřetězce jsou ε, a, b, aa, ab, ba, aaa, aab, aba, baa, $aaba$, $abaa$, $baaa$, $aabaa$, $abaaa$, $aabaaa$. Vlastní podřetězce jsou a, b, aa, ab, ba, aaa, aab, aba, baa, $aaba$, $abaa$, $baaa$, $aabaa$, $abaaa$.

Řešení příkladu 1.6.

(a) $(abc)^3 = abcabcabc$

(b) $\varepsilon^4 = \varepsilon$

(c) $\varepsilon\varepsilon = \varepsilon$

(d) $\varepsilon a = a$

(e) $\{ab\}\{cc, ca\}\{a\} = \{abcca, abcaa\}$

(f) $\{a\}^*\emptyset = \emptyset$

(g) $\{a\}^+ \cup \{\varepsilon\} = \{a\}^*$

(h) $\Sigma^* - \{a, b, c\}^* = \emptyset$

(i) $\{ab, bc, ac\} - \{cb\} = \{ab, bc, ac\}$

(j) $\{ab, cc\}^2 = \{abcc,\ ccab,\ abab,\ cccc\}$

(k) $\{a, b, c\}^3 = \{aaa,\ aab,\ aac,\ aba,\ abb,\ abc,\ aca,\ acb,\ acc,\ baa,\ bab,\ bac,\ bba,\ bbb,\ bbc,\ bca,\ bcb,\ bcc,\ caa,\ cab,\ cac,\ cba,\ cbb,\ cbc,\ cca,\ ccb,\ ccc\}$

(l) $\{ab, bc, bb, aa\} \cap \{cb, ab\} = \{ab\}$

(m) $\overline{\Sigma^*} = \emptyset$

Řešení příkladu 1.7.

(a) $\emptyset^* = \{\varepsilon\}$

(b) $\emptyset^+ = \emptyset$

(c) $\emptyset^3 = \emptyset$

(d) $\emptyset\emptyset = \emptyset$

(e) $\emptyset \cup \emptyset = \emptyset$

(f) $\{a\}\emptyset = \emptyset$

(g) $\{a\} \cup \emptyset = \{a\}$

(h) $\{\varepsilon\}^* = \{\varepsilon\}$

(i) $\{\varepsilon\}^+ = \{\varepsilon\}$

(j) $\{\varepsilon\} \cup \emptyset = \{\varepsilon\}$

(k) $\{\varepsilon\}\emptyset = \emptyset$

(l) $\{\varepsilon\}^5 = \{\varepsilon\}$

Řešení příkladu 1.8.

(a) $L_1 \ominus L_2 = \{x \mid x \in L_1$ a zároveň $x \notin L_2$, nebo $x \in L_2$ a zároveň $x \notin L_1\}$

(b) $L_1 \ominus L_2 = (L_1 - L_2) \cup (L_2 - L_1)$

Řešení příkladu 1.9. Ano, L je končený. Dokonce platí, že $L = \emptyset$. Vyplývá to ze slavné Velké Fermatovy věty [5].

Kapitola 2

Řešení příkladu 2.1. lexikální analýza, syntaktická analýza, sémentická analýza, generování vnitřního kódu, optimalizace, generování cílového kódu

Kapitola 3

Řešení příkladu 3.1.

(a) a^* – platný

(b) a^3 – neplatný (mocnina není v definici regulárních výrazů)

(c) aa – platný

(d) $abcd$ – neplatný ($d \notin \Sigma$)

(e) $ab + bc$ – platný

(f) ε – platný

(g) $ab^* - b$ – neplatný (rozdíl není v definici regulárních výrazů)

(h) ε^* – platný

(i) ε^ε – neplatný ($^\varepsilon$ není v definici regulárních výrazů)

(j) $a^+ + b^+ + c^+$ – platný

(k) Σ^* – neplatný ($\Sigma \notin \Sigma$)

Řešení příkladu 3.2.

(a) $a + aa + aaa = \{a, aa, aaa\}$

(b) $a^+ + \varepsilon = \{a\}^*$

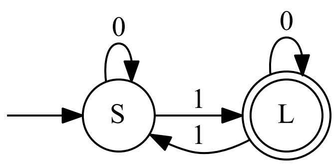
(c) $(a + b + c)^* = \{a, b, c\}^*$

(d) $abc = \{abc\}$

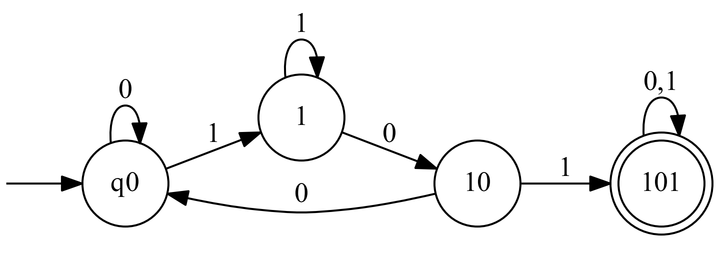
(e) $(\emptyset + \varepsilon)^* = \{\varepsilon\}$

(f) $(b + c)^*a = \{b, c\}^*\{a\}$

Řešení příkladu 3.3.



