# TOC

## Chomsky Classification

Type 0 => no restriction

Type 1 . $\phi A \psi \to \phi \alpha \psi$ , $\alpha \neq \Delta$

Type 2 $A \to \alpha$

Type 3 .

$L_0$ = Type 0 Language (unrestricted language)

$L_{csl}$ = Type 1 " or Context sensitive language

$L_{cfl}$ = Type 2 " or Context free language

$L_{rl}$ = Type 3 or regular language

## Operation on Languages

Concatenation : Union , Intersection , Transpose (Reversal)

**Thm** : All the language $L_0$, $L_{csl}$, $L_{cl}$, $L_{rl}$ are closed under the union operation.

**Soln** => Let $L_1$ and $L_2$ be 2 languages of same type (Type 1)

Let $G_1 = (V_N', \Sigma_1, P_1, S_1)$ and $G_2 = (V_N'', \Sigma_2, P_2, S_2)$ be the grammar such that $L(G_1) = L_1$ and $L(G_2) = L_2$

$V_N' \cap V_N'' = \phi$

set of terminals

$G = (V_N, \Sigma, P, S)$ starting symbol

set of variables production rules

Let $G_u = (V_N, \Sigma, P, S)$ be a grammar generating a language $L_u$.

We need to proove $L = L_1 \cup L_2$

$V_N = V_N' \cup V_N''$ $\Sigma = \Sigma_1 \cup \Sigma_2$

$P = P_1 \cup P_2 \cup \{S \to S_1 ; S \to S_2\}$

S is a starting symbol , $S \notin V_N' \cup V_N''$

$L_u = L_1 \cup L_2$

Scanned with CamScanner

Let $w$ be a string in $L_1 \cup L_2$.

$w \in L_1$ or $w \in L_2$.

$S_1 \xrightarrow{*} w$ $\quad$ $S_2 \xrightarrow{*} w$

$\Rightarrow S \underset{G_u}{\rightarrow} S_1 \xrightarrow{*} w$ $\quad$ or $\quad$ $S \underset{G_u}{\rightarrow} S_2 \xrightarrow{*} w$

$\therefore w \in L_u \Rightarrow L_1 \cup L_2 \subseteq L_u$

Let $w$ be a string in $L_u$.

then $S \xrightarrow{*} w$

If your first step in the derivation is $S \rightarrow S_1$

then $w \in L_1$

else if first step is $S \rightarrow S_2$ $\quad \therefore w \in L_1 \cup L_2$

then $w \in L_2$

<u>Theorem</u> : $L_0$, $L_{CF_1}$, $L_{CS_1}$, $L_{r_1}$ are closed under concatenation.

<u>Proof</u> $\Rightarrow$ $\quad G_1 = (V_N', \Sigma_1, P_1, S_1)$ $\quad (V_N' \cap V_N'' = \phi)$

$G_2 = (V_N'', \Sigma_2, P_2, S_2)$

$L(G_1) = L_1$ $\quad$ $L(G_2) = L_2$

$L(G_u) = L_1 L_2$ $\quad$ if $\Lambda \in L_1 \rightarrow \cancel{\ } L_2 \subseteq L_u$

$\Lambda \in L_2 \rightarrow \cancel{\ } L_1 \subseteq L_u$

$G_u = (V_N, \Sigma, P, S)$

$V_N = V_N' \cup V_N'' \cup \{S\}$ $\quad \Sigma = \Sigma_1 \cup \Sigma_2$

$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}$

we need to show $L_u = L_1 L_2$

If $w \in L_1 L_2$

$w = w_1 w_2$ $\quad$ where $w_1 \in L_1$, $w_2 \in L_2$

$S_1 \xrightarrow{*} w_1$ $\quad$ $S_2 \xrightarrow{*} w_2$

$S \underset{G_u}{\rightarrow} S_1 S_2 \xrightarrow{*} w_1 S_2 \xrightarrow{*} w_1 w_2$

$w \in L_u$ $\quad \therefore L_1 L_2 \subseteq L_u$

Let $w \in L_u \Rightarrow S \xrightarrow{*} w$

The first step is $S \to S_1 S_2$

<u>Type 1 or Type 3</u>

$\Lambda \in L_1 \cup L_2$

$L_1' = L_1 - \{\Lambda\}$ $\quad$ $L_2' = L_2 - \{\Lambda\}$

$$L_1 L_2 \begin{cases} L_1' L_2' \cup L_2^{*} & ; \Lambda \in L_1 ; \Lambda \notin L_2 \\ L_1'(L_2) \cup L_1 & ; \Lambda \notin L_1 ; \Lambda \notin L_2 \\ L_1'(L_2) \cup L_1 \cup L_2 & ; \Lambda \in L_1 ; \Lambda \in L_2 \end{cases}$$

---

## UNIT - 3

(Regular sets and Regular Expression)

<u>Regular Expressions</u> ⇒ are used for representing a set of strings in an algebraic ~~expr~~ fashion

<u>Formal definition of regular expression</u>

(i) Any terminal symbol $(a \in \Sigma)$, $\Lambda$, $\phi$ are regular expressions

(ii) union of two regular expressions is a regular expression

(iii) concatenation of two regular expressions is a regular expression

(iv) The iteration (Reflexive - transitive closure } or Kleene -Star) of a regular expression is a regular expression.

