Masarykova univerzita
Fakulta infomatiky

Rozhraní pro práci s konečnými automaty

Bakalářská práce

Viktor Sýkora

Brno, podzim 2008

Vedoucí práce: Mgr. Jaroslav Hrdina, Ph.D.

Prohlášení

Prohlašuji, že tato bakalářská práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Shrnutí

Cílem bakalářské práce je vytvořit hybridní studijní materiál - hybridní ve smyslu kombinace e-learningové pomůcky a sbírky příkladů. Tématem jsou vybrané kapitoly z látky regulárních jazyků. Práce se zaměřuje na studenty se specifickými nároky, kontrétně na nevidomé. Cílem je zlepšit možnosti studia nevidomých a poskytnout jim k tomu účinný nástroj.

Klíčová slova:

Regulární jazyk, konečný automat, lemma o vkládání, minimalizace konečného automatu, Nerodova věta, determinizace konečného automatu, odstranění ϵ-kroků konečného automatu, uzávěrové vlastnosti regulárních jazyků.

Obsah

Úvod 4
Popis sbírky příkladů 5
Popis doplňkového programu 6
Základní pojmy 9
Konstrukce konečných automatů 13
Lemma o vkládání pro regulární jazyky 23
Minimalizace konečných automatů 25
Nerodova věta 28
Determinizace konečných automatů 29
Odstranění ε-kroků 32
Uzávěrové vlastnosti 34
Řešení příkladů 34
Závěr 51
Příloha - obsah CD 52
Použité zdroje 53

Úvod

Tento hybridní studijní materiál nabízí spojení e-learningové pomůcky a běžné sbírky příkladů k problematice regulárních jazyků. Přesněji, poskytuje nutné teoretické základy, sadu řešených příkladů a jejich ekvivalent zadání v doprovodném programu. Nemá za cíl nahradit běžná učební skripta, ale naopak nabídnout spojení mezi teoretickými podklady a klasickou sbírkou příkladů. Takový materiál je zvlášť vhodný pro studenty se zrakovým postižením, protože student nemusí mít k dispozici několik různých zdrojů, které je nutné upravovat do potřebné formy. Cílem je tedy spojit teoretickou a praktickou část výuky, za účelem komfortnějšího studia zrakově postižených studentů.

I když je materiál určen především pro handicapované studenty, mohou jej stejně dobře využívat i studenti ostatní. Ke všem cvičením jsou k dispozici podrobná řešení, což umožňuje nejen ověření správnosti, ale i možnost řešení dvojím směrem, viz. kapitola Konstrukce konečných automatů.

Stěžejní částí bakalářské práce je e-learningový program, jakožto doplněk ke sbírce. Předpokladem je, že sbírka je studijní materiál a program nevidomým studentům nahrazuje "pero a papír". Spolu pak tvoří silný nástroj pro zvládnutí abstraktní látky regulárních jazyků. Umožňuje vytvořit libovolný konečný, deterministický automat a zobrazit jej na vizuální rozhraní, respektivě linearizovat do podoby vhodné pro nevidomé uživatele, s výstupem na Braillský řádek. Obě rozhraní, vizuální, respektivě řádkové, mají stejnou vyjadřovací sílu. Uživatel může vytvořit automat, upravovat jej, použít algoritmy pro výpočet slova nebo minimalizace automatu. Rozhraní bylo vytvořeno s důrazem na jednoduchost a intuitivnost, avšak dodržující standardy pro uživatelské rozhraní.

Cvičení týkající se konečných automatů jsou kompletně převedeny do programu a uživatel tak může souběžně pracovat se zadáním a zároveň tvořit řešení, bez nutnosti dalších zdrojů.

Program se ukazuje jako silná pomůcka nejen pro nevidomé studenty, ale i pro jejich lektory. Dosavadní způsob výuky konstrukce konečných automatů je zatěžován problémy se zápisem automatů a jejich následnou kontrolou. V tomto případě je program prostředníkem mezi vnímáním nevidomého studenta a jeho lektora. Nabízí možnost souběžné práce na daném příkladu, provedené změny se promítají na obě rozhraní. Lektor tedy pracuje s grafovým zápisem přechodové funkce, zatímco student čte lineární výstup na Braillském řádku. Lektor tak může jednoduše vidět, co student právě dělá a naopak. Tímto se rapidně zvyšuje efektivnost výuky.

Popis sbírky příkladů

Sbírka je doplňkovým studijním materiálem k teorii regulárních jazyků. Obsahuje řešené příklady k vybraným kapitolám: základní pojmy, konstrukce konečných automatů, lemma o vkládání pro regulární jazyky, minimalizace konečných automatů, Nerodova věta, determinizace konečných automatů, odstraňování ϵ-kroků a uzávěrové vlastnosti. Každá část je uvedena řešeným příkladem, ve kterém je podrobně vysvětlen postup řešení daného typu příkladů. Sbírka je určena všem studentům, jako procvičení látky regulárních jazyků.

V první kapitole jsou stručně popsány základní pojmy, jejichž znalost je nezbytná pro řešení příkladů. Část věnovaná jazykům poskytuje základní znalosti při práci s tímto formalismem: operace nad slovy, tj. zřetězení, mocnina, obraz, respektivě operace nad jazyky: množinové operace, zřetězení, mocniny, iterace, doplněk a obraz. V další podkapitole je definován konečný automat, způsoby zápisu přechodové funkce a související pojmy: rozšířená přechodová funkce a akceptování slova respektivě jazyka. Následuje zavedení pojmu gramatika a regulární gramatika. Dále jsou uvedeny pojmy pro dokazování neregulárnosti jazyka - lemma o vkládání, respektivě regulárnosti - Nerodova věta.

Druhá kapitola se zabývá konstrukcí konečných automatů. Je rozdělena do čtyř tématicky podobných cvičení. První jsou zaměřeny na tvorbu konečného automatu podle daného jazyka, přičemž jazyky v každém cvičení jsou specificky zadané: množinovými operacemi, zřetězením a iterací v první části, zatímco konkrétním počtem písmen ve slovech v části druhé, respektivě jazyky obsahující slova s určitým podslovem v části třetí. V příkladech poslední části kapitoly se postupuje opačně: automat je již zadán a úkolem je nalézt jazyk, který akceptuje. Hledané jazyky jsou podobné jazykům z předchozích cvičení.

Ve třetí kapitole Lemma o vkládání pro regulární jazyky jsou příklady na dokazování neregulárnosti daného jazyka. Řešený příklad krok po kroku ukazuje použití lemmatu o vkládání, od výběru slova po rozhodnutí, zda patří výsledné slovo do daného jazyka. Ke všem příkladům v této kapitole je k dispozici podrobné řešení, včetně výběru vhodného slova a jednotlivých konstant.

Čtvrtá kapitola se věnuje problematice minimalizace automatů. V řešeném příkladu jsou popsány kroky algoritmu, který ze vstupního automatu vytvoří tzn. redukt.

Pátá kapitola stručně přibližuje dokazování regulárnosti jazyků pomocí Nerodovy věty. K dispozici je opět podrobné řešení.

V šesté a sedmé kapitole jsou příklady na determinizaci, respektivě na odstraňování ϵ-kroků. Podobně jako v kapitole o minimalizaci, je i zde uveden řešený příklad, podrobně popisující jednotlivé algoritmy.

Poslední kapitola se věnuje uzávěrovým vlastnostem regulárních jazyků. K řešení příkladů je nutné znát, které operace jsou uzavřené na množinu těchto jazyků.

Druhou část sbírky tvoří řešení všech uvedených příkladů. Rozsah řešení daného příkladu je závislý na složitosti problému.

Popis doplňkového programu

Doplňkový program spolu se sbírkou tvoří hybridní studijní materiál, který usnadňuje učební proces všem studentům dané látky. Zvlášť přínosný je pro studenty nevidomé, kterým umožňuje pracovat s konečnými automaty efektivněji než doposud. Mají k dispozici výstup na Braillský řádek, na němž se zobrazuje linearizovaná podoba konečného automatu. Tato kapitola popisuje práci s přiloženým programem. Podrobně rozebírá jednotlivé přístupy tvorby automatů, tj. pomocí řádkového, respektivě vizuálního rozhraní.

Práce s automatem v řádkovém rozhraní

Toto rozhraní je určeno nevidomým studentům. Umožňuje pracovat s konečným automatem, s jeho linearizovanou podobou a pomocí klávesnice jej upravovat.

Přehled ovládání

- Založení nového automatu - *Ctrl+n.*
- Otevření existujícího automatu - *Ctrl+o.*
- Přepnutí na jiný otevřený automat - *Ctrl + šipka vlevo, vpravo nebo funkční klávesy F1 až F8.*
- Procházení automatem - *Šipka dolů, šipka nahoru.*
- Návrat do předchozí nabídky, respektivě do hlavní - *Klávesa Esc.*
- Editovat přechody - *Písmeno e.*
- Přidat stav - *Písmeno s.*
- Přidat písmeno - *První možnost: Písmeno s + šipka dolů nebo Písmeno e + šipka nahoru.*
- Přehled stavů - *Písmeno q.*
- Nastavení počátečního, respektivě koncového stavu - *Písmeno q + p, respektivě k.*
- Smazání stavu - *Písmeno q + d.*
- Zobrazení definice a názvu automatu - *Písmeno x.*
- Výpočet slova - *Písmeno w.*
- Minimalizace - *Písmeno m.*

Ukázkový příklad

Vytvořme automat $M = (\{1, 2, 3\}, \{a, b\}, \delta, 1, \{3\})$, který akceptuje jazyk všech slov nad Σ^*, které obsahují podslovo **ab**.

1. Založme nový automat (*Ctrl+n*): zobrazí se dialogové okno, ve kterém zadáme název automatu, například: **Podslovo ab**; klávesou *Enter* se nový automat uloží a dialogové okno zavře.

2. Nový automat nyní neobsahuje žádné stavy, žádná písmena abecedy a přechodová funkce není definovaná. Vstupní/výstupní řádek (dále jen řádek) nás nejdříve navede na přidání stavů, pak písmen abecedy a definici přechodové funkce. Na řádku se zobrazil text: *Přidat stavy? a/n* - stisk písmena *a* změní text řádku na *Zadejte názvy:* - nyní můžeme zadat názvy stavů, oddělené čárkami: **1,2,3** + klávesa *Enter*. Vytvořili jsme tedy 3 stavy.

