Unit-2

Properties of Regular Sets

It is required to determine whether the language is finite or not, which can be easily answered for regular language but not further other language family. We need to decide whether the given language is regular or not If the language is regular we can always show it by DFA, regular expression or regular grammars for it. But if it is not, we need to study the general properties of ~~reg~~ regular language. If we know such properties, and if we can show that the candidate language doesn't have such properties, then we can tell the language is not regular.

To prove that certain languages are not regular, we need to make use of pumping lemma. The pumping lemma works in similar to the pigeonhole principle. The pigeonhole principle based on the simple observation. Suppose there are n-objects and m boxes where n-objects is greater than m-boxes ($n > m$). If such cases, if all n objects are placed into m-boxes then at least one box ~~with~~ will have more than one objects. The pumping lemma is based on this principle. Using pumping lemma, we can easily proved that certain

HE-UO

language are non-regular. In addition, we can make use of pumping lemma to determine whether a language accepted by finite automata is regular or not.

Pumping Lemma:

The language accepted by Finite Automata is known as Regular language. Pumping lemma is used to prove that a language is not regular. It cannot be used to prove that a language is Regular.

pumping :- The word pumping refers to generating many inputs strings by pushing a symbol in an input string repeatedly.

Lemma :- The word lemma refers to the intermediate theorem in a proof. There are two Pumping lemma, that are defined for

- Regular languages
- Context Free languages

Pumping Lemma for Regular Languages

Theorem :- If A is a Regular language, then A has a Pumping length 'p' such that any string 's' where $|s| \geq p$ may be divided into three parts $s = xyz$ such that the following conditions must be true:

(i) $xy^iz \in A$ for every $i \geq 0$ i.e. $i = 0, 1, 2, 3, 4, \ldots$

(ii) $|y| > 0$

(iii) $|xy| \leq p$

In simple words, if a string y is 'pumped' or insert any number of times, the resultant string still remains in A.

Pumping Lemma is used as proof of the irregularity of a language. It means, that if a language is regular, it always satisfies the pumping lemma. If at least one string is made from pumping, not in language A, then A is not regular.
We use the contradiction method to prove that a language is not Regular.

Steps to prove that a language is not Regular using Pumping Lemma:

step1: Assume that language A is regular.
step2: It has to have a Pumping Length (say p).
step3: All strings longer than p can be pumped $|S| \geq p$.
step4: Now, find a string 'S' in A such that $|S| \geq p$
step5: Divide S into xcyz strings.

step6: Show that $xy^iz \notin A$ for same i.i.e. $i \neq 1$

step7: Then consider how S can be divided into xyz.

step8: Show that none of the above strings satisfies all three pumping conditions simultaneously.

step9: Cannot be pumped :: Contraction.

Q1. Show that $L = \{a^n b^n \mid n \geq 0\}$ is not regular

step1:- Let L be regular and n be the number of states in Finite Automata. Consider the string $X = a^n b^n$.

step2: Here, $|X| = 2n$ and is greater than n.

Split X into uvw such that

$|uv| \leq n$ and

$|v| \geq 1$

{ $|uv| \leq n$ and $u^i v^i w \in L$ }

as shown below

$X = \overbrace{aaaaa}^{b}\overbrace{a}^{r}\overbrace{bbbbbb}^{b}$ $\quad n = 6$

$\quad\quad\quad u \quad\quad v \quad\quad w$

where, $|u| = n-1 = 6-1 = 5$

$|v| = 1$

So, that, $|uv| = |u| + |v|$

$= n-1+1$

$= n$

and $w = n$

According to pumping lemma,

$uv^i w \in L$ for $i = 0, 1, 2, 3, \ldots$

Step3:- If $i = 0$ i.e; v doesnot appear and so number of 'a' will be less than the no. of 'b' and so the string 'X' doesnot contain same no. of 'a's followed by same no. of 'b's.

For $i = 2, 3, 4, 5, \ldots$, the no. of a's will be more than no. of b's. According to pumping lemma n number of a's should by followed by n number of b's. which is contradiction to the assumption i.e. the language is regular. So, the language $L = \{a^n b^n \mid n \geq 0\}$ is not regular.

