LFA: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie.

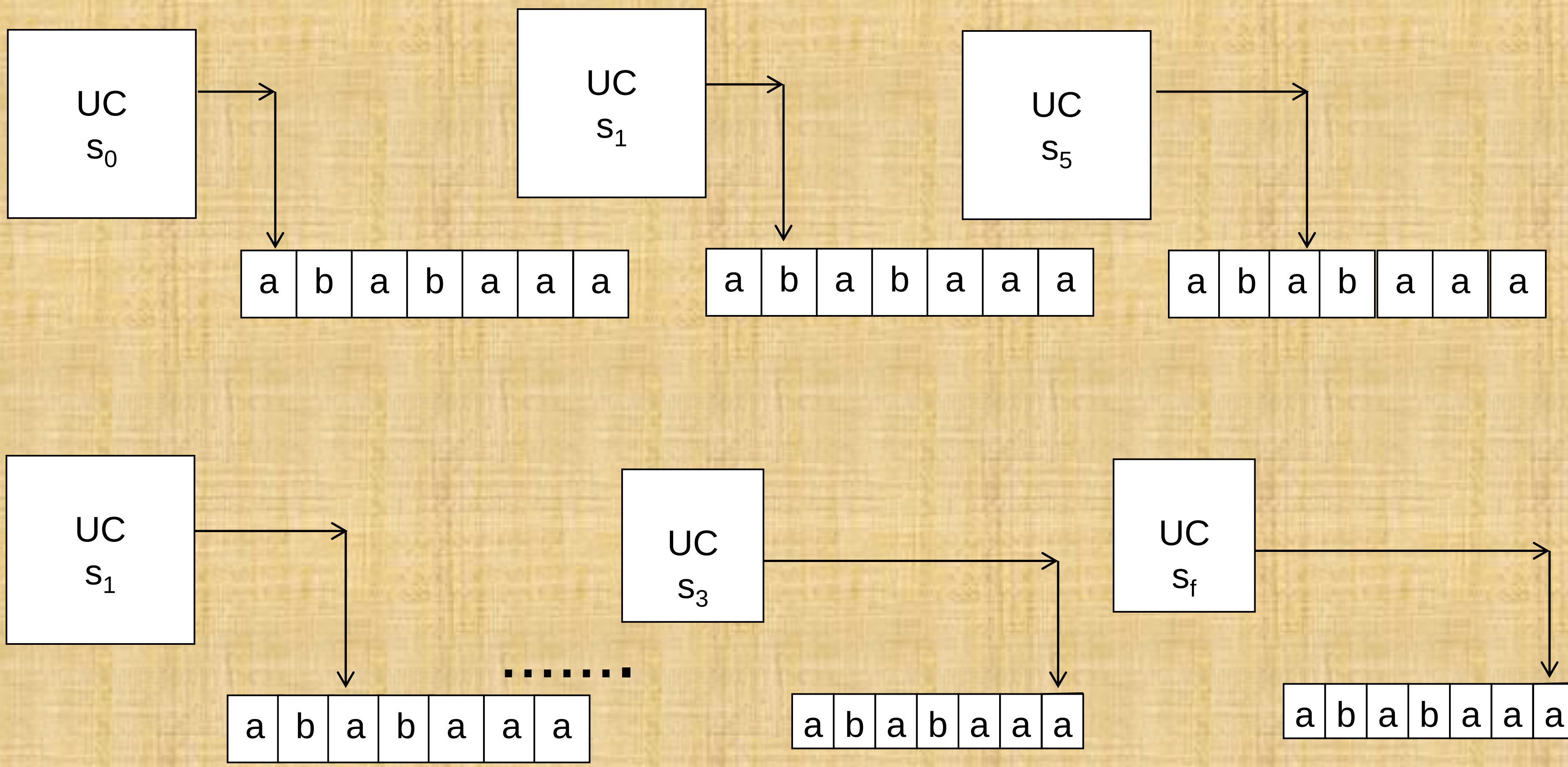
LFA: C3 – LIMBAJE REGULATE

Automatele finite: aplicatii

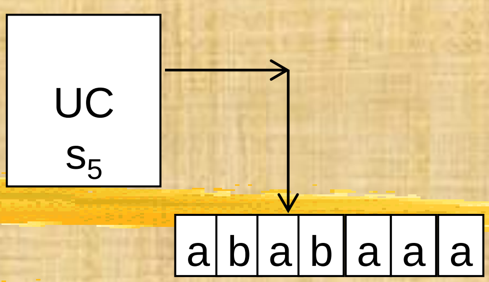
- ✓ procesarea vorbirii,
- ✓ recunoasterea optica a caracterelor,
- ✓ recunoasterea formelor,
- ✓ modele matematice pentru calculatoarele cu memorie finita (incorporate in aparatele electrocasnice, comutatoare/bariere electrice etc.).

LFA: C3 – LIMBAJE REGULATE

Automat



LFA: C3 – LIMBAJE REGULATE



Automat

Informal: cel mai simplu automat este un dispozitiv care

✓ dispune de:

- o unitate de calcul şi de control (decizie),
- o banda FINITA, utilizata ca dispozitiv de memorare,
- un cap de citire care se poate deplasa pe banda numai de la stanga la dreapta şi
- care poate numai citi simbolul curent de pe banda;

✓ şi care calculeaza astfel:

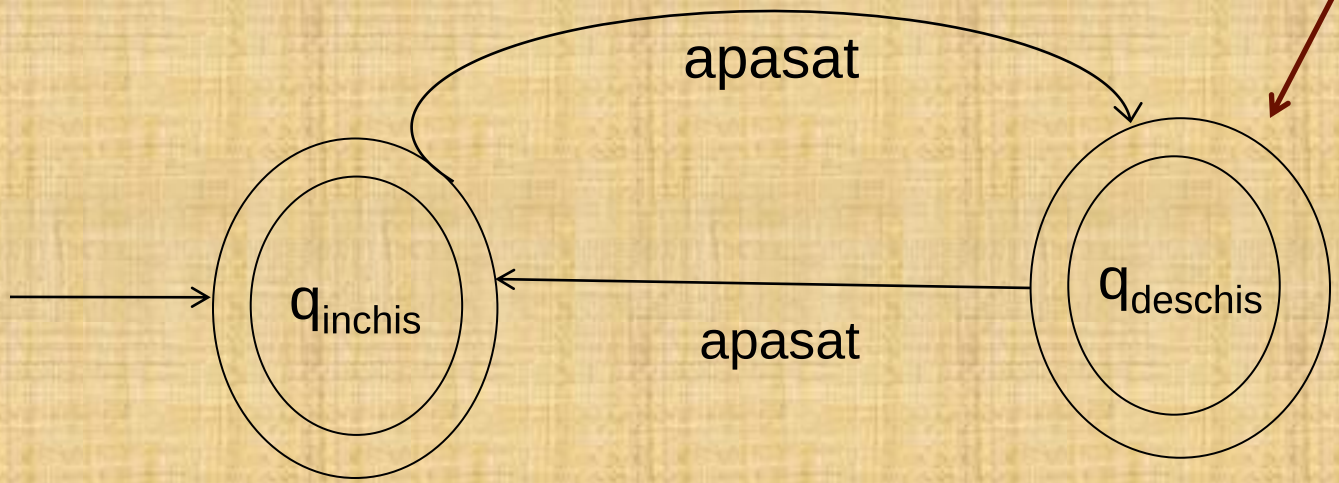
- incepe calculul aflat în starea initiala şi cu capul de citire postat pe prima celula din stanga,
- la fiecare pas de calcul, inclusiv primul, în functie de starea curenta şi de simbolul citit, trece în alta stare/ ramane în starea curenta şi comanda deplasarea capului de citire o celula la dreapta.

LFA: C3 – LIMBA

Acest comutator este un calculator cu 1! bit de memorie, suficient pt a memora in care dintre cele 2 stari se afla comutatorul

<u>Exemple 1:</u>

AF pt un comutator electric

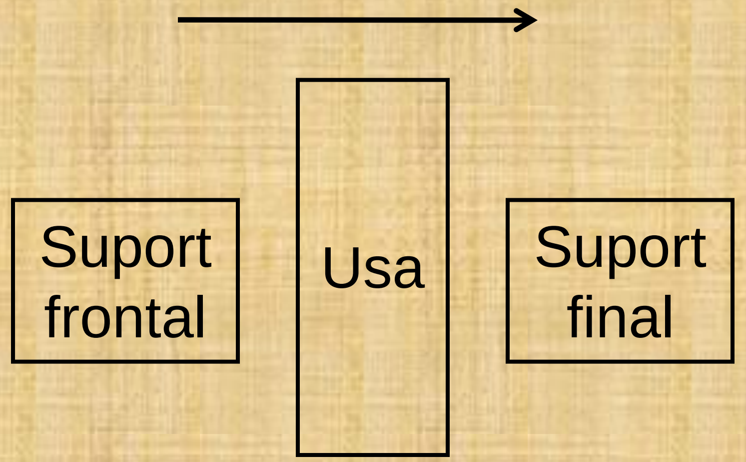


Ascensoare, termostate, masini de spalat etc.

