Automata and Grammars (BIE-AAG)

6. Regular expressions

Jan Holub

Department of Theoretical Computer Science

Faculty of Information Technology

Czech Technical University in Prague

© Jan Holub, 2011

Regular expressions

Definition

Regular expression E *over alphabet* Σ is defined as:

1. $\emptyset, \varepsilon, a$ are regular expressions for all $a \in \Sigma$.
2. If x, y are regular expressions over Σ, then:

   (a) $(x + y)$ (union, alternation),

   (b) $(x.y)$ (concatenation),

   (c) $(x)^*$ (Kleene star)

   are regular expressions over Σ. $\square$

Regular expressions

Definition

Value $v(x)$ *of regular expression* x is defined thusly:

1. $v(\emptyset) = \emptyset, v(\varepsilon) = \{\varepsilon\}, v(a) = \{a\}$,
2. $v(x + y) = v(x) \cup v(y)$,
   $v(x.y) = v(x).v(y)$,
   $v(x^*) = (v(x))^*$. $\square$

Regular expressions — Axioms

$A_1 : x + (y + z) = (x + y) + z$ (associativity of union),
$A_2 : x + y = y + x$ (commutativity of union),
$A_3 : x + \emptyset = x$ ($\emptyset$ is the identity element of union),
$A_4 : x + x = x$ (idempotence of union),
$A_5 : x.(y.z) = (x.y).z$ (associativity of concatenation),
$A_6 : \varepsilon x = x\varepsilon = x$ (ε is the identity element of conc.),
$A_7 : \emptyset x = x\emptyset = \emptyset$ ($\emptyset$ is the identity element of conc.),
$A_8 : x.(y + z) = x.y + x.z$ (left distributivity),
$A_9 : (x + y).z = x.z + y.z$ (right distributivity),
$A_{10} : x^* = \varepsilon + x^*x$
$A_{11} : x^* = (\varepsilon + x)^*$
$A_{12} : x = x\alpha + \beta \Rightarrow x = \beta\alpha^*$ (solution of left regular equation),
$A_{13} : x = \alpha x + \beta \Rightarrow x = \alpha^*\beta$ (solution of right regular equation).

Regular equations

Example

$L = \{$ even number of ones followed by suffix $010\,\}$.

$$(R)\;\; x = 11x + 010.$$

solution: $x = (11)^*010$

$(11)^*010 = 11(11)^*010 + 010$ $\qquad$ $(x = \alpha x + \beta \Rightarrow x = \alpha^*\beta)$

$(11)^*010 = (11(11)^* + \varepsilon)010$ $\qquad$ $(xy + y = (x + \varepsilon)y)$

$(11)^*010 = (11)^*010$ $\qquad$ $(xx^* + \varepsilon = x^*)$

Regular equations

Definition

Standard system of regular equations has this form:
$X_i = \alpha_{i0} + \alpha_{i1}X_1 + \alpha_{i2}X_2 + ... + \alpha_{in}X_n, 1 \le i \le n$, where $X_1, X_2, ..., X_n$ are variables and α_{ij} are regular expressions over alphabet Σ that does not contain $X_1, X_2, ..., X_n$. $\square$

Regular equations

Example

$$A = 1A + 1B$$
$$B = 0A + 0B + 0.$$

$A = 1^*1B$

$B = 01^*1B + 0B + 0.$

$B = (01^*1 + 0)B + 0$

$B = (01^*1 + 0)^*0 = (0(1^*1 + \varepsilon))^*0 = (01^*)^*0.$

Solution:

$$A = 1^*1(01^*)^*0$$
$$B = (01^*)^*0.$$

Derivatives of regular expressions

Definition

Derivative $\frac{d}{dx}$ of regular expression E with respect to string $x \in \Sigma^*$:
$\frac{dE}{dx} = E', v(E') = \{y : xy \in v(E)\}$

Derivatives of regular expressions

Definition

Derivative $\frac{d}{dx}$ of regular expression E with respect to string $x \in \Sigma^*$:

1. $\frac{dE}{d\varepsilon} = E$
2. for $a \in \Sigma$ it holds that:
   $\frac{d\varepsilon}{da} = \emptyset \quad \frac{d\emptyset}{da} = \emptyset$
   $$\frac{db}{da} = \begin{cases} \emptyset, \text{ if } a \neq b \\ \varepsilon, \text{ if } a = b \end{cases}$$
   $\frac{d(F+E)}{da} = \frac{dF}{da} + \frac{dE}{da}$
   $\frac{d(FE)}{da} = \frac{dF}{da}E + \{\frac{dE}{da} : \varepsilon \in v(F)\}$
   $\frac{d(E^*)}{da} = \frac{dE}{da}.E^*$
3. For $x = a_1a_2...a_n, a_i \in \Sigma$ it holds that
   $\frac{dE}{dx} = \frac{d}{da_n}(\frac{d}{da_{n-1}}(\dots\frac{d}{da_2}(\frac{dE}{da_1})\dots))$ $\square$

Derivatives of regular expressions

Example

Regular expression $y = (0+1)^*.1$.

$$\begin{aligned}
\frac{dy}{d\varepsilon} &= (0+1)^*.1 \\
\frac{dy}{d1} &= \frac{d(0+1)^*}{d1}.1 + \frac{d1}{d1} \\
&= \frac{d(0+1)}{d1}.(0+1)^*.1 + \varepsilon \\
&= (\frac{d0}{d1} + \frac{d1}{d1})(0+1)^*.1 + \varepsilon \\
&= (\emptyset + \varepsilon).(0+1)^*.1 + \varepsilon \\
&= (0+1)^*.1 + \varepsilon \\
\frac{dy}{d0} &= \frac{d(0+1)^*}{d0}.1 + \frac{d1}{d0} \\
&= \frac{d(0+1)}{d0}.(0+1)^*.1 + \emptyset \\
&= (\varepsilon + \emptyset).(0+1)^*.1 + \emptyset \\
&= (0+1)^*.1
\end{aligned}$$

Integral of regular expressions

Definition

Integral of regular expression E in respect to string $x \in \Sigma^*$ is defined thusly:

$$v(\int E\ dx) = \{xy : y \in h(E)\}.$$

For integration of regular expressions following rules apply:

1. $\int E d\varepsilon = E$
2. for $a \in \Sigma$ it holds that:
   $\int \varepsilon\ da = a,$
   $\int \emptyset\ da = \emptyset,$
   $\int b\ da = ab,$
   $\int (F+E)\ da = \int F\ da + \int E\ da,$
   $\int (F.E)\ da = aFE,$
   $\int E^*\ da = aE^*.$
3. for $x = a_1a_2\cdots a_n \in \Sigma^*$ it holds that:
   $\int E\ dx = \int \cdots [\int(\int E\ da_n)\ da_{n-1}] \cdots da_1.$

Integral of regular expressions

$$\frac{d}{dx}\int E\ dx = E,$$
$$\int \frac{dE}{dx}\ dx = E.$$

Integral with an integration constant Z:

$$\int E\ dx = xE + Z$$
$$\frac{dZ}{dx} = \emptyset$$

Integral of regular expressions

Example
Regular expression $(0+1)^*.1$.

$$\int (0+1)^*.1 \; d1 = 1.(0+1)^*.1 + Z_1,$$
$$\int (0+1)^*.1 \; d0 = 0.(0+1)^*.1 + Z_0.$$

Conversions of regular expressions

Definition

(a) regular expressions x, y are called *identical* (denoted by $x \equiv y$) if x a y are two exactly same strings of symbols.

(b) regular expressions x, y are called *equivalent* (denoted by $x = y$) if they have the same value, $v(x) = v(y)$, that is, the regular sets described by these equations are identical.