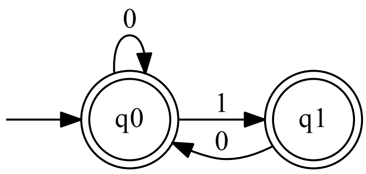
Řešení příkladu 3.4.



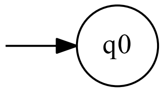
Řešení příkladu 3.5. $(a+b)^*aabb(a+b)^*$

Řešení příkladu 3.6. $(ab+b)^*(a+\varepsilon)$

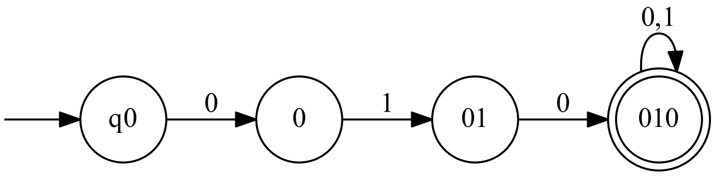
Řešení příkladu 3.7.



Řešení příkladu 3.8.

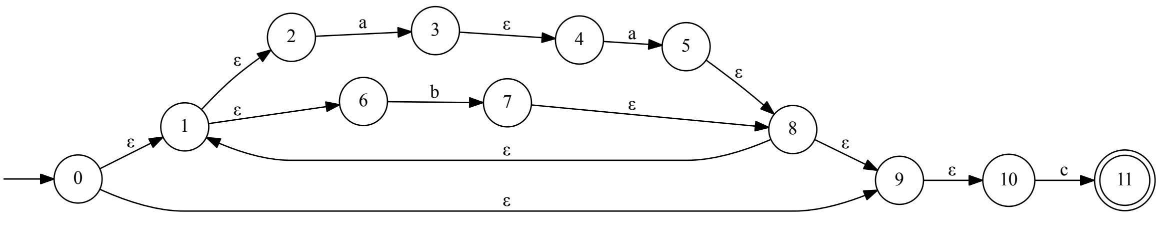


Řešení příkladu 3.9.

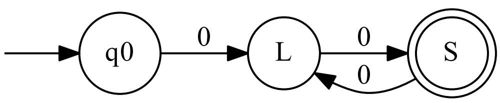


Řešení příkladu 3.10. $(a+b)^*aa$

Řešení příkladu 3.11.



Řešení příkladu 3.12.

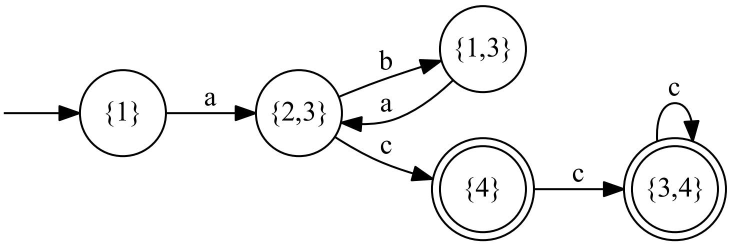


Řešení příkladu 3.13. Takový konečný automat neexistuje (potřebovali bychom nekonečný počet stavů).

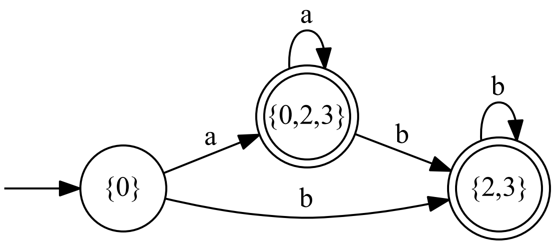
Řešení příkladu 3.14. $(a+bc^*b)^*bc^*$

Kapitola 4

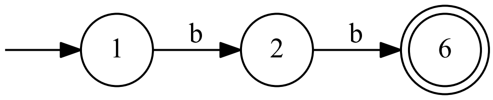
Řešení příkladu 4.1.



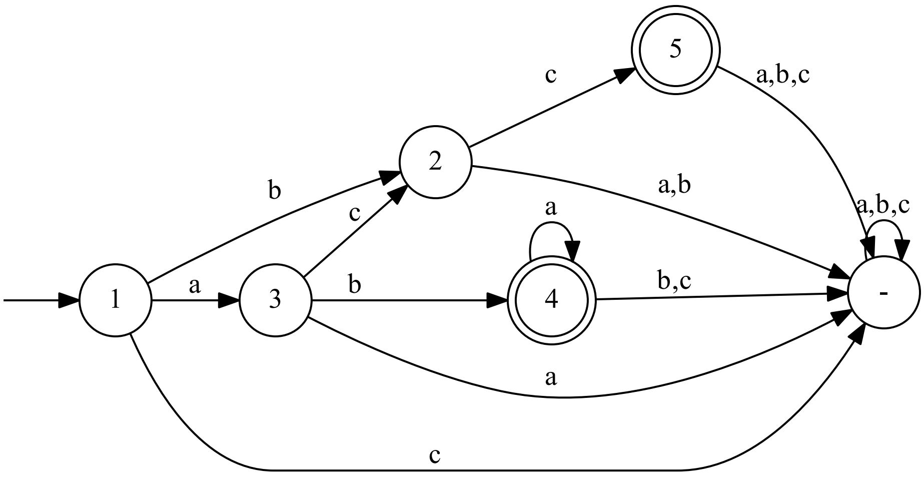
Řešení příkladu 4.2.