3. Na řádku je nyní text *Přidat písmena? a/n*. Postupujeme obdobně; formát zadaných písmen může vypadat takto: *a,b* nebo *ab* + *Enter*.

4. Nyní jsme schopni definovat přechodovou funkci. Všechny přechody mají zatím tvar *1 ->a N* apod. Ze stavu 1 se tedy pod písmenem *a* dostaneme do *N*, což znamená, že přechod není definovaný. Šipkami nahoru a dolů se můžeme pohybovat po jednotlivých přechodech a editovat je: *Editovat 1 ->a N? a/n* po stisku klávesy *a* zadáme název cílového stavu; v našem případě chceme akceptovat všechna slova s podslovem **ab**, proto pod písmenen *a* zadáme stav *2* + *Enter* a obdobně pod *b* do stavu *1*. Editujeme i přechody ostatních stavů: 2 ->a 2, 2 ->b do 3, 3 ->a 3, 3 ->b 3.

5. Definovali jsme přechodovou funkci. Po celou dobu jsme byli v editačním módu, tj. mohli jsme libovolně přidávat stavy, písmena a přechody. Nyní se přepneme do módu procházecího - stisk *Esc*. Na řádku se teď zobrazují přechody podle jednotlivých stavů, podobně jako v tabulkovém zápisu automatu. Mezi editačním a procházecím módem můžeme kdykoli přecházet: stisk *e* nás přepne na editaci přechodů a naopak *Esc* zpět na přehled.

6. Definovali jsme přechodovou funkci, nyní je nutné označit počáteční, respektivě koncové stavy. Klávesa *q* nás přepne na přehled stavů. Vertikálním pohybem šipek se zobrazují názvy stavů. Stiskem *p* se aktuální stav nastaví jako počáteční; stiskem *k* se aktuální stav nastaví jako koncový. Nastavme tedy stav *1* jako počáteční a stav *3* jako koncový. (Pokud bychom chtěli nějaký stav smazat, můžeme to nyní udělat pomocí klávesy *d*). *Esc* do předchozí nabídky.

7. Automat je nyní kompletní. Můžeme použít algoritmus na výpočet slova nebo minimalizaci.

8. **Výpočet slova:** stisk *w*. Zadejme slovo *bbabba*; řádek zobrazí jednotlivé přechody a na konci *ok* - slovo je akceptováno. Pokud by automat slovo neakceptoval, místo *ok* by se zobrazila *chyba* v místě, kde by automat nemohl přejít dále.

9. **Minimalizace:** stisk *m*. Vytvoří se a otevře nový automat, který je reduktem původního. K původnímu přejdeme pomocí *Ctrl + šípka vlevo*. Nový automat je označen přívlastkem minimalizovaný. V našem případě: *Podslovo ab - minimalizovaný*.

Poznámky: upravované automaty se ukládají po každém kroku. Nedochází tedy ke ztátě dat.

Práce s automatem ve vizuálním rozhraní

Ukázkový příklad

Vytvořme automat $M = (\{1, 2, 3\}, \{a, b\}, \delta, 1, \{3\})$, který akceptuje jazyk všech slov nad Σ^*, které obsahují podslovo **ab**.

1. V hlavní nabídce zvolíme Soubor -> Nový. Zobrazí se dialog, ve kterém zadáme název nového automatu: **Podslovo ab**.

2. Nyní máme prázdný automat. Přidáme po jednom stavy: buď pomocí tlačítka *Stav* v sekci *Přidat...* nebo pravým tlačítkem a volbou *Přidat stav*. Přidáme stavy *1*, *2* a *3*.
3. Přidáme písmena abecedy pomocí tlačítka *Písmeno* v sekci *Přidat...*.
4. Klepneme pravým tlačítkem myši na stav *1* a označíme jej jako počáteční. Stav *3* označíme jako koncový.
5. Nadefinujeme přechodovou funkci: buď pomocí tlačítka *Přechod* v sekci *Přidat...* nebo přetažením myši z jednoho stavu do druhého se stlačeným pravým tlačítkem.
6. Jakmile dokončíme přechodovou funkci můžemem použít algoritmy pro výpočet slova a minimalizaci (sekce *Alogoritmy* nebo hlavní menu).

Technické poznámky: Program vyžaduje operační systém Microsoft Windows 2000 a výš, s platformou Microsoft .NET Framework 3.5.

Základní pojmy

Jazyk

Definice jazyku

- **Abeceda** je libovolná konečná množina. Prvky nazýváme **znaky, písmena nebo symboly**.
- **Slovo** nad abecedou Σ je libovolná konečná posloupnost znaků z abecedy Σ.
- **Jazyk** nad abecedou Σ je libovolná množina slov nad Σ.
- ε je prázdné slovo, takové, že $\varepsilon \cdot \mathbf{u} = \mathbf{u} \cdot \varepsilon = \mathbf{u}$.

Operace nad slovy

- **Zřetězení slov** je operace definovaná předpisem $\mathbf{u} \cdot \mathbf{v} = \mathbf{uv}$.
- **i-tá mocnina slova** je definovaná induktivně:
  - $u^0 = \varepsilon$
  - $u^{(i+1)} = u \cdot u^i$
- **Zrcadlový obraz** slova $w = a_1..a_n$ je slovo $w^R = a_n..a_1$. Platí $(\varepsilon)^R = \varepsilon$

Operace nad jazyky

- Množinové operace **sjednocení** ($\cup$), **průnik** ($\cap$), **rozdíl** ($\setminus$).
- **Zřetězení** jazyků K, L je jazyk $K \cdot L = \{u \cdot v | u \in K, v \in L\}$. Platí $\emptyset \cdot L = L \cdot \emptyset = \emptyset$ a také $\{\varepsilon\} \cdot L = L \cdot \{\varepsilon\} = L$.
- **i-tá mocnina jazyka** je definovaná induktivně:
  - $L^0 = \{\varepsilon\}$
  - $L^{(i+1)} = L \cdot L^i$

  Platí: $\emptyset^0 = \{\varepsilon\}$; $\emptyset^i = \emptyset$ pro i $\in N$; $\varepsilon^i = \{\varepsilon\}$ pro i $\in N_0$
- **Iterace** jazyka L je jazyk $L^* = \bigcup_{i=0}^{\infty} L^i$. Platí $\emptyset = \{\varepsilon\}$.
- **Pozitivní iterace** jazyka L je jazyk $L^+ = \bigcup_{i=1}^{\infty} L^i$. Platí $\emptyset^* = \{\emptyset\}$.
- **Doplněk** jazyka L je jazyk $co - L = \Sigma^* \setminus L$.
- **Zrcadlový obraz** jazyka L je $L^R = \{w^R | w \in L\}$.
- Na všechny uvedené operace je třída regulárních jazyků **uzavřená**.

Lemma o vkládání (Pumping lemma)

L je **regulární** $\Rightarrow$
$\exists \in N$
$\forall w \in L.(|w| \geq n$
$\exists x, y, z.(w = xyz \wedge y \neq \varepsilon |xy| \leq n \wedge$
$\forall i \in N_0.xy^i z \in L))$

Nerodova věta

Nechť jazyk L je jazyk nad Σ. Potom jsou následující tvrzení ekvivalentní:

1. L je rozpoznatelný konečným automatem.
2. L je sjednocením některých tříd rozkladu určeného pravou kongruencí na Σ^* s konečným indexem.

Pravá kongruence

Relace $\sim$ je ekvivalence na Σ^*. Relace $\sim$ je **pravá kongruence**, pokud pro každé $u, v, w \in \Sigma^*$ platí: $u \sim v \Rightarrow uw \sim vw$.
Index relace $\sim$ je počet tříd rozkladu $\Sigma^*/\sim$

Chomského hierarchie jazyků

- **Typ 0 (frázové jazyky)**: pravidla v obecném tvaru.
- **Typ 1 (kontextové jazyky)**: každé pravidlo je tvaru $\alpha \rightarrow \beta$, kde $|\alpha| \leq |\beta|$ s výjimkou pravidla S $\rightarrow \varepsilon$, pokud se S nevyskytuje na pravé straně žádného pravidla.
- **Typ 2 (bezkontextové jazyky)**: každé pravidlo je tvaru A $\rightarrow \alpha$, kde $|\alpha| \geq 1$ s výjimkou pravidla S $\rightarrow \varepsilon$, pokud se S nevyskytuje na pravé straně žádného pravidla.
- **Typ 3 (regulární jazyky)**: každé pravidlo je tvaru A $\rightarrow$ aB nebo A $\rightarrow$ a s výjimkou pravidla S $\rightarrow \varepsilon$, pokud se S nevyskytuje na pravé straně žádného pravidla.

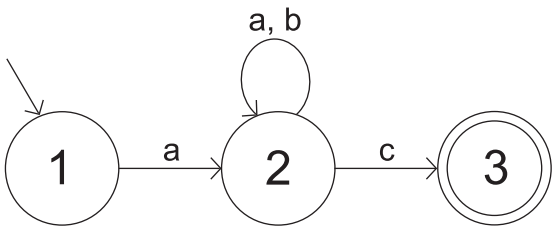
Konečné automaty

Definice: Konečný automat je pětice $(Q, \Sigma, \delta, q_0, F)$, kde

- Q je neprázdná konečná množina stavů
- Σ je vstupní abeceda
- $\delta : \ Q \times \Sigma \to Q$ je parciální přechodová funkce
- $q_0 \in Q$ je počáteční stav
- $F \subseteq Q$ je množina koncových stavů

Způsob zápisu přechodové funkce:

- Přechodový graf:



- Tabulka:

	a	b	c
$\to 1$	2	-	-
2	2	2	3
$\leftarrow 3$	-	-	-

Rozšířená přechodová funkce: $\hat{\delta} : Q \times \Sigma^* \to Q$ je parciální funkce definovaná induktivně vzhledem k délce slova ze Σ^*:

- $\hat{\delta}(q, \varepsilon) = q$ pro každý stav $q \in Q$.
- $\hat{\delta}(q, wa) = \begin{cases} \delta(\hat{\delta}(q, w), a)) & \text{je-li } \hat{\delta}(q, w) \text{ a } \delta(\hat{\delta}(q, w), a)) \text{ definováno.} \\ \perp & \text{jinak.} \end{cases}$

Akceptování:

- Slovo w je akceptováno automatem M právě když $\hat{\delta}(q_0, w) \in F$.
- Jazyk akceptovaný automatem M je $L(M) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$.