LFA: C3 – LIMBAJE REGULATE

Exemplu 2:

AF pt o usa automata pentru acces auto:

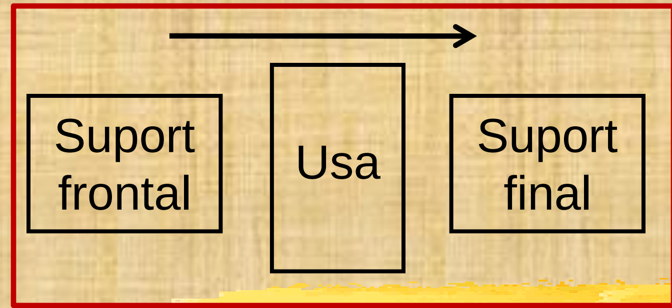


3 descrieri posibile

i. Descrierea in limbajul natural;

ii. Descrierea formala;

iii. Descrierea cu ajutorul diagramei de stare.

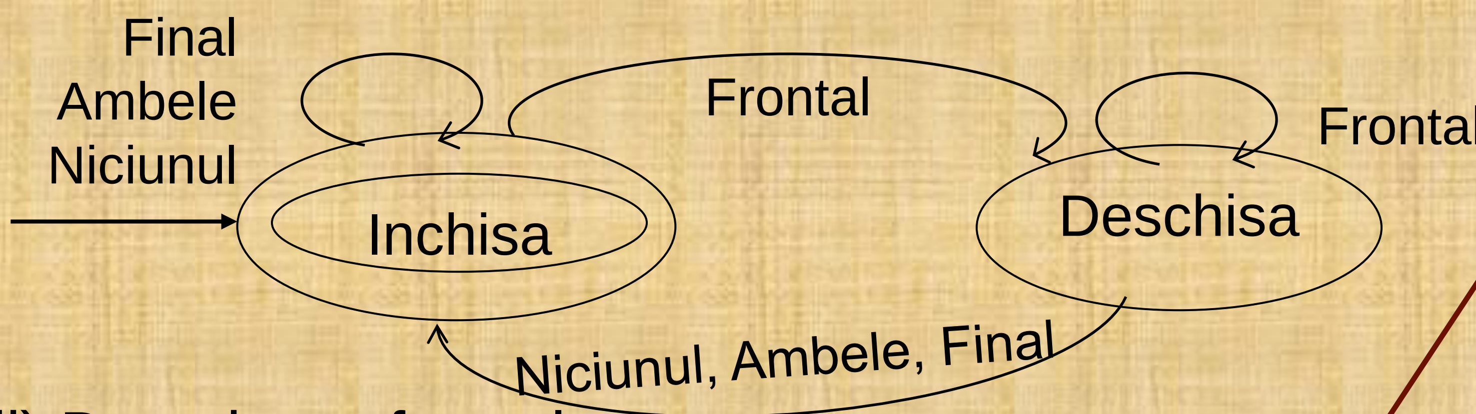


LFA: C3 – Limbaje regulate

(i) Descrierea in limbajul natural:

Aceasta usa automata dispune de un calculator cu 1! bit de memorie , suficient pt a memora in care dintre cele 2 stari se afla usa

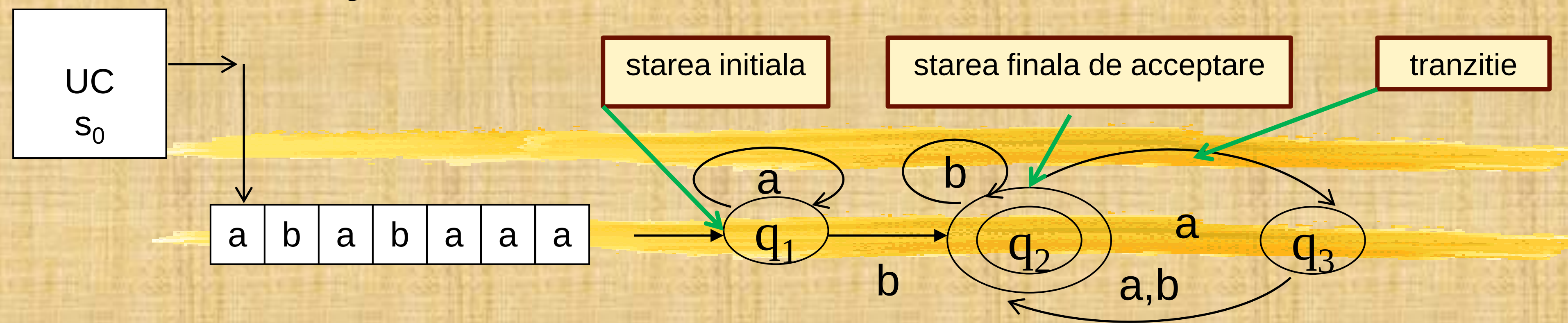
(ii) Descrierea cu ajutorul diagramei de stare



(iii) Descrierea formala

	Pe suportul frontal	*Pe suportul final*	*Pe ambele suporturi*	*Pe niciun suport*
Inchis	Deschis	Inchis	Inchis	Inchis
Deschis	Deschis	Inchis	Inchis	Inchis

LFA: C3 – LIMBAJE REGULATE



Observatie 3: Principiul de lucru

- ✓ Automatul finit (determinist) este un mecanism => e caracterizat de stari şi tranzitii intre stari
- ✓ date de intrare şi rezultate

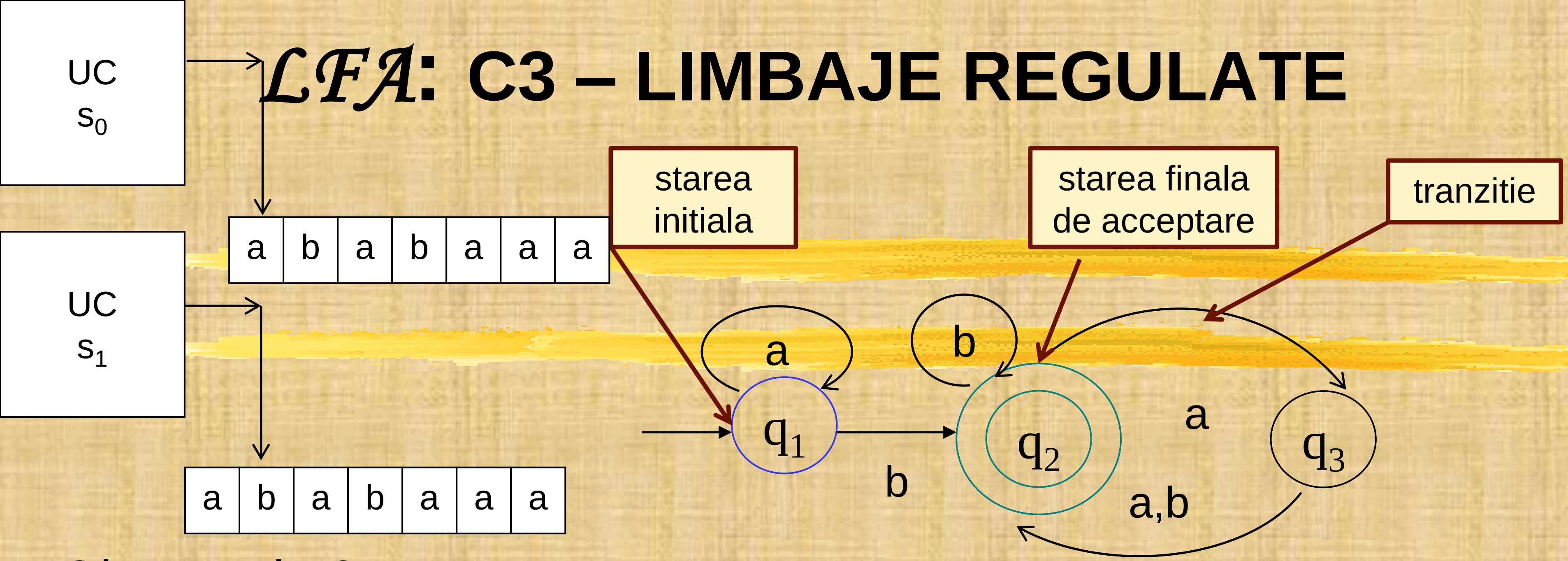
Date de intrare:

- ✓ o secventa FINITA de simboluri din alfabet, care sunt ”citite” unul cate unul;

In ce consta calculul/prelucrarea?

- ✓ aflat in starea initiala, automatul citeste un simbol din secventa primita ca intrare
- ✓ trece din starea curenta in alta stare (unic determinata)
- ✓ procedeaza in continuare la fel, pana la epuizarea secventei
- ✓ in acel moment (FINAL), accepta/respinge secventa in functie de tipul de stare in care se gaseste.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



<u>Observatie 3 (cont.)</u>

Ce determina trecerea intr-o (anumita) alta stare (calculul/prelucrarea)?

- ✓ starea curenta
- ✓ simbolul curent “citit”

Cand se termina calculul?