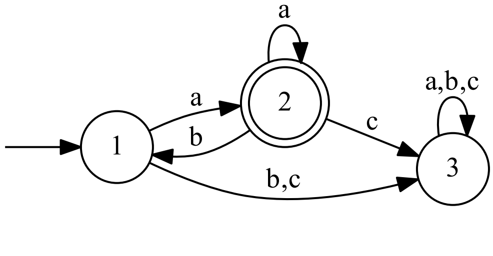
Řešení příkladu 4.3.



Řešení příkladu 4.4.

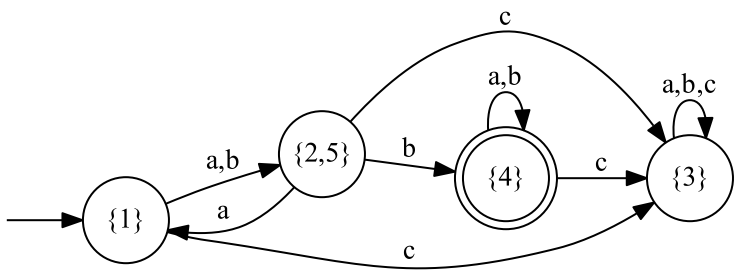


Řešení příkladu 4.5.



Řešení příkladu 4.6. Zadaný automat je již dobře specifikovaný konečný automat.

Řešení příkladu 4.7.



Kapitola 5

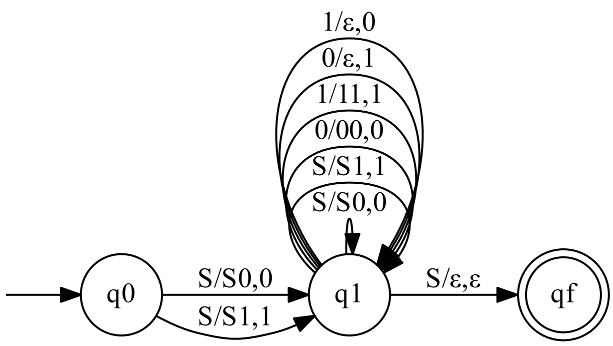
Řešení příkladu 5.1. *Lexéma* je lexikální jednotka daného programovacího jazyka, např. identifikátor, celočíselná konstanta, operátor sčítání, apod. *Token* je konkrétní reprezentace lexémy, např. identifikátor `fox` či celočíselná konstanta `6`. Dá se říct, že token je lexéma s případnými atributy.

Kapitola 6

Řešení příkladu 6.1. $G = \big(\{S\}, \{a, b\}, P, S\big)$, kde P obsahuje pravidla $S \to aSb$ a $S \to ab$.

Řešení příkladu 6.2. $G = \big(\{S\}, \{a, b\}, P, S\big)$, kde P obsahuje pravidla $S \to aSa$, $S \to bSb$ a $S \to \varepsilon$.

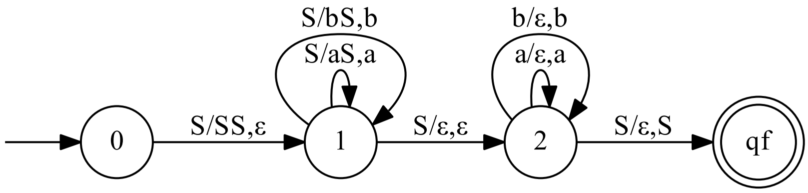
Řešení příkladu 6.3. Automat přijímá koncovým stavem (navíc má v koncovém stavu vždy prázdný zásobník).



Řešení příkladu 6.4. $G = \big(\{S, X, Y, Z, W\}, \{a, b, c\}, P, S\big)$, kde P obsahuje následující pravidla:

- $S \to XY \mid ZW$
- $X \to aXb \mid \varepsilon$
- $Y \to cY \mid \varepsilon$
- $Z \to aZ \mid \varepsilon$
- $W \to bWc \mid \varepsilon$

Řešení příkladu 6.5. Automat přijímá koncovým stavem (navíc má v koncovém stavu vždy prázdný zásobník).



Řešení příkladu 6.6. Zásobníkový automat přijímající vyprázdněním zásobníku

$$M = \big(\{s\}, \{a, b, c\}, \{S, A, B, C, D, a, b, c\}, R, s, S, \emptyset\big)$$

kde R obsahuje následující pravidla:

- $asa \to s$
- $bsb \to s$
- $csc \to s$
- $Ss \to cCbBaAs$
- $As \to CBs$
- $Bs \to s$
- $Cs \to Dbas$
- $Ds \to cs$

