The string w cannot be accepted by M, because $\delta(q_0, w) \in F$ [as F does not belong to $(Q - F)$].

So, $w \in L$, i.e., $\Sigma * - L \neq L$. But $\Sigma * - L$ is accepted by M' which is an FA.

Therefore, $\Sigma^* - L$ is a regular set.

Theorem 5.6: REs are closed under intersection operation.

Proof: From D'Morgan's theorem, we know

$$L_1 \cap L_2 = (L_1^C \cup L_2^C)^C.$$

We know that if L_1 and L_2 are regular, then L_1^C and L_2^C are also regular.

As L_1^C and L_2^C are regular, $L_1^C \cup L_2^C$ is also regular (RE is closed under union operation).

As $L_1^C \cup L_2^C$ is regular, so complement $(L_1^C \cup L_2^C)C$ is also regular.
So,$L_1 \cap L_2$ is also regular, i.e., the regular sets are closed under intersection.

Theorem 5.7: Two DFA are closed under cross product.

Proof: Let $D_1 = \{Q_1, \Sigma, \delta_1, q_01, F_1\}$ and $D_2 = \{Q_2, \Sigma, \delta_2, q_02, F_2\}$ are two DFA accepting two RE L_1 and L_2 respectively. Let us construct a new FA, D as follows.

$$D = \{Q, \Sigma, \delta, q_0, F\}$$

where

$$Q = Q_1 \times Q_2$$

$\delta((S_1, S_2), i/p) = (\delta_1(S_1, i/p), \delta_1(S_2, i/p))$ for all $S_1 \in Q_1, S_2 \in Q_2$ and $i/p \in \Sigma$.

$$q_0 = (q_01, q_02)$$
$$F = F_1 \times F_2$$

Clearly D is a DFA. Thus DFA are closed under cross product.

5.13 Decision Problems of Regular Expression

Decision problems are the problems which can be answered in 'yes' or 'no'. FA are one type of finite state machines which contain a finite number of memory elements. Thus, it can memorize only finite amount of information. FA can answer to those decision problems which require only a finite amount of memory.

The decision problems related to RE are

1. Whether a string x belongs to a regular expression R? ($x \in R$?)
2. Whether the language set of an FA M is empty? [$L(M) = \emptyset$?]
3. Whether the language set of an FA is finite? [L(M) is finite?]
4. Whether there exists any string accepted by two FA, M_1 and M_2?
5. Whether the language set of an FA M_1 is a subset of the language set of another FA M_2?[$L(M_1) \subseteq L(M_2)$?]
6. Whether two FA M_1 and M_2 accept the same language? [$L(M_1) = L(M_2)$]
7. Whether two REs R_1 and R_2 are from the same language set?
8. Whether an FA is the minimum state FA for a language L?

Proof

1. This problem is known as the membership problem. R can be converted to the equivalent FA M (see Section 5.5). x is applied on M. If M reaches its final state upon finishing x; $x \in R$ else $x \in R$.

270 | Introduction to Automata Theory, Formal Languages and Computation

2. An FA does not accept any string if it does not have any final state or if the final state is an inaccessible state. Let us calculate the set of state S_k reached from the beginning state q_0 upon applying a string of length k.

$$S_K = \begin{cases} q_0 & \text{if} \quad k = 0 \\ S_{K-1} \cup \{\delta(q, a) \text{ where } q \in S_{K-1} \text{ and } a \in \Sigma\} & \text{if} \quad k > 0 \end{cases}$$

If $k = 0$, it reaches the beginning state q_0. It reaches SK if it was in state $S_k - 1$ and 'a' as input is applied on $S_K - 1$.

Compare the set S_K.

a. for string length $k \geq 0$ whether a final state appears or

b. $S_K = S_K - 1$ (loop) for $k > 0$.

For the case (a), $L(M) \neq \emptyset$, and for case (b) $L(M) = \emptyset$.