Q2. Show that $L = \{ww^R \mid w \in (0+1)^*\}$ is not regular

Step1: Let L be regular and n be the no. of states in F.A. Consider the string

$X = ww^R$

Step2: Here, $|X| = 2n$ and is greater than n. split X into uvw such that

$|uv| \leq n$ and

$|v| \geq 1$

HK-57

as shown below

$X = \underbrace{001}_{u}\underbrace{1}_{v}\underbrace{100}_{w}$ where, $n=4$

$|u| = n-1 = 4-1 = 3$

$|v| = 1$

$|uv| = |u|+|v| = n-1+1 = n$

and $w = n$

According to pumping lemma

$uv^i w \in L$ for $i = 0, 1, 2, 3, \ldots$

step3: If $i=0$ i.e, v doesnot appear and so w is not be reverse of w^R.

for $i = 2, 3, 4, 5, \ldots$, the value of w and w^R is not contradiction so the language $L = \{ww^R / w \in (0+1)^*\}$ is not regular

Q.3. Show that $L = \{a^n / n \text{ is prime}\}$ is not regular.

Soln→

step1: Let L be regular and n be the numbers of states in FA. Consider the strings

$X = a^n$.

step2: Here, $|X| = n$. Split the X into uvw such that

$|uv| \leq n$

$|v| \geq 1$

as shown below

$X = \underbrace{aa}_{u}\underbrace{a}_{v}\underbrace{aaa}_{w}$

HK-54

where,

$|u| = n-3 = 5-3 = 2$

$|v| = 1$

So that, $|uv| = |u| + |v| = n-3+1 = n-2 \leq n$

$|v| \geq 1$

According to pumping lemma

$uv^i w \in L$ where $i = 0, 1, 2, 3, 4, \cdots$

step3: If $i=0$; V is doesnot present and n is not prime, similarly if $i = 2, 3, 5, 7$, etc, then n is not prime which is contradiction to the assumption that the language is regular. So the language $L = \{a^n \mid n$ is prime$\}$ is not regular.

Q.4. Show that $L = \{a^{n!} \mid n \geq 0\}$ is not regular.

Solⁿ⇒

step1: Let L be regular and n be the no. of states in FA. Consider the string

$x = a^{n!}$

step2: Here, $|x| > n$, so, we can split x into uvw such that

$|uv| \leq n$ and

$|v| \geq 1$

as, shown below

If $n=3$ then $n! = 6$

$x = \underbrace{aaa}_{u}\underbrace{a}_{v}\underbrace{aa}_{w}$

where,

$|u| = n-1$

$|v| = 1$

$|uv| = |u| + |v| = n-1+1 = n$

$|v| \geq 1$

and $w = n$

According to pumping lemma

$uv^i w \in L$ where $i = 0, 1, 2, 3, \cdots$

step3: If $i=0$ i.e. v is doesnot appear and n is not a factorial of n. If $i = 2, 3, 4, 5, \cdots$ then n is not ~~prime~~ factorial of n which is contradiction of the given language is regular. So the given language is not regular.

Q.5. Show that $L = \{a^i b^j \mid i > j\}$ is not regular.

Q.6. Show that $L = \{a^n b^l c^{n+l} \mid n, l \geq 0\}$ is not regular

Q.7. Show that $L = \{a^n \mid n = k^n \text{ for } k \geq 0\}$ is not regular

Q.8. Show that $L = \{0^n$ such that n is not a prime number$\}$ is not regular.

Q.9. Show that $L = \{0^n 1^n 2^n \text{ for } n \geq 0\}$ is not regular

✡ Closure properties of Regular sets/ language

If certain language are regular and a language L is found from them by certain operations, then L is also regular. These theorems are often called closure properties of the regular language.

(i) Closure under union, concatenation and star: Theorem:

If L_1 and L_2 are regular, then $L_1 \cup L_2$, $L_1 \cdot L_2$ and L_1^* also denote the regular language

proof:-

If L_1 and L_2 are regular languages corresponding to the regular expression R_1 & R_2 by definitions $R_1 + R_2$, $R_1 \cdot R_2$ and R_1^* are regular expressions and so, $L_1 \cup L_2$, $L_1 \cdot L_2$ and L_1^* denote the regular languages and therefore regular language are closed under union, concatenation and star closure.

(ii) Closure under complementation

Theorem:-

If L_1 and L_2 are regular, then the regular language is closed under complementation.

proof:- Let $M = \{Q, \Sigma, \delta, q_0, A\}$ be a DFA which accepts the language L_1. Now, we define $M_1 = \{Q_1, \Sigma_1, \delta_1, q_0, Q - A\}$, Here, non final states of M are the final states M_1 and final

states of M and non-final states of M_1. So the language which is rejected by M and is accepted by M_1 and in addition a language accepted by a DFA is regular. Therefore the language accepted by M_1 is regular. So, regular language is closed under complementation.

