FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Teorie programovacích jazyků

Dvourozměrné jazyky a digitální obrazy

Ak.rok: 2008/2009

Jiří Koutný

Abstrakt

Následující text je projektem do předmětu Teorie programovacích jazyků doktorského studijního programu (TJD). Jeho cílem je poskytnout čtenáři základní náhled do zobecnění teorie formálních jazyků do dvou rozměrů. V úvodu popisuje vlastnosti, které na teorii dvourozměrných jazyků klademe za tím účelem, aby celá teorie byla přirozeným rozšířením teorie jednorozměrných formálních jazyků. Stěžejní částí této práce je představení několika modelů pro přijímání dvourozměrných jazyků a modelu pro jejich generování. Poslední část této práce se věnuje souvislosti dvourozměrných jazyků a digitálního obrazu a nastiňuje možnosti praktického využití této teorie.

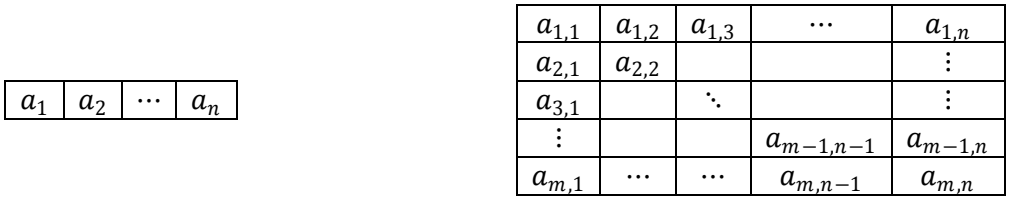
Klíčová slova

Abeceda, automat, dvourozměrný řetězec, dvourozměrný jazyk, gramatika, operace nad jazyky, obraz

Obsah

Dvourozměrné jazyky

Podstatou *dvourozměrných jazyků* je zobecnění konceptů a technik teorie formálních jazyků do dvou dimenzí. Neformálně řečeno, *dvourozměrný řetězec* je nazýván *obraz* a je definován jako dvourozměrné pole symbolů nad nějakou abecedou. *Dvourozměrný jazyk* pak je množina obrazů.



Obrázek 1 Schematické znázornění, vlevo jednorozměrný řetězec, vpravo dvourozměrný řetězec

Zobecnění do dvou dimenzí je možné provést několika různými způsoby, důsledkem tohoto faktu je, že pro generování a rozpoznávání dvourozměrných jazyků bylo zavedeno několik různých formálních modelů. Celá teorie dvourozměrných jazyků byla historicky motivována potřebami *rozpoznávání/přijímání dvourozměrných vzorů* (two-dimensional pattern matching), přičemž je známým faktem, že dvourozměrné vzory souvisí s celulárními automaty a dalšími modely paralelních výpočtů.

Požadavky na vlastnosti dvourozměrných jazyků

Podobně jako v případě jednorozměrných jazyků, tedy množin řetězců, je celá teorie postavena na konečně stavových modelech. Při hledání modelu popisujícího dvourozměrné jazyky bylo dalším přirozeným požadavkem to, aby třída rozpoznatelných dvourozměrných jazyků nějakým způsobem obsahovala i třídu rozpoznatelných jednorozměrných jazyků. Přesněji řečeno, když se omezíme na všechny dvourozměrné jazyky o rozměrech

- (1,n), tj. všechny jednorozměrné řetězce zapsané horizontálně
- (n,1), tj. všechny jednorozměrné řetězce zapsané vertikálně

pak dostaneme přesně všechny jednorozměrné jazyky. Dále požadujeme, aby postup přijímání dvourozměrného jazyka bylo přirozeným zobecněním některého postupu přijímání řetězce.

Dalším přirozeným požadavkem je definovat dvourozměrné jazyky pomocí *regulárních výrazů*, podobně jako tomu je v případě jazyků jednorozměrných, a zavést do teorie dvourozměrných jazyků regulární operace. Regulární výraz pak představuje způsob, jakým z několika elementárních dvourozměrných jazyků pomocí regulárních operací získat určitý dvourozměrný jazyk. Jak bude podrobněji uvedeno dále, na základě různých množin povolených regulárních operací můžeme získat různé třídy jazyků.

Generování a přijímání dvourozměrných jazyků

V teorii dvourozměrných jazyků hrají důležitou roli konečné automaty zobecněné do dvou dimenzí. Bylo zavedeno několik možných zobecnění, v tomto textu si popíšeme dvě z nich.

- Prvním možným zobecněním je klasickým konečným automatům přidat možnost pohybu v další dimenzi. Tímto způsobem byly zavedeny *čtyřsměrné konečné automaty*, které se pohybují po

dvourozměrné pásce. Takovým rozšířením však nevznikne model dostatečně silný, protože nezachovává některé důležité vlastnosti.

- Robustnějším modelem jsou tzv. *online teselační automaty*, které jsou neformálně definovány jako nekonečné dvourozměrné pole identických konečně stavových automatů a jde o speciální případ celulárního automatu. Ačkoliv není na první pohled zřejmé, že jde o zobecnění jednorozměrného modelu, pokud online teselační automat omezíme na velikost (1,n) nebo (n,1), opět získáme klasický konečný automat.

Výše uvedené modely založené na zobecnění konečného automatu do dvou dimenzí slouží k přijímání obrazů. Dále je třeba zavést nějaký model pro generování dvourozměrných jazyků, k tomu slouží gramatiky, které byly zobecněny do dvou rozměrů a korespondují ke klasickým bezkontextovým a regulárním gramatikám.

Jedním z dalších způsobů, jak dvourozměrné jazyky popsat, je využití *logických formulí*, ale bylo zavedeno i několik dalších různých modelů, které jsou schopny generovat nebo rozpoznávat různé třídy dvourozměrných jazyků. Tyto modely již přesahují rámec této práce, proto se jim v dalších kapitolách nebudeme věnovat. Základní informace a odkazy na další související prameny lze však nalézt například v (Giammaressi & Restivo, 2004) v kapitolách 6 a 7.

Základní definice

Na dalších stránkách předpokládáme, že čtenář je seznámen se základní terminologií a vlastnostmi teorie jednorozměrných formálních jazyků, kterou lze nastudovat např. z (Meduna, 2000).

Nechť Σ je konečná abeceda.

Definice 1: Dvourozměrný řetězec (nebo *obraz*) nad abecedou Σ je dvourozměrné obdélníkové pole prvků z abecedy Σ. Množinu všech dvourozměrných řetězců nad abecedou Σ označujeme Σ^{**}. *Dvourozměrný jazyk* nad abecedou Σ pak je podmnožina Σ^{**}.

Mějme obraz nad abecedou $p \in \Sigma^{**}$, nechť $l_1(p)$ značí počet řádků p a $l_2(p)$ značí počet sloupců p.