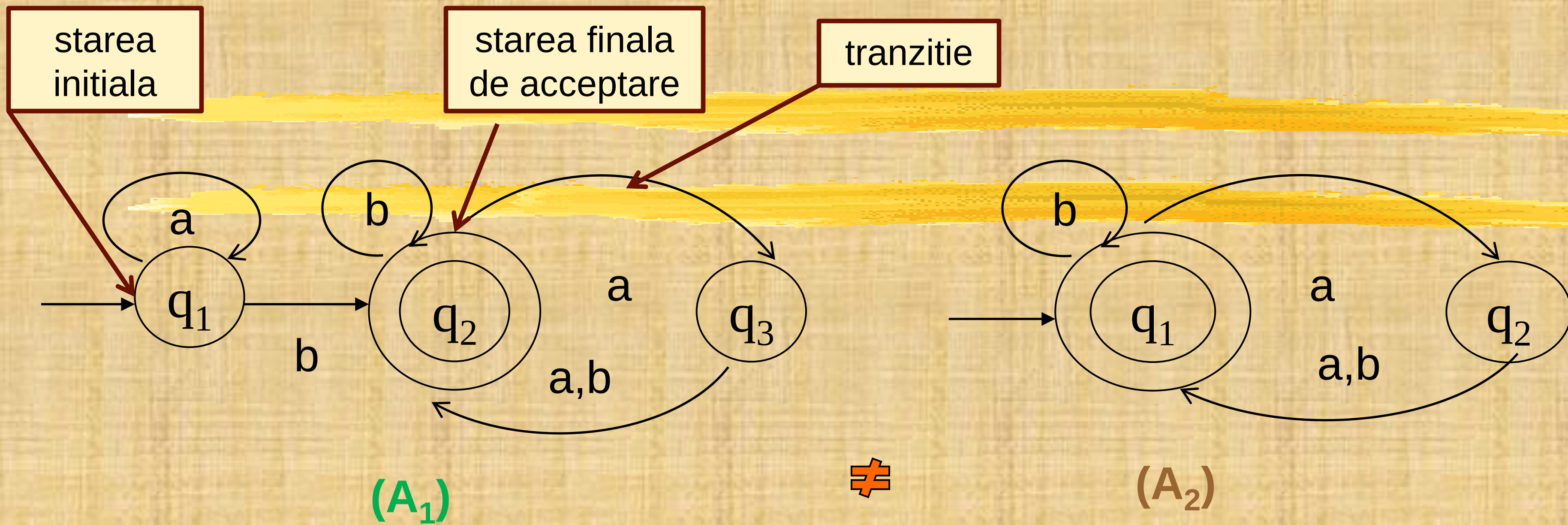
- ✓ cand au fost citite toate simbolurile din secventa de intrare

Cum se termina calculul (ce produce automatul)?

- ✓ la “terminarea” secventei, automatul ajunge intr-una dintre starile “finale”, deci **automatul accepta secventa,**
- ✓ la “terminarea” secventei, automatul ajunge intr-una dintre starile “nefinale”, deci **automatul nu accepta secventa**;

<u>Observatie 4: Conventii de reprezentare.</u> →

LFA: C3 – LIMBAJE REGULATE



√ b, ab, bb, abab, ababaa, abaab, ……

⊗ a, ba, ababa, …..

$(q_1,b) \to q_2$; $(q_1,a) \to (q_1,b) \to q_2$; $(q_1,a) \to (q_1,b) \to (q_2,a) \to (q_3,b) \to q_2$;

$(q_1,a) \to (q_1,b) \to (q_2,a) \to (q_3,b) \to (q_2,a) \to (q_3,a) \to q_2$; etc

$(q_1,a) \to q_1$; $(q_1,b) \to (q_2,a) \to q_3$; $(q_1,a) \to (q_1,b) \to (q_2,a) \to (q_3,b) \to (q_2,a) \to q_3$;

=>

$L(A_1) = L_1 = \{ w \in \{a,b\}^* \mid w = \alpha b(aa)^n, \alpha \in \{a,b\}^* \}$

$L(A_2) = L_1 \cup \{ \varepsilon \} \cup \{ (aa)^n \mid n \in N \}$

=> e necesara o definitie formala a AFD

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 5: Automat finit determinist

AFD = (Q, Σ, δ, s, F), unde:

Q = multime finita, nevida (stari),

Σ = multime finita, nevida, numita alfabet de intrare (simboluri),

$\delta : Q \times \Sigma \rightarrow Q$, numita functia de tranzitie,

$s \in Q$, numita starea initiala,

$F \subseteq Q$ numita multimea starilor finale (de acceptare);

Notatie 6

$\mathcal{A}$ = { A | A este un automat finit determinist }

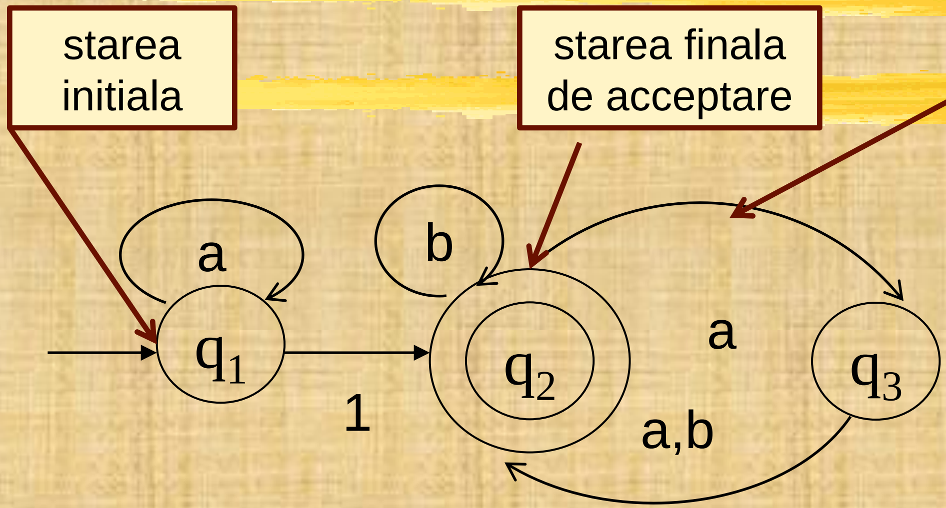
Observatie 7

Pentru a descrie calculul efectuat de un AFD extindem functia δ printr-o definitie inductiva astfel:

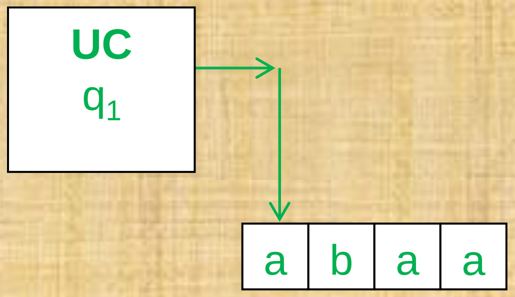
$\delta : Q \times \Sigma^* \rightarrow Q : \quad \delta(s, \varepsilon) = s$

$\delta(s,wa) = \delta(\delta(s,w),a), \forall w \in \Sigma^*, a \in \Sigma$.

LFA: C3 – LIMBAJE REGULATE



Fie:



$\delta(q_1,abaa) =$

$\delta(\delta(q_1,aba),a) =$

$\delta(\delta(\delta(q_1,ab),a),a) =$

$\delta(\delta(\delta(\delta(q_1,a),b),a),a) =$

$\delta(\delta(\delta(q_1,b),a),a) =$

$\delta(\delta(q_2,a),a) =$

$\delta(q_3,a) = q_2$.

<u>Exemplu 8</u>

A_1: $Q = \{q_1,q_2,q_3\}$;

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

$s = q_1$;

$F = \{q_2\}$

δ :

	a	b
q_1	**q_1**	**q_2**
q_2	**q_3**	**q_2**
q_3	**q_2**	**q_2**

LFA: C3 – LIMBAJE REGULATE

Exemplu 8

A_1: $Q = \{A,B,C,D,E,F,G\}$;

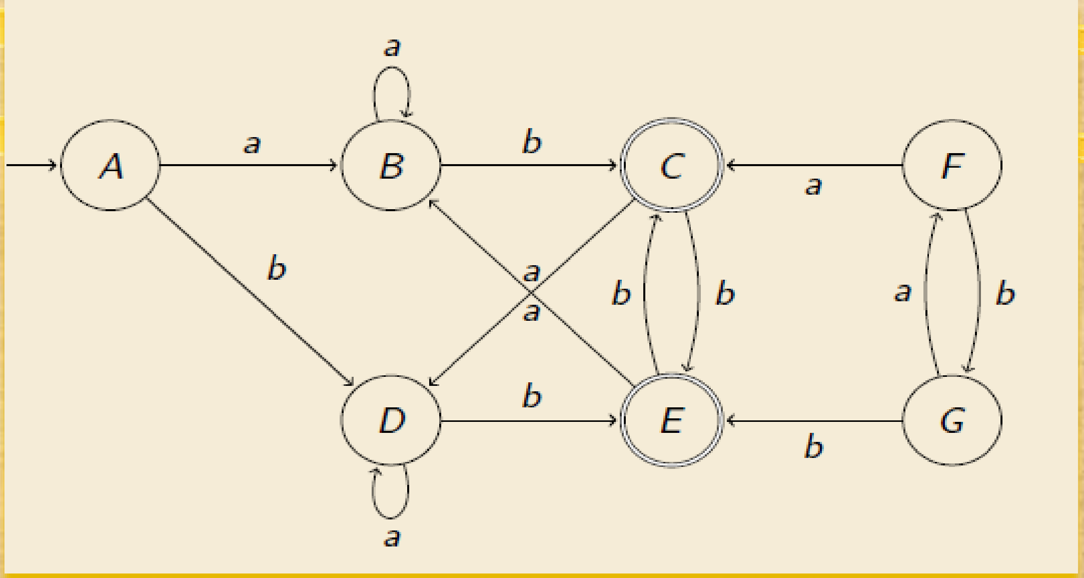
$\Sigma = \{a,b\}$;

$s = A$;

$F = \{C,E\}$

δ :

	a	b
A	B	D
B	B	C
C	D	E
D	D	E
E	B	C
F	C	G
G	F	E



⌘ √ abab, ……

⌘ ⊗ abba, …..