Řetězec *abcababcc* je přijat následující posloupností přechodů:

$$\begin{array}{rcll}
Ssabcababcc & \vdash & cCbBaAsabcababcc & [Ss \to cCbBaAs] \\
 & \vdash & cCbBaCBsabcababcc & [As \to CBs] \\
 & \vdash & cCbBaCsabcababcc & [Bs \to s] \\
 & \vdash & cCbBaDbasabcababcc & [Cs \to Dbas] \\
 & \vdash & cCbBaDbsbcababcc & [asa \to s] \\
 & \vdash & cCbBaDscababcc & [bsb \to s] \\
 & \vdash & cCbBacscababcc & [Ds \to cs] \\
 & \vdash & cCbBasababcc & [csc \to s] \\
 & \vdash & cCbBsbabcc & [asa \to s] \\
 & \vdash & cCbsbabcc & [Bs \to s] \\
 & \vdash & cCsabcc & [bsb \to s] \\
 & \vdash & cDbasabcc & [Cs \to Dbas] \\
 & \vdash & cDbsbcc & [asa \to s] \\
 & \vdash & cDscc & [bsb \to s] \\
 & \vdash & ccscc & [Ds \to cs] \\
 & \vdash & csc & [csc \to s] \\
 & \vdash & s & [csc \to s]
\end{array}$$

Řešení příkladu 6.7. Rozšířený zásobníkový automat přijímající koncovým stavem

$$M = \big(\{s, f\}, \{a, b, c\}, \{S, A, B, C, D, a, b, c, \#\}, R, s, \#, f\big)$$

kde R obsahuje následující pravidla:

- $\#Ss \to f$
- $sa \to as$
- $sb \to bs$
- $sc \to cs$
- $AaBbCcs \to Ss$
- $BCs \to As$
- $s \to Bs$
- $abDs \to Cs$
- $cs \to Ds$

Řetězec *abcababcc* je přijat následující posloupností přechodů:

$$
\begin{array}{rcll}
\#sabcababcc & \vdash & \#Bsabcababcc & [s \to Bs] \\
& \vdash & \#Basbcababcc & [sa \to as] \\
& \vdash & \#Babscababcc & [sb \to bs] \\
& \vdash & \#Babcsababcc & [sc \to cs] \\
& \vdash & \#BabDsababcc & [cs \to Ds] \\
& \vdash & \#BCsababcc & [abDs \to Cs] \\
& \vdash & \#Asababcc & [BCs \to As] \\
& \vdash & \#Aasbabcc & [sa \to as] \\
& \vdash & \#AaBsbabcc & [s \to Bs] \\
& \vdash & \#AaBbsabcc & [sb \to bs] \\
& \vdash & \#AaBbasbcc & [sa \to as] \\
& \vdash & \#AaBbabscc & [sb \to bs] \\
& \vdash & \#AaBbabcsc & [sc \to cs] \\
& \vdash & \#AaBbabDsc & [cs \to Ds] \\
& \vdash & \#AaBbCsc & [abDs \to Cs] \\
& \vdash & \#AaBbCcs & [sc \to cs] \\
& \vdash & \#Ss & [AaBbCcs \to Ss] \\
& \vdash & f & [\#Ss \to f]
\end{array}
$$

Řešení příkladu 6.8. Stačí zaručit, že se bude lišit počet a a b nebo počet b a c. Gramatika proto na začátku provede "rozdvojení" do dvou větví, kde v první se zaručí, že počet a a b bude různý, a ve druhé se zaručí, že počet b a c bude různý. Dále se v každé větvi provede další "rozdvojení", kde v prvním bude počet jednoho symbolu menší než počet druhého, a ve druhé větvi naopak (např. v první větvi se zaručí, že počet a bude menší než počet b, a ve druhé větvi se zaručí, že počet a bude větší než počet b).

Řešením je

$$G = \big(\{S_1, S_1', S_2, S_2', \bar{a}, \bar{b}, \bar{c}, A_1, A_1', B_1, B_1', A, C\}, \{a, b, c\}, P, S\big)$$

kde P obsahuje následující pravidla:

- $S \to S_1 \mid S_1' \mid S_2 \mid S_2'$
- $\bar{a} \to a \mid \varepsilon$
- $\bar{b} \to b \mid \varepsilon$
- $\bar{c} \to c \mid \varepsilon$
- $A \to Aa \mid \varepsilon$
- $C \to Cc \mid \varepsilon$

- $S_1 \to A_1 bC$
- $A_1 \to \bar{a} A_1 b \mid \varepsilon$
- $S_1' \to aA_1'C$
- $A_1' \to aA_1\bar{b} \mid \varepsilon$

- $S_2 \to AB_1 c$
- $B_1 \to \bar{b} B_1 c \mid \varepsilon$
- $S_2' \to AbB_1'$
- $B_1' \to bB_1'\bar{c} \mid \varepsilon$

Řešení příkladu 6.9. Takový deterministický zásobníkový automat neexistuje.

Řešení příkladu 6.10. Takový zásobníkový automat neexistuje.

Kapitola 7

Řešení příkladu 7.1.

