1. What are terminals and non terminal Symbols in grammar

Ans:- Terminals: The terminal Symbols are the elementary symbol of the language defined by a formal gramar. A terminal Symbol which doesn't appear on the left hand side of any production. A grammar Contains a set of terminal Symbol such as Symbol +, -, */, etc. The digits form 0 to 9, The bold faud letters such as id.

Non-terminal: The Non-terminal Symbol [or Syntactic variable) are replaced by groups of terminal Symbol according to the production rules. The following are non terminal. The Cover Case names such as expression operator, operand, statement, etc. The letter S is the start Symbol, The grammar symbol Can be terminal or non-terminal. The Capital letters near the end of the alphabets Such as X, Y, Z etc represent grammar Symbol.

3. Construct a DFA to accept the string abba

Ans:-

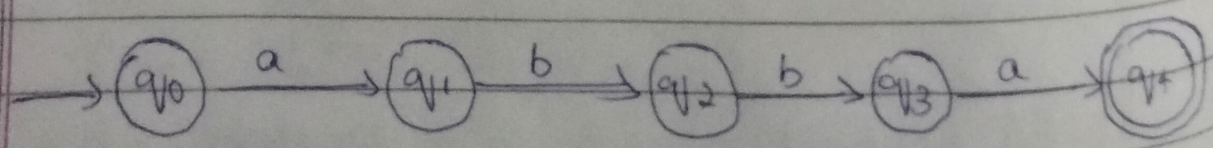
Step 1: Identify the minimum string

minimum string = {abba}

Step 2: Identify the alphabets

$\Sigma = \{a, b\}$

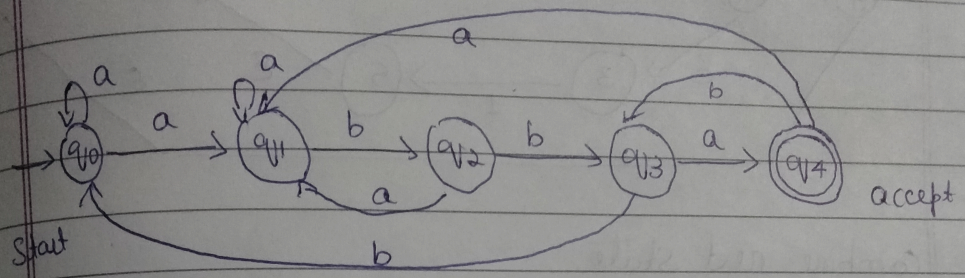
Step 3: Construct a skeleton DFA



start accept

Step 4: Identify the skeleton transition not defined in Step 3

$\delta(q_0,b) = ?$, $\delta(q_1,a) = ?$, $\delta(q_2,a) = ?$, $\delta(q_3,b) = ?$
$\delta(q_4,a) = ?$, $\delta(q_4,b) = ?$



Step 5: Construct the DFA using transition in step 3 & step 4

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

$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$\Sigma = \{a, b\}$

q_0 is the start state

q_4 is final state

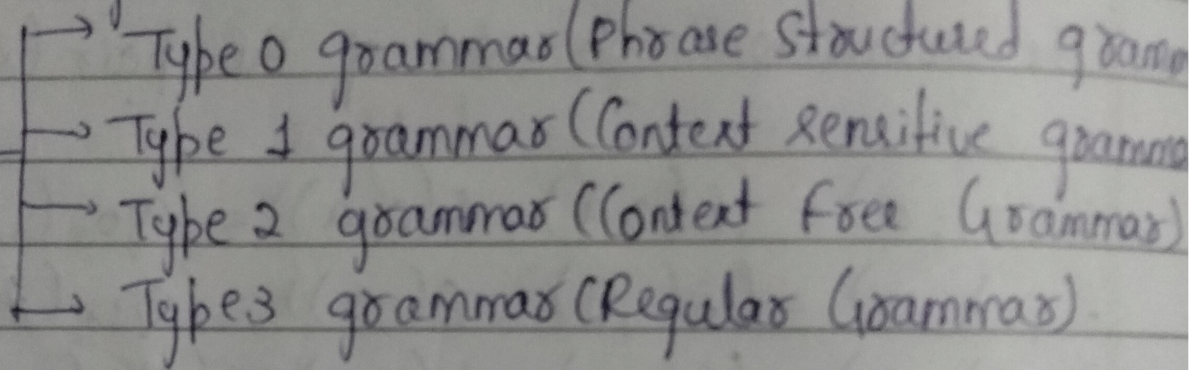
	δ	a	b
→	q_0	q_1	q_0
	q_1	q_1	q_2
	q_2	q_1	q_3
	q_3	q_4	q_0
*	q_4	q_1	q_2