Definice 2: *Velikost obrazu* je definována jako dvojice $(l_1(p), l_2(p))$. Obraz je *prázdný* právě tehdy, když jeho velikost je $(0,0)$, a takový obraz označujeme symbolem λ. Obrazy o velikostech $(0, n)$ nebo $(n, 0)$, kde $n > 0$ nejsou definovány.

Množinu všech obrazů o velikosti (m, n), kde $m, n > 0$, nad abecedou Σ označujeme jako $\Sigma^{m \times n}$.

Jestliže $1 \leq i \leq l_1(p)$ a $1 \leq j \leq l_2(p)$, pak $p(i, j)$, nebo jednoduše $p_{i,j}$ označuje *symbol v p na souřadnicích* (i, j).

Příklad 1: Nechť $\Sigma = \{a\}$ je abeceda. Množina obrazů složených ze symbolů a, které mají tři sloupce, je dvourozměrným jazykem nad abecedou Σ. Formálně

$$L = \{p | p \in \Sigma^{**} \wedge l_2(p) = 3\}$$

Příklad 2: Nechť $\Sigma = \{a\}$ je abeceda a nechť L je podmnožina Σ^{**} taková, že všechny obrazy v ní obsažené mají tvar „čtverce“. Formálně

$$L = \{p | p \in \Sigma^{**} \wedge l_1(p) = l_2(p)\}$$

Příklad 3: Nechť $\Sigma = \{0,1\}$ je abeceda a nechť L je podmnožina Σ^{**} taková, že všechny obrazy v ní obsažené mají tvar „čtverce“, kde všechny prvky na hlavní diagonále jsou rovny 1, zatímco všechny ostatní prvky jsou rovny 0. Formálně

$$L = \{p | p \in \Sigma^{**} \wedge l_1(p) = l_2(p) \wedge p(i,i) = 1 \wedge p(i,j) = 0, pro\ i \neq j, i,j = 1..l_1(p)\}$$

Obraz patřící do jazyka z předchozího příkladu můžeme graficky zobrazit takto:

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1

Obrázek 2 Obraz dle definice jazyka z Příkladu 3.

Základní operace nad dvourozměrnými řetězci/jazyky

Nyní můžeme definovat operace nad dvourozměrnými řetězci a následně i nad dvourozměrnými jazyky. Na rozdíl od jednorozměrných řetězců a jazyků rozlišujeme několik různých typů konkatenace obrazů a dvourozměrných jazyků.

Nechť p a q jsou dva obrazy nad abecedou Σ a nechť velikost p je (m,n) a velikost q je (m',n'), kde $m,n,m',n' > 0$.

$$p = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix} \qquad q = \begin{bmatrix} q_{1,1} & \cdots & q_{1,n'} \\ \vdots & \ddots & \vdots \\ q_{m',1} & \cdots & q_{m',n'} \end{bmatrix}$$

Obrázek 3 Obrazy *p* a *q* pro definici konkatenace

Definice 3: *Sloupcová konkatenace* p a q, označme ji $p \oslash$ q, je parciální operace, definovaná pouze pokud $m = m'$, definována jako:

$$p \oslash \mathrm{q} = \left[\begin{array}{ccc|ccc} p_{1,1} & \cdots & p_{1,n} & q_{1,1} & \cdots & q_{1,n'} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} & q_{m',1} & \cdots & q_{m',n'} \end{array}\right]$$

Obrázek 4 Sloupcová konkatenace obrazů *p* a *q*

Definice 4: *Řádková konkatenace* p a q, označme ji $p \ominus \mathrm{q}$, je parciální operace, definovaná pouze pokud $n = n'$, definována jako:

$$p \ominus \mathrm{q} = \begin{array}{|ccc|} \hline p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \\ \hline q_{1,1} & \cdots & q_{1,n'} \\ \vdots & \ddots & \vdots \\ q_{m',1} & \cdots & q_{m',n'} \\ \hline \end{array}$$

Obrázek 5 Řádková konkatenace obrazů *p* a *q*

Navíc platí, že sloupcová i řádková konkatenace obrazu p a prázdného obrazu λ je definována vždy a prázdný obraz λ je neutrálním prvkem obou operací.

Stejně jako v případě jednorozměrných řetězců můžeme pomocí definice konkatenace řetězců definovat konkatenaci jazyků, můžeme sloupcovou a řádkovou konkatenaci obrazů rozšířit na sloupcovou a řádkovou konkatenaci dvourozměrných jazyků.

Definice 5: Nechť L_1, L_2 jsou dva dvourozměrné jazyky nad abecedou Σ. *Sloupcová konkatenace jazyků* L_1, L_2, označme ji $L_1 \oslash L_2$ je definována jako

$$L_1 \oslash L_2 = \{p \oslash \mathrm{q} | \mathrm{p} \in L_1 \wedge q \in L_2\}$$

Řádková konkatenace jazyků L_1, L_2, označme ji jako $L_1 \ominus L_2$, je definována jako

$$L_1 \ominus L_2 = \{p \ominus \mathrm{q} | \mathrm{p} \in L_1 \wedge q \in L_2\}$$

Pomocí iterace konkatenace můžeme definovat odpovídající tranzitivní uzávěry vzhledem k řádkům/sloupcům, který můžeme vnímat jako iteraci dvourozměrného jazyka.

Definice 6: Nechť L je dvourozměrný jazyk.

- *Sloupcová iterace* jazyka L je definována jako

$$L^{*\oslash} = \bigcup_{i \geq 0} L^{i\oslash}$$

  kde $L^{0\oslash} = \lambda$, $L^{1\oslash} = L$, $L^{n\oslash} = L \oslash L^{(n-1)\oslash}$.

- *Řádková iterace* jazyka L je definována jako

$$L^{*\ominus} = \bigcup_{i \geq 0} L^{i\ominus}$$

  kde $L^{0\ominus} = \lambda$, $L^{1\ominus} = L$, $L^{n\ominus} = L \ominus L^{(n-1)\ominus}$.

Poznamenejme, že pokud budeme obě předchozí operace aplikovat za sebou, pak je můžeme zaměnit, formálně $(L^{*\ominus})^{*\oslash} = (L^{*\oslash})^{*\ominus} = L^{**}$, což odpovídá faktu, že Σ^{**} značí množinu všech možných obrazů nad abecedou Σ.

Příklad 4: Uvažujme jazyk L z Příkladu 1, což je množina obrazů, které mají tři sloupce. Výsledkem sloupcové iterace jazyka L je množina obrazů, jejichž počet sloupců je násobkem tří.

U obrazů má smysl definovat některé operace, které v případě jednorozměrných řetězců smysl nemají. Příkladem takové operace je rotace nebo řádkově-sloupcová kombinace.