Gramatiky

Definice: Gramatika G je čtveřice (N, Σ, P, S), kde

- N je neprázdná konečná množina neterminálů.
- Σ je konečná množina terminálů, taková, že $N \cap \Sigma = \emptyset$. Množinu všech symbolů gramatiky $N \cup \Sigma$ značíme V
- P je konečná množina pravidel $\alpha \to \beta$, kde $\alpha \in V^*NV^*$ a $\beta \in V^*$.
- S je počáteční neterminál (kořen gramatiky).

Regulární gramatika: pravidla jsou tvaru A $\to$ aB nebo A $\to$ a s výjimkou pravidla S $\to \varepsilon$, pokud se S nevyskytuje na pravé straně žádného pravidla.

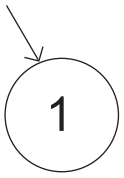
Konstrukce konečných automatů

V této kapitole si procvičíme konstrukci konečných automatů. V prvním cvičení se zaměříme na jazyky zadané pomocí operací **zřetězení, sjednocení, průnik, iterace, pozitivní iterace**. Tady si můžeme všimnout, že operace **zřetězení** vždy přidá nový stav, **iterace** tvoří smyčku nad stavem a **pozitivní iteraci** lze chápat jako $a \cdot a^*$. Ve druhém cvičení jsou jazyky zadány pomocí množinového zápisu a důraz je kladen na specifický počet písmen ve slovech. Třetí cvičení procvičuje zejména jazyky jejichž slova obsahují konkrétní podslovo. Všimněme si příkladu 3.*c*, který při řešení využívá komplementární automat. Ve cvičení 4 postupujeme obráceně a hledáme jazyk, který automat s přechodovou funkcí zadanou grafem a tabulkou akceptuje. V řešeném příkladu na straně 13 je popsán postup vytvoření odpovídajícího jazyka.

1. Sestrojte konečný automat rozpoznávající jazyky zadané pomocí operací **zřetězení, sjednocení, průnik, iterace, pozitivní iterace**.

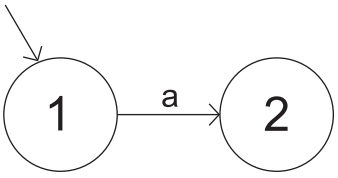
   - **Řešený příklad:** Konečný automat $M = (\{1, 2, 3\}, \{a, b\}, \delta, 1, \{3\})$ akceptující jazyk $\{a\} \cdot \{a \cup b\} \cdot \{a \cup b\}^*$
     Nejprve je třeba si ujasnit jaká slova konstruovaný automat akceptuje. V tomto případě jsou to slova začínající vždy na písmeno a, pak následuje písmeno a nebo b a končí libovolnou kombinací písmen a a b. Přechodovou funkci δ vytvoříme následovně:

     (a) Začneme počátečním stavem 1,

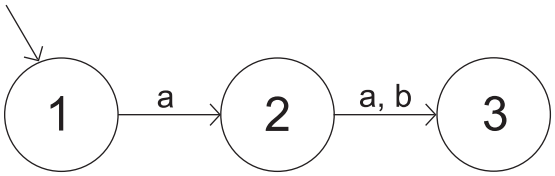
	a	***b***
$\rightarrow 1$	-	-

     (b) víme, že slovo musí začínat písmenem a, proto přidáme nový přechod a nový stav 2, ve kterém se automat nachází pokud první znak, který přečetl bylo a,

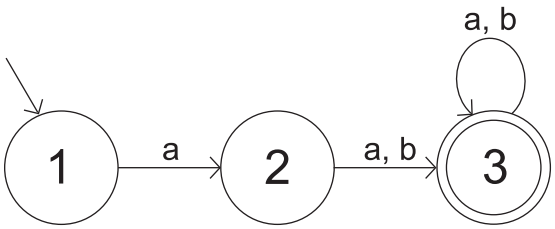
	a	***b***
$\rightarrow 1$	2	-
2	-	-

(c) nyní je automat ve stavu odpovídají jazyku $\{a\}$, následuje $\cdot(\{a\} \cup \{b\})$. Na vstupu tedy očekáváme a nebo b, aby se automat v této chvíli nezasekl, potřebuje přejít pod a nebo pod b do dalšího stavu a tedy přečíst vstupní písmeno,



	a	***b***
→ 1	2	-
2	3	3
3	-	-

(d) nyní na vstupu očekáváme $\cdot(\{a\} \cup \{b\})^*$, to znamená libovolnou kombinaci a a b. Zároveň přečtené už odpovídá jazyku L, proto jej můžeme akceptovat:



	a	***b***
→ 1	2	-
2	3	3
← 3	3	3

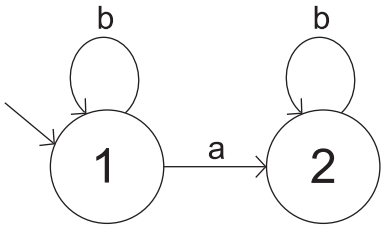
Příklady ke cvičení 1:

(a) $L = \{\{a\} \cdot \{a, b\}^* \cdot \{c\}\}$
(b) $L = \{(\{a\} \cup \{b\}) \cdot \{a\}\}$
(c) $L = \{\{b\} \cdot (\{a\}^+) \cdot (\{c\}^+) \cdot \{a\}\}$
(d) $L = \{\{a, b\}^2 \cdot (\{b, c\}^+) \cdot \{a\}^+ \cdot \{c\}\}$
(e) $L = \{(\{b\} \cdot \{c\})^+ \cup \{\{a\} \cdot (\{b, c\}^+)\}\}$

2. Sestrojte konečný automat rozpoznávající jazyk zadaný množinovým zápisem:

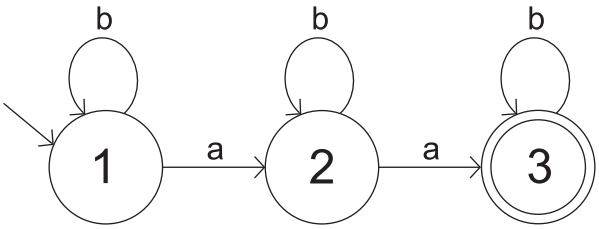
   - **Řešený příklad:** V tomto cvičení budeme konstruovat konečné automaty $M = (Q, \Sigma, \delta, q_0, \{q_F\})$ pro jazyky, jejichž slova budou obsahovat určitý počet písmen, například: $L = \{w | w \in \{a, b\}^* : \#_a(w) = 2\}$. Abychom zajistili akceptovaní jen slov s počtem písmen a právě 2, musí mít automat právě 2 přechody pod a, aby si přečtená písmena a mohl pamatovat. Na počet písmen b nemáme žádné požadavky, proto jich může být libovolně mnoho.

     (a) Automat může číst libovolná b, a po přečtení a se přesune do stavu 2, kde opět může číst libovolný počet b, tj. všechna slova nad Σ^* obsahující 1 písmeno a,



	a	***b***
→ 1	2	1
2	-	2

     (b) s dalším přečtením písmena a lze slovo už akceptovat nebo číst další b.



	a	***b***
→ 1	2	1
2	3	2
← 3	-	3

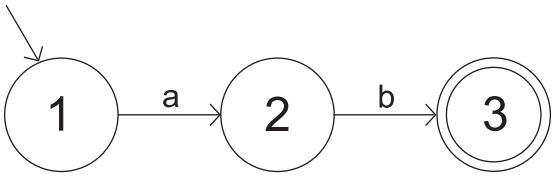
Příklady ke cvičení 2:

(a) $L = \{w | w \in \{a,b\}^* : \#_a(w) \geq 4\}$
(b) $L = \{w | w \in \{a,b\}^* : |w| \ mod \ 4 = 0\}$
(c) $L = \{w | w \in \{a,b\}^* : |w| \ mod \ 4 = 1\}$
(d) $L = \{w | w \in \{a,b\}^* : \#_a(w) = \#_b(w)\}$
(e) $L = \{w | w \in \{a,b\}^* : \#_a(w) = 3k; k \geq 0\}$
(f) $L = \{w | w \in \{a,b\}^* : \#_a(w)$ je sudý$\}$
(g) $L = \{w | w \in \{a,b,c\}^* : (\#_a(w) + \#_b(w)$ je sudý$)$ a w končí na „c“$\}$

3. Sestrojte konečný automat rozpoznávající jazyk zadaný množinovým zápisem slov obsahující konkrétní podslovo:

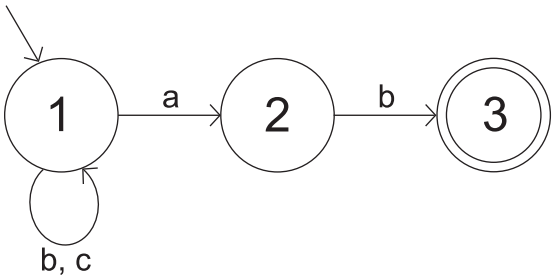
   - **Řešený příklad:** $L = \{w | w \in \{a, b, c\}^* : \text{w obsahuje podslovo } \mathbf{ab}\}$
     Při konstrukci přechodové funkce automatu, který akceptuje slova s určitým podslovem, nejdříve sestrojíme automat akceptující samotné podslovo a poté přechodovou funkci ztotálníme.