(v) If R is a regular expression then (R) is also a regular expression.

Any Combination of the above operations when applied on regular expressions give a regular expression.

union $R_1 + R_2$ $\quad$ Concatenation $R_1 R_2$

Iteration $R^*$

a or r.e.a

# TOC

~~Format~~ . Iteration > Concatenation > Union

exp > $\cdot$ > $+$

$a \in \Sigma$ , r.e a

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

$ab = \{ab\}$

$a^* = \{\Lambda, a, aa, aaa, \ldots\}$

**Identities :** Let P, Q, R be r.e

(i) $\phi + R = R$

(ii) $\phi R = R\phi = \phi$

(iii) $\Lambda R = R = R\Lambda = R$

(iv) $\Lambda^* = \Lambda$ & $\phi^* = \Lambda$

(v) $R + R = R$

(vi) $R^* R^* = R^*$

(vii) $R^* R = RR^* = R^+$

(viii) $(R^*)^* = R^*$

(ix) $\boxed{\Lambda + RR^* = R^*}$

(x) $(PQ)^* P = P(QP)^*$

(xi) $(P+Q)^* = (P^* Q^*)^*$

$= (P^* + Q^*)^*$ r.e.

(xii) $(P+Q)R = PR + QR$

and

$P(Q+R) = PQ + PR$

## ★ Arden's Theorem

**Theorem :** Let P and Q be 2 regular expression such that P does not contain $\Lambda$. Then, the following equation in R $\boxed{R = Q + RP}$ — (1)

has a unique solution given by $R = QP^*$

**Proof :** Substitute $R = QP^*$ on RHS

$Q + RP = Q + QP^*P$

$= Q[\Lambda + P^*P]$   $\Lambda + RR^* = R^*$

$= QP^*$

$= R = LHS$

$QP^*$ is a soln of (1)

Uniqueness

Substitution $R = Q + RP$ on RHS of (1)

$R = Q + RP$

011*

$= Q + (Q + RP)P$

$= Q + QP + RP^2$

$= Q + QP + (Q + RP)P^2$

$= Q + QP + QP^2 + RP^3$

$\vdots$

$R = \overbrace{Q + QP + QP^2 + \dots + QP^i}^{a} + \overbrace{RP^{i+1}}^{b}$ — (2)

Any solⁿ of (1) is also a solⁿ of (2)

Let $w$ be a string of length '$i$' in $R$.

So, $w$ must also be contained on RHS of (2)

$w$ is contained in $Q + QP + QP^2 + \dots + QP^i$ and not in $RP^{i+1}$ ($\because$ any string in $RP^{i+1}$ is of length $\geq (i+1)$)

$\therefore R \subseteq QP^*$

Converse => If any string is contained in $R$, it will be there in $QP^*$

Let $w$ be any string in $QP^*$

$\therefore$ $w$ is contained in $QP^k$ for some $k \geq 0$

=> $w$ is contained in $Q + QP + QP^2 + \dots QP^i$

=> $w$ is contained in $R$

$w \subseteq R$

Ques: Give a regular expression representing a set of strings where every zero is followed by atleast 2 ones.

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

eg : $(011 + 1)^*$

- **Simplify :-**

$\Lambda + [1^*(011)^*](1^*(011)^*)^*$ $\quad P = 1^*(011)^*$

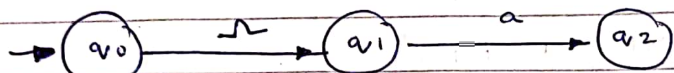
$= [1^*(011)^*]^*$ $\quad \Lambda + P^*.P = P^*$

~~= [1+011]*~~ $\quad = [1+011]^*$

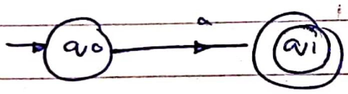
## Transition systems containing empty moves

i.e $\to$ NDFA with $\Lambda$ moves $\to$ Remove $\Lambda$-moves $\to$ DFA

- Given a transition system with $\Lambda$ moves
$\downarrow$
System without $\Lambda$-moves
$\downarrow$



