Regular Expressions

Deepak D’Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

11 August 2011
Outline

1. Regular Expressions
2. Kleene’s Theorem
3. Equation-based alternate construction
Examples of Regular Expressions

Expressions built from $a$, $b$, $\epsilon$, using operators $+$, $\cdot$, and $\ast$.

- $(a^{\ast} + b^{\ast}) \cdot c$
  "Strings of only $a$'s or only $b$'s, followed by a $c$.”

- $(a + b)^{\ast}abb(a + b)^{\ast}$
  “contains $abb$ as a subword.”

- $(a + b)^{\ast}b(a + b)(a + b)$
  “3rd last letter is a $b$.”

- $(b^{\ast}ab^{\ast}a)^{\ast}b^{\ast}$
Examples of Regular Expressions

Expressions built from $a$, $b$, $\epsilon$, using operators $+$, $\cdot$, and $\ast$.

- $(a^* + b^*) \cdot c$
  "Strings of only $a$'s or only $b$'s, followed by a $c$."

- $(a + b)^* a b b (a + b)^*$
  "contains $a b b$ as a subword."

- $(a + b)^* b (a + b) (a + b)$
  "3rd last letter is a $b$."

- $(b^* a b^* a)^* b^*$
  "Even number of $a$'s."
Examples of Regular Expressions

Expressions built from $a$, $b$, $\epsilon$, using operators $+$, $\cdot$, and $\ast$.

- $(a^* + b^*) \cdot c$
  “Strings of only $a$’s or only $b$’s, followed by a $c$.”

- $(a + b)^* abb(a + b)^*$
  “contains $abb$ as a subword.”

- $(a + b)^* b(a + b)(a + b)$
  “3rd last letter is a $b$.”

- $(b^* ab^* a)^* b^*$
  “Even number of $a$’s.”

- Ex. Give regexp for “Every 4-bit block of the form $w[4i, 4i + 1, 4i + 2, 4i + 3]$ has even parity.”
Examples of Regular Expressions

Expressions built from $a$, $b$, $\epsilon$, using operators $+$, $\cdot$, and $\ast$.

- $(a^* + b^*) \cdot c$
  “Strings of only $a$’s or only $b$’s, followed by a $c$."

- $(a + b)^* abb(a + b)^*$
  “contains $abb$ as a subword.”

- $(a + b)^* b(a + b)(a + b)$
  “3rd last letter is a $b$."

- $(b^* ab^* a)^* b^*$
  “Even number of $a$’s.”

- Ex. Give regexp for “Every 4-bit block of the form $w[4i, 4i + 1, 4i + 2, 4i + 3]$ has even parity.”
  $(0000 + 0011 + \cdots + 1111)^*(\epsilon + 0 + 1 + \cdots + 111)$
Formal definitions

- Syntax of regular expressions over an alphabet $A$:

\[ r ::= \emptyset | a | r + r | r \cdot r | r^* \]

where $a \in A$.
- Semantics: associate a language $L(r) \subseteq A^*$ with regexp $r$.

\[
L(\emptyset) = \{\}\nL(a) = \{a\}
L(r + r') = L(r) \cup L(r')
L(r \cdot r') = L(r) \cdot L(r')
L(r^*) = L(r)^*.
\]
Formal definitions

- Syntax of regular expressions over an alphabet $A$:
  \[ r ::= \emptyset \mid a \mid r + r \mid r \cdot r \mid r^* \]
  where $a \in A$.

- Semantics: associate a language $L(r) \subseteq A^*$ with regexp $r$.

  \[
  \begin{align*}
  L(\emptyset) &= \{\} \\
  L(a) &= \{a\} \\
  L(r + r') &= L(r) \cup L(r') \\
  L(r \cdot r') &= L(r) \cdot L(r') \\
  L(r^*) &= L(r)^*.
  \end{align*}
  \]

- Question: Do we need $\epsilon$ in syntax?
Formal definitions

- Syntax of regular expressions over an alphabet $A$:

$$r ::= \emptyset \mid a \mid r + r \mid r \cdot r \mid r^*$$

where $a \in A$.

- Semantics: associate a language $L(r) \subseteq A^*$ with regexp $r$.

$$L(\emptyset) = \{\}$$
$$L(a) = \{a\}$$
$$L(r + r') = L(r) \cup L(r')$$
$$L(r \cdot r') = L(r) \cdot L(r')$$
$$L(r^*) = L(r)^*.$$

- Question: Do we need $\epsilon$ in syntax?
  No. $\epsilon \equiv \emptyset^*$. 
Example: Semantics of regexp

\[(a^* + b^*) \cdot c\]
Example: Semantics of regexp

\((a^* + b^*) \cdot c\)
Example: Semantics of regexp

\((a^* + b^*) \cdot c\)
Example: Semantics of regexp

$$(a^* + b^*) \cdot c$$
Example: Semantics of regexp

\((a^* + b^*) \cdot c\)
Kleene’s Theorem: $RE = DFA$

Class of languages defined by regular expressions coincides with regular languages.