     (a) Automat akceptující slovo *ab*,



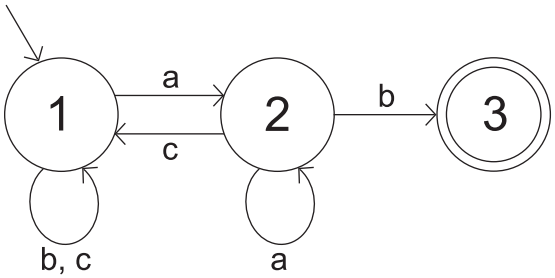
	a	***b***	***c***
→ 1	2	-	-
2	-	3	-
←3	-	-	-

     (b) postupně doplníme přechody pod ostatními písmeny,



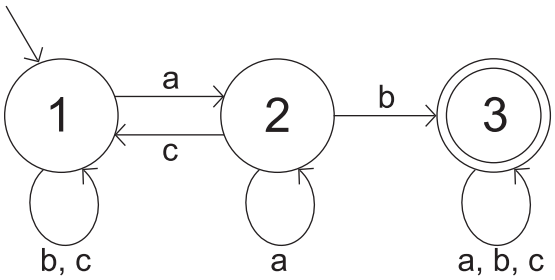
	a	***b***	***c***
→ 1	2	1	1
2	-	3	-
←3	-	-	-

     (c) přechod ze stavu 2 pod *c* by mohl porušit podslovo *ab*, proto vrátí automat do stavu 1, písmen *a* je možné přečíst libovolně mnoho a podslovo se neporuší,



	a	***b***	***c***
→ 1	2	1	1
2	2	3	1
←3	-	-	-

(d) ve stavu 3 už je zajištěno, že doposud přečtené slovo obsahuje *ab*, proto můžeme číst libovolný znak.



	a	***b***	***c***
→ 1	2	1	1
2	2	3	1
←3	3	3	3

Příklady ke cvičení 3:

(a) $L = \{w | w \in \{a, b, c\}^* : w$ obsahuje podslovo **acab**$\}$

(b) $L = \{w | w \in \{a, b\}^* : w$ obsahuje podslovo **abba**$\}$

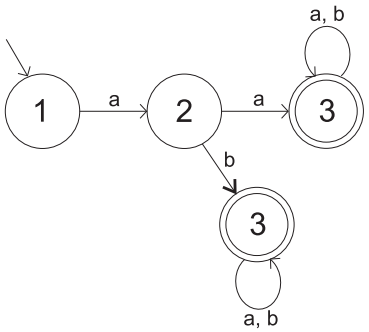
(c) $L = \{w | w \in \{a, b\}^* : w$ neobsahuje podslovo **abb**$\}$

(d) $L = \{w | w \in \{a, b\}^* : w$ je složeno pouze ze dvojic stejných písmen; jazyk obsahuje prázdné slovo$\}$

(e) $L = \{w | w \in \{a, b\}^* : w$ obsahuje podslovo **ac** nebo **bc**$\}$

4. Najděte jazyk, který akceptuje konečný automat s přechodovou funkcí zadanou grafem a tabulkou:

   - **Řešený příklad:** $M = (\{1,2,3,4\}, \{a,b\}, \delta, 1, \{3,4\})$



   Zjistíme jakými cestami se můžeme dostat do koncových stavů:

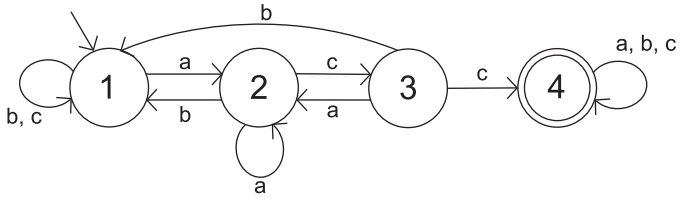
   (a) Po stavech 1, 2, 3 můžeme akceptovat $L_1 = \{a \cdot a \cdot (a,b)^*\}$, neboli všechna slova, která začínají dvěma písmeny a.

   (b) Po stavech 1, 2, 4 můžeme akceptovat $L_2 = \{a \cdot b \cdot (a,b)^*\}$, neboli všechna slova, která začínají dvěma písmeny ab.

   Výsledný jazyk, který akceptuje automat M je $L_1 \cup L_2$.

Příklady ke cvičení 4:

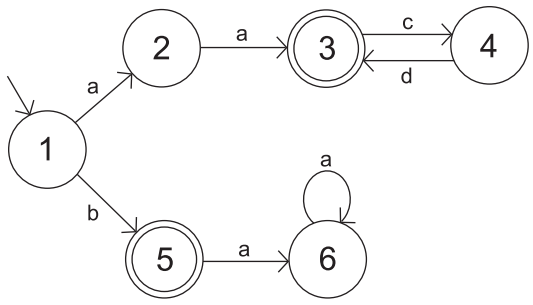
(a) Mějme konečný automat $M = (\{1,2,3,4\}, \{a,b,c\}, \delta, 1, \{4\})$ zadaný grafem:



(a) tabulkou:

	a	***b***	***c***
→ 1	2	1	1
2	2	1	3
3	2	1	4
←4	4	4	4

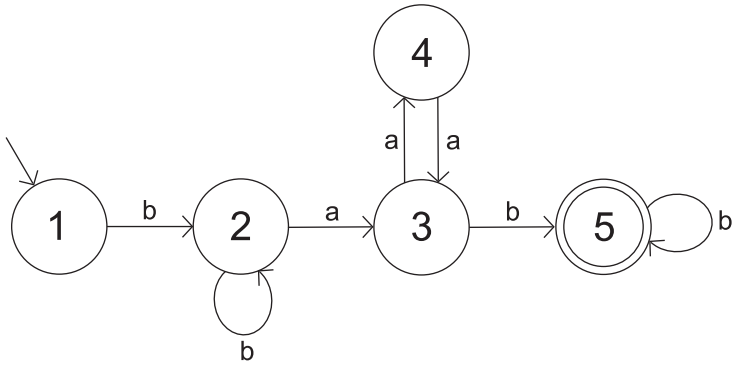
(b) Konečný automat $M = (\{1, 2, 3, 4, 5, 6\}, \{a, b, c, d\}, \delta, 1, \{3, 5\})$ zadaný grafem:



(b) tabulkou:

	a	***b***	***c***	***d***
→ 1	2	5	-	-
← 2	3	-	-	-
3	-	-	4	-
4	-	-	-	3
← 5	6	-	-	-
6	6	-	-	-

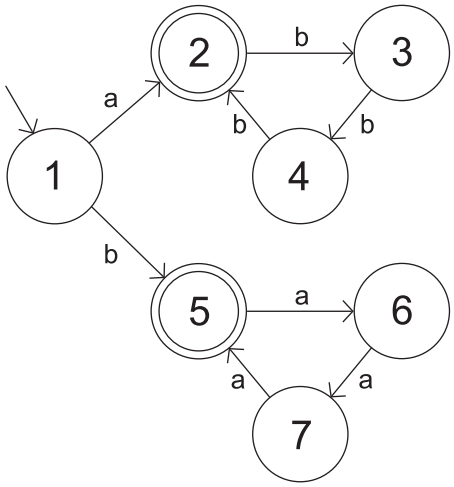
(c) Konečný automat $M = (\{1, 2, 3, 4, 5\}, \{a, b\}, \delta, 1, \{5\})$ zadaný grafem:



(c) tabulkou:

	a	***b***
→ 1	-	2
2	3	2
3	4	5
4	3	-
← 5	-	5

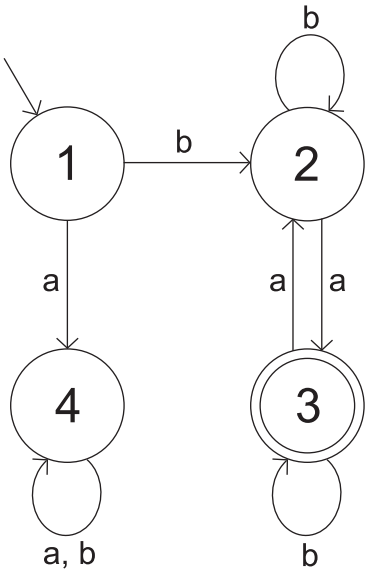
(d) Konečný automat $M = (\{1,2,3,4,5,6,7\}, \{a,b\}, \delta, 1, \{2,5\})$ zadaný grafem:



(d) tabulkou:

	a	***b***
$\rightarrow 1$	2	5
$\leftarrow 2$	-	3
3	-	4
4	-	2
$\leftarrow 5$	6	-
6	7	-
7	5	-

(e) Konečný automat $M = (\{1,2,3,4\}, \{a,b\}, \delta, 1, \{3\})$ zadaný grafem:



(e) tabulkou:

	a	b
$\rightarrow 1$	4	2
2	3	2
$\leftarrow 3$	2	3
4	4	4

Lemma o vkládání pro regulární jazyky

Definice lemmatu o vkládání je uvedena v základních pojmech na straně 4. Při důkazu neregulárnosti jazyka se však používá její obměna (negovaná verze).

1. Pomocí lemmatu o vkládání dokažte, že jazyk není regulární:

   - **Řešený příklad**: $L = \{a^n b^n \mid n \geq 0\}$
     Při důkazu neregulárnosti jazyka pomocí lemmy o vkládání je nejdůležitější správná volba slova, připomeňme:
     - Pro libovolné $n \in N$,
     - existuje slovo w z jazyka L, takové že $|w| \geq n$

       Je vhodné volit slova obsahující nějakou závislost počtu písmen, například rovnost, která může být po volbě konstanty i porušena. V tomto případě volíme $w = a^n b^n$

     - v dalším kroku uvažujeme všechna rozdělení slova taková, že platí: $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

       $x = a^k \quad k \geq 0$
       $y = a^l \quad l \neq 0$
       $z = a^{n-k-l} b^n$

       V tomto případě je takové rozdělení jen jedno.

     - abychom dokázali, že jazyk L není regulární, musíme najít takové, že $i \in N_0$ takové, že $xy^i z \notin L$
       $i = 0 \Rightarrow xy^0 z = a^k a^{n-k-l} b^n = a^{n-l} b^n$, protože $l \geq 1$ neplatí rovnost počtu písmen a a $b \Rightarrow xy^0 z \notin L \Rightarrow$ jazyk L **není** regulární.

Příklady ke cvičení 1:

(a) $L = \{a^n b^n \mid n \geq 0\}$
(b) $L = \{w \mid w \in \{a, b\}^* : \#_a(w) = \#_b(w)\}$
(c) $L = \{a^n b^m\} \mid n \leq m \leq 2n\}$
(d) $L = \{a^i b^j \mid i < j;\ i, j \in N\}$
(e) $L = \{w \cdot w \mid w \in \{a, b\}^*\}$
(f) $L = \{w \cdot w^R \mid w \in \{a, b\}^*\}$
(g) $L = \{a^i b^j\} \mid i \neq j;\ i, j \in N\}$

(h) $L = \{a^{n^2}\} \mid n \geq 0\}$
(i) $L = \{a^{2^n}\} \mid n \geq 0\}$
(j) $L = \{c^i a^j b^k \mid j \leq k;\ i, j, k \in N\}$

Minimalizace konečných automatů

1. Zkonstruujte minimální automat ke konečnému automatu zadaném tabulkou. (Zkontrolujte, zda jsou všechny stavy dosažitelné.)