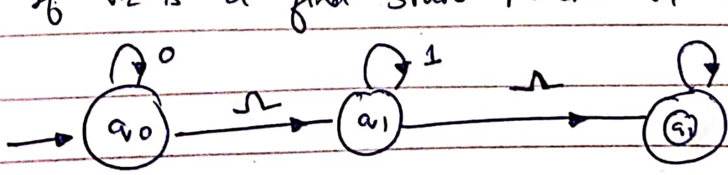

empty moves are unnecessary
$\Downarrow$

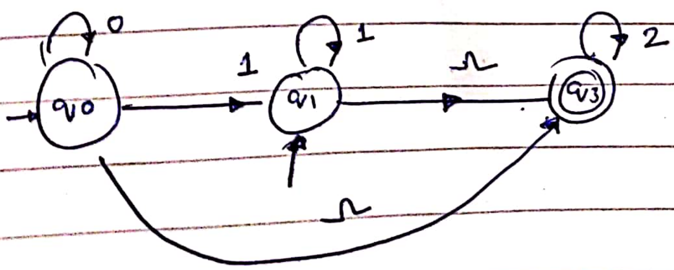



**Procedure $\Rightarrow$**

- Suppose we want to remove an empty move from state $V_1$ to state $V_2$, then we follow the following steps $\Rightarrow$

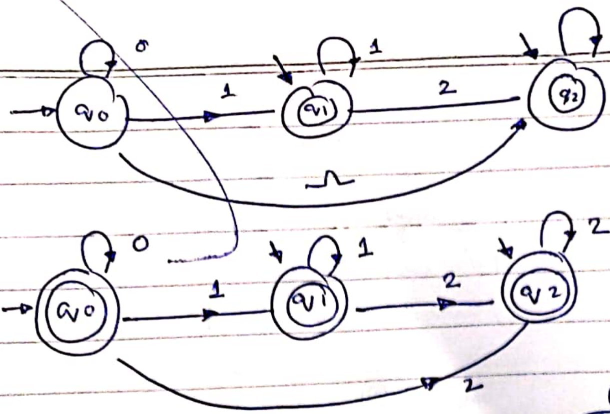
1) Find all the edges starting from $V_2$
2) Duplicate all these edges now starting from $V_1$.
3) If $V_1$ is an initial state, make $V_2$ also an initial state.
4) If $V_2$ is a final state, make $V_1$ also a final state.



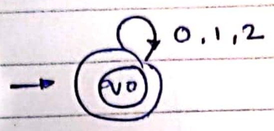

Scanned with CamScanner

A+B




but this is not minimized.

minimized version =>




## Kleen's theorem

**Statement** - If R is a regular expression over $\Sigma$ representing $L \subseteq \Sigma^*$ then there exists an NDFA with empty moves such that

$L = T(M)$

**Proof** => We use PMI on no of characters in the regular expression R.
[$a \in \Sigma$, $\Lambda$, $\phi$, operators $*$, $+$]

**Base step** If # of characters in R is 1.

$R = a \in \Sigma$ or $R = \Lambda$ or $R = \phi$


for $a \in \Sigma$ — for $\Lambda$ — for $\phi$

Assume $\exists$ an NDFA with $\Lambda$ moves for r.e R with n characters.

**To prove:** the result for r.e with $(n+1)$ characters.

Based on the last operator applied in the regular expression R.

| | | |
|---|---|---|
| $R = P^*$, r.e P<br>length of $P = n$ | $R = P + Q$ P, Q r.e.<br>length of P, Q are less than n | $R = PQ$ (P, Q r.e)<br>length of P and Q are less than n. |

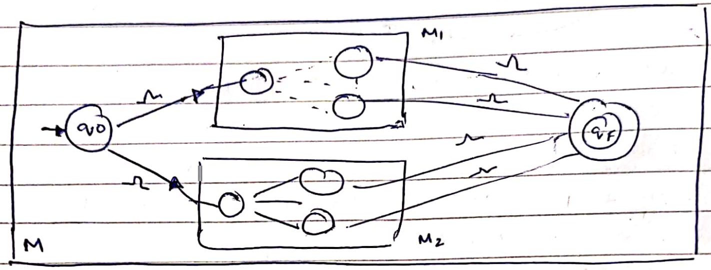
Here P and Q are r.e of length $\leq n$

So, by Induction hypothesis $\exists$ NDFA's.