(c) regular expressions x, y are called *similar* if they can be converted into each other using the following identities:
$x + x = x$
$x + y = y + x$
$(x + y) + z = x + (y + z)$
$x + \emptyset = x$
$x.\emptyset = \emptyset.x = \emptyset$
$x.\varepsilon = \varepsilon.x = x$

Conversions of regular expressions

Theorems in the axiomatic thoery of regular expressions

$T_1 :\quad \emptyset^* = \varepsilon$
$T_2 :\quad x^* + x = x^*$
$T_3 :\quad (x^*)^* = x^*$
$T_4 :\quad (x + y)^* = (x^*y^*)^*$
$T_5 :\quad x^*y = y + x^*xy$
$T_6 :\quad x^*y = y + xx^*y$
$T_7 :\quad x^*y = (x^n)^*.(y + xy + x^2y + \ldots + x^{n-1}y)$
$T_8 :\quad (\varepsilon \in v(x)) \Rightarrow (xx^* = x^*)$
$T_9 :\quad (xy)^*x = x(yx)^*$
$T_{10} :\quad (x + y)^* = (x^* + y^*)^*$

Conversions of regular expressions

Example
Regular expressions
$x = \varepsilon + 1^*(011)^*(1^*(011)^*)^*$
$y = (1 + 011)^*$
Are they equivalent?

$$\begin{aligned} x &= \varepsilon + 1^*(011)^*(1^*(011)^*)^* \\ &= (1^*(011)^*)^* \\ &= (1 + 011)^* \\ &= y. \end{aligned}$$

$$\varepsilon + xx^* = x^*$$
$$(x^*y^*)^* = (x + y)^*$$

Conversions of regular expressions

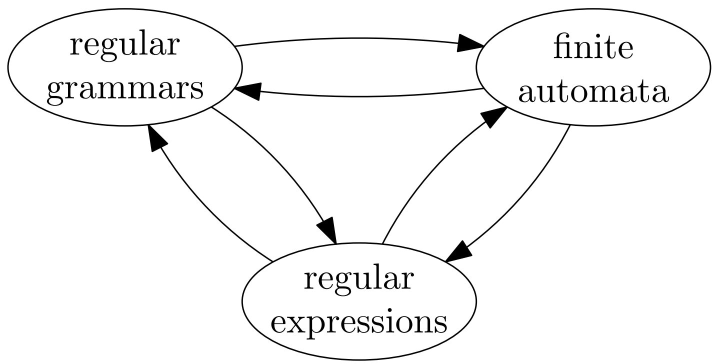
Example

Is the expression $(\varepsilon + \emptyset)(0+1)^*1 + (0+1)^*\emptyset$ similar to expression $(0+1)^*1$?

$$
\begin{aligned}
&(\varepsilon + \emptyset)(0+1)^*1 + (0+1)^*\emptyset = && x.\emptyset = \emptyset\\
&= (\varepsilon + \emptyset)(0+1)^*1 + \emptyset = && x + \emptyset = x\\
&= (\varepsilon + \emptyset)(0+1)^*1 = && x + \emptyset = x\\
&= \varepsilon(0+1)^*1 = && \varepsilon.x = x\\
&= (0+1)^*1. && \square
\end{aligned}
$$

Relations between formal systems of RE

Relations between formal systems for description of RE



Relationship between RG and FA

Algorithm Construction of NFA for a given right regular grammar

Input: Right regular grammar $G = (N, T, P, S)$.

Output: NFA $M = (Q, T, \delta, q_0, F)$ such that $L(G) = L(M)$.

Method:

1. Set of input symbols of automaton M is equal to T.
2. Set of states $Q = N \cup \{A\}, A \notin N$.
3. Mapping δ:
   if $B \to aC \in P$, then $\delta(B, a)$ contains C,
   if $B \to a \in P$, then $\delta(B, a)$ contains A.
4. $q_0 = S$.
5. $F = \{S, A\}$, if $S \to \varepsilon \in P$,
   $F = \{A\}$, if $S \to \varepsilon \notin P$.