Definice 7: Nechť p je obraz. *Pravotočivá rotace* obrazu p, označme ji p^R, je definována jako:

$$p^R = \begin{array}{|ccc|}\hline p_{m,1} & \cdots & p_{1,1} \\ \vdots & \ddots & \vdots \\ p_{m,n} & \cdots & p_{1,n} \\ \hline\end{array}$$

Obrázek 6 Pravotočivá rotace obrazu *p*

Rotace obrazu může být přirozeně rozšířena na rotaci dvourozměrného jazyka, ten pak označujeme jako L^R.

Definice 8: Nechť Σ je konečná abeceda a nechť $S_1, S_2 \subseteq \Sigma^*$ jsou dva jednorozměrné jazyky nad abecedou Σ. *Řádkově-sloupcová kombinace* jazyků S_1 a S_2 je dvourozměrný jazyk $L = S_1 \oplus S_2 \subseteq \Sigma^{**}$ takový, že obraz $p \in \Sigma^{**}$ patří do jazyka L právě tehdy když řetězce odpovídající řádkům patří do S_1 a řetězce odpovídající sloupcům patří do S_2.

Definice 9: Nechť p je obraz o velikosti (m, n). *Blok* (nebo *podobraz*) obrazu p je obraz p', který je podpolem obrazu p. Tj., pokud (m', n') je velikost obrazu p', pak $m' \leq m$ a $n' \leq n$ a existuji celé čísla h, k, $(h \leq m - m', k \leq n - n')$, že $p'(i,j) = p(i + h, j + k)$ pro všechna $0 \leq i \leq m'$ a $0 \leq j \leq n'$.

Definice 10: Pro libovolný obraz p o velikosti (m, n) definujeme obraz $\hat{p}$ o velikosti $(m + 2, n + 2)$, který získáme tak, že obraz p obklopíme speciálními okrajovými symboly #, kde $\# \notin \Sigma$. Obraz $\hat{p}$ je zobrazen na následujícím obrázku.

$$\hat{p} = \begin{array}{|ccccc|}\hline \# & \# & \cdots & \# & \# \\ \# & p_{1,1} & \cdots & p_{1,n} & \# \\ \vdots & & \ddots & & \vdots \\ \# & p_{m,1} & \cdots & p_{m,n} & \# \\ \# & \# & \cdots & \# & \# \\ \hline\end{array}$$

Obrázek 7 Obklopující obraz obrazu *p*

Definice 11: Nechť Σ a Γ jsou dvě konečné abecedy, $p \in \Gamma^{**}$ je obraz a $\pi: \Gamma \longrightarrow \Sigma$ je projekce definovaná následovně: Projekce π obrazu p je obraz $p' \in \Sigma^{**}$ takový, že $p'(i,j) = \pi(p(i,j))$, pro všechna $1 \leq i \leq l_1(p), 1 \leq j \leq l_2(p)$.

Definice dalších vlastností obrazů a rovněž další příklady dvourozměrných jazyků lze nalézt například v (Giammaressi & Restivo, 2004), kapitola 2.

Regulární výrazy

Základní operace na dvourozměrných jazycích, které byly uvedeny v předchozí kapitole, mohou být použity na elementární dvourozměrné jazyky za účelem získání tříd dvourozměrných jazyků.

Definice 12: Mějme abecedu Σ. Pak prázdný jazyk $\emptyset$ a každý jazyk $\{\boxed{a}\}$, kde $a \in \Sigma$, nazýváme *atomickými jazyky* nad abecedou Σ. Nechť $\mathcal{R}$ je množina *regulárních operací* definovaná jako $\mathcal{R} = \{\ominus, \oslash, *\ominus, *\oslash, \cup, \cap, {}^{c}\}$.

Jazyk nad abecedou Σ je regulární, pokud ho lze získat z nějakého atomického jazyka pomocí konečně mnoha aplikací operací z $\mathcal{R}$. Regulární výraz je pak předpis, který udává, jakým způsobem je daný jazyk pomocí regulárních operací z atomických jazyků získán.

Definice 13: *Regulární výraz* nad abecedou Σ je definován rekurzivně následovně:

- $\emptyset$ a každý symbol $a \in \Sigma$ je regulární výraz.
- Pokud α a β jsou regulární výrazy, pak
    - $(\alpha) \cup (\beta)$,
    - $(\alpha) \cap (\beta)$,
    - ${}^{c}(\alpha)$,
    - $(\alpha) \oslash (\beta)$,
    - $(\alpha) \ominus (\beta)$,
    - $(\alpha)^{*\oslash}$,
    - $(\alpha)^{*\ominus}$

  jsou regulární výrazy.

Každý regulární výraz nad abecedou Σ značí dvourozměrný jazyk nad abecedou Σ:

- $\emptyset$ značí prázdný jazyk
- a značí jazyk $\{\boxed{a}\}$
- $(\alpha) \cup (\beta)$ značí sjednocení jazyků α a β
- $(\alpha) \cap (\beta)$ značí jejich průnik
- $(\alpha) \oslash (\beta)$ a $(\alpha) \ominus (\beta)$ značí jejich sloupcovou a řádkovou konkatenaci
- $(\alpha)^{*\oslash}$ a $(\alpha)^{*\ominus}$ značí jejich sloupcovou a řádkovou iteraci
- ${}^{c}(\alpha)$ značí doplněk jazyka α

Definice 14: Dvourozměrný jazyk $L \subseteq \Sigma^{**}$ je regulární, pokud existuje regulární výraz nad abecedou Σ, který ho značí. Třída regulárních dvourozměrných jazyků se pak označuje jako $\mathcal{L}(RE)$.

Příklad 5: Nechť $\Sigma = \{a, b\}$. Regulární výraz

$$(((a \ominus \mathrm{b})^{*\ominus}) \oslash ((\mathrm{b} \ominus \mathrm{a})^{*\ominus}))^{*\oslash}$$

značí jazyk všech „šachovnic“, které mají sudou délku strany, tj. jsou to obrazy v tomto tvaru:

a	b	a	b	a	b	a	b
b	a	b	a	b	a	b	a
a	b	a	b	a	b	a	b
b	a	b	a	b	a	b	a
a	b	a	b	a	b	a	b
b	a	b	a	b	a	b	a
a	b	a	b	a	b	a	b
b	a	b	a	b	a	b	a

Obrázek 8 Obraz dle definice jazyka z Příkladu 5

Speciální regulární dvourozměrné jazyky

Je zajímavé uvažovat následující podmnožiny množiny regulárních operaci $\mathcal{R}$:

- $\mathcal{R}_1 = \{\ominus, \oslash, *\ominus, *\oslash, \cup, \cap\}$
- $\mathcal{R}_2 = \{\ominus, \oslash, \cup, \cap, ^c\}$

Regulární výrazy, které obsahují pouze operace z $\mathcal{R}_1$ nazýváme *regulární výrazy bez doplňku* (CFRE — complementation-free regular expressions). Jazyky, které tyto regulární výrazy značí, jsou regulární jazyky bez doplňku a třídu těchto jazyků označujeme jako $\mathcal{L}(CFRE)$.