LFA: C3 – LIMBAJE REGULATE

Definitie 9

L(A) = limbajul recunoscut de AFD A

= { w $\in\Sigma^*$| δ(s,w)=q$\in$F }

= multimea secventelor peste Σ care aduc A intr-o stare finala

Observatie 10: acceptare vs. recunoastere

Fie AFD A_3= (Q, Σ, δ, s, $\emptyset$)

=> L(A_3) = $\emptyset$

i.e.: automatul nu **accepta** nicio secventa peste alfabetul sau de intrare – pentru ca nu are nicio stare finala F = $\emptyset \subseteq$ Q

dar **recunoaste** totusi un limbaj, şi anume limbajul vid!!.

LFA : C3 – LIMBAJE REGULATE

Cum proiectam un AFD?

Ideea metodica a proiectarii unui AFD:

"proiectantul devine un AFD"

Sa pp. ca primim un limbaj L si vrem sa proiectam AFD A care sa il recunoasca

Metoda de mai sus presupune ca proiectantul primeste o fraza ***f*** si i se cere sa spuna daca ***f*** $\in$L sau ***f*** $\notin$L

Ca un AFD, proiectantul "vede" simbolurile din fraza unul cate unul si – dupa citirea fiecarui smb – trebuie sa fie in stare sa spuna daca fraza citita **pana in acel moment** $\in$L sau $\notin$L

i.e.: proiectantul – la fel ca un AFD –

- ✓ are o memorie limitata
- ✓ nu stie cand ajunge la "capatul" frazei si
- ✓ trebuie sa aiba mereu un raspuns pregatit. ->

LFA : C3 – LIMBAJE REGULATE

Cum proiectam un AFD? (cont.)

Elementul esential in aceasta strategie:

CE INFORMATIE DESPRE FRAZA CITITA TREBUIE MEMORATA DE AFD?

De ce nu memoram toata fraza citita?

1. limbajul: infinit :

   automatul: numar finit de stari, deci memorie finita

2. nu este necesar :

   e suficient sa memoram ”informatia cruciala”

CARE ESTE INSA INFORMATIA CRUCIALA ?!?

aceasta depinde de limbajul respectiv =>

stabilirea ei: elementul dificil si creativ in proiectarea unui AFD.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

<u>Exemplu 11</u>

Fie $\Sigma = \{0,1\}$ si $L = \{w \in \{0,1\}^{+} \mid \#_1(w)=2k+1, k \in N\}$

Fie secventa de intrare

01011100100000111100011111000011100011111110000111100001:

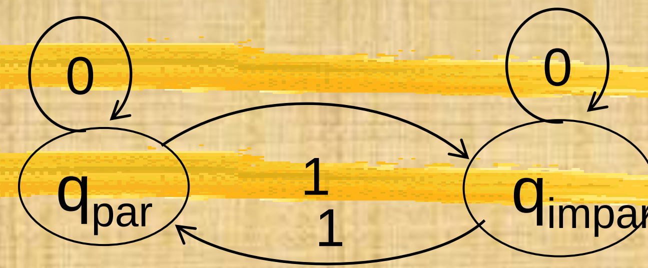
Pas 1: stabilim informatia de memorat:

- nr de smb 1 citite pana la momentul crt este sau nu impar?
- la citirea unui nou smb:
  - daca acesta este 0 -> raspunsul trebuie lasat neschimbat;
  - daca acesta este 1 -> raspunsul trebuie comutat

Pas 2: reprezentam informatia de memorat ca o lista finita de posibilitati:

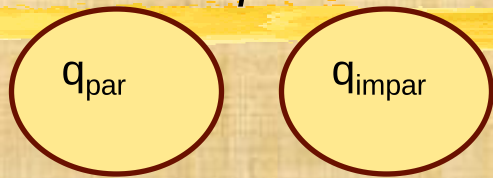
- numar par de simboluri 1, pana acum;
- numar impar de simboluri 1, pana acum. ->

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



Pas 3: asignam fiecarei posibilitati cate o stare:

- q_{par}
- q_{impar} . ->

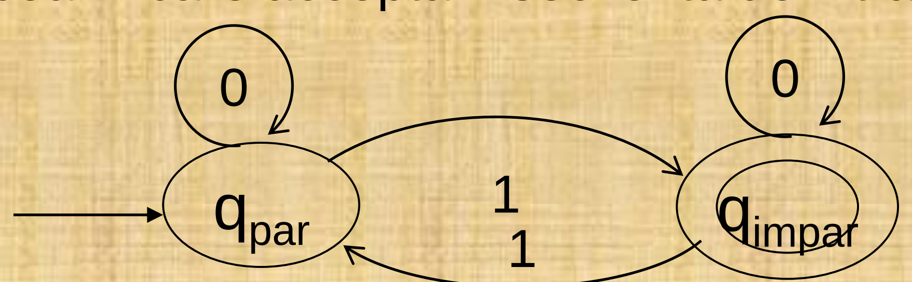


Pas 4: definim tranzitiile, examinand modul in care se trece de la o posibilitate la alta la citirea fiecarui tip de simbol din Σ:

- la citirea unui simbol 1 se trece din orice stare in cealalta stare,
- la citirea unui simbol 0 se ramane in aceeasi stare,

Pas 5: stabilirea starii initiale si a multimii starilor finale, examinand modul in care se intra/se paraseste fiecare posibilitate:

- initial se citesc 0 simboluri -> AFD porneste din starea q_{par}.
- starea finala trebuie sa fie cea in care acceptam secventa de intrare => starea finala este q_{impar}.



$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

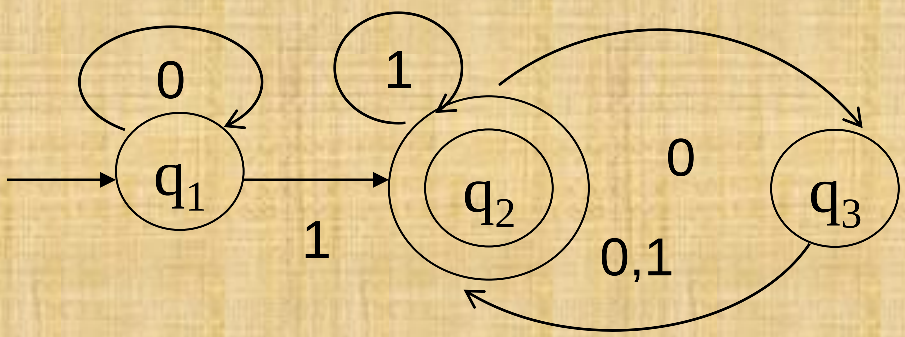
Definitie 12: Calculul efectuat de un AFD

Fie A = (Q, Σ, δ, s, F) un AFD

$w = w_1 w_2 \ldots w_n : \forall\, 1 \le i \le n: w_i \in \Sigma$

Atunci, A accepta w ddaca $\exists\, r_o, r_1, \ldots, r_n \in Q$ astfel încât:

1. $r_o = s$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}, \forall\, 0 \le i \le n-1$,
3. $r_n \in F$;



Exemplu 13

Fie automatul de mai sus;

el accepta secventa 010100 pentru ca exista secventa de stari

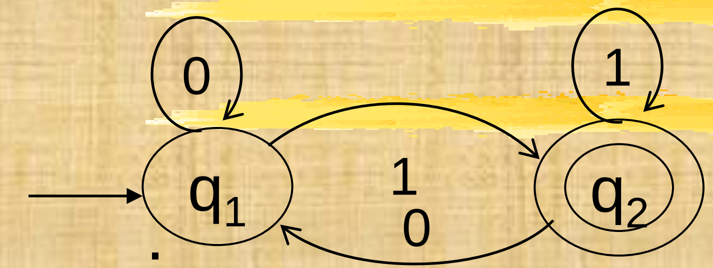
$q_1, q_1, q_2, q_3, q_2, q_3, q_2$, care indeplineste toate cele 3 conditii:

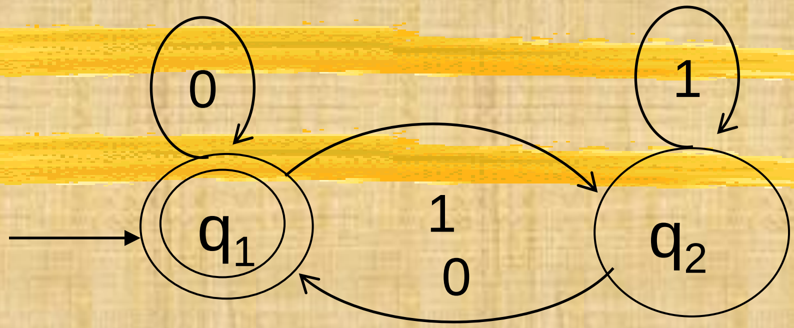
$\delta(q_1, 0) = q_1, \delta(q_1, 1) = q_2, \delta(q_2, 0) = q_3, \delta(q_3, 1) = q_2, \delta(q_2, 0) = q_3, \delta(q_3, 0) = q_2$

Definitie 14

$L \in \mathcal{L}_3 \Leftrightarrow \exists\, A \in \mathcal{A}$ astfel incat L(A) = L .