   - **Řešený příklad**:

	a	***b***
→ 1	2	1
← 2	1	4
3	2	3
4	1	2

   – Nejprve tabulku rozdělíme do tříd na koncové a nekoncové stavy, podle původní tabulky doplníme přechody, například: ze stavu 1 se pod a dostaneme do třídy II, protože stav 2 leží v této třídě, ze stavu 1 pod b do třídy I, protože stav 1 leží ve třídě I, atd.

	a	***b***
I		
1	II	I
3	II	I
4	I	II
II		
2	I	I

   – třídu **I** lze dále rozdělit, protože přechody u stavu **4** se liší od ostatních. Přidáme tedy novou třídu a tabulku vyplníme stejným postupem jako v předchozím bodě:

	a	***b***
I		
1	III	I
3	III	I
II		
4	I	III
III		
2	I	II

   – ve třídě **I** mají stavy **1** a **3** stejné přechody, můžeme je tedy sloučit v jeden stav. Ostatní třídy mají po jednom stavu, proto je není potřeba dále dělit anebo slučovat:

	a	***b***
I	III	I
II	I	III
III	I	II

– Iniciální resp. koncové stavy jsou ty třídy, které obsahují alespoň jeden původní iniciální resp. koncový stav. Minimální automat, ekvivalentní původnímu vypadá tedy takto:

	a	***b***
→ I	III	I
II	I	III
← III	I	II

Příklady ke cvičení 1:

(a)

	a	***b***
→ 1	2	-
2	3	4
← 3	6	5
4	3	2
← 5	6	3
← 6	2	-
7	6	1

(b)

	a	***b***
→ 1	2	3
2	5	2
3	3	5
← 4	12	2
← 5	7	8
6	4	9
7	12	11
8	4	6
9	10	8
← 10	3	2
← 11	12	6
12	3	10

(c)

	a	***b***
1	3	1
→ 2	9	4
3	5	1
← 4	9	4
5	8	5
6	5	4
← 7	6	9
8	10	10
9	7	9
10	8	1

Nerodova věta

Nerodova věta se používá při důkazu, že **je** daný jazyk regulární. Zavádí relaci **pravá kongruence**, jejíž předpis je uveden jak v základních pojmech na straně 3. Postup důkazu je ukázán v následujícím řešeném příkladu.

1. Pomocí Nerodovy věta dokažte, že jazyk není regulární.

   - **Řešený příklad**: $L = \{\{a^i b^i\} \mid i \geq 0\}$
     Chceme dokázat, že jazyk není L regulární:
     Důkaz sporem: předpokládáme, že jazyk L vznikl sloučením některých tříd rozkladu na $\{a, b\}^*$ podle pravé kongruence na Σ^*, která má konečný index k. Jinými slovy, chceme ukázat, že **index** pravé kongruence **není** konečný.

     Relace **pravá kongruence** je ekvivalence na Σ^* definovaná takto: pro $u, v, w \in \Sigma^*$ platí: $u \sim v \Leftrightarrow uw \sim vw$

     - Volíme $k + 1$ slov, které nemusí být z jazyka L. Protože očekáváme konečný index k, musí tedy alespoň 2 slova být ve stejné třídě. Pokusíme se taková 2 slova najít a ukázat, že nejsou spolu v relaci pravé kongruence.
       $a^1, a^2, a^3, ..., a^k, a^{k+1} \Rightarrow \exists i, j \in N; i < j$, takové, že platí $a^i \sim a^j$
     - pak ale musí platit i $a^i w \sim a^j w$, pokud ale zvolíme $w = b^i$, pak neplatí $a^i b^i \sim a^j b^i \Rightarrow$ spor s předpokladem, že k je konečný index $\Rightarrow$ jazyk L **není** regulární.

Příklady ke cvičení 1:

(a) $L = \{\{a^n b^n\} \mid n \leq m \leq 2n; m, n > 0\}$
(b) $L = \{ww^R \mid w \in \{a, b\}^+\}$
(c) $L = \{a^n \mid n = 2^i; i \in N_0\}$

Determinizace konečných automatů

1. Determinizujte NFA a převeďte do kanonického tvaru:

   - **Řešený příklad:**

	a	b
$\rightarrow 1$	$\{2, 3\}$	$\emptyset$
$\leftarrow 2$	$\{3\}$	$\{1, 3\}$
3	$\{1, 2\}$	$\emptyset$

   Při determinizaci konečného automatu postupujeme tak, že pro každou množinu přechodů pod konkrétním písmenem vytvoříme nový odpovídající stav automatu (pokud ještě neexistuje). Přechody z takto vytvořených stavů vzniknou sloučením přechodů stavů, které byly prvky původní množiny.

   Nejprve opíšeme první řádek přechodové tabulky a stav 1 přejmenujeme na $\{1\}$:

	a	b
$\rightarrow \{1\}$	$\{2, 3\}$	$\emptyset$

   vytvořme nový stav s názvem $\{2, 3\}$, jehož množina přechodů vznikne sloučením množin stavů 2 a 3:

	a	b
$\rightarrow \{1\}$	$\{2, 3\}$	$\emptyset$
$\{2, 3\}$	$\{1, 2, 3\}$	$\{1, 3\}$

   získali jsme další nové stavy $\{1, 2, 3\}$ a $\{1, 3\}$, postupujeme tedy obdobně:

	a	b
$\rightarrow \{1\}$	$\{2, 3\}$	$\emptyset$
$\{2, 3\}$	$\{1, 2, 3\}$	$\{1, 3\}$
$\{1, 2, 3\}$	$\{1, 2, 3\}$	$\{1, 3\}$
$\{1, 3\}$	$\{2, 3\}$	$\emptyset$

   nyní doplňme původní stavy 2 a 3, všiměme si, že s přidáním stavu $\{3\}$ jsme přidali další nový stav $\{1, 2\}$, postupujeme stejně jako v předchozím kroku:

	a	***b***
→ {1}	{2, 3}	∅
{2}	{3}	{1, 3}
{3}	{1, 2}	∅
{1, 3}	{2, 3}	∅
{2, 3}	{1, 2, 3}	{1, 3}
{1, 2, 3}	{1, 2, 3}	{1, 3}
{1, 2}	{2, 3}	{1, 3}

za koncové stavy označíme takové, které obsahují alespoň 1 původní koncový stav. Například stav {2, 3} obsahuje stav 2, proto jej označme také jako koncový. Podobně u ostaních stavů:

	a	***b***
→ {1}	{2, 3}	∅
←{2}	{3}	{1, 3}
{3}	{1, 2}	∅
←{1, 2}	{2, 3}	{1, 3}
{1, 3}	{2, 3}	∅
←{2, 3}	{1, 2, 3}	{1, 3}
←{1, 2, 3}	{1, 2, 3}	{1, 3}

nakonec provedeme kanonizaci. Začneme v iniciálním stavu a postupně přejmenujeme všechny přechody dalším volným písmenem abecedy:

	a	***b***
→ A	B	∅
←B	C	D
C	C	D
D	B	∅
E	F	D
F	G	∅
G	B	D

Příklady ke cvičení 1:

(a)

	a	***b***	***c***
→ 1	{2, 3}	{3, 4}	{1}
←2	{3}	{4}	{2}
3	{1, 2, 3}	{1}	{3, 4}
4	{1}	{1}	{3, 4}

(b)

	a	***b***	***c***
→ 1	{1, 2}	{1}	{1}
←2	∅	{3}	∅
3	∅	∅	{4}
4	{5}	∅	∅
5	∅	{6}	∅
6	{7}	∅	∅
←7	∅	∅	∅

Odstranění ε-kroků

1. Převeďte na NFA bez ε-kroků:

   - **Řešený příklad:**

	a	b	ε
$\rightarrow 1$	$\{2\}$	$\emptyset$	$\{2, 3\}$
2	$\{2\}$	$\{2, 3\}$	$\emptyset$
$\leftarrow 3$	$\emptyset$	$\emptyset$	$\{2\}$

   Nejprve napočítejme množinu D_ε pro každý stav. D_ε bude obsahovat všechny stavy, do kterých se můžeme dostat z aktuálního stavu pod ε:

	a	b	ε	D_ε
$\rightarrow 1$	$\{2\}$	$\emptyset$	$\{2, 3\}$	$\{1, 2, 3\}$
2	$\{2\}$	$\{2, 3\}$	$\emptyset$	$\{2\}$
$\leftarrow 3$	$\emptyset$	$\emptyset$	$\{2\}$	$\{2, 3\}$

   nové přechody vytvořme následovně: pro stav 1: D_ε stavu 1 je $\{1, 2, 3\}$, sjednocením přechodů všech prvků pod a dostaneme množinu$\{2\}$, sjednotíme D_ε všech prvků v této množině a výsledkem je nový přechod stavu 1 pod a:

	a	b
$\rightarrow 1$	$\{2\}$	

   stejně tak pro všechny ostatní stavy:

	a	b
$\rightarrow 1$	$\{2\}$	$\{2, 3\}$
2	$\{2\}$	$\{2, 3\}$
3	$\{2, 3\}$	$\{2, 3\}$

Příklady ke cvičení 1:

(a)

	a	***b***	***c***	ε
$\rightarrow 1$	$\{2\}$	$\{3\}$	$\{3\}$	$\{4\}$
2	$\{4\}$	$\{1\}$	$\{4\}$	$\emptyset$
3	$\{1, 4\}$	$\{4\}$	$\emptyset$	$\emptyset$
$\leftarrow 4$	$\emptyset$	$\emptyset$	$\emptyset$	$\{2, 3\}$

(b)

	a	***b***	***c***	ε
$\rightarrow 1$	$\{2\}$	$\{2\}$	$\emptyset$	$\{2, 4\}$
$\leftarrow 2$	$\{3\}$	$\{3\}$	$\{2\}$	$\emptyset$
3	$\emptyset$	$\emptyset$	$\{2\}$	$\{2, 5\}$
4	$\emptyset$	$\emptyset$	$\{5\}$	$\emptyset$

Uzávěrové vlastnosti

1. Pomocí uzávěrových vlastností rozhodněte, zda je jazyk L složený z jazyků L_1 a L_2 regulární:

   - **Řešený příklad:** $L = (L_1 \cdot L_2) \cup L_1$; jazyky L_1 a L_2 jsou regulární.
     Víme, že třída regulárních jazyků je uzavřená na operace **zřetězení, sjednocení, průnik, rozdíl, iterace, pozitivní iterace, komplement**. Postupně tedy rozebereme jazyk L a zjistíme, zda je regulární:

     (a) $(L_1 \cdot L_2)$: oba jazyky L_1 i L_2 jsou regulární, z uzávěrových vlastností tedy plyne, že i jejich zřetězení je regulární.
     (b) $(L_1 \cdot L_2) \cup L_1$: v přechozím kroku jsme zjistili, že $L_1 \cdot L_2$ je regulární. L_1 je regulární, a proto opět z uzávěrových vlastností plyne, že sjednocení regulárních jazyků je regulární.