Příklad 6: Jazyk z Příkladu 5 patří do $\mathcal{L}(CFRE)$, protože je vyjádřen regulárním výrazem bez doplňku.

Regulární výrazy obsahující pouze operace z $\mathcal{R}_2$ nazýváme *regulární výrazy bez iterace* (SFRE—star-free regular expressions). Jazyky, které tyto regulární výrazy značí, jsou regulární jazyky bez iterace a třídu těchto jazyků označujeme jako $\mathcal{L}(SFRE)$.

Příklad 7: Uvažujme dvourozměrný jazyk L všech obrazů nad abecedou $\Sigma = \{a, b\}$ takových, že existuje alespoň jeden řádek, který obsahuje dva po sobě následující výskyty symbolu a. Jazyk $L \in \mathcal{L}(SFRE)$, protože může být vyjádřen regulárním $^c(\emptyset) \ominus (^c(\emptyset) \oslash (\mathrm{a} \oslash \mathrm{a}) \oslash {}^c(\emptyset)) \ominus {}^c(\emptyset)$, což je regulární výraz bez iterace.

Automaty

V této kapitole popíšeme dva různé typu automatů, které čtou dvourozměrnou pásku. Prvním z nich je *čtyřsměrný automat*, což je sekvenční model, a druhým je dvourozměrný *online teselační automat*, což je model vycházející z celulárních automatů.

Čtyřsměrné automaty

V roce 1967 Blum a Hewit zavedli model konečného automatu, který čte dvourozměrnou pásku. Nazvali ho *čtyřsměrný automat* a jde o rozšíření klasického konečného automatu, který přijímá řetězce, tím způsobem, že je umožněn pohyb ve čtyřech směrech — Vlevo, Vpravo, Nahoru a Dolů.

Definice 15: Nedeterministický (deterministický) čtyřsměrný konečný automat, označujeme ho jako 4NFA (4DFA), je sedmice $\mathcal{A} = (\Sigma, Q, \Delta, q_0, q_a, q_r, \delta)$, kde:

- Σ je vstupní abeceda
- Q je konečná množina stavů
- $\Delta = \{R, L, U, D\}$ je množina „směrů“
- $q_0 \in Q$ je „startovací“ stav
- $q_a, q_r \in Q$ jsou „přijímací“ (accepting) a „zamítací“ (rejecting) stav
- $\delta: Q - \{q_a, q_r\} \times \Sigma \longrightarrow 2^{Q \times \Delta}$ ($\delta: Q - \{q_a, q_r\} \times \Sigma \longrightarrow Q \times \Delta$) je přechodová funkce

Podobně jako v případě jednorozměrných jazyků jde o model konečného řízení pomocí množiny stavů Q, který čte vstupní obraz. Pohyb čtecí hlavy závisí na přechodové funkci δ:, která z aktuálního stavu se symbolem na aktuální pozici na pásce přejde do nového stavu a přesune čtecí hlavu o jednu pozici v daném směru. V případě dosažení stavu q_a nebo q_r, 4FA zastaví, protože pro tyto stavy není definován žádný přechod.

Používá se konvence, že 4FA čte ohraničený obraz $\hat{p}$ a když se čtecí hlava dostane na ohraničující symbol #, pak se v následujícím kroku ihned vrátí zpět do p. Jinými slovy, automat “ví”, že je blízko okraje a nikdy se mimo p nedostane.

Čtyřsměrný konečný automat přijímá (rozpoznává) obraz $p \in \Sigma^{**}$, jesliže se ze startovací pozice $(1,1)$ a startovacího stavu může přesunovat a může zastavit ve přijímacím stavu q_a. Kromě výše uvedeného rozšíření možných směrů přechodu, existují dva další podstatné rozdíly od klasického konečného automatu:

- Není vyžadováno, aby 4FA přečetl všechny symboly vstupního obrazu.
- 4FA se může na libovolnou pozici vstupního obrazu pomocí konečně stavového řízení libovolněkrát vrátit.

Příklad 8: Nechť $\Sigma = \{0,1\}$ je abeceda a nechť $L \subseteq \Sigma^{**}$ je dvourozměrný jazyk, pro jehož každý obraz platí, že první sloupec je roven poslednímu sloupci. Formálně

$$L = \{p | p \in \Sigma^{**} \wedge p(i,1) = p(i, l_2(p), i = 1, \dots, l_1(p)\}$$

Jazyk L je přijímán čtyřsměrným deterministickým konečným automatem, který pracuje následovně. Prochází vstupní obraz p řádek po řádku zleva doprava, shora dolů a současně kontroluje, zda všechny pozice obsahují symboly ze Σ a že první symbol každého řádku se rovná poslednímu symbolu stejného řádku.

Příklad 9: Nechť $\Sigma = \{0,1\}$ je abeceda a nechť L je podmnožina Σ^{**} taková, že všechny obrazy v ní obsažené mají tvar „čtverce“ o liché délce strany, kde na prvek centrální pozici je roven 1. Příklad takového obrazu následuje:

1	1	0	0	0
0	0	1	0	0
1	0	**1**	0	1
1	1	1	1	1
1	0	0	1	0

Obrázek 9 Obraz dle definice jazyka z Příkladu 9

Jazyk L je přijímán čtyřsměrným nedeterministickým konečným automatem, který pracuje následovně. Ve vstupním obraze p se ze startovní pozice $(1,1)$ pohybuje po diagonále (tj. jeden krok dolů, jeden krok doprava). V některém okamžiku, který je určen nedeterministicky, si tento 4NFA zapamatuje symbol nacházející se na aktuální pozici a následně se začne pohybovat dolů po druhé diagonále (tj. jeden krok dolů, jeden krok doleva). Pokud se takový 4NFA dostane do levého dolního roku obrazu p, pak obraz p má tvar čtverce o liché délce strany, na jehož centrální pozici se nachází onen zapamatovaný symbol. Pokud zapamatovaný symbol je 1, pak automat obraz p přijme, v opačném případě obraz p odmítne.

Třídu jazyků přijímaných čtyřsměrnými nedeterministickými automaty je označujeme $\mathcal{L}(4NFA)$ a třídu jazyků přijímaných čtyřsměrnými deterministickými automaty označujeme $\mathcal{L}(4DFA)$. Ačkoliv 4FA jsou přirozeným rozšířením konečných automatů do dvou dimenzí, nezachovávají většinu důležitých vlastností konečných automatů pro jednorozměrné řetězce.