3. ... with Epsilon Transition

production
- S is the start Symbol

5 Explain in detail chomsky hierarchy

Noam chomsky who is the founder of formal language theory has classified the grammar into various categories



(i) Type 0 Grammar

A grammar $G = (V, T, P, S)$ is said to be type 0 grammar / unrestricted grammar / phrase structure grammar if all the production are of the form $\alpha \rightarrow \beta$ where $\alpha \in (V \cup T)^+$ and $\beta \in (V \cup T)^*$

The only restriction is that α can not be ε i.e, ε can not appear on the left hand side of any production family of grammar more

other types. It is also called type 0 language / recursively enumerable language. Only turing machine can recognize the language

$S \longrightarrow aAb/\epsilon$

$aA \longrightarrow bAA$

$bA \longrightarrow a$

(ii) Type 1 Grammar / Context sensitive Grammar

A grammar $G = (V, T, P, S)$ is said to be type 1 grammar Context sensitive grammar if all the productions are of the form $\alpha \rightarrow \beta$ as in type 0 grammar. There is a restriction on length of β. The length of α i.e, $|\beta| \geq |\alpha|$ & α and $\beta \in (V \cup T)^*$ i.e, ϵ can not appear on the left hand right hand side of any production. It is ϵ-free grammar

Linear bounded automata (LBA) can be constructed to recognize the language generated from this grammar

$S \longrightarrow aAb$

$aA \longrightarrow bAA$

$bA \longrightarrow aa$

(iii) Type 2 Grammar / Context free Grammar

A grammar $G(V, T, P, S)$ is said to be type 2 grammar or Context free grammar if all the productions are of the form $A \rightarrow \alpha$ where $\alpha \in (V \cup T)^*$ & A is non terminal. The ϵ can appear on the right hand side of any production. Pushdown Automata (PDA) can be Constructed to recognize the language generated from this grammar

$S \longrightarrow aB \mid bA \mid \varepsilon$

$A \longrightarrow aA \mid b$

$B \longrightarrow bB \mid a \mid \varepsilon$

(iv) Type 3 Grammar / Regular Grammar

The grammar $G = (V, T, P, S)$ is said to type 3 grammar / Regular gramar of the grammar is right linear / Left Linear. A grammar G is said to be right linear if all the production are of the form

$A \longrightarrow wB$

or $A \longrightarrow w$

where A, B are variable & $w \in T^*$, i.e T^* i.e, w & string of terminals. A grammar G is said to the left linear

$A \longrightarrow Bw$ @ $A \longrightarrow w$

where $A, B \in V$ & $w \in T^*$

$S \longrightarrow aaB \mid bbA \mid \in$

$A \longrightarrow aA \mid b$

$B \longrightarrow bB \mid a \mid \varepsilon$

is a right linear Grammar

$S \longrightarrow Baa \mid Abb \mid \varepsilon$

$A \longrightarrow Aa \mid b$

$B \longrightarrow Bb \mid a \mid \varepsilon$

is a left linear grammar

Consider the following grammar

$S \longrightarrow aA$

$A \longrightarrow aB \mid b$

$B \longrightarrow Ab \mid a$

In this grammar each production is either left linear & right linear. But, the grammar is not either left linear / right linear. Such type of grammar is