     Jazyk L je **regulární**.

   Příklady ke cvičení 1:

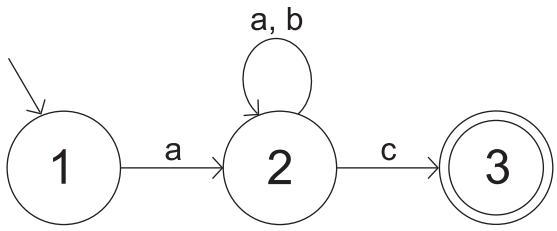
   (a) $L = L_1 \cdot (L_1 \cup L_2)$; L_1 je regulární; L_2 je neregulární.
   (b) $L = L_1 \cdot L_2 \cdot L_3$; L_1 je regulární; L_2 je regulární. L_3 je neregulární.
   (c) $L = co - L_1 \cdot (L_2 \cap L_1)$; L_1 je regulární; L_2 je neregulární.
   (d) $L = (L_1 \cup L_2) \setminus (L_1 \cap L_2)$; L_1 je regulární; L_2 je regulární.
   (e) $L = (L_1 \cup L_2) \setminus (L_1 \cap L_2)$; L_1 je neregulární; L_2 je regulární.
   (f) $L = (L_1 \cup L_2) \setminus (L_1 \cap L_2)$; L_1 je regulární; L_2 je neregulární.

2. Rozhodněte o pravdivosti následujících implikací. Své rozhodnutí zdůvodněte.

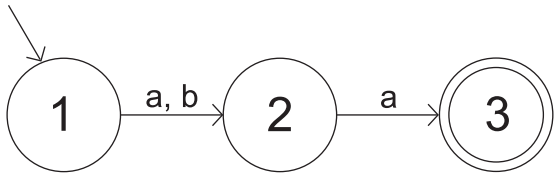
   (a) L_1 je regulární, L_2 neregulární $\Rightarrow L_1 \cap L_2$ je neregulární.
   (b) L_1 je regulární, L_2 neregulární $\Rightarrow L_1 \cap L_2$ je regulární.
   (c) L_1 je regulární, L_2 neregulární $\Rightarrow L_1 \setminus L_2$ je neregulární.
   (d) L_1 je regulární, L_2 neregulární $\Rightarrow L_1 \setminus L_2$ je regulární.
   (e) L_1 je regulární, L_2 neregulární $\Rightarrow L_2 \setminus L_1$ je neregulární.
   (f) L_1 je regulární, L_2 neregulární $\Rightarrow L_2 \setminus L_1$ je regulární.

Konstrukce konečných automatů - řešení

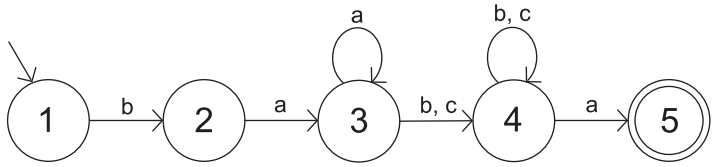
1. (a) $M = (\{1, 2, 3\}, \{a, b, c\}, \delta, 1, \{3\})$



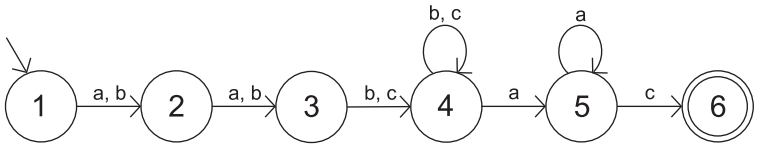
(b) $M = (\{1, 2, 3\}, \{a, b\}, \delta, 1, \{3\})$



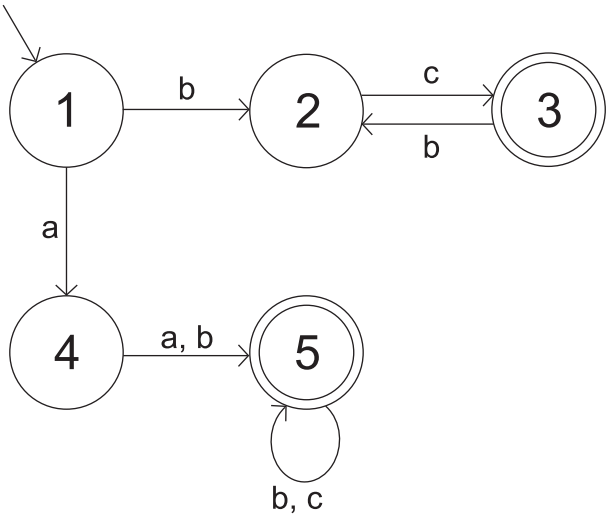
(c) $M = (\{1, 2, 3, 4, 5\}, \{a, b, c\}, \delta, 1, \{5\})$



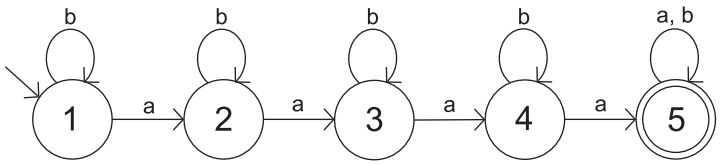
(d) $M = (\{1, 2, 3, 4, 5, 6\}, \{a, b, c\}, \delta, 1, \{6\})$



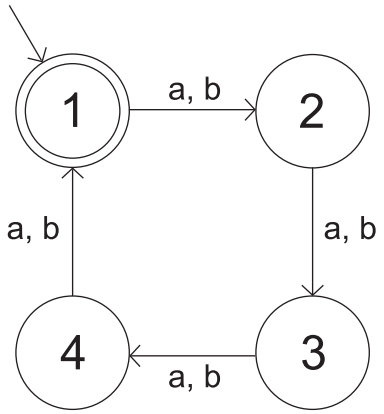
(e) $M = (\{1, 2, 3, 4, 5\}, \{a, b, c\}, \delta, 1, \{3, 5\})$



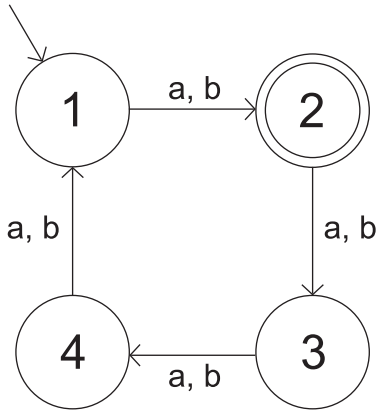
2. (a) $M = (\{1, 2, 3, 4, 5\}, \{a, b\}, \delta, 1, \{5\})$



(b) $M = (\{1, 2, 3, 4\}, \{a, b\}, \delta, 1, \{1\})$

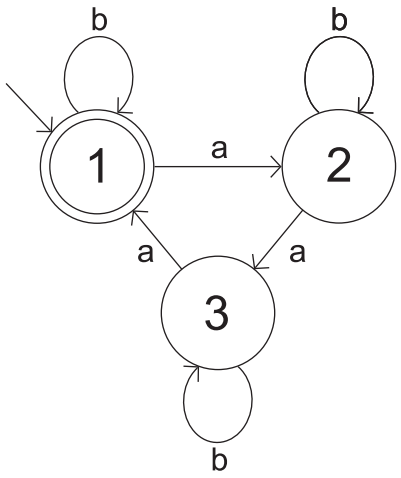


(c) $M = (\{1, 2, 3, 4\}, \{a, b\}, \delta, 1, \{2\})$

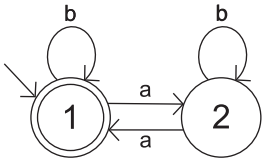


(d) Nelze sestrojit: L není **regulární jazyk**.