Teorém 1: $\mathcal{L}(4DFA) \subset \mathcal{L}(4NFA)$.

Teorém 2: $\mathcal{L}(4DFA)$ a $\mathcal{L}(4NFA)$ nejsou uzavřeny vůči řádkové a sloupcové konkatenaci, ani vůči iteraci.

Teorém 3: $\mathcal{L}(4DFA)$ a $\mathcal{L}(4NFA)$ jsou uzavřeny vůči boolovskému sjednocení a průniku. $\mathcal{L}(4DFA)$ je navíc uzavřena vůči doplňku.

Důkazy Teorémů 1, 2 a 3 lze nalézt v (Giammaressi & Restivo, 2004), kapitola 4.

Online teselační automat

Čtyřsměrný automat z předchozí kapitoly pracuje sekvenčně. V této kapitole uvedeme celulární automat, který pracuje nad celou páskou současně. Neformálně je dvourozměrný celulární automat pole buněk, přičemž každá z těchto buněk je v daném časovém okamžiku v nějakém stavu. V každém kroku každá buňka přejde do nového stavu v závislosti na stavu sousedních buněk.

Dvourozměrný online teselační automat zavedli Inoue a Nakamura. Jde o omezený dvourozměrný celulární automat, kde jednotlivé buňky nemění své stavy v každém kroku, ale „vlna“ přechodů přejde skrz pole diagonálně.

Definice 16: Nedeterministický (deterministický) dvourozměrný online teselační automat, označujeme ho jako 2OTA (2DOTA), je definován jako pětice $\mathcal{A} = (\Sigma, Q, I, F, \delta)$, kde:

- Σ je vstupní abeceda
- Q je konečná množina stavů
- $I \subseteq Q$ ($I = \{i\}$, i ∈ Q) je množina počátečních stavů
- $F \subseteq Q$ je množina koncových stavů
- $\delta: Q \times Q \times \Sigma \longrightarrow 2^Q$ ($\delta: Q \times Q \times \Sigma \longrightarrow Q$) je přechodová funkce

2OTA $\mathcal{A}$ pracuje na obrazu $p \in \Sigma^{**}$ tak, že každé pozici (i, j) obrazu p přiřadí stav z Q. Tento stav je dán přechodovou funkcí δ a závisí na stavech přiřazených k pozici $(i-1, j)$ a $(i, j-1)$ obrazu p a na symbolu $p(i, j)$.

#	#	#	#	#	#	#	#
#							#
#							#
#							#
#					$p(i-1, j)$		#
#				$p(i, j-1)$	$p(i, j)$		#
#							#
#	#	#	#	#	#	#	#

Obrázek 10 Ilustrace k definici 16

V čase $t = 0$ je počáteční stav q_0 přiřazen všem pozicím prvního řádku a prvního sloupce obrazu $\hat{p}$. Výpočet pak proběhne během $l_1(p) + l_2(p) - 1$ kroků. Začne v čase $t = 1$ přečtením pozice $p(1,1)$ a přiřazením stavu $\delta(q_0, q_0, p(1,1))$ na pozici $(1,1)$. V čase $t = 2$ jsou stavy přiřazeny současně na pozice $(1,2)$ a $(2,1)$ a tak dále po diagonále. V čase $t = k$ jsou současně přiřazeny stavy na všechny pozice (i, j) takové, že $i + j - 1 = k$. 2OTA $\mathcal{A}$ přijímá obraz p, pokud existuje výpočet $\mathcal{A}$ na obrazu p takový, že stav přiřazený pozici $(l_1(p), l_2(p))$ patří do množiny koncových stavů F.

Příklad 10: Nechť $\Sigma = \{a\}$ je abeceda a nechť $L \subseteq \Sigma^{**}$ je jazyk všech „čtverců“ nad abecedou Σ, formálně viz Příklad 2. Nedeterministický dvourozměrný online teselační automat může jazyk L přijímat následovně. Pozicím na hlavní diagonále obrazu p přiradí stav 1, pozicím pozicím nad hlavní diagonálou obrazu p přiřadí stav 2 a pozicím pod hlavní diagonálou obrazu p přiřadí stav 3. Pak je obraz p přijat právě tehdy, když pozice v pravém dolním rohu obrazu p obsahuje stav 1. Formálně: Jazyk L je přijímán 2OTA $\mathcal{A} = (\Sigma, Q, I, F, \delta)$, kde:

- $Q = \{0,1,2,3\}$
- $I = \{0\}$
- $F = \{1\}$
- $\delta(0,0,a) = \delta(2,3,a) = 1$
  $\delta(0,1,a) = \delta(0,2,a) = \delta(2,1,a) = \delta(2,2,a) = 2$
  $\delta(1,0,a) = \delta(3,0,a) = \delta(1,3,a) = \delta(3,3,a) = 3$

Třídu jazyků, které jsou přijímány pomocí 2OTA, značíme $\mathcal{L}(2OTA)$ a třídu jazyků, které přijímá přijímaných pomocí 2DOTA značíme $\mathcal{L}(2DOTA)$.

Teorém 4: Třída $\mathcal{L}(2DOTA)$ je ostře pod třídou $\mathcal{L}(2OTA)$.

Odkaz na důkaz Teorému 4 lze nalézt v (Giammaressi & Restivo, 2004), kapitola 4.

Bez důkazů uvedeme několik důležitých vlastností třídy $\mathcal{L}(2OTA)$, přičemž důkazy jsou v kapitole 4 (Giammaressi & Restivo, 2004).

Teorém 5: Třída $\mathcal{L}(2OTA)$ je uzavřena vůči projekci, řádkové i sloupcové konkatenaci.

Teorém 6: Třída $\mathcal{L}(4NFA)$ je podtřídou třídy $\mathcal{L}(2OTA)$.

Gramatiky

Předchozí modely založené na automatech sloužily k přijímání tříd dvourozměrných jazyků. Pro jejich generování bylo zavedeno také několik různých modelů. V tomto textu se zaměříme především na gramatiky zobecněné do dvou rozměrů, které jsou založeny na bezkontextových a regulárních gramatikách a obsahují dvě množiny pravidel — množinu pravidel horizontálních a množinu pravidel vertikálních.

Tyto modely pracují tím způsobem, že nejprve na základě množiny horizontálních pravidel vygenerují řetězec symbolů σ a pak pomocí paralelní aplikace vertikálních pravidel z jednotlivých symbolů řetězce σ vygenerují obdélníkový obraz.