3. Construct a DFA to accept the string abba

Ans:-

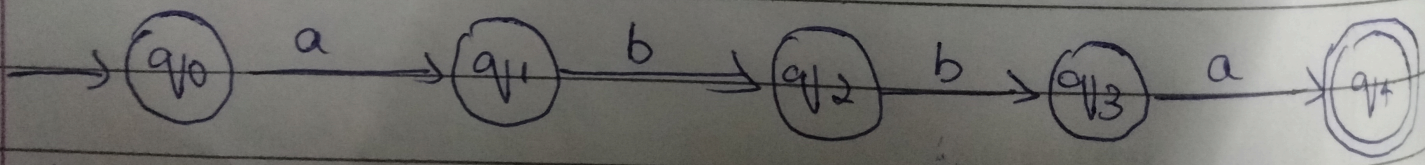
Step 1: Identify the minimum string

minimum string = {abba}

Step 2: Identify the alphabets

$\Sigma = \{a, b\}$

Step 3: Construct a skeleton DFA

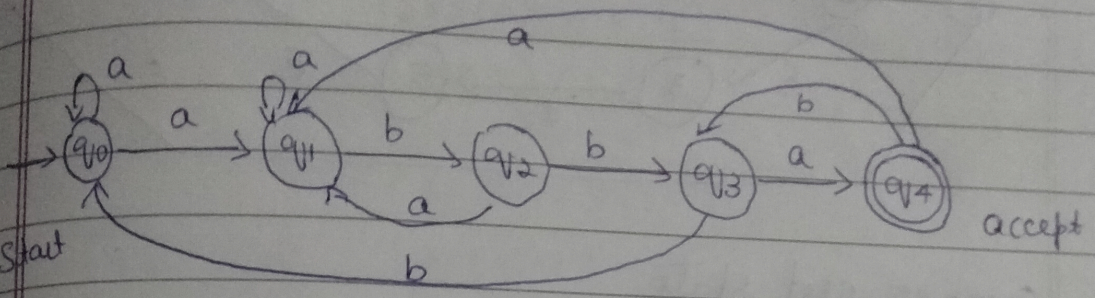


Start

Accept

in Step 3

$\delta(q_0,b) = ?$, $\delta(q_1,a) = ?$, $\delta(q_2,a) = ?$, $\delta(q_3,b) = ?$
$\delta(q_4,a) = ?$, $\delta(q_4,b) = ?$



Step 5: Construct the DFA using transition in step 3 & Step 4

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

$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$\Sigma = \{a, b\}$

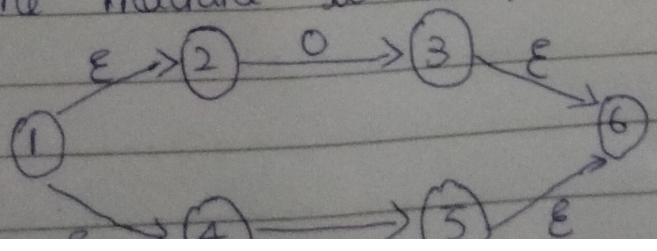
q_0 is the start state

q_4 is final state

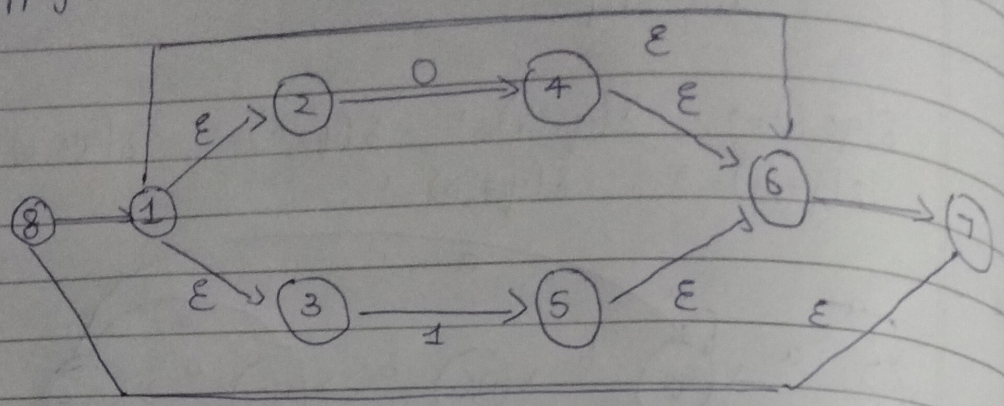
	δ	a	b
→	q_0	q_1	q_0
	q_1	q_1	q_2
	q_2	q_1	q_3
	q_3	q_4	q_0
*	q_4	q_1	q_2

4. Construct a NFA with Epsilon Transition $(0+1)^* 1 (0+1)$

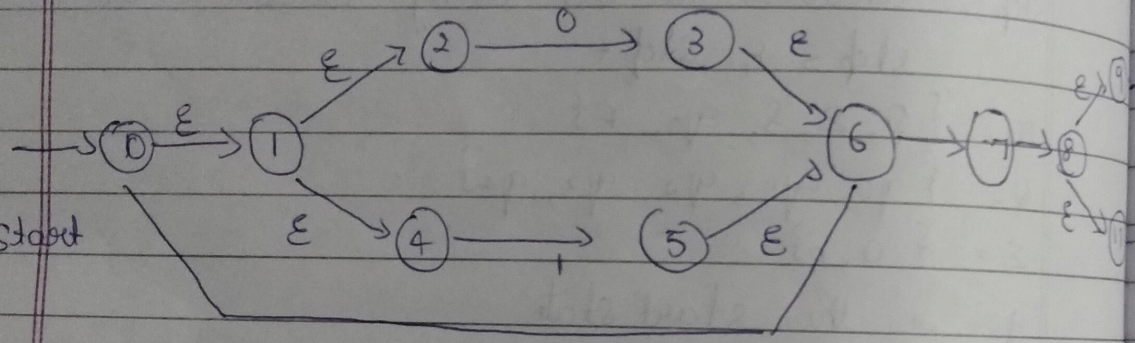
Step 1. The machine to accept '1'