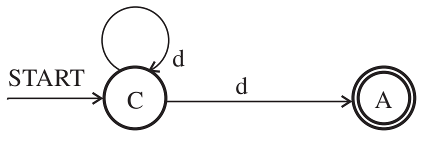
Relationship between RG and FA

Example

$G = (\{C\}, \{d\}, \{C \to d \mid dC\}, C)$.

$M = (\{C, A\}, \{d\}, \delta, C, \{A\})$, where δ:

δ	d
C	$\{C, A\}$
A	

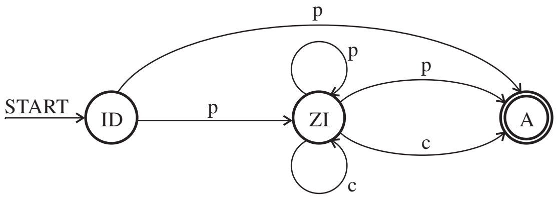


Relationship between RG and FA

Example

$G = (\{ID, ZI\}, \{p, c\}, P, ID)$, where P contains rules:
$ID \rightarrow p\,ZI \mid p$
$ZI \rightarrow p\,ZI \mid c\,ZI \mid p \mid c$.
(Grammar generates identifiers according to the usual definition (p – alphabet letter, c – digit).)

$M = (\{ID, ZI, A\}, \{p, c\}, \delta, ID, \{A\})$, kde δ



δ	p	c
ID	$\{ZI, A\}$	
ZI	$\{ZI, A\}$	$\{ZI, A\}$
A		

Relationship between RG and FA

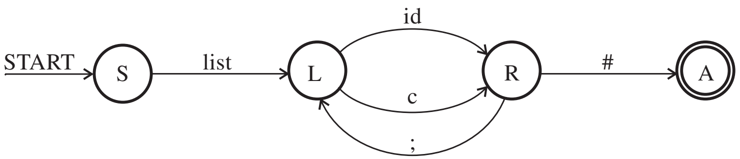
Example

Language describing strings of the form: $list\ \ id; c; id; id; \ldots; c; c; id\#$

$G = (\{S, L, R\}, \{list, id, c, ;, \#\}, P, S)$, where P:
$S \rightarrow list\ L,\ L \rightarrow id\ R \mid c\ R,\ R \rightarrow ;\ L \mid \#$

$M = (\{S, L, R, A\}, \{list, id, c, \#, ;\}, \delta, S, \{A\})$, where δ:

δ	$list$	id	c	$;$	$\#$
S	L				
L		R	R		
R				L	A
A					



Relationship between RG and FA

Algorithm Construction of NFA for left regular grammar.
Input: Left regular grammar $G = (N, T, P, S)$.
Output: NKA $M = (Q, T, \delta, q_0, F)$ such that $L(G) = L(M)$.
Method:

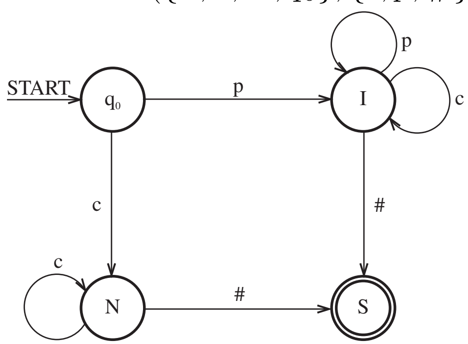
1. Set of input symbols of automaton M is equal to T.
2. Set of states $Q = N \cup \{q_0\}$.
3. Mapping δ:
   If $A \rightarrow Ba \in P$, then $\delta(B, a)$ contains A,
   if $A \rightarrow a\ \in P$, then $\delta(q_0, a)$ contains A.
4. q_0 is the initial state of autmoaton M.
5. If $S \rightarrow \varepsilon \in P$, then $F = \{S, q_0\}$,
   in the converse case $F = \{S\}$.