Definice 17: Dvourozměrná, pravě lineární gramatika (2RLG) je definována sedmicí $G = (V_h, V_v, \Sigma_I, \Sigma, S, R_h, R_v)$, kde:

- V_h je konečná množina horizontálních proměnných
- V_v je konečná množina vertikálních proměnných
- $\Sigma_I \subseteq V_v$ je konečná množina „horizontálních terminálů“
- Σ je konečná množina terminálů
- $S \in V_h$ je startovací symbol
- R_h je konečná množina horizontálních pravidel ve tvaru $V \longrightarrow AV'$ nebo $V \longrightarrow A$, kde $V, V' \in V_h, A \in \Sigma_I$
- R_h je konečná množina vertikálních pravidel ve tvaru $W \longrightarrow AW'$ nebo $W \longrightarrow a$, kde $W, W' \in V_v, a \in \Sigma$

Derivační krok pak má dvě fáze:

- V první fázi je gramatikou $G_h = (V_h, \Sigma_I, S, R_h)$ vygenerován (jednorozměrný) jazyk $H(G)$ nad abecedou Σ_I. Řetězce jazyka $H(G)$ jsou pak považovány za horní řádek obrazu, který se bude generovat.

- V druhé fázi je pak každý symbol řetězce z jazyka $H(G)$ považován za startovací symbol a generování podle vertikálních pravidel z R_h probíhá paralelně nad všemi symboly tohoto řetězce. Pravidla z R_h musí být aplikována paralelně z toho důvodu, aby bylo zajištěno, že všechna terminální pravidla (tj. pravidla ve tvaru $V_i \longrightarrow a_i$) budou ve všech sloupcích aplikována současně.

Příklad 11: Nechť $G = (V_h, V_v, \Sigma_I, \Sigma, S, R_h, R_v)$ je gramatika, kde:

- $V_h = \{S, T\}$
- $V_v = \{A, B, C, D\}$
- $\Sigma_I = \{A, B\}$
- $\Sigma = \{0,1\}$
- $R_h = \{S \longrightarrow AT, T \longrightarrow BS, T \longrightarrow B\}$
- $R_v = \{A \longrightarrow 1C, C \longrightarrow 0A, C \longrightarrow 0, B \longrightarrow 0D, D \longrightarrow 1B, D \longrightarrow 1\}$

V první fázi gramatika G generuje jazyk $H(G) = \{AB\}^+$. V druhé fázi gramatika odstartuje na řetězci z jazyka $H(G)$, který je považován za horní řádek výsledného obrazu, a budou aplikována pravidla z R_v. Tímto způsobem získáme jazyk L vygenerovaná gramatikou G, což je množina „šachovnic“ o sudé délce strany. Příkad takového obrazu je na následujícím obrázku:

1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1

Obrázek 11 Ilustrace k Příkladu 11

Třídu jazyků generovaných dvourozměrnými pravě lineárními gramatikami označujeme $\mathcal{L}(2RLG)$.

Teorém 7: Třída $\mathcal{L}(2RLG)$ je vlastní podtřídou třídy $\mathcal{L}(4DFA)$.

Teorém 8: Třída $\mathcal{L}(2RLG)$ je uzavřena vůči projekci.

Důkazy teorémů 7 a 8 je uveden v (Giammaressi & Restivo, 2004), kapitola 5.

Dvourozměrné jazyky je také možné popsat pomocí logických formulí, což však přesahuje rámec tohoto textu. Úvodu do souvislosti dvourozměrných jazyků a logických formulí se věnuje například kapitola 6 (Giammaressi & Restivo, 2004), kde je možné nalézt i odkazy na další související literaturu.

Další velkou kapitolou teorie dvourozměrných jazyků jsou, podobně jako v případě jednorozměrných jazyků, meze rozhodnutelnosti. S tímto tématem souvisí především tzv. tiling systémy a domino systémy. V této práci se však budeme věnovat spíše praktickému využití dvourozměrných jazyků a jejich souvislosti

s digitálními obrazy. Zájemce o problematiku rozpoznatelnosti dvourozměrných jazyků odkážeme na poměrně obsáhlé kapitoly 6—11 v (Giammaressi & Restivo, 2004).

Digitální obraz a formální jazyky

V předchozí kapitole jsme definovali dvourozměrný jazyk jako množinu dvourozměrných řetězců nad nějakou abecedou. Dvourozměrné řetězce můžeme interpretovat jako digitální obraz (proto se jim také říká obrazy). Digitální šedotónový obraz s rozlišením $m \times n$ je reprezentován funkcí, která každé pozici přiřadí nějakou číselnou hodnotu, kterou můžeme považovat za úroveň šedi. V případě obrazu o rozlišení $2^n \times 2^n$ můžeme libovolný jeho bod adresovat pomocí řetězce nad abecedou Σ, která má 4 symboly, $\Sigma = \{0,1,2,3\}$.

V této kapitole vysvětlíme, jakým způsobem lze obraz s konečným rozlišením reprezentovat funkcí $f: \Sigma^n \longrightarrow \mathbb{R}$. Dále vysvětlíme, jak lze dvourozměrné řetězce, které obsahují daný obraz v několika různých rozlišeních zároveň, tzv. multiresolution obrazy, reprezentovat funkcí $g: \Sigma^* \longrightarrow \mathbb{R}$. Také popíšeme algoritmus, který pro daný multiresolution obraz nalezne minimální FA (pokud takový FA existuje), který daný obraz reprezentuje.

Černobílé obrázky a konečné automaty

Uvažujme čtvercový obraz s rozlišením $2^n \times 2^n$ (kde typicky $7 \leq n \leq 11$), kde každému pixelu (pozici) tohoto obrazu je přiřazena pravdivostní hodnota — hodnota 0 reprezentuje černou barvu, hodnota 1 barvu bílou. Pro účely popisu obrazu pomocí automatu a pro účely manipulace s obrazem je vhodné každému pixelu obrazu přiřadit adresu.

Definice 18: Mějme dvourozměrný obraz p o rozlišení $2^n \times 2^n$. Pak slovo w, jehož délka $|w| = n$, nad abecedou $\Sigma = \{0,1,2,3\}$ udává adresu každého pixelu v obrazu p takto:

- Symbol ε označuje adresu celého čtvercového obrazu p.
- Kvadranty obrazu p jsou adresovány jednou číslicí dle následujícího obrázku:

p:

1	3
0	2

- Čtyři kvadranty podobrazu, jehož adresa je w, jsou adresovány pomocí řetězců $w0, w1, w2$ a $w3$.

Příklad 12: Mějme obraz p s rozlišením 16×16 pixely. Přiřazení adres všem jeho pixelům je zobrazeno na následujícím obrázku, přičemž pixel adresou 203 je zobrazen na černém pozadí.

