Section: Regular Languages

Regular Expressions

Method to represent strings in a language

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

Example:

\[(a + b)^* \circ a \circ (a + b)^* = (a+b)^* a (a+b)^*\]

Example:

\[(aa)^* \quad \exists = \exists a^*\]

Strings with an even number of a's
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:

\[ ab^* + c = (a \cdot (b)^*) + c \]
Examples:

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

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

3. Regular expression for all integers (including negative) $\leq 3, \ 0, 1, 2, \ldots, 9$.

\[0 + (\pm \%)(1+2+3+\ldots+9)(0+1+2+\ldots+9)^*\]
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:

Suppose \( r \) and \( s \) are R.E.

1. \( r + s \)
2. \( r \circ s \)
3. \( r^* \)
Example

$ab^* + c$

Use JFLAP to convert the regular expression to an NFA
Convert this NFA to a DFA

Then convert to a min DFA
Theorem Let $L$ be regular. Then $\exists$ R.E. $r$ s.t. $L=L(r)$.

Proof Idea: remove states sucessively until two states left

\* Proof:

$L$ is regular

$\Rightarrow \exists$ DFA $M$ s.t. $L=L(M)$

1. Assume $M$ has one final state and $q_0 \not\in F$

2. Convert to a generalized transition graph (GTG), all possible edges are present. If no edge, label with $\emptyset$
Let $r_{ij}$ stand for label of the edge from $q_i$ to $q_j$
3. If the GTG has only two states, then it has the following form:

In this case the regular expression is:

\[ r = (r_{ii}r_{ij}r_{ji}r_{ji})^*r_{ii}r_{ij}r_{jj} \]
4. If the GTG has three states then it must have the following form:
<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>

**remove state** \( q_k \)
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:

\[
\begin{align*}
    r + r &= r \\
    s + r^* s &= r^* s \\
    r + \emptyset &= r \\
    r \emptyset &= \emptyset \\
    \emptyset^* &= \emptyset \\
    r \lambda &= r \\
    (\lambda + r)^* &= r^* \\
    (\lambda + r)r^* &= r^* \\
\end{align*}
\]

and similar rules.
Example:
Using JFLAP
convert this NFA to
regular expression

First complete all
the arcs, add
Emptyset to any arc
missing
If you remove state q1, here are the transition replacements that will keep the information of going through state q1

Here is the resulting NFA with regular expressions on the arcs

Here is the resulting regular expression

```
((aa*b)*(a+aa*b)b)*(aa*b)*(a+aa*b)
```
Example 2: 4 states!
Remove State q3

Results in

Now remove state q1
Here is the regular expression:

\[((a(a+aa)\ast b)\ast (a+aa(a+aa)\ast (b+ab))b)\ast (a+aa)\ast b)\ast (a+aa(a+aa)\ast (b+ab))\]
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:

\[G=(\{S\}, \{a,b\}, S, P), P=\]
\[S \rightarrow \text{ab}S\]
\[S \rightarrow \lambda\]
\[S \rightarrow \text{Sab}\]
Example 2:

\[ G = (\{S,B\}, \{a,b\}, S, P), \quad P = \]

\[ S \rightarrow aB \mid bS \mid \lambda \]

\[ B \rightarrow aS \mid bB \]

\[ S \rightarrow aB \rightarrow abB \rightarrow abaS \rightarrow aba \]

\[ L = \{ \text{even number of } a's \text{ over } b's \} \]

\[ \exists Z = \exists a, b Z \]

right-linear

Regular

even no b's
odd no a's
Theorem: \( L \) is a regular language iff \( \exists \) regular grammar \( G \) s.t. \( L=\operatorname{L}(G) \).

Outline of proof:

\( \iff \) Given a regular grammar \( G \)
\hspace{1cm} Construct NFA \( M \)
\hspace{1cm} Show \( \operatorname{L}(G)=\operatorname{L}(M) \)

\( \implies \) Given a regular language
\hspace{1cm} \( \exists \) DFA \( M \) s.t. \( \operatorname{L}=\operatorname{L}(M) \)
\hspace{1cm} Construct reg. grammar \( G \)
\hspace{1cm} Show \( \operatorname{L}(G) = \operatorname{L}(M) \)
Proof of Theorem:

\[ \iff \]

Given a regular grammar \( G \)

\[ G = (V, T, S, P) \]

\[ \begin{align*}
V &= \{ V_0, V_1, \ldots, V_y \} \\
T &= \{ v_0, v_1, \ldots, v_z \} \\
S &= V_0
\end{align*} \]

Assume \( G \) is right-linear

(see book for left-linear case).

Construct NFA \( M \) s.t. \( L(G) = L(M) \)

If \( w \in L(G) \), \( w = v_1 v_2 \ldots v_k \)
$M = (V \cup \{V_f\}, T, \delta, V_0, \{V_f\})$

$V_0$ is the start (initial) state

For each production, $V_i \rightarrow aV_j$,

For each production, $V_i \rightarrow a$,

Show $L(G) = L(M)$

Thus, given R.G. $G$, $L(G)$ is regular
(⇒⇒) Given a regular language L
∃ DFA M s.t. L=L(M)
M=(Q,Σ,δ,q₀, F)
Q={q₀, q₁, ..., qₙ}
Σ = {a₁, a₂, ..., aₘ}

Construct R.G. G s.t. L(G) = L(M)
G=(Q,Σ,q₀,P)
if δ(qᵢ,aⱼ)=qₖ then
qᵢ → aⱼ qₖ ∈ P

if qₖ ∈ F then
qₖ → ∅

Show w ∈ L(M) ⇐⇒ w ∈ L(G)
Thus, L(G)=L(M).

QED.
Example

\[ G = (\{S, B\}, \{a, b\}, S, P), \quad P = \]
\[ S \rightarrow aB \mid bS \mid \lambda \]
\[ B \rightarrow aS \mid bB \]
Example:

```
Construct Regular
q0 -> a q1
q1 -> a q0 | b q1 | \lambda
```