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE





<u>Exemple 15</u>

1. $L_1 = \{w \in \{0,1\}^* \mid w = w_1 w_2 \ldots w_k 1,\ k \in N\}$

Putem verifica pentru:

√ 10101, 0001, ……

⊗ 0000, 1010, ….. =>

=> $A_1 = (\{q_1, q_2\}, \{0,1\}, \delta, q_1, \{q_2\})$,

δ	0	1
q_1	q_1	q_2
q_2	q_1	q_2

Fie acum:

√ 0000, 1010, …..

⊗ 10101, 0001, …… =>

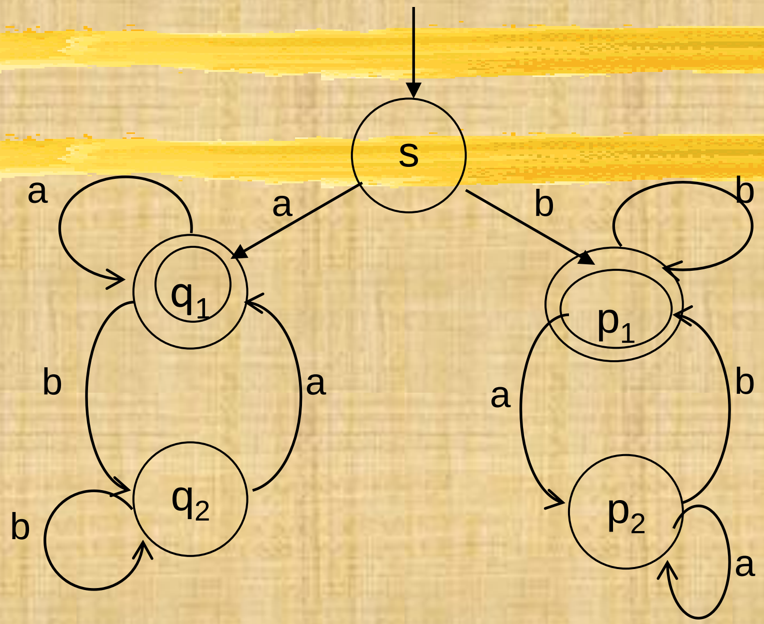
2. $L_2 = \{w \in \{0,1\}^* \mid w = w_1 w_2 \ldots w_k 0,\ k \in N\}$

δ	0	1
q_1	q_1	q_2
q_2	q_1	q_2

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

.

3. Fie A_3:



Observam simetria =>

simulam un calcul (pentru ramura stanga):

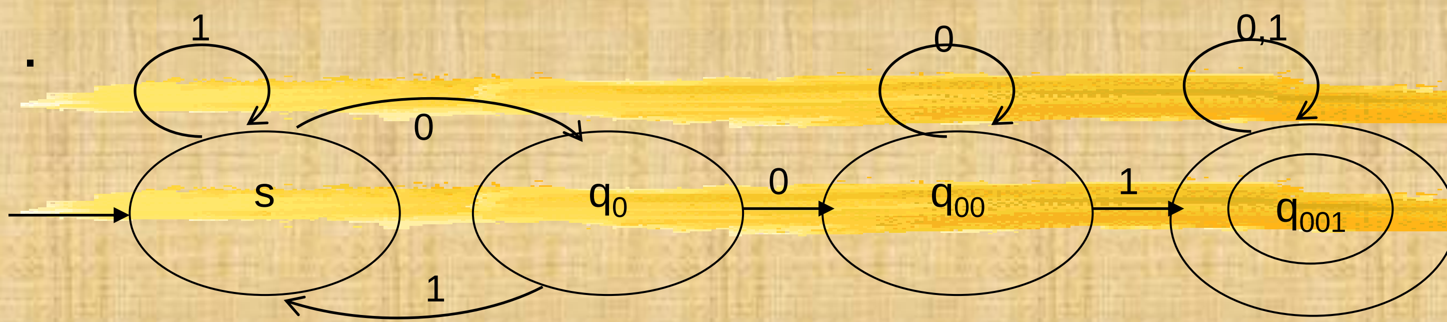
aa…abb…baa…abb…baa..a….

=> a^n, $a^nb^ma^k$, $a^nb^ma^kb^ua^v$ => $a^{n1}b^{m1}a^{k1}a^{n2}b^{m2}a^{k2}\ldots a^{nx}b^{mx}a^{kx}$

=> L_3= {w$\in${a,b}* | w incepe şi se termina cu a} $\cup$

$\cup$ {w$\in${a,b}* | w incepe şi se termina cu b} .

LFA : C3 – LIMBAJE REGULATE



4. Vrem sa construim un AFD care sa recunoasca toate cuvintele binare care contin subcuvantul 001:

$L_4 = \{w \in \{0,1\}^* \mid \exists\ x,y \in \{0,1\}^* \text{ a.i. } w=x001y\}$

=> trecem peste prefixele formate numai din 1 (pastram starea initiala, s)

cand gasim un 0 semnalam cu o noua stare q_0

daca intalnim 0 din nou semnalam cu o noua stare, q_{00}

1 reluam cautarea intorcandu-ne in s

daca intalnim 1 semnalam cu o noua stare q_{001} şi o declaram finala (nu conteaza cate simboluri 0 sau 1 mai intalnim in continuare, acceptam pt ca am gasit deja subcuvantul cautat)

0 ramanem pe loc in asteptarea unui 1 (daca il gasim trecem in starea finala, daca nu, AFD nu accepta secv.) .

LFA: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 16

Fie A, B $\subseteq \Sigma^*$; definim urmatoarele operatii:

- ✓ **reuniunea** : $A \cup B = \{ \omega \in \Sigma^* \mid \omega \in A \text{ sau } \omega \in B \}$,
- ✓ **concatenarea** : $A \circ B = \{\omega\upsilon \in \Sigma^* \mid \omega \in A \text{ şi } \upsilon \in B\}$,
- ✓ **operatia** star : $A^* = \{\omega_1\omega_2 \ldots\omega_n \in \Sigma^* \mid \omega_k \in A,\ \forall 1 \leq k \leq n,\ n \in \mathcal{N}\}$;

Observatii 17

- ❑ Cele 3 operatii: **operatii regulate**
  - ✓ specifice clasei limbajelor formale,
  - ✓ utilizate pentru a studia proprietatile limbajelor (regulate);
- ❑ Operatia star
  - ✓ este singura unara,
  - ✓ $\forall A \subseteq \Sigma^*$: A^* contine ε (n>0 sau n=0!);

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Exemplu 18

Fie Σ = {a,b, c,…,z}, A = {telefon, mobil, fax}, B = {fix, mobil}

=> A $\cup$ B = {telefon, mobil, fax, fix}

A $\circ$ B = {telefonfix,telefonmobil, mobilfix,mobilmobil, faxfix, faxmobil}

B* = {ε, fix, mobil, fixfix, fixmobil, mobilfix, mobilmobil, fixfixfix, fixfixmobil, fixmobilfix, fixmobilmobil, fixfixfixfix, …. }.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Teorema 19

$\mathcal{L}_3$ este inchisa la reuniune (ie.: $L_1, L_2 \in \mathcal{L}_3$ => $L = L_1 \cup L_2 \in \mathcal{L}_3$)

Demonstratie (constructiva)

Ideea dem.:

ip.: $L_1, L_2 \in \mathcal{L}_3$ => $\exists\ A_i = (Q_i, \Sigma_i, \delta_i, s_i, F_i), \in \mathcal{A}$ a. i. $L_i = L(A_i)$, i=1,2

cum $L = L_1 \cup L_2$ ->

trebuie sa construim un AFD A care sa accepte oridecateori A_1, respectiv A_2 accepta

-> A trebuie sa se bazeze pe A_1, A_2: simuleaza intai A_1 şi, daca el nu accepta, simuleaza A_2

-> **eroare**: daca A l-a simulat intai pe A_1 şi el nu a acceptat, A nu poate relua secventa pt A_2

-> alta strategie: A simuleaza **simultan,** pe fiecare simbol din secventa de intrare, pe A_1 şi A_2

-> **dificultate**: trebuie sa memoram starile prin care trece A in timpul celor 2 simulari;

se poate face cu memoria finita a unor AFD?!?

DA, pt ca avem de memorat tot un numar finit de perechi de stari: $|Q_1| x |Q_2|$!!

=> aceste perechi de stari vor constitui multimea de stari ale lui A

starile finale de acceptare ale A sunt acele perechi de stari din A_1 respectiv A_2 ,care

contin cel putin o stare finala de acceptare (pentru A_1, respectiv A_2).