Relationship between RG and FA

Example

left regular grammar $G = (\{S, I, N\}, \{c, p, \#\}, P, S)$, where P:
$S \rightarrow I\# \mid N\#, I \rightarrow p \mid Ip \mid Ic, N \rightarrow c \mid Nc.$

NFA $M = (\{S, I, N, q_0\}, \{c, p, \#\}, \delta, q_0, \{S\})$, where δ:



	c	p	$\#$
S			
I	I	I	S
N	N		S
q_0	N	I	

Relationship between RG and FA

Algorithm Construction of right regular grammar for a given NFA

Input: NFA $M = (Q, T, \delta, q_0, F)$.

Output: Right regular grammar $G = (N, T, P, S)$, $L(M) = L(G)$.

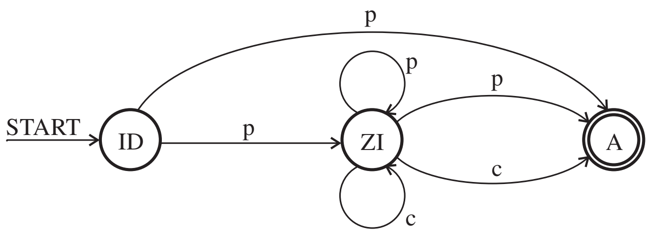
Method:

1. $N = Q$.
2. P is constructed thusly:
   (a) If $\delta(B, a)$ contains C, then $B \rightarrow aC$ is in P.
   (b) If $\delta(B, a)$ contains C and $C \in F$, then $B \rightarrow a$ is in P.
   (c) Symbol $S = q_0$.
   (d) If $q_0 \in F$ and S is not on the right-hand side of any rule, then $S \rightarrow \varepsilon$ is in P and the initial symbol is S.
   (e) If $q_0 \in F$ and S is on the right-hand side, then we add rules of form $S' \rightarrow \alpha$ into P, where α are the right-hand sides of the rules in the form $S \rightarrow \alpha$, S' is the initial symbol and rule $S' \rightarrow \varepsilon$ is added to P.

Relationship between RG and FA

Example

We create a right regular grammar for NFA:



Resulting regular grammar $G = (\{\mathit{ID}, \mathit{ZI}, \mathit{A}\}, \{p, c\}, P, \mathit{ID})$, where P:

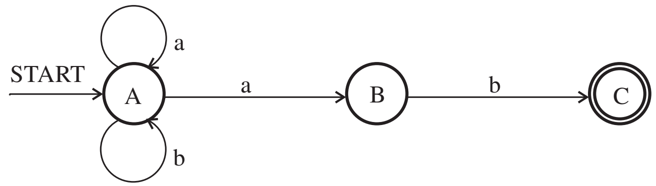
$\mathit{ID} \rightarrow p\mathit{ZI} \mid p \mid p\mathit{A}$

$\mathit{ZI} \rightarrow p\mathit{ZI} \mid p\mathit{A} \mid c\mathit{ZI} \mid c\mathit{A} \mid p \mid c$.

Relationship between RG and FA

Example

We create a right regular grammar for NFA:



$G = (\{A, B, C\}, \{a, b\}, P, A)$, kde P:

$A \rightarrow aA \mid aB \mid bA$

$B \rightarrow bC \mid b$.

Finite transducers

Theorem

If we have $RTG = (N, T, D, R, S)$, then there exists $FT = (Q, T, D, \delta, q_0, F)$ such that $Z(RTG) = Z(FT)$.

Proof: For a given $RTG = (N, T, D, R, S)$ we create $FT = (Q, T, D, \delta, q_0, F)$, where $Q = N \cup \{X\}$, $X \notin N$.
Mapping δ is defined thuslY ($y \in D^*, B, C \in N$):

$(C, y) \in \delta(B, a)$ if $B \rightarrow ayC \in R, \forall a \in T$,
$(X, y) \in \delta(B, a)$ if $B \rightarrow ay \in R, \forall a \in T$,

$q_0 = S$
$F = \{S, X\}$ if $S \rightarrow \varepsilon \in R$
$F = \{X\}$ if $S \rightarrow \varepsilon \notin R$.