Apply kleene star



Combine next state



5. Construct a DFA to accept the string of 0's & 1's representing zero modulo 5

Step 1 : $r = 2$, $d = \{0, 1\}$ $k = 5$

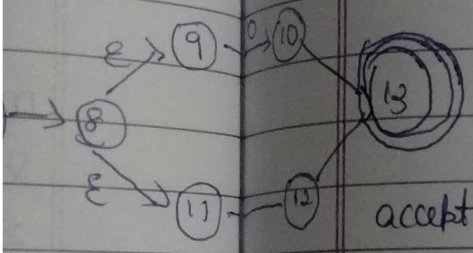
After dividing k, the possibe remainder

Step 2 : $i = 0, 1, 2, 3, 4$

Step 3 : $\delta(q_i, a) = q_j$ where $j = (r * i + d) \bmod k$

with $r = 2$ & $k = 5$

remainder	d	(2*i+d) % 5	$\delta(q_i, d) = q_j$
i = 0	0	(2*0+0) mod 5 = 0	$\delta(q_0, 0) = q_0$
	1	(2*0+1) % 5 = 1	$\delta(q_0, 1) = q_1$
i = 1	0	(2*1+0) % 5 = 2	$\delta(q_1, 0) = q_2$
	1	(2*2+1) % 5 = 3	$\delta(q_1, 1) = q_3$
i = 2	0	(2*2+0) % 5 = 4	$\delta(q_2, 0) = q_4$
	1	(2*2+1) % 5 = 0	$\delta(q_2, 1) = q_0$
i = 3	0	(2*3+0) % 5 = 1	$\delta(q_3, 0) = q_1$
	1	(2*3+1) % 5 = 2	$\delta(q_3, 1) = q_2$
i = 4	0	(2*4+0) % 5 = 3	$\delta(q_4, 0) = q_3$
	1	(2*4+1) % 5 = 4	$\delta(q_4, 1) = q_4$



step 4 : $m = (Q, \Sigma, \delta, q_0, F)$

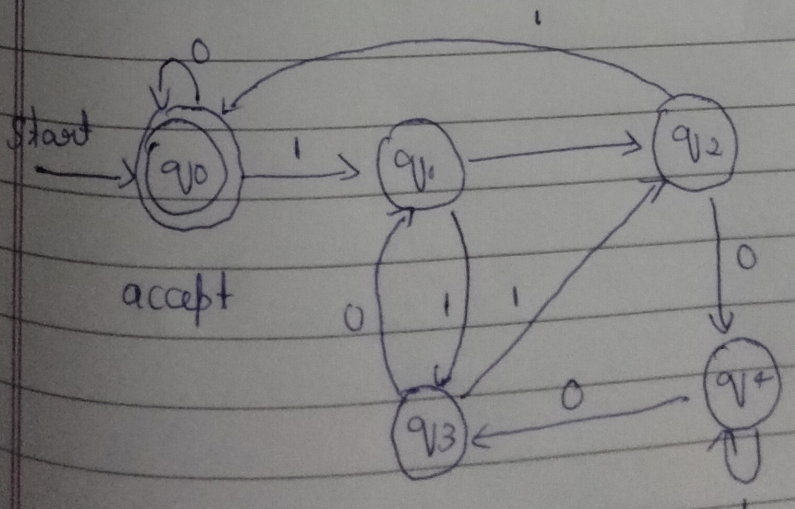
$Q = \{q_0, q_1, q_2, q_3, q_4\}$

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

q_0 is the start state

$F = \{q_0\}$

δ = Transition state



→ Transition table

δ	0	1
→ q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_4	q_0
q_3	q_1	q_2
q_4	q_3	q_4

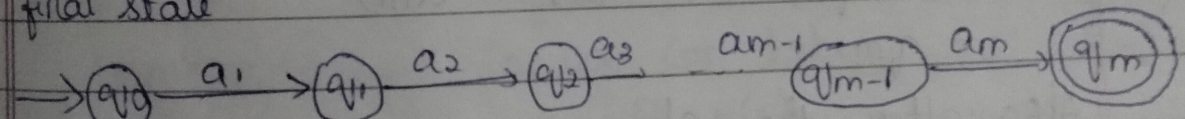
Theorem: Let $m = (Q, \Sigma, \delta, q_0, F)$ be a FA & has n number of states. Let L be the regular language accepted by M. Let for every string x in L, there exists a constant n such that $|x| \geq n$. If string x can be broken into three substring u, v & w such that $x = uvw$