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Demonstratie formala:

Construim $A = (Q, \Sigma, \delta, s, F)$, care recunoaste $L = L_1 \cup L_2 = L(A_1) \cup L(A_2)$,

unde $A_1 = (Q_1, \Sigma_1, \delta_1, s_1, F_1)$, $A_2 = (Q_2, \Sigma_2, \delta_2, s_2, F_2)$, astfel:

$Q = \{(q_1,q_2) \mid q_1 \in Q_1$ şi $q_2 \in Q_2\} = Q_1 \times Q_2$

$\Sigma = \Sigma_1 \cup \Sigma_2$

$\delta : Q \times \Sigma \to Q, \quad \delta((q_1,q_2), a) = (\delta_1(q_1,a), \delta_2(q_2,a))$

$s = (s_1,s_2)$

$F = \{(q_1,q_2) \mid q_1 \in F_1$ sau $q_2 \in F_2\} = (F_1 \times Q_2) \cup (Q_1 \times F_2)$;

Mai trebuie: $L(A_1) \cup L(A_2) \subseteq L(A)$ şi $L(A) \subseteq L(A_1) \cup L(A_2)$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Demonstratie formala (cont.):

$L(A_1) \cup L(A_2) \subseteq L(A)$: evident, cf Def. 12: calculul efectuat de un AFD:

Fie $A = (Q, \Sigma, \delta, s, F)$ un AFD şi $w = w_1 w_2 \dots w_n$: $\forall\ 1 \le i \le n$: $w_i \in \Sigma$

Atunci, A accepta w ddaca $\exists\ r_o, r_1, \dots, r_n \in Q$ astfel incat:

(1.) $r_o = s$, **(2.)** $\delta(r_i, w_{i+1}) = r_{i+1}$, $\forall\ 0 \le i \le n-1$, **(3.)** $r_n \in F$;

Fie $w = w_1 w_2 \dots w_n \in L(A_1)$ =>

$\exists\ r_o, r_1, \dots, r_n \in Q_1$ a.i. $r_o = s_1$, $\delta(r_i, w_{i+1}) = r_{i+1}$, $\forall 0 \le i \le n-1$, $r_n \in F_1$ =>

oricare ar fi starile de pe pozitia a 2a din perechile (r, q), $r \in Q_1$ şi $q \in Q_2$, ajungem in starea finala

$(r_n, q_n) \in (F_1 \times Q_2) \subseteq (F_1 \times Q_2) \cup (Q_1 \times F_2) = F$ **=>** $w \in L(A)$;

Fie $w = w_1 w_2 \dots w_n \in L(A_2)$: analog;

Reciproc: fie $w = w_1 w_2 \dots w_n \in L(A)$ =>

$\exists\ (r_1, q_1), (r_2, q_2), \dots, (r_n, q_n) \in Q = Q_1 \times Q_2$ a.i.

$(r_1, q_1) = (s_1, s_2)$,

$\delta((r_i, q_i), w_{i+1}) = (\delta_1(r_i, w_{i+1}), \delta_2(q_i, w_{i+1})) = (r_{i+1}, q_{i+1})$, $\forall 0 \le i \le n-1$,

$(r_n, q_n) \in F$;

Dar $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$ => distingem cazurile:

$(r_n, q_n) \in (F_1 \times Q_2)$ => $r_n \in F_1$ => $w \in L(A_1)$,

$(r_n, q_n) \in (Q_1 \times F_2)$ => $q_n \in F_2$ => $w \in L(A_2)$,

$(r_n, q_n) \in (F_1 \times Q_2) \cap (Q_1 \times F_2)$) => $(r_n, q_n) \in (F_1 \times F_2)$ => $r_n \in F_1, q_n \in F_2$ =>$w \in L(A_1) \cap L(A_2)$

=> $w \in L(A_1) \cup L(A_2)$ q.e.d.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Propozitie 20

$\mathcal{L}_3$ este inchisa la intersectie, diferenta şi complementara (ie.: L_1, $L_2 \in \mathcal{L}_3$ => $(L_1 \cap L_2)$, $(L_1 - L_2)$, $(\Sigma - L_1) \in \mathcal{L}_3$)

Demonstratie

Acelasi rationament (constructie), dar:

AFD care recunoaste $L = L_1 \cap L_2$ are ca multime de stari finale, multimea:

$F = \{(q_1,q_2) \mid q_1 \in F_1$ ***şi*** $q_2 \in F_2\} = F_1 \times F_2$

AFD care recunoaste $L = L_1 - L_2$ are ca multime de stari finale, multimea:

$F = \{(q_1,q_2) \mid q_1 \in F_1$ şi $q_2 \notin F_2\} = F_1 \times (Q_2 - F_2)$

AFD care recunoaste $\Sigma - L_1$ are ca multime de stari finale, multimea:

$F = \{ q_1 \mid q_1 \in (Q_1 - F_1) \}$ q.e.d.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Observatii 21

- ✓ Intersectia, diferenta şi complementara NU sunt operatii regulate!
- ✓ AFD care recunosc $L_1 \cup L_2$, respectiv $L_1 \cap L_2$ au $|Q_1|$ x $|Q_2|$ stari.

Propozitie 22

Fie $L_1 \in \mathcal{L}_3$ şi $L_2 \subseteq \Sigma^*$ oarecare

=> catul la dreapta $L_1 / L_2 = \{w \in \Sigma^* | \exists y \in L_2 : wy \in L_1\} \in \mathcal{L}_3$

Demonstratie

Fie $A=(Q, \Sigma, \delta, s, F)$ a.i. $L(A)=L_1$;

definim $A'=(Q, \Sigma, \delta, s, F')$ astfel: $F'= \{q \in Q \mid \exists y \in L_2 : \delta(q,y) \in F\}$

=> $\delta(s,w) \in F'$ ddaca $\exists y \in L_2 : wy \in L_1$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

$s : \Sigma \to \mathcal{P}(\Psi^*)$

$s(\varepsilon) = \{\varepsilon\}$,

$s(a\beta) = s(a)s(\beta)$, $\forall a \in \Sigma$, $\forall \beta \in \Sigma^*$

card (s(a)) = 1, $\forall a \in \Sigma$ => **[omo]morfism**:

Fie un limbaj $L \subseteq \Sigma^*$; atunci definim prin: $s(L) = \bigcup_{\alpha \in L} s(\alpha)$

limbajul obtinut din L prin **substitutie canonica**

fie $s:\{a,b\} \to \mathcal{P}(\{0,1,x\}^*)$ $s(a)= \{0x\}$, $s(b)=\{x11\}$

daca $L=\{a,b, aa, ab, ba, bb\}$ =>

$s(L) = \{0x, x11, 0x0x, 0xx11, x110x, x11x11\}$.

Propozitie 23

Fie $L \in \mathcal{L}_3$ si h: $\Sigma^* \to \Psi^*$ un morfism => $h^{-1}(L) \in \mathcal{L}_3$

Demonstratie

Fie $A=(Q, \Sigma, \delta, s, F)$ a.i. $L(A)=L\subseteq\Sigma^*$

definim $A'=(Q, \Psi, \delta', s, F)$ astfel: $\delta'(q,a)=\delta(q,h(a))$;

se dem. prin inductie asupra $w \in L$ ca $\delta'(s,w) = \delta(s,h(w))$

(i.e. A' accepta w ddaca A accepta h(w)).

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

<u>Observatie 24</u>

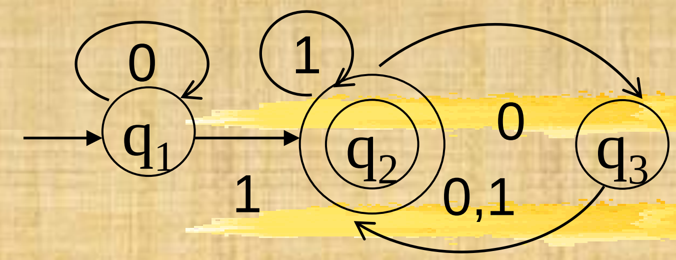
Incercam sa folosim pentru demonstrarea inchiderii $\mathcal{L}_3$ la concatenare (şi operatia star) aceeasi tehnica utilizata pentru reuniune (şi intersectie),

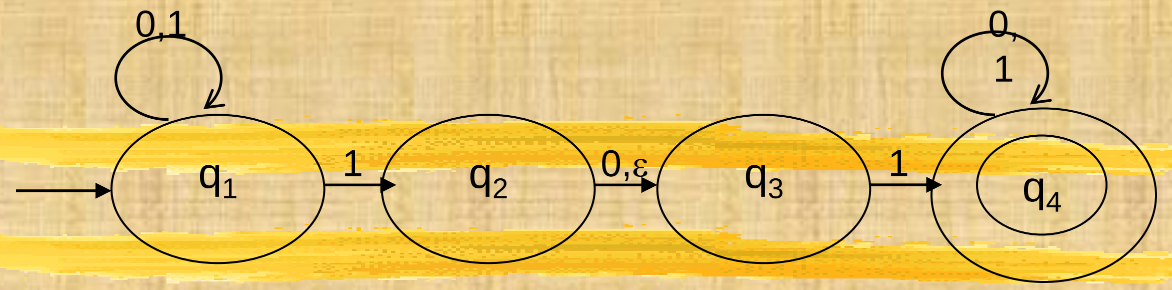
-> dificultate: AFD **A** care trebuie sa recunoasca $A_1 \cdot A_2$ (deci sa accepte o secventa de tipul $w=w_1w_2$) trebuie sa accepte numai cand A_1, respectiv A_2 accepta w_1, respectiv w_2 **simultan**,

ori, **A** nu stie unde trebuie sa “sparga” w pentru a obtine w_1 şi w_2 şi a incepe simularea!