Zásobník	Vstup	Pravidlo
$\$E$	$i+i\$$	$1\colon E \to TE'$
$\$E'T$	$i+i\$$	
$\$E'T'F$	$i+i\$$	$4\colon T \to FT'$
$\$E'T'F$	$i+i\$$	$8\colon F \to i$
$\$E'T'i$	$i+i\$$	
$\$E'T'$	$+i\$$	$6\colon T' \to \varepsilon$
$\$E'$	$+i\$$	$2\colon E' \to +TE'$
$\$E'T+$	$+i\$$	
$\$E'T$	$i\$$	$4\colon T \to FT'$
$\$E'T'F$	$i\$$	$8\colon F \to i$
$\$E'T'i$	$i\$$	
$\$E'T'$	$\$$	$6\colon T' \to \varepsilon$
$\$E'$	$\$$	$3\colon E' \to \varepsilon$
$\$$	$\$$	

Výsledkem prediktivní syntaktické analýzy je (1) informace, zda byla analýza úspěšná, čili zda lze vstupní řetězec vygenerovat danou gramatikou, a (2) levý rozbor. V tomto příkladu syntaktická analýza proběhla úspěšně a levý rozbor je 148624863.

Řešení příkladu 7.2.

	a	b	c	$\$$
S	1			
A	2	3	3	
B		4	5	
C			6	

Ano, G je LL gramatika.

Řešení příkladu 7.3. 12456

Řešení příkladu 7.4.

	a	b	c	$\$$
S		1	1	
A		2	$2,3$	
B	5	4	5	5
C			6	

Ne, G není LL gramatika.

Řešení příkladu 7.5. 1245645

Řešení příkladu 7.6. $H = \big(\{X, X', Y, Z\}, \{a, b, c\}, P, X\big)$, kde P obsahuje následující pravidla:

$X \to aX'$ $\quad X' \to \varepsilon$ $\quad Z \to bb$

$X' \to bY$ $\quad Y \to cZ$

Řešení příkladu 7.7. $H = \big(\{X, X', Y, Z\}, \{a, b, c\}, P, X\big)$, kde P obsahuje následující pravidla:

$X \to cX'$ $\qquad$ $X' \to \varepsilon$ $\qquad$ $Y \to \varepsilon$

$X' \to abZX'$ $\qquad$ $Z \to YaaY$

Kapitola 8

Řešení příkladu 8.1.

Zásobník	Operace	Vstup	Pravidlo
$\$$	$<$	$i * (i + i)\$$	
$\$ < i$	$>$	$*(i + i)\$$	$4\colon E \to i$
$\$E$	$<$	$*(i + i)\$$	
$\$ < E*$	$<$	$(i + i)\$$	
$\$ < E* < ($	$<$	$i + i)\$$	
$\$ < E* < (< i$	$>$	$+i)\$$	$4\colon E \to i$
$\$ < E* < (E$	$<$	$+i)\$$	
$\$ < E* < (< E+$	$<$	$i)\$$	
$\$ < E* < (< E+ < i$	$>$	$)\$$	$4\colon E \to i$
$\$ < E* < (< E + E$	$>$	$)\$$	$1\colon E \to E + E$
$\$ < E* < (E$	$=$	$)\$$	
$\$ < E* < (E)$	$>$	$\$$	$3\colon E \to (E)$
$\$ < E * E$	$>$	$\$$	$2\colon E \to E * E$
$\$E$		$\$$	

Výsledkem precedenční analýzy je (1) informace, zda byla analýza úspěšná, čili zda lze vstupní řetězec vygenerovat danou gramatikou, a (2) pravý rozbor. V tomto příkladu syntaktická analýza proběhla úspěšně a pravý rozbor je 444132.

Řešení příkladu 8.2.

	$\diamond$	$\bullet$	$\sqcap$	(	)	i	$\$$
$\diamond$	$>$	$<$	$<$	$<$	$>$	$<$	$>$
$\bullet$	$>$	$>$	$<$	$<$	$>$	$<$	$>$
$\sqcap$	$>$	$>$	$<$	$<$	$>$	$<$	$>$
(	$<$	$<$	$<$	$<$	$=$	$<$	
)	$>$	$>$	$>$		$>$		$>$
i	$>$	$>$	$>$		$>$		$>$
$\$$	$<$	$<$	$<$	$<$		$<$	

Řešení příkladu 8.3. 55355142

Řešení příkladu 8.4.

Zásobník	Stav	Vstup	Akce	Pravidlo
$\langle \$, 0\rangle$	0	$i + i\$$	$\alpha[0, i] = s5$	
$\langle \$, 0\rangle\langle i, 5\rangle$	5	$+i\$$	$\alpha[5, +] = r6, \beta[0, F] = 3$	$6\colon F \to i$
$\langle \$, 0\rangle\langle F, 3\rangle$	3	$+i\$$	$\alpha[3, +] = r4, \beta[0, T] = 2$	$4\colon T \to F$
$\langle \$, 0\rangle\langle T, 2\rangle$	2	$+i\$$	$\alpha[2, +] = r2, \beta[0, E] = 1$	$2\colon E \to T$
$\langle \$, 0\rangle\langle E, 1\rangle$	1	$+i\$$	$\alpha[1, +] = s6$	
$\langle \$, 0\rangle\langle E, 1\rangle\langle +, 6\rangle$	6	$i\$$	$\alpha[6, i] = s5$	
$\langle \$, 0\rangle\langle E, 1\rangle\langle +, 6\rangle\langle i, 5\rangle$	5	$\$$	$\alpha[5, \$] = r6, \beta[6, F] = 3$	$6\colon F \to i$
$\langle \$, 0\rangle\langle E, 1\rangle\langle +, 6\rangle\langle F, 3\rangle$	3	$\$$	$\alpha[3, \$] = r4, \beta[6, T] = 9$	$4\colon T \to F$
$\langle \$, 0\rangle\langle E, 1\rangle\langle +, 6\rangle\langle T, 9\rangle$	9	$\$$	$\alpha[9, \$] = r1, \beta[0, E] = 1$	$1\colon E \to E + T$
$\langle \$, 0\rangle\langle E, 1\rangle$	1	$\$$	$\alpha[1, \$] = ☺$	

Pravý rozbor řetězce $i + i$ je 642641.