3. In the pumping lemma (Section 4.11), it is discussed that for accepting a string of length$\geq n$ (the number of states of an FA), it has to traverse at least one loop. The language accepted by an FA is finite if it has length$< n$. Formulate an algorithm for testing as follows.

Give an input to M, the strings of length n or$> n$ in increasing order. If for a string 's' of length in between n and $< 2n$, it reaches M then L(M) is infinite; else L(M) is finite. Infinite means there exists x, y, z such that $s = xyz$, where $|xy| \leq n, |y| > 0$ and $xy^iz \in L$ for each $i \geq 0$ (pumping lemma). Here, y^i is the looping portion, which can generate an infinite number of strings.

4. For this decidable problem, construct an FA M accepting $L(M_1) \cap L(M_2)$. Then apply the decision problem 2 on M.

5. For this decidable problem, construct an FA M accepting $L(M_1) - L(M_2)$. Then, apply decision problem 2 on M. It is true if $L(M_1) - L(M_2) = \emptyset$. (If $L(M_1)$ is a subset of $L(M_2)$, $L(M_1) - L(M_2)$ will produce null.)

6. Two sets A and B are the same if $A \subseteq B$ and $B \subseteq A$. Construct an FA, M, accepting $L(M_1) - L(M_2)$ and M’ accepting $L(M_2) - L(M_1)$ (reducing it to problem v). Problem vi is decidable if $L(M_1) - L(M_2) = \emptyset$ and $L(M_2) - L(M_1) = \emptyset$.

7. There exists an algorithm to convert an RE to FA (see Section 5.5). Reduce this problem to problem (vi).

8. Minimize an FA, M, using 3.13 and generate M’. If the number of states of M and M’ are the same, then it is minimized; else it is not. Hence, decidable.

5.14 ‘Grep’ and Regular Expression

REs have been an integral part of Unix since the beginning. Ken Thompson used the RE in the early computer text editor ‘QED’ and the Unix editor ‘ed’ for searching text. ‘grep’ is a find string command in Unix. It is an acronym for ‘globally search a regular expression and print it’. The command searches through files and folders (directories in Unix) and checks which lines in those files match a given RE. ‘grep’ gives output the filenames and the line numbers or the actual lines that matched the RE. Let us start with some simple grep commands.

```
$ grep RE chapter5
```

This command searches the string ‘RE’ in the file ‘chapter5’.

```
$ grep ‘RE’ −i chapter5
```

This command searches the strings with case insensitive.

```
$ grep h??a chapter5
```

This command displays all strings of length 4 starting with 'h' and ending with 'a'.

If we enter into the internal operation of 'grep', we will see that we are searching for a string that belongs to an RE. Let us take the last grep which searches for four length strings starting with 'h' and ending with 'a'. For this case, the RE is h$(ch)^+(ch)^+a$, where 'ch' is any symbol that belongs to a file (Generally symbols available in the keyboard).

5.15 Applications of Regular Expression

RE is mainly used in lexical analysis of compiler design. In the programming language, we need to declare a variable or identifier. That identifier must start with a letter followed by any number of letters or digits. The structure of the identifier is represented by RE. The definition of an identifier in a programming language is

$$\text{letter} \rightarrow A|B|...|Z|a|b|...|z$$
$$\text{digit} \rightarrow 0|1|...|9$$
$$\text{id} \rightarrow \mathit{letter}(\mathit{letter}|\mathit{digit})*$$

The definition of an unsigned number in a programming language is

$$\text{digit} \rightarrow 0|1|...|9$$
$$\text{digits} \rightarrow \mathit{digit}+$$
$$\text{opt-fraction} \rightarrow (.\mathit{digits})*$$
$$\text{opt-exponent} \rightarrow (E(+|-) * \mathit{digits})*$$
$$\text{unsigned-num} \rightarrow \text{digits opt-fraction opt-exponent}$$

The other application of RE is pattern searching from a given text.