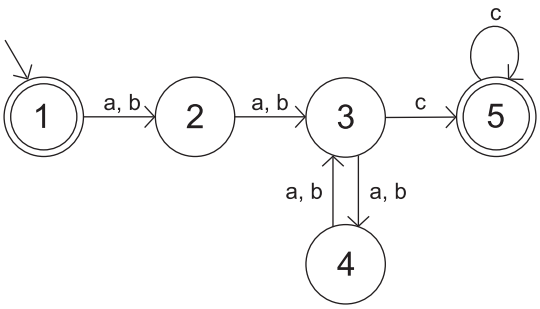
(e) $M = (\{1, 2, 3\}, \{a, b\}, \delta, 1, \{1\})$



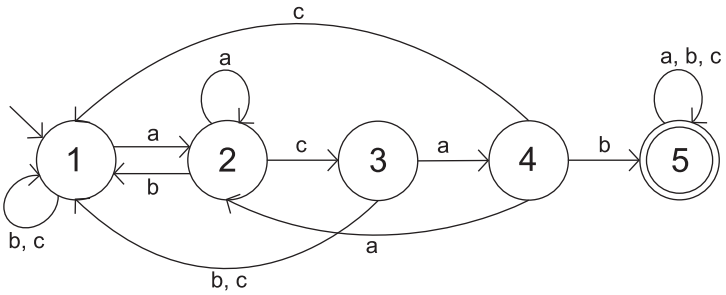
(f) $M = (\{1, 2\}, \{a, b\}, \delta, 1, \{1\})$



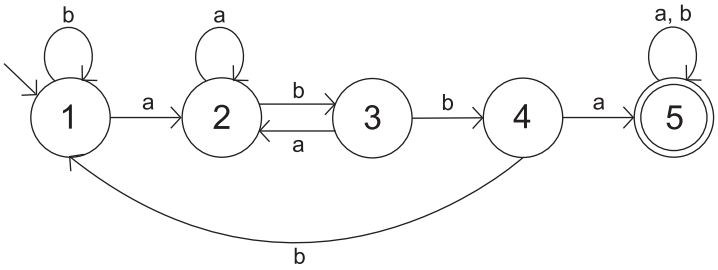
(g) $M = (\{1, 2, 3, 4, 5\}, \{a, b, c\}, \delta, 1, \{1, 5\})$



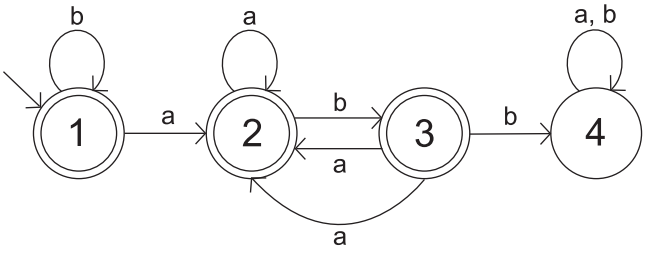
3. (a) $M = (\{1, 2, 3, 4, 5\}, \{a, b, c\}, \delta, 1, \{5\})$



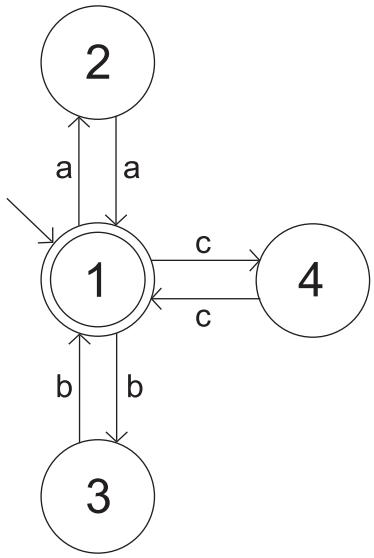
(b) $M = (\{1, 2, 3, 4, 5\}, \{a, b, c\}, \delta, 1, \{5\})$



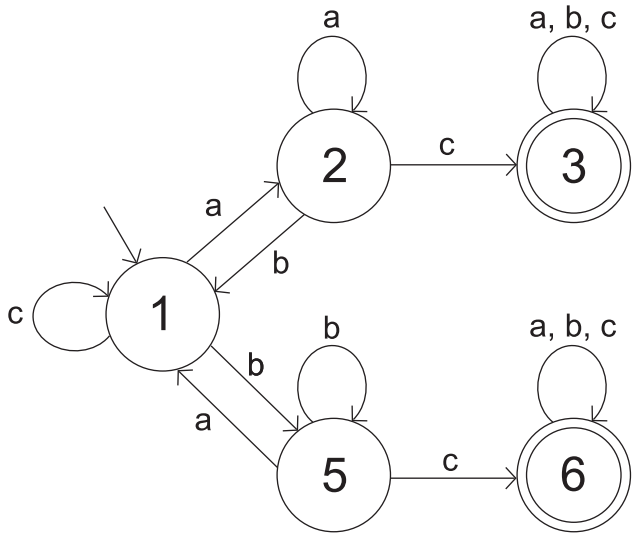
(c) $M = (\{1, 2, 3, 4\}, \{a, b\}, \delta, 1, \{1, 2, 3\})$



(d) $M = (\{1, 2, 3, 4\}, \{a, b, c\}, \delta, 1, \{1\})$



(e) $M = (\{1, 2, 3, 4, 5, 6\}, \{a, b, c\}, \delta, 1, \{3, 6\})$



4. (a) $L = \{w | w \in \{a, b, c\}^* : w$ obsahuje podslovo **acc**$\}$
   (b) $L = \{\{b\} \cup \{\{a\} \cdot \{a\} \cdot (\{c\} \cdot \{d\})^*\}\}$
   (c) $L = \{w | w \in \{a, b\}^* : w$ začíná a končí na „b“, zároveň obsahuje lichý počet „a“ v řadě za sebou$\}$
   (d) $L = \{w | w \in \{a, b\}^*$: pokud w začíná na „a“, pak obsahuje **3k** písmen „b“ nebo pokud začíná na „b“, pak obsahuje **3k** písmen „a“; kde k $\geq 0\}$
   (e) $L = \{w | w \in \{a, b\}^* : w$ začíná na „b“ a končí na „a“; $\#_a(w)$ je lichý$\}$

Lemma o vkládání pro regulární jazyky - řešení

1. (a) $L = \{c^i d^i \mid i \geq 0\}$
   - Pro libovolné $n \in N$,
   - volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = c^n d^n$
   - pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

     $x = c^k \quad k \geq 0$
     $y = c^l \quad l \neq 0$
     $z = c^{n-k-l} d^n$

   - volíme $i \in N_0$, takové že $xy^i z \notin L : i = 2 \Rightarrow xy^2 z = c^k c^{2l} c^{n-k-l} d^n = c^{n+l} d^n \notin L$, protože $l \geq 1 \Rightarrow$ jazyk L **není** regulární.

   (b) $L = \{w \mid w \in \{a, b\}^* : \#_a(w) = \#_b(w)\}$
   - Pro libovolné $n \in N$,
   - volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^n b^n$
   - pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

     $x = a^k \quad k \geq 0$
     $y = a^l \quad l \neq 0$
     $z = a^{n-k-l} b^n$

   - volíme $i \in N_0$, takové že $xy^i z \notin L : i = 0 \Rightarrow xy^0 z = a^k a^{n-k-l} b^n = a^{n-l} b^n \notin L$, protože $l \geq 1 \Rightarrow$ jazyk L **není** regulární.

   (c) $L = \{a^n b^m\} \mid n \leq m \leq 2n\}$
   - Pro libovolné $n \in N$,
   - volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^n b^n$
   - pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

     $x = a^k \quad k \geq 0$
     $y = a^l \quad l \neq 0$
     $z = a^{n-k-l} b^n$

- volíme $i \in N_0$, takové že $xy^iz \notin L : i = 2 \Rightarrow xy^2z = a^k a^{2l} a^{n-k-l} b^n = a^{n+l} b^n \notin L$, protože $l \geq 1 \Rightarrow$ jazyk L **není** regulární.

(d) $L = \{a^i b^j \mid i < j;\ i, j \in N\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^n b^{n+1}$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

$x = a^k \quad k \geq 0$
$y = a^l \quad l \neq 0$
$z = a^{n-k-l} b^{n+1}$

- volíme $i \in N_0$, takové že $xy^iz \notin L : i = 2 \Rightarrow xy^2z = a^k a^{2l} a^{n-k-l} b^{n+1} = a^{n+l} b^{n+1} \notin L$, protože $l \geq 1$, počet a je tedy nejméně roven počtu $b \Rightarrow$ jazyk L **není** regulární.

(e) $L = \{w \cdot w \mid w \in \{a, b\}^*\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^n b a^n b$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

$x = a^k \quad k \geq 0$
$y = a^l \quad l \neq 0$
$z = a^{n-k-l} b a^n b$

- volíme $i \in N_0$, takové že $xy^iz \notin L : i = 2 \Rightarrow xy^2z = a^k a^{2l} a^{n-k-l} b a^n b = a^{n+l} b a^n b \notin L$, protože $l \geq 1$; jazyk L **není** regulární.

(f) $L = \{w \cdot w^R \mid w \in \{a, b\}^*\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^n b b a^n$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

$x = a^k \quad k \geq 0$
$y = a^l \quad l \neq 0$

$z = a^{n-k-l}bba^n$

- volíme $i \in N_0$, takové že $xy^iz \notin L : i = 0 \Rightarrow xy^0z = a^k a^{n-k-l}bba^n = a^{n-l}bba^n \notin L$, protože $l \geq 1$; jazyk L **není** regulární.

(g) $L = \{a^ib^j\} \mid i \neq j;\ i, j \in N\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^nb^{n+n}$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

$x = a^k \quad k \geq 0$
$y = a^l \quad l \neq 0$
$z = a^{n-k-l}b^{n+n}$

- volíme $i \in N_0$, takové že $xy^iz \notin L$: vyjádříme i pomocí rovnosti $k+li+n-k-l = n+n \Rightarrow i = \frac{n}{l}+1 \Rightarrow xy^{\frac{n}{l}+1}z = a^k a^{(\frac{n}{l}+1)l}a^{n-k-l}b^{n+n} = a^{n+n}b^{n+n} \notin L \Rightarrow$ jazyk L **není** regulární.

(h) $L = \{a^{n^2} \mid n \geq 0\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^{n^2}$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

$x = a^k \quad k \geq 0$
$y = a^l \quad l \neq 0$
$z = a^{n^2-k-l}$

- volíme i $\in N_0$, takové že $xy^iz \notin L : i = 0 \Rightarrow xy^0z = a^{n^2-l}$, což je jistě kratší než a^{n^2} a delší než $a^{(n-1)^2} \Rightarrow$ slovo xy^0z nemůže patřit do jazyka L $\Rightarrow$ L **není** regulární.

(i) $L = \{a^{2^n} \mid n \geq 0\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = a^{2^n}$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| \leq n$

  $x = a^k \quad k \geq 0$
  $y = a^l \quad l \neq 0$
  $z = a^{2^n - k - l}$

- volíme i $\in N_0$, takové že $xy^iz \notin L : i = 2 \Rightarrow xy^2z = a^{2^n + l}$, pak tedy $2^n < 2^n + l \leq 2^{n+1}$, což platit nemusí $\Rightarrow$ L **není** regulární.