Kapitola 9

Řešení příkladu 9.1. K pravidlům v gramatice jsou přiřazeny tzv. *sémantické akce*, které jsou vykonávány při aplikaci daného pravidla při syntaktické analýze. Mezi tyto akce patří např. generování vnitřního kódu, práce s tabulkou symbolů či jakákoliv jiná akce, která je potřeba. Samotný překlad je tudíž řízen syntaxí programu, která udává akce, které se provedou.

Řešení příkladu 9.2.

(1) Syntaktický analyzátor vytvoří abstraktní syntaktický strom, který je převeden na (3AK).

(2) Syntaktický analyzátor vytvoří postfixovou reprezentaci programu, která je převedena na 3AK.

(3) Syntaktický analyzátor vytvoří 3AK přímo.

Kapitola 10

Řešení příkladu 10.1. První základní blok:

```
        int a = 0;
```

Druhý základní blok:

```
    L1:
        a += 1;
        printf("%d", a);
```

Třetí základní blok:

```
    L2:
        if (a < 5) goto L1;
```

Čtvrtý základní blok:

```
        a = rand();
```

Pátý základní blok:

```
    L3:
        if (a > 10) goto L2;
```

Šestý základní blok:

```
    L4:
        printf("%d", a);
```

Řešení příkladu 10.2. Není, protože výsledný kód není funkčně ekvivalentní původnímu (výsledek výrazu `b = c / a * b` se obecně může lišit od výsledku výrazu `b = c / AB;`, protože v prvním případě se provede nejdříve dělení a pak až násobení, ale ve druhém případě se dělí vynásobená hodnota).

Řešení příkladu 10.3.

Řádek	Instrukce	Stav	Další použití
1	`v = a / c`	`a,c,v:L`	`a:N; c:3; v:2`
2	`w = v - b`	`b,w:L; v:D`	`b,v:N; w:3`
3	`u = w * c`	`c,u,w:L`	`c:N; u,w:4`
4	`d = u + w`	`d:L; u,w:D`	`d,u,w:N`

Kapitola 11

Řešení příkladu 11.1. Nechť L je regulární jazyk. Pak existuje $k \geq 1$ takové, že pokud $z \in L$ a $|z| \geq k$, pak existují u, v a w takové, že $z = uvw$ a jsou splněny následující tři vlastnosti:

(1) $v \neq \varepsilon$,

(2) $|uv| \leq k$,

(3) pro každé $i \geq 0$ platí, že $uv^iw \in L$.

Řešení příkladu 11.2. Protože pumping lemma představuje pouze nutnou podmínku pro to, aby daný jazyk byl regulární. Jinými slovy, každý regulární jazyk tuto podmínku splňuje, ale existují i některé ne-regulární jazyky, které ji taktéž splňují.

Řešení příkladu 11.3. Postupujte obdobně jako v řešeních příkladů v materiálech ke třetímu demonstračnímu cvičení [3].

Řešení příkladu 11.4. Nechť $M_1 = (Q_1, \Sigma_1, R_1, s_1, F_1)$ a $M_2 = (Q_2, \Sigma_2, R_2, s_2, F_2)$ jsou dva konečné automaty. Bez újmy na obecnosti můžeme předpokládat, že $Q_1 \cap Q_2 = \emptyset$ (množiny stavů obou automatů jsou disjunktní) a že oba automaty jsou úplné. Sestrojme konečný automat

$$M = \big(Q, \Sigma, R, s, F\big)$$

kde

- $Q = Q_1 \cup Q_2 \cup \{s\}$, kde s je nový stav,
- $\Sigma = \Sigma_1 \cup \Sigma_2$,
- $R = R_1 \cup R_2 \cup \{s \rightarrow s_1,\ s \rightarrow s_2\}$,
- $F = F_1 \cup F_2$.

Zřejmě $L(M) = L(M_1) \cup L(M_2)$. Rigorózní důkaz identity $L(M) = L(M_1) \cup L(M_2)$ by se prováděl indukcí a je nad rámec předmětu IFJ.

Řešení příkladu 11.5. Nechť $M_1 = (Q_1, \Sigma_1, R_1, s_1, F_1)$ a $M_2 = (Q_2, \Sigma_2, R_2, s_2, F_2)$ jsou dva konečné automaty. Bez újmy na obecnosti můžeme předpokládat, že $Q_1 \cap Q_2 = \emptyset$ (množiny stavů obou automatů jsou disjunktní) a že oba automaty jsou úplné. Sestrojme konečný automat

$$M = \big(Q, \Sigma, R, s_1, F_2\big)$$

kde

- $Q = Q_1 \cup Q_2$,
- $\Sigma = \Sigma_1 \cup \Sigma_2$,

- $R = R_1 \cup R_2 \cup \{f \to s_2 \mid f \in F_1\}$.