What We Have Learned So Far

1. An RE can be defined as a language or string accepted by an FA.
2. Any terminal symbols, null string ($\wedge$), or null set (Φ) are RE.
3. Union, concatenation, iteration of two REs, or any combination of them are also RE.
4. The Arden's theorem states that if P and Q are two REs over Σ, and if P does not contain Λ, then the equation $R = Q + RP$ has a unique (one and only one) solution $R = QP*$.
5. The Arden's theorem is used to construct an RE from a given FA by the algebraic method.
6. If any FA contains any $\in$ (null) move or transaction, then that FA is called NFA with $\in -moves$.
7. The $\in -closure$ of a state is defined as the set of all states S, such that it can reach from that state to all the states in S with input $\in$ (i.e., with no input).
8. The pumping lemma for RE is used to prove that certain sets are not regular.
9. A set is closed (under an operation) if and only if the operation on two elements of the set produces another element of the set.
10. Closure is a property which describes when we combine any two elements of the set, the result is also included in the set.
11. REs are closed under union, complementation, and intersection operation.

Solved Problems

1. Describe the following REs in English language.

   $a)a(a+b)*abb$ $\quad b)(a+b)*aba(a+b)*$ $\quad c)(0+1)*1(0+1)*0(0+1)*$

   Solution:

   - The language starts with 'a' and ends with 'abb'. In the middle of 'a' and 'b', there is any combination of 'a' and 'b'.

     Hence, the RE described in English language is

     {Set of any combination of 'a' and 'b' beginning with 'a' and ending with 'abb'}.
   - The expression is divided into three parts: $(a+b)*$, aba, and $(a+b)*$. In each element of the language set, there is aba as a substring. In English language, the RE is described as {Set of any combination of 'a' and 'b' containing 'aba' as substring}.
   - The expression is divided into five parts: $(0+1)*, 1, (0+1)*, 0$ and $(0+1)*$. In each element of the language set, there is 1 and 0, where 1 appears first. In English language, the RE is described as

     {Set of any combination of '0' and '1' containing at least one 1 and one 0 where 1 appears first}

2. Find the RE for the following:

a) The set of language of all strings of 0 and 1 containing exactly two 0's

[UPTU 2004]

b) The set of languages of any combination of 'a' and 'b' beginning with 'a'.

c) The set of all strings of 0 and 1 that do not end with 11.

d) The set of languages of any combination of '0' and '1' containing at least one double symbol.

e) The set of all strings over a, b in which the number of occurrences of 'a' is divisible by 3.

[UPTU 2004]

f) The set of all strings where the 10th symbol from the right end is a 1.

[JNTU 2007]

g) The set of languages of any combination of '0' and '1' containing no double symbol.

a) The language contains exactly two 0's. But the language consists of 0 and 1. 1 may appear at first or in the middle or at the last. Thus, the language is $L = 1 * 01 * 01 *$.

b) The set of any combination of 'a' and 'b' is denoted by $(a + b)*$. The RE is

$$L = a(a + b)*.$$

c) The strings may end with 00, 01, or 10. Thus, the language is $L = (0 + 1) * (00 + 01 + 10)$.

d) The language consists of two symbols '0' and '1'. A double (same) symbol means either 00 or 11. The part of the language containing at least one double symbol is denoted by $(00 + 11)$. So, the language of any combination of '0' and '1' containing at least one double symbol is expressed as $L = (0 + 1) * (00 + 11)(0 + 1)*$.

e) The number of 'a' is divisible by 3 means that the number of 'a' may be 0, 3, 6, 9, 12, ... The number of 'b' may be 0, 1, 2, 3,...... . The RE is $(b * ab * ab * ab*)*$.

f) The 10th symbol from the right hand side should be 'a', whereas the other symbols may be 'a' or 'b'. A string of length n has $(n - 10)$ symbols from start and the last 9 symbols are of 'a' or 'b'. The RE is $(a + b) * a \ (a + b)10$.

g) The language consists of two symbols '0' and '1'. A double (same) symbol means either 00 or 11. According to the condition, in the language, 00 or 11 will not appear. The language