Pumping Lemma for regular language

Statement: If language A is regular language the A has a pumping length 'p' such that any string 's' where $|s| \geq p$ any be divided into 3 points $s = xyz$ such that following conditions must be true.

(a) $xy^iz \in A$ for every $i \geq 0$

(b) $|y| > 0$ (length of y)

(c) $|xy| \leq p$ (length of x and y)

Steps to be followed while proving language is not reular by pumping lemma:

(a) Assume that A is regular.

(b) It has to have a pumping length (say P).

(c) All strings longer than P can be pumped.

(d) Now find a string 's' in A such that $|S| \geq P$.

(e) Divide S into x, y, z.

(f) Show that, $xy^iz \in A$ for some i.

(g) Then consider all ways that S can be divided into x, y, z.

(h) Show that none of these can satisfy all the 3 pumping conditions at the same time.

(i) S can't be pumped = contradiction.

(j) We should learn it by some examples.

1. **Using pumping lemma prove that the language $A = \{a^n b^n / n \geq 0\}$ is not regular.**

Solution:

(Theoretically, we can conclude that this language is not regular as the language generates strings which contains certain number of a's followed by equal number of b's. And to do that, finite state machine need to have some memory for storing counted number of a but in actual it doesn't have memory. So, if a language is not recognized by finite state machine, it is not regular.)

Using pumping lemma, let's assume that language A is regular. Now, it must follow following conditions.

(a) $XY^iz \in A$ for every $i \geq 0$ ... ... (i)

(b) $|y| > 0$ ... ... (ii)

(c) $|xy| \leq p$ where pis pumping length... ... ... (iii)

Let, $P = 7$ (say)

Let, $S \in A = a^P b^P$

$= a^7b^7$

$=$ aaaaaaabbbbbbb

Dividing S into x, y and z. Gets these cases.

Case – I

When y is in the 'a' part.

i.e. $S = \underbrace{a}_{x} \quad \underbrace{aaaa}_{y} \quad \underbrace{aabbbbbbb}_{z}$

Here, condition 2 and 3 are satisfied as $|y| = 4 > 0$ and $|xy| = 5 < 7$.

For condition 1

$xy^iz = xy^2z \qquad [i = 2 \text{ say}]$

Then,

S = a aaaa aaaa aabbbbbbb

So, number of 'a' = 11

number of 'b' = 7

$\therefore$ Number of 'a' $\neq$ number of 'b'.

The first condition is not satisfied.

i.e. $xy^2z \notin A$

Case – II:

When y in the both 'a' and 'b' part

i.e. $S = \underbrace{aaaaaa}_{x} \quad \underbrace{ab}_{y} \quad \underbrace{bbbbbb}_{z}$

Let $S = xy^2z$ (i = 2)

$= aaaaaa\ abab\ bbbbbb$

Here, pattern is not followed i.e. $a^n b^n$.

So, condition 1 is not satisfied and condition 3 as well as $|xy| = 8 > 7$

Case - III

When y is in the 'b' part.

i.e. $S = \underbrace{aaaaaaabb}_{x} \quad \underbrace{bbb}_{y} \quad \underbrace{bb}_{z}$

Let, $S = xy^2z$

$= aaaaaaabb \quad bbbbbb \quad bb$

Here, also condition 1 is not satisfied as, number of a's ≠ number of b's,

So, none of the cases satisfy the all 3 conditions of regular language as stated by pumping lemma. Therefore language A is not regular.

Proving Langauge not to be Regular

It is shown that the class of language known as regular language has at least four different descriptions. They are the language accepted by DFA's, by NFA's, by Є-NFA, and defined by RE.
Not every language is Regular. To show that a langauge is not regular, the powerfull technique used is known as Pumping Lemma.

Pumping Lemma

Statement: Let L be a regular language. Then, there exists an integer constant n so that for any $x \in L$ with $|x| \geq n$, there are strings u, v, w such that $x = uvw$, $|uv| \leq n$, $|v| > 0$.
Then $uv^kw \in L$ for all $k \geq 0$.

Note: Here y is the string that can be pumped i.e repeating y any number of times or deleting it, keeps the resulting string in the language.

→ It is a negative test

VVI

Pumping Lemma for Regular language

The pumping lemma is used to determine whether the language is regular or not i.e. it is used to determine the class of language of FA.