Zřejmě $L(M) = L(M_1)L(M_2)$. Rigorózní důkaz identity $L(M) = L(M_1)L(M_2)$ by se prováděl indukcí a je nad rámec předmětu IFJ.

Řešení příkladu 11.6. Nechť $M_1 = (Q_1, \Sigma_1, R_1, s_1, F_1)$ a $M_2 = (Q_2, \Sigma_2, R_2, s_2, F_2)$ jsou dva konečné automaty. Bez újmy na obecnosti můžeme předpokládat, že $Q_1 \cap Q_2 = \emptyset$ (množiny stavů obou automatů jsou disjunktní) a že oba automaty jsou úplné.

Idea důkazu je taková, že budeme zároveň simulovat oba automaty, a řetězec přijmeme, pokud by jej přijal jak M_1, tak M_2. Sestrojme konečný automat

$$M = \big(Q, \Sigma, R, s, F\big)$$

kde

- $Q = Q_1 \times Q_2$ ($\times$ označuje kartézský součin [4]),
- $\Sigma = \Sigma_1 \cup \Sigma_2$,
- $R = \big\{(p_1, p_2)a \to (q_1, q_2) \mid p_1 a \to q_1 \in R_1,\ p_2 a \to q_2 \in R_2\big\}$,
- $s = (s_1, s_2)$
- $F = \big\{(f_1, f_2) \mid f_1 \in F_1,\ f_2 \in F_2\big\}$.

Zřejmě $L(M) = L(M_1) \cap L(M_2)$. Rigorózní důkaz identity $L(M) = L(M_1) \cap L(M_2)$ by se prováděl indukcí a je nad rámec předmětu IFJ.

Řešení příkladu 11.7. Nechť $M = (Q, \Sigma, R, s, F)$ je konečný automat. Bez újmy na obecnosti můžeme předpokládat, že M je úplný. Sestrojme konečný automat

$$N = \big(Q', \Sigma, R', s', F'\big)$$

kde

- $Q' = Q \cup \{s'\}$, kde s' je nový stav,
- $R' = \{qa \to p \mid pa \to q \in R\}$,
- $F' = \{s\}$.

Zřejmě $L(N) = \text{reversal}\big(L(M)\big)$. Rigorózní důkaz identity $L(N) = \text{reversal}\big(L(M)\big)$ by se prováděl indukcí a je nad rámec předmětu IFJ.

Kapitola 12

Řešení příkladu 12.1. $G' = \big(\{S, B, C, \bar{a}, \langle Ca\rangle, \langle CC\rangle\}, \{a, b, c\}, P', S\big)$, kde P' obsahuje pravidla

- $S \to \langle Ca\rangle B$
- $\langle Ca\rangle \to C\bar{a}$
- $\bar{a} \to a$
- $B \to \langle CC\rangle\langle CC\rangle \mid b$
- $\langle CC\rangle \to CC$
- $C \to c$

Řešení příkladu 12.2. $G' = \big(\{S, A, B, \bar{a}, \bar{b}\}, \{a, b\}, P', S\big)$, kde P' obsahuje pravidla

- $S \to a\bar{a}BA$
- $\bar{a} \to a$
- $A \to bB\bar{b}\bar{b}$
- $\bar{b} \to b$
- $B \to a\bar{b}$

Řešení příkladu 12.3. Nechť L je bezkontextový jazyk. Pak existuje $k \geq 1$ takové, že pokud $z \in L$ a $|z| \geq k$, pak existují u, v, w, x a y takové, že $z = uvwxy$ a jsou splněny následující tři vlastnosti:

(1) $vx \neq \varepsilon$,

(2) $|vwx| \leq k$,

(3) pro každé $i \geq 0$ platí, že $uv^iwx^iy \in L$.

Řešení příkladu 12.4. Protože pumping lemma představuje pouze nutnou podmínku pro to, aby daný jazyk byl bezkontextový. Jinými slovy, každý bezkontextový jazyk tuto podmínku splňuje, ale existují i některé ne-bezkontextové jazyky, které ji taktéž splňují.

Řešení příkladu 12.5. Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Sestrojme bezkontextovou gramatiku $H = (N, T, P', S)$, kde

$$P' = \{A \to \text{reversal}(x) \mid A \to x \in P\}$$

Zřejmě $L(H) = \text{reversal}\big(L(G)\big)$. Rigorózní důkaz identity $L(H) = \text{reversal}\big(L(G)\big)$ by se prováděl indukcí a je nad rámec předmětu IFJ.

Kapitola 13

Řešení příkladu 13.1. Turingův stroj nejdříve zkontroluje, zda je vstup tvořen posloupností a následovanou posloupností b a končící posloupností c. Pokud tomu tak není, pak vstup zamítne. Po kontrole se přesune zpět na začátek vstupu. Nyní přepíše nejlevější a na A, poté nejlevější b na B, a nakonec nejlevější c na C. Pokud některý z těchto přepisů nelze provést (např. chybí b), pak vstup zamítne. Toto se opakuje tak dlouho, dokud se na vstupu nevyskytují žádné symboly a, b a c, což znamená, že vstup byl korektní, a stroj vstupní řetězec přijme. V opačném případě vstup zamítne (např. přebývají symboly c).