111	113	131	133	311	313	331	333
110	112	130	132	310	312	330	332
101	103	121	123	301	303	321	323
100	102	120	122	300	302	320	322
011	013	031	033	211	213	231	233
010	012	030	032	210	212	230	232
001	003	021	023	201	203	221	223
000	002	020	022	200	202	220	222

Obrázek 12 Ilustrace k Příkladu 12

Abychom byli schopni plně specifikovat černobílý obraz s rozlišením $2^m \times 2^m$, potřebujeme specifikovat buď funkci $\Sigma^m \longrightarrow \{0,1\}$ nebo množinu adres všech pixelů, které mají být černé, tj. jazyk $L \subseteq \Sigma^m$. V praxi je často vhodné uvažovat multiresolution obrazy.

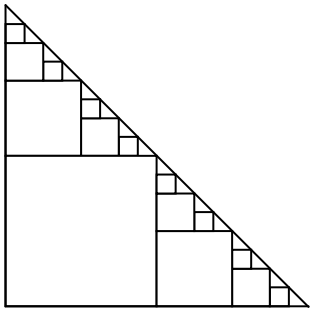
Definice 19: Multiresolution obraz je obraz, který je specifikován pro všechna možná rozlišení současně.

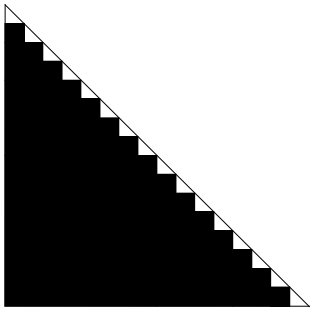
Příklad 13: Šachovnice o rozměrech 2×2 vypadá ve všech rozlišeních $2^m \times 2^m, m \geq 1$ stejně. Je-li velikost šachovnice rovna m, pak množinu adres všech černých polí této šachovnice specifikuje výraz $\{1,2\}\Sigma^{m-1}$, jde-li o multiresolution obraz, pak $\{1,2\}\Sigma^*$.

Příklad 14: Chceme-li specifikovat šachovnici o rozměrech 8×8, můžeme ji získat tak, že do všech čtyřech kvadrantů čverce o rozměrech 8×8 vložíme šachovnici o rozměrech 2×2. Pak množinu adres všech černých polí této šachovnice specifikuje regulární výraz $\Sigma^2\{1,2\}\Sigma^*$. Všimněme si, že výraz $\Sigma^2\{1,2\}\Sigma^*$ je konkatenací dvou regulárních výrazů, Σ^2 a $\{1,2\}\Sigma^*$. Obecně platí, že je-li obraz L popsán konkatenací dvou jazyků, $L = L_1L_2$, pak obraz L lze vždy získat vložením obrazu L_2 do všech čtverců, které jsou adresovány jazykem L_1. Proto můžeme šachovnici o rozměrech 8×8 získat také vložením 4 kopií šachovnice o rozměrech 4×4, popsaných regulárním výrazem $\Sigma\{1,2\}\Sigma^*$, do čtverců s adresami $0,1,2$ a 3, lze popsat konkatenací Σ a $\Sigma\{1,2\}\Sigma^*$.

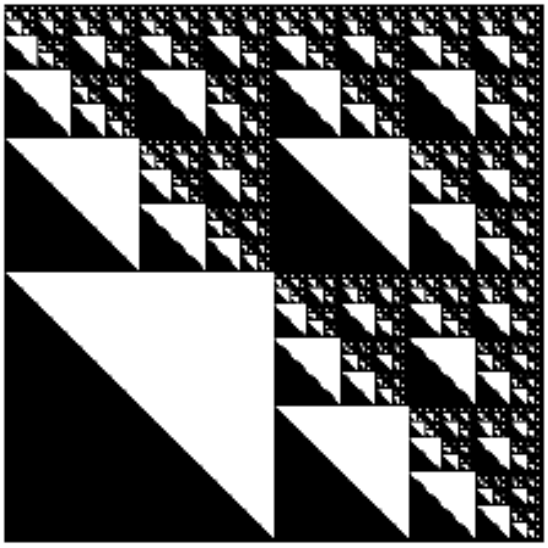
Příklad 15: Jazyk $L_1 = \{1,2\}^*0$ představuje adresy všech nekonečně mnoha čtverců zobrazených na následujícím obrázku vlevo.

Příklad 16: Budeme-li mít jazyk $L_2 = \Sigma^*$ představující adresy všech černých čtverců, pak konkatenací $L_1L_2 = \{1,2\}^*0\Sigma^*$ „vybarvíme“ všechny čtverce předchozího příkladu, výsledek je zobrazen na následujícím obrázku vpravo.





Příklad 17: Vložením trojúhelníku popsaného jazykem L_1L_2 z předchozího příkladu do všech čtverců s adresami $L_3 = \{1,2,3\}^*0$ získáme obraz $L_3L = \{1,2,3\}^*0\{1,2\}^*0\Sigma^*$. Několik prvních vložení znázorňuje následující obrázek.



Obrázek 13 Ilustrace k Příkladu 17, převzato z (Mindek, Finite State Automata and Image Recognition, 2004)

Nezbytnou podmínkou, aby mohl být multiresolution obraz reprezentován regulární množinou, je, aby tento obraz měl konečný počet různých podobrazů ve všech podčtvercích s adresami z Σ^*. Dále uvidíme, že tato podmínka je také postačující. Obrazy, které mohou být perfektně (tj. s nekonečnou přesností) popsány regulárními výrazy (a tím pádem i konečnými automaty), jsou obrazy charakteru fraktálů, jejichž typickou vlastností je soběpodobnost (definice viz např. (Žára, Beneš, Sochor, & Felkel, 2005), kapitola 8). Pomocí regulárních výrazů však může být aproximován každý obraz, nicméně čím menší chybu aproximace požadujeme, tím větší bude výsledný automat (regulární výraz), který obraz popisuje.

Algoritmus 1: Konstrukce konečného automatu, který specifikuje vstupní obraz — pokud takový automat existuje, tj. pokud daný obraz má jen konečný počet různých podobrazů.

Vstup: Dvourozměrný obraz I.
Notace: I_w značí takovou zvětšenou část obrazu I, která je adresována slovem w. Část obrazu, která je reprezentována stavem x označujeme jako ψ_x.

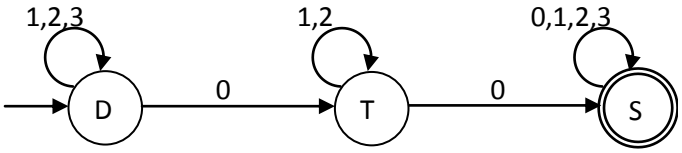
Algoritmus:

1. $i = j = 0$
2. Vytvoř stav 0 a nastav $\psi_0 = I$
3. Předpokládejme, že $\psi_i = I_w$. Zpracuj stav i, tj. pro $k = 0,1,2,3$:
    - Pokud $I_{wk} = \psi_q$ pro nějaké q, pak vytvoř přechod se symbolem k ze stavu i do stavu q; jinak nastav $j = j + 1, \psi_j = I_{wk}$ a vytvoř přechod se symbolem k ze stavu i do nového stavu j.