It is a relationship between the length of string (n) and the no. of states (m) of the given finite automata.

It is a powerful tool for proving the certain language non-regularity.

Statement

Let m be the total no. of states and n be the length of strings such that $m \leq n$. Let L be the regular Language and $w \in L$ then we decompose w into 3 sub strings namely x, y, z such that:

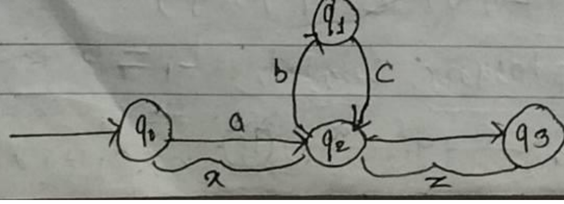
(i) $y \neq \epsilon$ (empty) $[y > 0]$

(ii) $|xy| \leq n$

(iii) string $xy^iz \in L$ for all $i \geq 0$

Note: Here y is the string that can be pumped ie repeating y any number of times or deleting it, keeps the resulting string in the Language.

(eg.)



∄ It is a negative test

VVI

Pumping Lemma for Regular language

The pumping lemma is used to determine whether the language is regular or not i.e. it is used to determine the class of language of FA.

It is a relationship between the length of string (n) and the no. of states (m) of the given finite automata.

It is a powerful tool for proving the certain language non-regularity.

$L \rightarrow$ PLT $\xrightarrow{\text{pass}}$ Reg. (undecidable)

PLT $\xrightarrow{\text{fail}}$ non Reg (decidable)

$L \rightarrow$ PLT $\xrightarrow{\text{pass}}$ undecidable Reg.

$\downarrow$ fail

non Reg (decidable)

Proof:

Let $q_1, q_2, \dots, q_m$ be the m states of finite automata and $w = a_1, a_2, \dots, a_n$ be the string.

Let 'L' be the regular language and $w \in L$.

Let us decompose w into three substrings x, y, z.

$w = a_1, a_2, \dots, a_n$

$x \quad y \quad z$

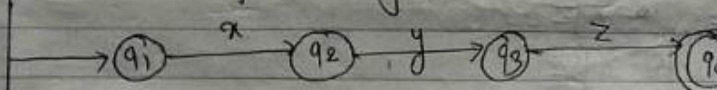
$x = a_1, a_2, \dots, a_i \leftarrow a$

$y = a_{i+1}, a_{i+2}, \dots, a_j$

$z = a_{j+1}, a_{j+2}, \dots, a_n$

Also, we can say that $w = xyz \in L$

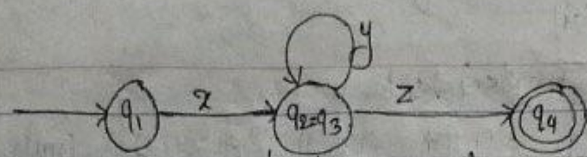
Now, FA for $w = xyz$ is:



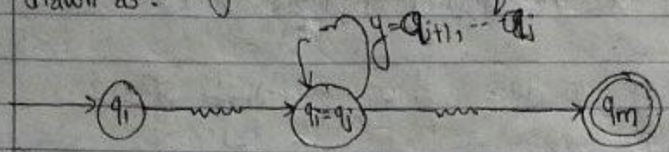
Here, $|w| = 3$ &

total states $(m) = 4$, which violates the rule for pumping lemma.

Hence, using Pigeonhole principle we can say that any 2 states of FA must coincide. Let states q_2 and q_3 coincide then the model is:



Now, the generalized form of above automata can be drawn as:



$x = q_1, \cdots q_i$ $\qquad$ $z = q_{j+1}, q_{j+2}, \cdots q_m$

Here

$$\boxed{1 \le \underbrace{i \le j}_{xy} \le m}$$

$\therefore\ |xy| \le m$

$1 \le |xy| \le n$

Now,

when $i = 0,\ w = xz$

$i = 1,\ w = xyz$

$i = 2,\ w = xy^2z$

$\vdots$

$i = n,\ w = \cancel{xz}\ xy^n z$

Similarly

$xy^i z$ for all $i \geq 0$

$$\boxed{\therefore\ xy^i z \in L}$$

Proved

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

Sol^n

Let L be a regular language, w is a string such that $w \in L$. Let $w = a^p b^p$

pumping length = p

According to pumping Lemma w can be decomposed into 3 substrings x, y and z.