Řešení příkladu 13.2. $G = \big(\{S, A, A', X, X_a, X_b, X_\varepsilon\}, \{a, b\}, P, S\big)$, kde P obsahuje pravidla

- $S \to AXA'$
- $AX \to aAX_a \mid bAX_b \mid X_\varepsilon$
- $X_aA' \to XaA'$
- $X_bA' \to XbA'$
- $X_\varepsilon A' \to \varepsilon$
- $aX \to Xa$
- $bX \to Xb$
- $aX_a \to X_aa$
- $bX_a \to X_ab$
- $aX_b \to X_ba$
- $bX_b \to X_bb$
- $aX_\varepsilon \to X_\varepsilon a$
- $bX_\varepsilon \to X_\varepsilon b$

Neformálně, G funguje tak, že po každém vygenerovaném symbolu vlevo od A se pomocí X vygeneruje příslušný symbol vlevo od A'. Pokud se např. vygeneruje a, pak se X změní na X_a. Toto X_a se pak přesune od A až k A'. Po vygenerování a před A' se X_a změní zpět na X, přesune se zpátky k A, a může být vygenerován další symbol. Po ukončení generování dojde ke zrušení všech tří neterminálů A, X a A'.

Řetězec $aabaab$ lze vygenerovat následovně:

$$
\begin{aligned}
S &\Rightarrow AXA' \\
&\Rightarrow aAX_aA' \\
&\Rightarrow aAXaA' \\
&\Rightarrow aaAX_aaA' \\
&\Rightarrow aaAaX_aA' \\
&\Rightarrow aaAaXaA' \\
&\Rightarrow aaAXaaA' \\
&\Rightarrow aabAX_baaA' \\
&\Rightarrow aabAaX_baA' \\
&\Rightarrow aabAaaX_bA' \\
&\Rightarrow aabAaaXbA' \\
&\Rightarrow aabAaXabA' \\
&\Rightarrow aabAXaabA' \\
&\Rightarrow aabX_\varepsilon aabA' \\
&\Rightarrow aabaX_\varepsilon abA' \\
&\Rightarrow aabaaX_\varepsilon bA' \\
&\Rightarrow aabaabX_\varepsilon A' \\
&\Rightarrow aabaab
\end{aligned}
$$

Řešení příkladu 13.3. Jelikož je každá bezkontextová gramatika zároveň neomezenou gramatikou, tak řešením je gramatika

$$G = \big(\{S\}, \{a, b\}, P, S\big)$$

kde P obsahuje pravidla $S \to aSb$ a $S \to ab$.

Řešení příkladu 13.4. $G = \big(\{S,\ X\},\ \{a,\ b,\ c\},\ P,\ S\big)$, kde P obsahuje pravidla

- $S \to aaX$
- $X \to abcX \mid ac \mid ca$

Řešení příkladu 13.5. (a) Např. $\{a^n \mid n \geq 0\}$ nebo $\{a, b, c\}$.

(b) Např. $\{a^n b^n \mid n \geq 0\}$ nebo $\big\{w\,\mathrm{reversal}(w) \mid w \in \{a, b\}^*\big\}$.

(c) Např. $\{a^n b^n c^n \mid n \geq 0\}$ nebo $\big\{ww \mid w \in \{a, b\}^*\big\}$.

Řešení příkladu 13.6. $G = \big(\{S,\ A,\ B,\ B_r,\ C\},\ \{0\},\ P,\ S\big)$, kde P obsahuje následující pravidla:

- $S \to CS \mid B_r$
- $CA \to BC$
- $CB \to ABC$
- $CB_r \to AB_r$
- $A \to 0$
- $B \to 0$
- $B_r \to 0$

Řešení je převzato z [1] (v tomto článku je taktéž vysvětleno, jak G funguje).

Literatura

[1] Holzer, M.; Rossmanith, P.: A simpler grammar for Fibonacci numbers. *The Fibonacci Quarterly*, ročník 35, č. 5, 1996: s. 465–466.
Dostupné na URL: `http://www.fq.math.ca/Scanned/34-5/holzer.pdf`

[2] Meduna, A.: *Automata and Languages: Theory and Applications*. Springer, Londýn, 2000, ISBN 978-1-85233-074-3.

[3] Meduna, A.; Lukáš, R.: Přednášky z předmětu Formální jazyky a překladače [online]. 2011, [cit. 2011-09-01].
Dostupné na URL: `http://www.fit.vutbr.cz/study/courses/IFJ/public/materials/`

[4] Wikipedia: Cartesian product [online]. 2011, [cit. 2011-09-09].
Dostupné na URL: `http://en.wikipedia.org/wiki/Cartesian_product`

[5] Wikipedia: Fermat's Last Theorem [online]. 2011, [cit. 2011-08-27].
Dostupné na URL: `http://en.wikipedia.org/wiki/Fermat_last_theorem`

[6] Wikipedia: Fibonacci number [online]. 2011, [cit. 2011-09-09].
Dostupné na URL: `http://en.wikipedia.org/wiki/Fibonacci_number`