Regular Expressions

Method to represent strings in a language

+ union (or)
  ○ concatenation (AND) (can omit)
  * star-closure (repeat 0 or more times)

Example:

(a + b)* ◯ a ◯ (a + b)*

Example:

(aa)*

Definition: Given $\Sigma$,

1. $\emptyset, \lambda, a \in \Sigma$ are R.E.
2. If $r$ and $s$ are R.E. then
   - $r + s$ is R.E.
   - $rs$ is R.E.
   - $(r)$ is a R.E.
   - $r^*$ is R.E.
3. $r$ is a R.E. iff it can be derived from (1) with a finite number of applications of (2).

Definition: $L(r) =$ language denoted by R.E. $r$.

1. $\emptyset, \{\lambda\}, \{a\}$ are L denoted by a R.E.
2. if $r$ and $s$ are R.E. then
   - (a) $L(r + s) = L(r) \cup L(s)$
   - (b) $L(rs) = L(r) \circ L(s)$
   - (c) $L(( r )) = L(r)$
   - (d) $L(( r )^*) = ( L( r)^* )$

Precedence Rules

* highest
  ○
  +

Example:

$ab^* + c =$
Examples:

1. $\Sigma = \{a, b\}, \{w \in \Sigma^* \mid w \text{ has an odd number of } a\text{'s followed by an even number of } b\text{'s}\}$.

2. $\Sigma = \{a, b\}, \{w \in \Sigma^* \mid w \text{ has no more than } 3 \text{ } a\text{'s and must end in } ab\}$.

3. Regular expression for positive and negative integers

Section 3.2 Equivalence of DFA and R.E.

**Theorem** Let $r$ be a R.E. Then $\exists$ NFA $M$ s.t. $L(M) = L(r)$.

- **Proof:**
  1. Suppose $r$ and $s$ are R.E.
  2. $r+s$
  3. $r*\quad a$

**Example**

$ab^* + a$

**Theorem** Let $L$ be regular. Then $\exists$ R.E. $r$ s.t. $L=L(r)$.

Proof Idea: remove states sucessively, generating equivalent generalized transition graphs (GTG) until only two states are left (one initial state and one final state).

- **Proof:**
  1. Assume $M$ has one final state and $q_0 \notin F$
  2. Convert to a generalized transition graph (GTG), all possible edges are present. If no edge, label with
  3. Let $r_{ij}$ stand for label of the edge from $q_i$ to $q_j$
  4. If the GTG has only two states, then it has the following form:
In this case the regular expression is:

\[ r = (r_{ii}^*r_{ij}r_{jj}^*r_{ji})^*r_{ii}^*r_{ij}r_{jj}^* \]

4. If the GTG has three states then it must have the following form:

In this case, make the following replacements:

<table>
<thead>
<tr>
<th>REPLACE</th>
<th>WITH</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_{ii} )</td>
<td>( r_{ii} + r_{ik}r_{kk}r_{ki} )</td>
</tr>
<tr>
<td>( r_{jj} )</td>
<td>( r_{jj} + r_{jk}r_{kk}r_{kj} )</td>
</tr>
<tr>
<td>( r_{ij} )</td>
<td>( r_{ij} + r_{ik}r_{kk}^*r_{kj} )</td>
</tr>
<tr>
<td>( r_{ji} )</td>
<td>( r_{ji} + r_{jk}r_{kk}^*r_{ki} )</td>
</tr>
</tbody>
</table>

After these replacements, remove state \( q_k \) and its edges.

5. If the GTG has four or more states, pick a state \( q_k \) to be removed (not initial or final state).

For all \( o \neq k, p \neq k \) use the rule

\( r_{op} \) replaced with \( r_{op} + r_{ok}r_{kk}r_{kp} \)

with different values of \( o \) and \( p \).

When done, remove \( q_k \) and all its edges. Continue eliminating states until only two states are left. Finish with step 3.

6. In each step, simplify the regular expressions \( r \) and \( s \) with:
\[ r + r = r \\
\frac{s + r^*s}{r} = r \\
\frac{r\emptyset}{r} = \emptyset \\
\frac{\emptyset^*}{r} = r \\
\frac{(\lambda + r)^*}{(\lambda + r)r^*}{r} = \] and similar rules.

Example:

![Diagram](image)

Section 3.3

Grammar \( G=(V,T,S,P) \)
- \( V \) variables (nonterminals)
- \( T \) terminals
- \( S \) start symbol
- \( P \) productions

Right-linear grammar:

all productions of form
- \( A \rightarrow xB \)
- \( A \rightarrow x \)
where \( A,B \in V \), \( x \in T^* \)

Left-linear grammar:

all productions of form
- \( A \rightarrow Bx \)
- \( A \rightarrow x \)
where \( A,B \in V \), \( x \in T^* \)

Definition:

A regular grammar is a right-linear or left-linear grammar.

Example 1:
G=({S}, {a,b}, S, P), P=
S → abS
S → λ
S → Sab

Example 2:

G=({S,B}, {a,b}, S, P), P=
S → aB | BS | λ
B → aS | bB

Theorem: L is a regular language iff ∃ regular grammar G s.t. L=L(G).

Outline of proof:

(⇐) Given a regular grammar G
Construct NFA M
Show L(G)=L(M)

(⇒) Given a regular language
∃ DFA M s.t. L=L(M)
Construct reg. grammar G
Show L(G) = L(M)

Proof of Theorem:

(⇐) Given a regular grammar G
G=({V,T,S}, P)
V={V_0, V_1, \ldots, V_y}
T={v_0, v_1, \ldots, v_z}
S=V_0
Assume G is right-linear
(see book for left-linear case).
Construct NFA M s.t. L(G)=L(M)
If w∈L(G), w=v_1v_2 \ldots v_k

M=(V∪{v_f}, T, δ, V_0, {v_f})
V_0 is the start (initial) state
For each production, V_i → aV_j,
For each production, $V_i \rightarrow a$,

Show $L(G) = L(M)$
Thus, given R.G. G,
$L(G)$ is regular

$(\Rightarrow)$ Given a regular language $L$
$\exists$ DFA $M$ s.t. $L = L(M)$
$M = (Q, \Sigma, \delta, q_0, F)$
$Q = \{q_0, q_1, \ldots, q_n\}$
$\Sigma = \{a_1, a_2, \ldots, a_m\}$

Construct R.G. G s.t. $L(G) = L(M)$
$G = (Q, \Sigma, q_0, P)$
if $\delta(q_i, a_j) = q_k$ then

if $q_k \in F$ then

Show $w \in L(M) \iff w \in L(G)$
Thus, $L(G) = L(M)$.
QED.

Example

$G = (\{S, B\}, \{a, b\}, S, P), P =$
$S \rightarrow aB \mid bS \mid \lambda$
$B \rightarrow aS \mid bB$

Example:

![DFA Diagram](attachment:DFA.png)