$v \neq \varepsilon$ i.e, $|v| \geq 1$. $|uv| \leq n$

then $uv^i w$ is in L for $i \geq 0$

Proof: Let $m = (Q, \Sigma, \delta, q_0, F)$ be an FA & Let L is the language accepted by DFA & is regular. Let $x = a_1 a_2 a_3 \ldots a_m$ where $m > n$ & each a_i is in Σ. Here, n represent the state of DFA

Since we have m input symbol, naturally we should have m+1 state in the sequence $q_0, q_1, q_2 \ldots q_m$, where q_0 will be start state & q_m will be the final state

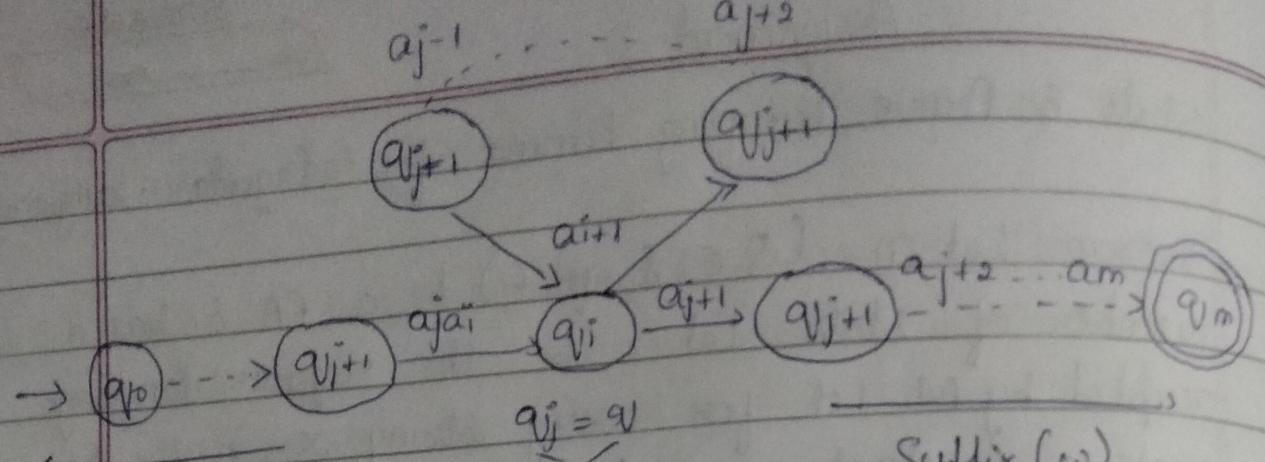


Since $|x| \geq n$, by the pigeon hole principle it is not possible to have distinct transitions, Once of the state can have a loop. Let the string x is indo 3 substrings

The first group is string prefix $a_1, a_2 \ldots a_i$ i.e, $a_1 a_2 \ldots a_i$ the second group is the loop string $a_{i+1} a_{i+2} \ldots a_j$ i.e, $v = a_{i+1} a_{i+2} \ldots a_{j-1} a_j$

The third group is the loop string sufix from $a_{j+1} a_{j+2} \ldots a_m$ i.e.,

$w = a_i \ldots a_{i+2} \ldots a_m$



But when $i=1$ the string uvw is accepted by DFA when $i=2$, the string uvw is accepted by DFA. So, if $i>0$, the machine goes from q_0 to q_j on I/P string u, Circles from q_i to q_j based on the value if i then goes to accepting state on I/P string w, if the string n is split into sub string uvw, then for all $i \geq 0$

$uv^i w \in L$

This can be expressed using transitions

$\delta(q_0, a_1 a_2 \ldots a_{i-1}\ a_i a_{j+1}\ a_{j+2} \ldots a_m)$

$= \delta(\delta(q_0, a_1 a_2 \ldots a_{i+1}\ a_i), a_{j+1}\ a_{j+2} \ldots a_m)$

$= \delta(q_i\ a_{j+1}\ a_{j+2} \ldots a_m)$

$= \delta(q_k, a_{k+1}\ a_{k+2} \ldots a_m)$

$= q_m$

Also after the string $a_1 a_2 \ldots a_i$ the machine will be in state q_{i-1} Since q_i and q_j are same $a_{i+1}\ a_{i+2} \ldots a_j$ any no of time & the machine will stay in q_j only finally, if the I/P string is $a_{j+1}\ a_{j+2} \ldots a_m$ The machine enters into final state q_m