Proof that $Z(RTG) = Z(FT)$: by induction by the length of derivative in RTG and the length of sequence of transitions in FT. $\square$

Finite transducers

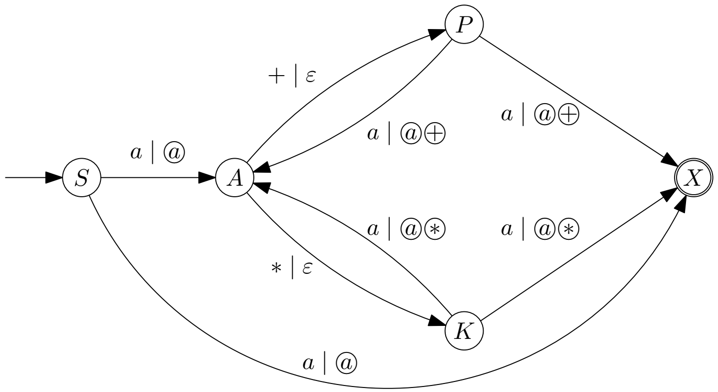
Example

$RTG = (\{S, A, P, K\}, \{a, +, *\}, \{@, \oplus, \circledast\}, R, S)$, where R:

$S \to a@A \qquad A \to *K \qquad S \to a@ \qquad A \to +P$

$K \to a@\circledast A \qquad P \to a@\oplus A \qquad K \to a@\circledast \qquad P \to a@\oplus$

$FT = (\{S, X, A, P, K\}, \{a, +, *\}, \{@\oplus\circledast\}, \delta, S, \{X\})$



Finite transducers

Example

$FT = (\{N, J, D\}, \{0, 1\}, \{\textcircled{0}, \textcircled{1}\}, \delta, N, \{N\})$, where δ:

$\delta(N, 0) = \{(N, \textcircled{0})\} \qquad \delta(J, 1) = \{(N, \textcircled{1})\}$

$\delta(N, 1) = \{(J, \textcircled{0})\} \qquad \delta(D, 0) = \{(J, \textcircled{1})\}$

$\delta(J, 0) = \{(D, \textcircled{0})\} \qquad \delta(D, 1) = \{(D, \textcircled{1})\}$

$RTG = (\{S, N, J, D\}, \{0, 1\}, \{\textcircled{0}, \textcircled{1}\}, R, S)$, where R:

$S \to \varepsilon \qquad N \to 0\textcircled{0}N \qquad J \to 0\textcircled{0}D \qquad D \to 0\textcircled{1}J$

$S \to 0\textcircled{0}N \qquad N \to 1\textcircled{0}J \qquad J \to 1\textcircled{1}N \qquad D \to 1\textcircled{1}D$

$S \to 1\textcircled{0}J \qquad N \to 0\textcircled{0} \qquad J \to 1\textcircled{1} \qquad S \to 0\textcircled{0}$

Finite transducers

Theorem

If FT is a transducer, then there exists a regular translation grammar RTG such that $Z(FT) = Z(RTG)$.

Proof: For a given $FT = (Q, T, D, \delta, q_0, F)$ we create $RTG = (N \cup \{S\}, T, D, R, S)$, where $S \notin N$, thusly:

1. $N = Q$.
2. We create a set of rules R', for all $\forall a \in T$ and $y \in D^*$:
   $B \to ayC$, when $(C, y) \in \delta(B, a)$,
   $B \to ay$, when $(C, y) \in \delta(B, a)$ and $C \in F$,
   $S \to \varepsilon$, when $q_0 \in F$.
3. $R = R' \cup \{S \to x : q_0 \to x \in R'\}$.

Proof of equivalence $Z(RTG) = Z(FT)$: by induction on the length of derivative in RTG and on the length of sequence of transitions in FT.$\square$