$x = a^q$

$y = a^r$

$z = a^{p-(q+r)} b^p$, $r > 0$.

Now, $xy^2z = a^q y^2 z$

$= a^q a^{2r} . a^{p-(q+r)} b^p$

$= a^{q+2r+p-q-r} b^p$

$= a^{r+p} . b^p$

Since, $r > 0$, $(p+r) > p$ which is not of the form $a^p b^p$.

Therefore, $xy^2z \notin L$

Hence, our assumption that L is regular is wrong.

Proved

Question : Show that the language $L = \{1^{n^2} : n > 0\}$ is not regular.

Soln Let L be a regular language, w is a string such that $w \in L$. Let $w = 1^{p^2}$

According to pumping lemma w can be decomposed into 3 substrings x, y, z.

$$x = 1^a$$
$$y = 1^b$$
$$z = 1^{p^2-(a+b)}$$

~~Let n=2~~ $i = 1$

Then, $w = xy^{i}z$

So,

$$xy^2z = 1^a \cdot 1^{b} \cdot 1^{p^2-(a+b)}$$
$$= 1^{\cancel{a}+\cancel{b}+p^2-\cancel{a}-\cancel{b}}$$
$$= \cancel{1^{b+p^2}} \; 1^{p^2}$$

It is in the form of 1^{n^2} so ~~it~~ L is a regular language.

Closure

~~Closure~~ properties of regular sets or regular languages

~~(1) Union~~

If R_1 and R_2 are REs then R_1+R_2, $R_1 \cdot R_2$, R^* are also RE

Let L_1 and L_2 be two RE such that $L_1 \in M_1$ and $L_2 \in M_2$ where M_1 and M_2 are respective finite state machine for L_1 and L_2.

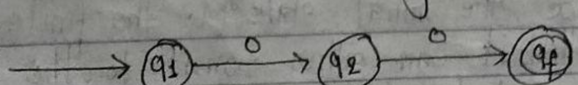
Let $M_1 = \{Q_1, \Sigma_1, \delta_1, q_1, f_1\}$ &
$M_2 = \{Q_2, \Sigma_2, \delta_2, q_2, f_2\}$

(1) for union

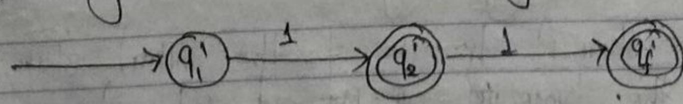
we have to show that the union of L_1 and L_2 ie, L_1+L_2 is also regular. To show that we have to design a finite automata such that it processes L_1+L_2.

~~let~~ $\Sigma = \{0,1\}$

Let $L_1 = 00$ is a regular then ϵ-NFA model is:

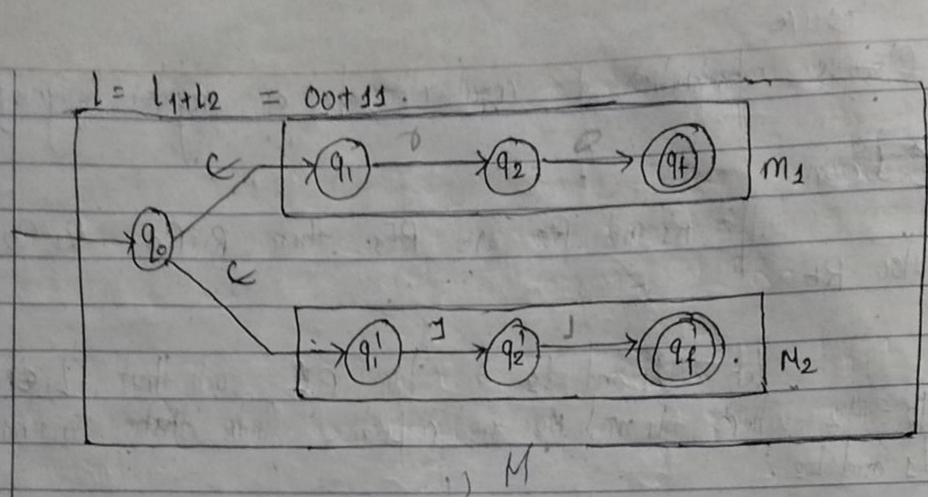


Similarly, $L_2 = 11$ is a regular then model is:



Now, the union is:

$l = l_1 + l_2 = 00 + 11$.