(iii) Closure under intersection

Theorem:-

Let L_1 and L_2 are regular, then the regular language is closed under intersection.

proof:-

Let $M_1 = \{Q_1, \Sigma, \delta_1, q_1, A_1\}$ accepts the language L_1 and $M_2 = \{Q_2, \Sigma, \delta_2, q_2, A_2\}$ accepts the language L_2 and now we define $M_3 = \{Q_3, \Sigma, \delta_3, q_3, A_3\}$ which recognizes the language $L_1 \cap L_2$ as given below:

$$Q_3 = Q_1 \times Q_2$$

$$q_3 = (q_1, q_2)$$

$$A_3 = \{(p,q) \mid p \in A_1 \text{ and } q \in A_2\}$$

The transition function δ_3 is given as

$$\boxed{Q_3 \times \Sigma \rightarrow Q_3}$$ This can be defined as

$$\delta_3((p,q), a) = (\delta_1(p,a), \delta_2(q,a))$$

HK-58

It is clear from δ_3 that
$\delta_3((p,q),w) = (\delta_1(p,w), \delta_2(q,w))$ and the string w is accepted only if
$\delta_3((p,q),w) \in A_3$
which implies that
$\delta_1(p,w) \in A_3$ and
$\delta_2(q,w) \in A_3$ i.e. if and only if
$w \in L_1 \cap L_2$

So, the regular language is closed under intersection

(iv) Closure under difference

Theorem:- If L_1 and L_2 are regular, then the language is closed under difference.

proof:-

$M_1 = \{Q_1, \Sigma, \delta_1, q_1, A_1\}$ accepts L_1 and
$M_2 = \{Q_2, \Sigma, \delta_2, q_2, A_2\}$ accepts L_2
Now define, $M_3 = (Q_3, \Sigma, \delta_3, q_3, A_3\}$ that recognizes $L_1 - L_2$ as follows.

$Q_3 = Q_1 \times Q_2$
$q_3 = (q_1, q_2)$
$A_3 = \{(p,q) \mid p \in A_1 \text{ and } q \in A_2\}$

$\delta_3 : Q_3 \times \Sigma$ to Q_3 is defined by
$\delta_3((p,q),a) = \delta_1((p,a), \delta_2(q,a))$
It is clear from δ_3 that

HK-59

$\delta_3^*((p,q),w) = (\delta_1^*(p,w), \delta_2(q,w))$ and the string w is accepted only if

$\delta_3((q_1,q_2),w) \in A_3$ which implies

$(\delta_1^*(q_1,w), \delta_2^*(q_2,w)) \in A_3$

This will happen only if $\delta_1^*(q_1,w) \in A_3$ and $\delta_2(q_2,w) \notin A_3$ i.e; if and only if $w \in L_1 - L_2$. So, the regular language.

(v) Closure under homomorphism

Let Σ and Σ^1 are set of alphabets. The function

$h: \Sigma \to \Sigma^{1*}$ is called homomorphism where a single letter is replaced by a string.

If $w = a_0 a_1 \cdots a_n$

$h(w) = h(a_0)h(a_1)\cdots h(a_n)$

If L is made of alphabets Σ then

$h(L) = \{h(w) | w \in L\}$ is called homomorphic image.

Eg. Let $\Sigma = \{0,1\}$ and $\Sigma^1 = \{p,1,2\}$ and

$h(0) = 01$

$h(1) = 112$ and what is $h(010)$.

If $L = \{00, 0101\}$ what is the homomorphic image of L.

Soln $h(010) = h(0)h(1)h(0)$

$= 01\,112\,01$

$H(L) = (00, 010)$

$= (0101, 01112\,01)$ Ans

HK-60

☆ Decision properties of Regular language

(i) Testing Emptiness of a Regular language
(ii) Testing membership of a Regular language

(i) Testing Emptiness of a Regular language

If there is any path from the start state to some accepting state, the language is non-empty. If the accepting states are all seperated from the start state, the language is empty. The accepting state is surely reachable from the start state if there is an arc from q to p with any input symbol (or @ $\in$ in NFA), then p is reachable in this way we can compute set of reachable. If any accepting state is among them; we answered 'no' i.e. the language of the automaton is not empty otherwise we answered "yes" i.e., the language of the automaton is empty

(ii) Testing membership of a Regular language

Given a string w and Regular language L, we need to determine whether w is in L or Not. w is represented explicitly where as L is represented by an automaton or regular expression.

HK-61

If L is represented by DFA, the algorithm is simple. Simulate the DFA processing the string of input symbol w, Beginning in the start state. If the DFA ends in accepting state, the answer is "yes" i.e. $w \in L$; otherwise answer is "no" i.e, $w \notin L$.

✡ Homomorphism

→ A function that maps each symbol in an alphabet to a string, preserving the concatenation of strings, essentially translating one language to another while maintaining its structure.

HK-62