4. Pokud $i = j$, pak jsou všechny stavy již zpracovány a algoritmus končí; jinak nastav $i = i + 1$ a pokračuj bodem 3.

Neformálně řečeno, Algoritmus 1 končí, pokud existuje automat, který perfektně popisuje vstupní obraz a jeho výsledkem je minimální deterministický automat.

Příklad 18: Uvažujme jazyk z příkladu 17. Algoritmus 1 zkonstruuje následující deterministický minimální konečný automat:



Nejprve je vytvořen stav D. Pro symbol 0 je vytvořen nový stav T, pro symboly $1,2,3$ se cyklí ve stavu D. Pak je zpracován stav T tak, že pro symbol 0 je vytvořen nový stav S a pro symboly $1,2$ se cyklí ve stavu T. Jelikož třetí kvadrant je pro T prázdný, nevychází se stavu T žádná přechod se symbolem 3. Nakonec je zpracován stav S tak, že všechny symboly v tomto stavu cyklí.

V této kapitole jsme uvažovali pouze černobílé digitální obrazy, kde plně postačuje hodnotu pixelu definovat jako pravdivostní hodnotu ($0/1$, $černá/bílá$, $nepravda$/pravda). Praxe si ale vyžaduje práci s minimálně šedotónovými obrazy, kde hodnota pixelu je dána obecně reálným číslem, které je digitalizováno obvykle na hodnoty z intervalu 0 $až$ $2^k - 1$, kde typicky $k = 8$. Touto problematikou se zabývá kapitola třetí a čtvrtá (Culik & Kari, 2004), kde lze nalézt podrobnější informace nejen o šedotónových obrazech a jejich souvislosti s tzv. váženými konečnými automaty, ale také úvod do vážených konečných převodníků, které v praxi mohou sloužit pro jisté transformace mezi obrazy.

Konečné automaty a rozpoznávání obrazu

V poslední kapitole tohoto textu na základě definic dvourozměrného jazyka, co by množiny dvourozměrných řetězců (obrazů), adresace jeho pixelů a algoritmu konstrukce konečného automatu pro stručně zmíníme jednu z možných aplikací teorie dvourozměrných jazyků.

Jedná se zejména o rozpoznávání obrazu pomocí konečného automatu. Chceme-li obraz zachytit precizně, je potřeba velkého rozlišení, což znamená použití velkého množství pixelů a tím pádem velké množství dat. Při analýze obrazu pak může být obtížné všechna tato data zpracovat přímo, takže v praxi se proces analýzy často rozděluje do dvou fází:

1. Nalezení důležitých rysů obrazu (feature extraction). Tato fáze probíhá buď automaticky nebo manuálně a často bez potřeby hlubšího pochopení obsahu obrazu. Klasickými příklady rysů jsou hrany, rohy, regiony apod. Je zřejmé, že kvalita a spolehlivost metod extrakce rysů má významný vliv na výkon celého procesu analýzy obrazy. K extrakci rysů existuje velké množství literatury, např. (Nixon & Aguado, 2002), podrobnější informace by však byly mimo hlavní téma tohoto textu.

2. Rysy extrahované v prvním kroku jsou použity pro analýzu obrazu. Například práce (Mindek, Finite State Automata and Image Recognition, 2004) uvádí algoritmy pro:
    - Konstrukci automatu sloužícího pro rozpoznávání. Jde o Algoritmus 1 uvedený v předchozí kapitole této a jeho detailní popis lze nalézt na straně 9 díla (Mindek, Finite State Automata and Image Recognition, 2004). Připomeňme, že tento algoritmus končí, jestliže existuje automat, který perfektně (přesněji: s chybou dané velikosti) specifikuje vstupní obraz a jeho výstupem je automat s minimálním počtem stavů.
    - Rekonstrukci obrazu na základě automatu. Jde o postup, který na základě automatu z Algoritmu 1 rekonstruuje obraz. Podrobný popis tohoto postupu lze nalézt v (Mindek, Finite State Automata and Image Recognition, 2004), kapitola 4.

Problematika rozpoznávání vzorů na základě konečných automatů je poměrně obsáhlá, podrobnější informace a odkazy na další literaturu lze nalézt například v (Mindek, Finite State Automata and Image Recognition, 2004) a (Žďárek & Melichar, 2006).

Závěr

Účelem tohoto textu bylo čtenáři jednak přehledně podat základní náhled do celé obsáhlé teorie dvourozměrných jazyků a také mu poskytnout odkazy na relevantní literaturu, v níž je možno najít mnohem podrobnější informace, včetně kompletního znění důkazů teorémů, které jsou v této práci uvedeny bez důkazů.

V úvodních kapitolách byly uvedeny základy teorie dvourozměrných jazyků, operací nad nimi, modely pro jejich přijímání a generování, v druhé části textu byla pospána jejich souvislost s digitálními obrazy a v závěru jsme se velmi stručně zmínili o jejich možném použití v praxi.

Citovaná literatura

Culik, K., & Kari, J. (2004). Digital images and formal languages. In S. A. Rozenberg G., *Handbook of Formal Languages, vol 3 Beyond Word* (pp. 599-616). Springer.

Giammaressi, D., & Restivo, A. (2004). Two-dimensional languages. In S. A. Rozenberg G., *Handbook of formal languages, vol.3 Beyond words* (pp. 215-267). Springer.

Meduna, A. (2000). *Automata and languages: theory and applications.* London: Springer-Verlag.

Mindek, M. (2004). Finite State Automata and Image Recognition. *CEUR Workshop Proceedings*, (pp. 141-151). 2004.

Mindek, M., & Burda, M. (2006). Storage, Indexing and Recognition with Finite State Automata. *IMECS* (pp. 609-613). HongKong: IAENG.

Nixon, M., & Aguado, A. (2002). *Feature Extraction in Computer Vision and Image Processing.* Newnes.

Techet, J., Masopust, T., & Meduna, A. (2009, 12 1). *Transducers and Translation Grammars*. Retrieved 2 4, 2009, from TID: Moderní teoretická informatika: http://www.fit.vutbr.cz/~meduna/work/doku.php?id=lectures:phd:tid:tid

Žára, J., Beneš, B., Sochor, J., & Felkel, P. (2005). *Moderní počítačová grafika.* Brno: Computer Press.

Žďárek, J., & Melichar, B. (2006). On Two-Dimensional Pattern Matching by Finite Automata. *In Implementation and Application of Automata - CIAA 2005, LNCS 3845* (pp. 329-340). Heidelberg: Springer.