=> trebuie introdusa o noua tehnica: nedeterminismul !

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE





Conceptual, diferentele dintre un AFD si un AFN sunt:

1) $\forall\ q \in Q$: $\forall\ a \in \Sigma$:

in AFD pleaca o singura sageata pentru fiecare simbol de intrare,

in AFN pleaca

1,

0, sau

mai multe sageti, etichetate cu diferite smb. de intrare;

2) Sagetile sunt etichetate:

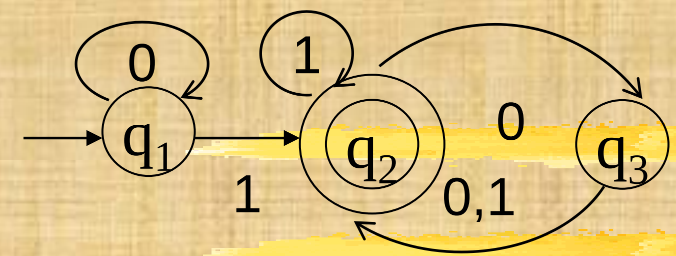
in AFD: cu simboluri din Σ,

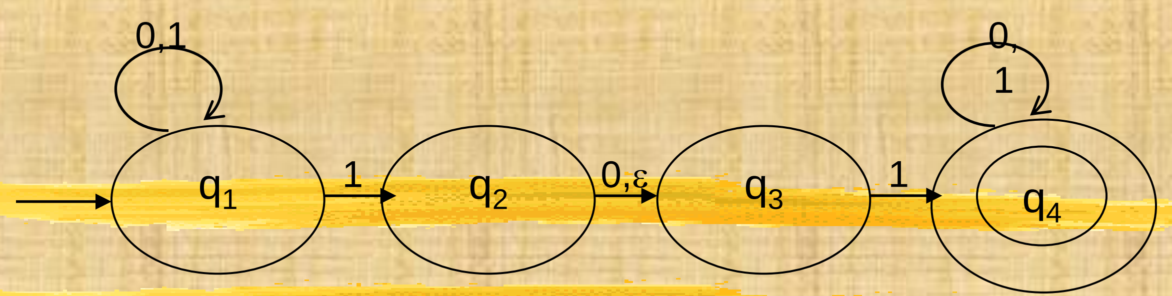
in AFN:

cu simboluri din Σ,

cu simboluri din Σ si/sau cu simbolul vid, **ε**.

3) Modul de calcul ->





Conceptual, diferentele dintre un AFD si un AFN sunt (cont.):

3) Modul de calcul

⌘ Pp. ca AFN se afla in starea $q_i \in Q$ si citeste simbolul a $\in \Sigma \Rightarrow$

AFN SE MULTIPLICA intr-un numar de exemplare **n**, egal cu numarul de stari $q_{i1}, q_{i2}, \ldots q_{in}$ in care poate trece si CONTINUA CALCULUL IN PARALEL, pentru fiecare dintre posibilitati,

⌘ Daca, in continuare, dintr-una dintre starile in care a trecut, fie ea $q_{ik} \in Q$, AFN poate trece in mai multe stari $q_{ik1}, q_{ik2}, \ldots q_{ikm} \Rightarrow$

acel exemplar se multiplica la randul lui in **m** exemplare etc.,

⌘ Daca insa noul simbol citit cand AFN se afla in starea q_i nu apare pe niciuna dintre sagetile care ies din starea $q_{ik} \Rightarrow$

acel exemplar “moare”, impreuna cu toata ramura de calcul respectiva;

⌘ Pentru ca secventa de intrare sa fie recunoscuta de AFN este suficient ca o singura ramura de calcul (un singur exemplar din AFN) sa ajunga intr-o stare finala;

⌘ Daca din starea $q_i \in Q$ pleaca o sageata etichetata cu simbolul **ε** $\Rightarrow$

AFN se multiplica de asemenea intr-un numar de exemplare egal cu numarul de sageti etichetate cu **ε**, daca exista mai multe astfel de sageti, plus un exemplar care “ramane pe loc” in aceeasi stare $q_i \in Q$. Apoi se continua ca mai sus.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 25: Automat finit nedeterminist

AFN = (Q, Σ, δ, s, F), unde:

Q = multime finita, nevida (stari),

Σ = multime finita, nevida, numita alfabet de intrare (simboluri),

$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$, numita functia de tranzitie,

$s \in Q$, numita starea initiala,

$F \subseteq Q$ numita multimea starilor finale (de acceptare);

Notatii 26

$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$,

$\mathcal{AN} = \{ N \mid N$ este un automat finit nedeterminist $\}$

Observatie 27

Pentru a descrie calculul efectuat de un AFN extindem functia δ printr-o definitie inductiva astfel:

$\delta : Q \times (\Sigma \cup \{\varepsilon\})^* \rightarrow \mathcal{P}(Q) : \quad \delta(s, \varepsilon) = \{s\}$

$\delta \qquad \delta(s,wa) = \cup_{q \in \delta(s,w)} \delta(q,a), \ \forall\, w \in \Sigma^*, a \in \Sigma$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 28: Calculul efectuat de un AFN

Fie AN = (Q, Σ, δ, s, F) un AFN

$w = w_1 w_2 \ldots w_n : \forall\ 1 \leq i \leq n: w_i \in \Sigma_\varepsilon$

Atunci, AN accepta w ddaca $\exists\ r_o, r_1, \ldots, r_n \in Q$ astfel incat:

1. $r_o = s$,
2. $r_{i+1} \in \delta(r_i, w_{i+1}), \forall\ 0 \leq i \leq n-1$,
3. $r_n \in F$

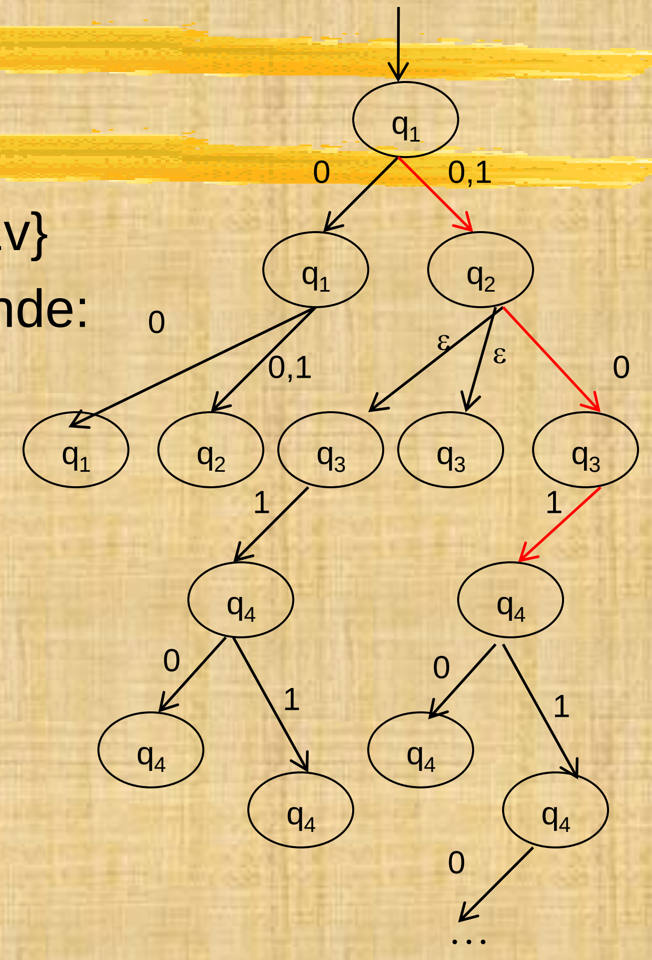
Definitie 29

L(N) = limbajul recunoscut de AFN N

= **{ w $\in\Sigma^*$| δ(s,w) $\cap$ F$\neq\emptyset$ }**

= multimea secventelor peste Σ care aduc N intr-o stare finala.