Proof

- $RE \rightarrow DFA$: Use closure properties of regular languages.
- $DFA \rightarrow RE$: 
DFA $\rightarrow$ RE: Kleene’s construction

- Let $\mathcal{A} = (Q, s, \delta, F)$ be given DFA.
- Define $L_{pq} = \{w \in A^* \mid \hat{\delta}(p, w) = q\}$.
- Then $L(\mathcal{A}) = \bigcup_{f \in F} L_{sf}$.
- For $X \subseteq Q$, define $L^X_{pq} = \{w \in A^* \mid \hat{\delta}(p, w) = q \text{ via a path that stays in } X \text{ except for first and last states}\}$
- Then $L(\mathcal{A}) = \bigcup_{f \in F} L^Q_{sf}$. 

\[
\begin{align*}
\text{Let } \mathcal{A} = (Q, s, \delta, F) \text{ be given DFA.} \\
\text{Define } L_{pq} = \{w \in A^* \mid \hat{\delta}(p, w) = q\}. \\
\text{Then } L(\mathcal{A}) = \bigcup_{f \in F} L_{sf}. \\
\text{For } X \subseteq Q, \text{ define } L^X_{pq} = \{w \in A^* \mid \hat{\delta}(p, w) = q \text{ via a path that stays in } X \text{ except for first and last states}\} \\
\text{Then } L(\mathcal{A}) = \bigcup_{f \in F} L^Q_{sf}. 
\end{align*}
\]
DFA $\rightarrow$ RE: Kleene’s construction

Advantage:

$$L_{pq}^{X \cup \{r\}} = L_{pq}^X + L_{pr}^X \cdot (L_{rr}^X)^* \cdot L_{rq}^X.$$
DFA $\rightarrow$ RE: Kleene’s construction (2)

Method:

- Begin with $L_{sf}^Q$ for each $f \in F$.
- Simplify by using terms with strictly smaller $X$’s:

$$L_{pq}^{X\cup\{r\}} = L_{pq}^{X} + L_{pr}^{X} \cdot (L_{rr}^{X})^* \cdot L_{rq}^{X}.$$

- For base terms, observe that

$$L_{pq}^\emptyset = \begin{cases} 
\{ a \mid \delta(p, a) = q \} & \text{if } p \neq q \\
\{ a \mid \delta(p, a) = q \} \cup \{ \epsilon \} & \text{if } p = q.
\end{cases}$$

- Exercise: convert NFA/DFA’s below to RE’s:
DFA $\rightarrow$ RE: Kleene’s construction (2)

Method:

- Begin with $L_s^f$ for each $f \in F$.
- Simplify by using terms with strictly smaller $X$’s:

$$L_{pq}^{X \cup \{r\}} = L_{pq}^X + L_{pr}^X \cdot (L_{rr}^X)^* \cdot L_{rq}^X.$$ 

- For base terms, observe that

$$L\{\} = \begin{cases} \{a \mid \delta(p, a) = q\} & \text{if } p \neq q \\ \{a \mid \delta(p, a) = q\} \cup \{\epsilon\} & \text{if } p = q. \end{cases}$$

- Exercise: convert NFA/DFA’s below to RE’s:
DFA $\rightarrow$ RE using system of equations

- **Aim:** to construct a regexp for

$$L_q = \{w \in A^* \mid \widehat{\delta}(q, w) \in F\}.$$ 

- Note that $L(A) = L_s$.

- **Example:**

Set up equations to capture $L_q$'s:

$$x_e = b \cdot x_e + a \cdot x_o$$
$$x_o = a \cdot x_e + b \cdot x_o + \epsilon.$$ 

- Solution is a RE for each $x$, such that languages denoted by LHS and RHS RE's coincide.
Aim: to construct a regexp for

\[ L_q = \{ w \in A^* \mid \hat{\delta}(q, w) \in F \} \].

Note that \( L(A) = L_s \).

Example:

Set up equations to capture \( L_q \)'s:

\[
\begin{align*}
x_e &= b \cdot x_e + a \cdot x_o \\
x_o &= a \cdot x_e + b \cdot x_o + \epsilon.
\end{align*}
\]

Solution is a RE for each \( x \), such that languages denoted by LHS and RHS RE’s coincide.
Solving a system of equations

- $L_q$’s are a solution to the system of equations
- In general there could be many solutions to equations. Consider $x = A^*x$.
- In this case, $L_q$’s can be seen to the least solution to the equations.
Least solution to the system of equations

- Equations can be viewed as:

\[
\begin{bmatrix}
  x_e & x_o
\end{bmatrix}
= \begin{bmatrix}
  b & a \\
  a & b
\end{bmatrix}
\begin{bmatrix}
  x_e \\
  x_o
\end{bmatrix} + \begin{bmatrix}
  \epsilon & \emptyset
\end{bmatrix}
\]

- System of equations have the general form:

\[ X = AX^T + B. \]

- Claim: \( A^*B \) is the least solution to the equations above. [See Kozen, Supplementary Lecture A].

- Definition of \( A^* \):

\[
\begin{bmatrix}
  a & b \\
  c & d
\end{bmatrix}^*
= \begin{bmatrix}
  (a + bd^*c)^* & (a + bd^*c)^*bd^* \\
  (d + ca^*b)^*ca^* & (d + ca^*b)^*
\end{bmatrix}
\]