(j) $L = \{c^i a^j b^k \mid j \leq k;\ i, j, k \in N\}$

- Pro libovolné $n \in N$,
- volíme slovo w z jazyka L, takové že $|w| \geq n$; $w = ca^n b^n$
- pro všechna rozdělení slova $w = xyz$, kde $|x| \geq 0, |y| \geq 1, |xy| + 1 \leq n$

  i. $x = ca^k \quad k \geq 0$
  $y = a^l \quad l \neq 0$
  $z = a^{n-k-l}b^n$
  – volíme $i \in N_0$, takové že
  $xy^iz \notin L : i = 2 \Rightarrow xy^2z = ca^k a^{2l} a^{n-k-l} b^n = ca^{n+l}b^n \notin L.$

  ii. $x = \varepsilon \quad k \geq 0$
  $y = ca^k \quad l \neq 0$
  $z = a^{n-k}b^n$
  – volíme $i \in N_0$, takové že
  $xy^iz \notin L : i = 0 \Rightarrow xy^0z = (ca^k)^0 a^{n-k} b^n = a^{n-k}b^n \notin L.$

Minimalizace konečných automatů - řešení

1. (a)

	a	***b***
→ 1	3	2
2	2	2
3	4	3
← 4	5	4
← 5	3	2

(b)

	a	***b***
→ 1	2	3
2	4	2
3	3	4
← 4	3	2

(c)

	a	***b***
1	1	1
→ 2	4	5
3	1	5
4	6	4
← 5	4	5
← 6	3	4

Nerodova věta - řešení

1. (a) $L = \{\{a^n b^n\} \mid n \leq m \leq 2n; m, n > 0\}$
   Důkaz sporem: předpokládáme, že jazyk L vznikl sloučením některých tříd rozkladu na $\{a, b\}^*$ podle pravé kongruence, která má konečný index k.
   - Zvolíme následujících $k + 1$ slov:
     $a^1, a^2, a^3, ..., a^k, a^{k+1} \Rightarrow \exists i, j \in N; i < j$, takové, že platí $a^i \sim a^j$
   - zvolíme $w = b^{2j} \Rightarrow$ neplatí $a^i b^{2j} \sim a^j a^{2j}$, protože $a^i b^{2j} \notin L$ a zároveň $a^j a^{2j} \in L \Rightarrow$ spor s předpokladem, jazyk L **není** regulární.

   (b) $L = \{ww^R \mid w \in \{a, b\}^+\}$
   Důkaz sporem: předpokládáme, že jazyk L vznikl sloučením některých tříd rozkladu na $\{a, b\}^*$ podle pravé kongruence, která má konečný index k.
   - Zvolíme následujících $k + 1$ slov:
     $a^1, a^2, a^3, ..., a^k, a^{k+1} \Rightarrow \exists i, j \in N; i < j$, takové, že platí $a^i \sim a^j$
   - zvolíme $w = a^i \Rightarrow$ neplatí $a^i a^i \sim a^j a^i$, protože $a^i a^i \in L$ a zároveň $a^j a^i \notin L \Rightarrow$ spor s předpokladem, jazyk L **není** regulární.

   (c) $L = \{a^n \mid n = 2^i; i \in N_0\}$ Důkaz sporem: předpokládáme, že jazyk L vznikl sloučením některých tříd rozkladu na $\{a, b\}^*$ podle pravé kongruence, která má konečný index k.
   - Zvolíme následujících $k + 1$ slov:
     $a^{2^1}, a^{2^1}, ..., a^{2^k}, a^{2^{k+1}} \Rightarrow \exists i, j \in N; i < j$, takové, že platí $a^{2^i} \sim a^{2^j}$
   - zvolíme $w = a^{2^i} \Rightarrow$ neplatí $a^{2^i} a^{2^i} \sim a^{2^j} a^{2^i}$, protože $a^{2^i} a^{2^i} \in L$ a zároveň $a^{2^j} a^{2^i} \notin L \Rightarrow$ spor s předpokladem, jazyk L **není** regulární.

Determinizace konečných automatů - řešení

1. (a)

	a	***b***	***c***
→A	B	C	A
←B	D	E	F
C	D	A	C
←D	D	G	H
E	D	G	G
←F	D	E	F
G	D	G	G
←H	D	G	H
←I	J	K	I
J	D	I	C
K	A	A	C

(b)

	a	***b***	***c***
→ A	B	A	A
←B	B	C	∅
C	∅	∅	D
D	E	∅	∅
E	∅	F	∅
F	G	∅	∅
←G	∅	∅	∅
←H	∅	C	∅

Odstranění ε-kroků - řešení

1. (a)

	a	b	c
$\rightarrow 1$	$\{2\}$	$\{3\}$	$\{3\}$
2	$\{2, 3, 4\}$	$\{1, 4\}$	$\{2, 3, 4\}$
3	$\{1, 2, 3, 4\}$	$\{2, 3, 4\}$	$\emptyset$
$\leftarrow 4$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{2, 3, 4\}$

(b)

	a	b	c
$\rightarrow 1$	$\{2, 3, 5\}$	$\{2, 3, 5\}$	$\{2, 5\}$
$\leftarrow 2$	$\{2, 3, 5\}$	$\{2, 3, 5\}$	$\{2\}$
3	$\{2, 3, 4, 5\}$	$\{2, 3, 4, 5\}$	$\{2, 5\}$
4	$\emptyset$	$\emptyset$	$\{5\}$
$\leftarrow 4$	$\{4, 5\}$	$\{4, 5\}$	$\{5\}$

Uzávěrové vlastnosti - řešení

1. (a) L **není** regulární.
   (b) L **není** regulární.
   (c) L **je** regulární.
   (d) L **je** regulární.
   (e) L **není** regulární.
   (f) L **není** regulární.

2. (a) **Neplatí**. Protipříklad: $L_1 = \{a^i : i > 0\}; L_2 = \{a^j b^j : j > 0\}$, pak $L_1 \cap L_2$ je $\emptyset$, což **je** regulární.
   (b) **Neplatí**. Protipříklad: $L_1 = \Sigma^*; L_2 = \{a^i b^i : i > 0\}$, pak $L_1 \cap L_2$ je $\{a^i b^i : i > 0\}$, což **není** regulární.
   (c) **Neplatí**. Protipříklad: $L_1 = \emptyset; L_2 = \{a^i b^i : i > 0\}$, pak $L_1 \setminus L_2$ je $\emptyset$, což **je** regulární.
   (d) **Neplatí**. Protipříklad: $L_1 = \Sigma^*; L_2 = \{a^i b^i : i > 0\}$, pak $L_1 \setminus L_2$, což **není** regulární.
   (e) **Neplatí**. Protipříklad: $L_1 = \Sigma^*; L_2 = \{a^i b^i : i > 0\}$, pak $L_2 \setminus L_1$ je $\emptyset$, což **je** regulární.
   (f) **Neplatí**. Protipříklad: $L_1 = \emptyset; L_2 = \{a^i b^i : i > 0\}$, pak $L_2 \setminus L_1$ je $\{a^i b^i : i > 0\}$, což **není** regulární.

Závěr

Hybridní studijní materiály se ukazují jako dobrá pomůcka pro handicapované studenty. Oproti materiálům v papírové formě mají nesporné výhody: odpadá nutnost tisknout materiály do Braillova písma, je možné používat doplňující zařízení, například: Braillské čtečky nebo čtecí zařízený, které zobrazený text převádí do zvukové podoby. Je tedy možné vytvořit velmi robustní software, ale zároveň dovolit handicapovaným uživatelům plnohodnotně využívat všech jeho možností. Problémem však stále je zobrazení libovolného obrazu do formy vhodné pro nevidomé uživatele. Na Fakultě informatiky Masarykovy univerzity se docent Ivan Kopeček zabývá převodem obrazu na odpovídající zvukové schéma. Každé barvě by měl odpovídat určitý zvuk, popřípadě směs zvuků.

V mé bakalářské práci jsem převod obrazu řešit nemusel. Výhodou teorie konečných automatů je možnost zápisu přechodové funkce pomocí grafu. Takový graf lze zapsat jako seznam uzlů a k nim odpovídající hrany. Linearizace takového formalismu je násobně jednodušší než zpracování obrazového výstupu. Nicméně, bylo nutné vyřešit problém komunikace nevidomého uživatele s programem. Najít postup, který by byl zároveň efektivní a intuitivní. Jako nejlepší řešení se ukázala kombinace dialogového systému s imperativními prvky, tj. uživatel má k dispozici sadu příkazů, kterými program ovládá a při provádění daného příkazu se program uživatele dotazuje na další postup. Program byl také testován studentem Fakulty informatiky MU Jiřím Fencem a jeho poznatky, připomínky k funkcionalitě byly zahrnuty do celkového řešení, za účelem zlepšení interakce řádkového rozhraní.

Další rozšíření by se mohlo týkat obsahu látky bezkontextových jazyků. Možnost zpracování zásobníkových automatů, popřípadě algoritmů syntaktické analýzy. Zde ovšem vyvstávají problémy se zobrazením takového algoritmu na jednořádkovém zařízení. Problém se týká již linearizace algoritmu pro převod regulárních výrazů na konečný automat a naopak.

Zvolená cesta, tedy kombinace použití e-learningových pomůcek, se jeví jako správná. Netriviální problémy ale mohou vznikat nikoli v samotném programování takové pomůcky, ale v návrhu vhodného výstupu k řešení problému. Spolupráce při vývoji s handicapovanými uživateli je v tomto smyslu zásadní.

Příloha - obsah CD

- Spustitelný, přeložený zdrojový kód: *Thesis.exe,*
- datový soubor formátu XML,
- gramatika datového souboru XML,
- zdrojový kód (Microsoft Visual Studio 2008 Solution),
- elektronická podoba sbírky ve formátu PDF.

Použité zdroje

Sharp, John: *Microsoft Visual C# 2008 Krok za krokem.* Computer Press, listopad 2008.

Microsoft Developer Network Library. http://www.msdn.microsoft.com.

Strejček, Jan: *Učební skripta k předmětu Automaty a gramatiky.* Fakulta informatiky Masarykovy univerzity. 2007, http://www.fi.muni.cz/ xstrejc/IB102/

Lipovčan, Marek: *Optimalizace aplikací distribuce teTeX.* Bakalářská práce, Fakulta informatiky Masarykovy univerzity. 2003, http://www.fi.muni.cz/ xpavlov/xml/examples/bc4/bc4.xhtml.