Scanned with CamScanner

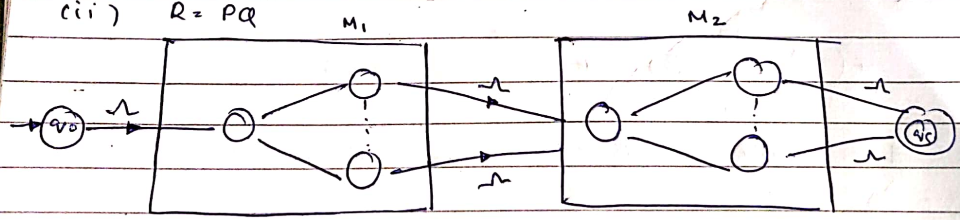
Let $M_1, M_2$ be $L(P) = T(M_1)$ and $L(Q) = T(M_2)$


(i) $P + Q$

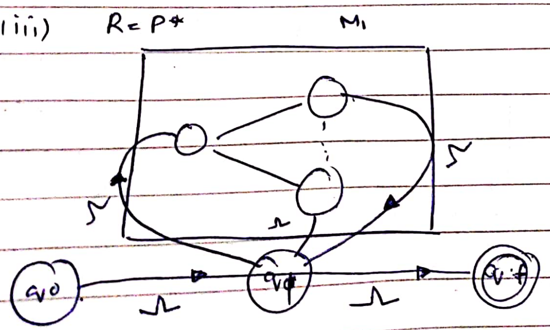



Let $q_0$ and $q_F$ be new initial and final state respectively such that $q_0$ and $q_f$ don't belong to $M_1$ and $M_2$. Adding $\Lambda$ moves from $q_0$ to initial states of $M_1$ and $M_2$ & from final states of $M_1$ and $M_2$ to $q_f$.

(ii) $R = PQ$




(iii) $R = P^*$




Scanned with CamScanner

Non deterministic Tran System to Deterministic System

DFA → $(Q, \Sigma, \delta, q_0, F)$ $\delta : Q \times \Sigma \to Q$

NDFA → $(Q, \Sigma, \delta, q_0, F)$ $\delta : Q \times \Sigma \to 2^Q$

Transition System → $(Q, \Sigma, \delta, Q_0, F)$

( ↓ more than one initial state )

| | a | b |
|---|---|---|
| → $q_0$ | | $q_1, q_2$ |
| → $q_1$ | | $q_0$ |
| $q_2$ | $q_0, q_1$ | |

| | a | b |
|---|---|---|
| → $[q_0, q_1]$ | $\phi$ | $[q_0, q_1, q_2]$ |
| $[q_0, q_1, q_2]$ | $[q_0, q_1]$ | $[q_0, q_1, q_2]$ |

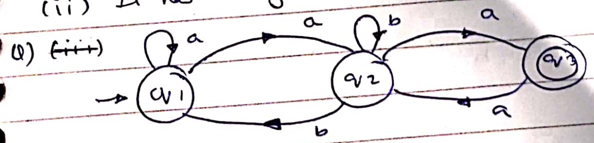
→ Algebraic method using Arden's Theorem

The following assumptions are made regarding the given transition system,

(i) The transition graph does not have empty moves.

(ii) It has only one initial state

Q) ~~(iii)~~ Find r.e accepted by given system.




$\cancel{q_1}$

$R = Q + RP$

$\Rightarrow R = QP^*$

$q_1 = q_1 a + q_2 b + \Lambda$

$q_2 = q_2 b + q_3 a + q_1 a +$

$q_3 = q_2 a$

$\Rightarrow q_2 = q_1 a + q_2 b + q_2 aa$

$q_2 = q_1 a + q_2 (b + aa)$

$\Rightarrow q_2 = q_1 a (b + aa)^*$

Scanned with CamScanner

$q_1 = q_1 a + q_1 a (b+aa)^* b + \Lambda$

$q_1 = \Lambda + q_1 [a + a(b+aa^*)b)$

$q_1 = \Lambda + (a + a(b+aa)^* b)^*$

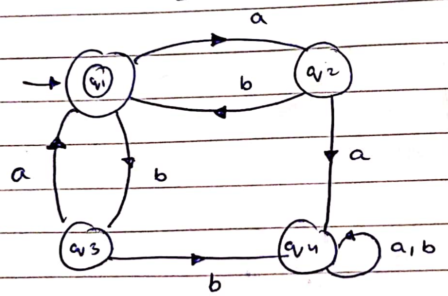
$q_1 = (a + a(b+aa)^* b)^*$

$q_2 = q_1 a (b+aa)^*$

$= [a + a(b+aa)^* b]^* a (b+aa)^*$

$q_3 = q_2 a$

$= [a + a(b+aa)^* b]^* a (b+aa)^* a$




$q_1 = q_2 b + q_3 a + \Lambda$

$q_2 = q_1 a$ *

$q_3 = q_1 b$

$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$

$q_4 = (q_1 a) a + (q_1 b) b + q_4 (a+b)$

$q_4 = [q_1 aa + q_1 bb] + q_4 [a+b]$

Scanned with CamScanner