Let $M = \{Q, \Sigma, \delta, q_0, f\}$
where,

Q = finite set of states $= Q_1 \cup Q_2 \cup q_0$
$\Sigma = \Sigma_1 \cup \Sigma_2$
$\delta = \delta_1 \cup \delta_2 \cup \{\delta(q, \epsilon) \rightarrow q_1, \delta(q, \epsilon) \rightarrow q_1'\}$
q_0 = initial state $= q_0$
f = final state $= F_1 \cup F_2 = \{q_f, q_f'\}$

This finite automata M processes the union of l_1 and l_2. Let ~~of~~ q_0 be the initial state of M. The finite automata M is designed such that it can either transit to the initial state of M_1 or to the initial state of M_2 by non-deterministically consuming empty string (ϵ).

When it transits to the initial state of M_1, it initiates the processing of M_1 and, when it transmits to the initial state of M_2 it ~~mi~~ ~~mi~~ imitates the processing of M_2 resulting in the processing of $L_1 + L_2$. Hence, we can conclude that the class of language of FA is closed

(iv) Complement: L is RL then $\bar{L}$ is also RL. $\boxed{\bar{L} = \Sigma^* - L}$

Let L = set of all strings containing 1100 as substring.

Then $\bar{L}$ = " " not " 1100 as substring.

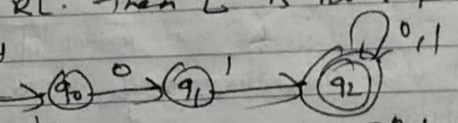
(v) Intersection: under union. If L_1 & L_2 are two RL. then $L_1 \cap L_2$ is also a R.L.

$L_1 = a^*$. $L_2 = b^*$ the $L_1 \cap L_2 = a^* \cap b^*$

~~for concatenation~~ We have to show that

(vi) Reversal: If L is a RL. then L^R is also a RL.

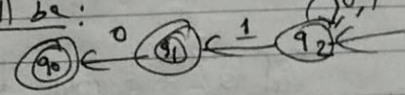
Let assume FA as



$L = \{0, 01, 001\}$

$L^R = (0, 10, 110)$

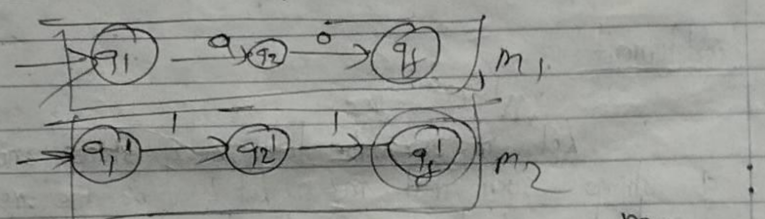
∴ The L^R will be:

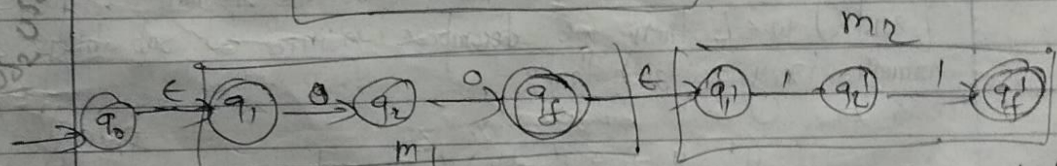


(2) concatenation: →

$L_1 = 00$, $L_2 = 11$

∴ $L_1 L_2$ is also Regular Expression. i.e. 00.11





$\delta: \delta_1 \cup \delta_2 \cup \{\delta(q, \epsilon) \to q_1, \delta(q_f, \epsilon) \to q_1'\}$

q_0 — initial state

Q — Finite set of states $Q_1 \cup Q_2 \cup q_0$

$\Sigma = \Sigma_1 \cup \Sigma_2$

$F = F_1 \cup F_2 = \{q_f, q_f'\}$

(3) Kleene closure similarly, $L_1 = 0$ is regular, $L_1^* = 0^*$ is also re

$F = F_1$, q = initial state

$\delta: \delta_1 \cup \{\delta(q_1, \epsilon) \to q_1, \delta(q, \epsilon) \to q$

$\delta(q_f, \epsilon) \to q_1$