$\mathcal{FA}$: C3 – LIMBAJE REGULATE



Exemplu 30

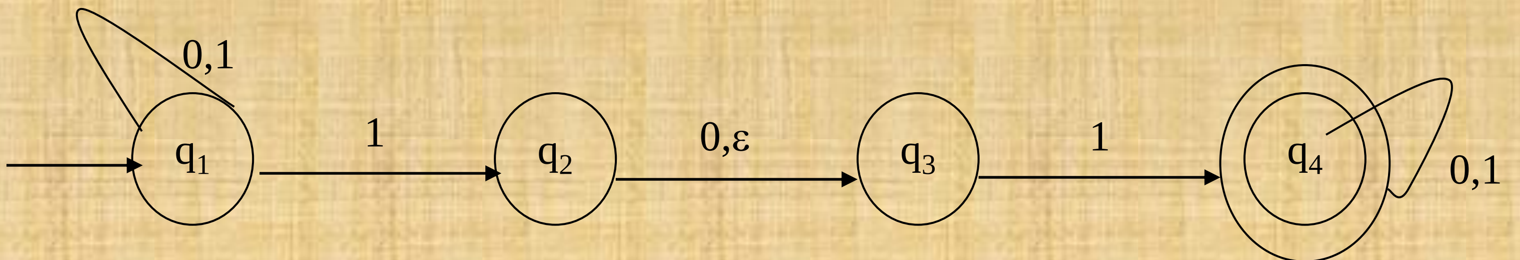
AFN care recunoaste limbajul:

$L_2 = \{ w \in \Sigma_\varepsilon^* \mid \exists\ u,v \in \Sigma_\varepsilon^*: w=u101v$ sau $w=u11v\}$

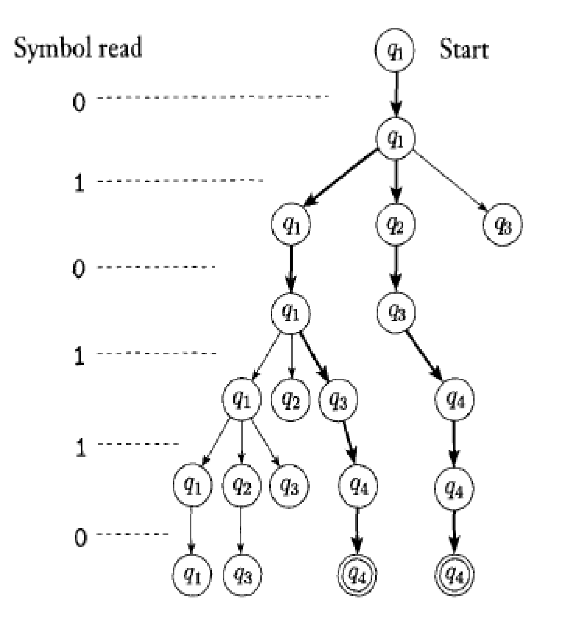
$\Rightarrow AFN_1 = (\{q_1, q_2, q_3, q_4\}, \{\varepsilon, 0, 1\}, \delta, q_1, \{q_4\})$, unde:

δ	ε	0	1
q_1	Φ	$\{q_1\}$	$\{q_1, q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$	Φ
q_3	Φ	Φ	$\{q_4\}$
q_4	Φ	$\{q_4\}$	$\{q_4\}$

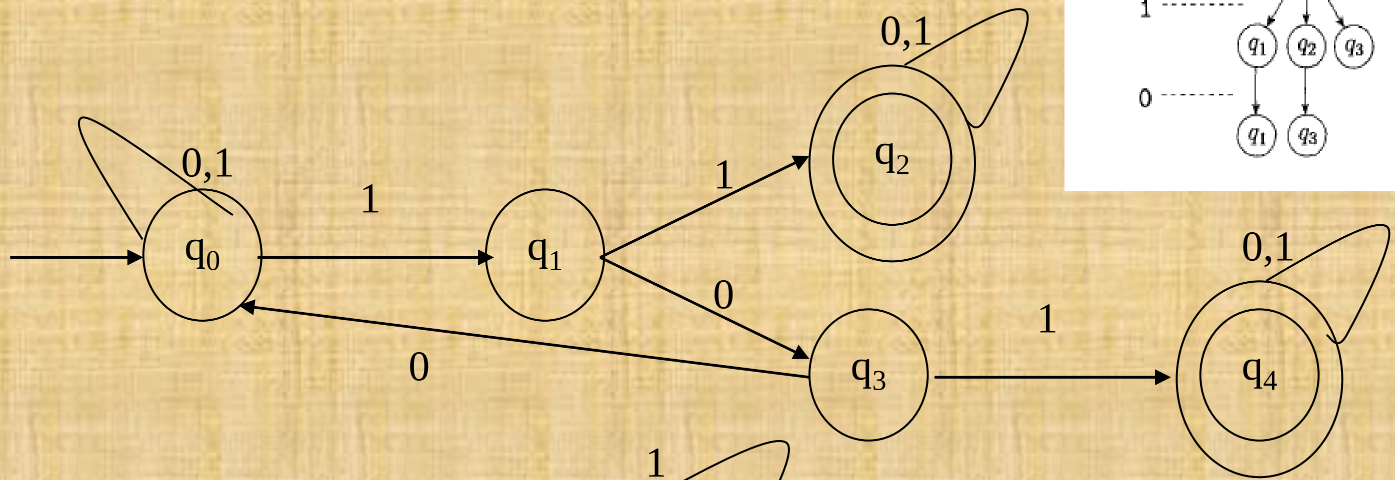
(a)



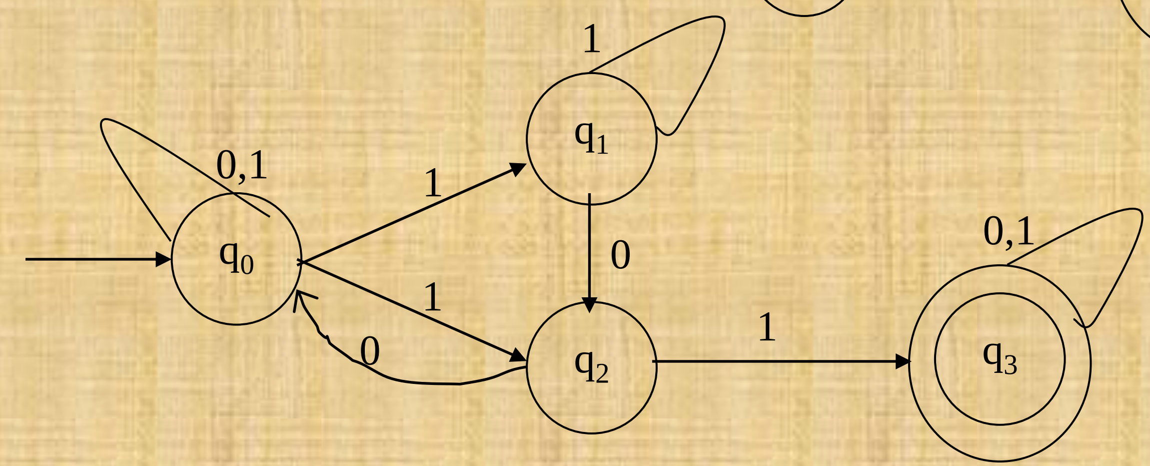
$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



(b)



(c)



Teorema 31

AFN $\Leftrightarrow$ AFD

Demonstratie

"$\Leftarrow$"

Evident: orice AFD se converteste intr-un AFN in care fiecare multime de stari in care poate trece automatul consta dintr-o singura stare;

"$\Rightarrow$"

Fie AN=(Q, Σ, δ, q_0, F) $\in \mathcal{AN}$;

el se poate converti intr-un AFD, A=(Q', Σ', δ', q'_0, F') $\in \mathcal{A}$, astfel:

Q' = $\mathcal{P}$(Q), $\Sigma' = \Sigma$, $q_0' = \{q_0\}$,

F' = { R$\in$Q'=$\mathcal{P}$(Q) | R contine cel putin o stare finala a lui AFN },

$\forall R \in Q' = \mathcal{P}(Q)$ si $\forall a \in \Sigma' = \Sigma$: $\delta'(R,a) = \{q \in Q \mid \exists r \in R: q \in \delta(r,a)\} = \cup_{r \in R} \delta(r,a)$

Daca $\exists$ tranzitii etichetate cu ε, mai definim

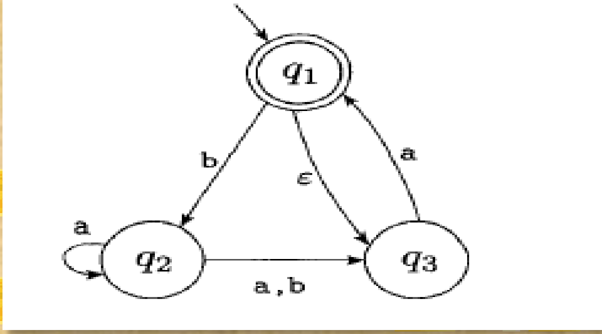
Vid(R) = R $\cup$ {q$\in$Q | q poate fi atinsa din R cu ajutorul a 1 sau mai multe tranzitii etichetate cu ε} $\Rightarrow$ pentru a include starile in care se trece cu tranzitii etichetate cu ε redefinim:

$\delta'(R,a) = \{q \in Q \mid \exists r \in R: q \in Vid(\delta(r,a))\} = \cup_{r \in R} Vid(\delta(r,a))$

$q_0' = Vid(\{q_0\})$ q.e.d.

Corolar 32: $\forall L \subseteq \Sigma^*$: $L \in \mathcal{L}_3$: $\Leftrightarrow \exists AN \in \mathcal{AN}$: L(AN) = L.

LFA : C3 – LIMBAJE REGULATE



Fie AFN=(Q, Σ, δ, s, F) -> AFD=(Q', Σ', δ', s', F') astfel:

Q' = $\mathcal{P}$(Q), Σ' = Σ, q_0'={q_0},

F' = { R∈Q'=$\mathcal{P}$(Q) | R contine cel putin o stare finala a lui AFN },

∀R∈Q' si a ∈ Σ': δ'(R,a) = { q ∈Q | ∃r∈R: q∈ δ(r,a) }= $\cup_{r\in R}$ δ(r,a).

Daca ∃ tranzitii etichetate cu ε, mai definim

Vid(R) = R ∪ {q∈Q | q poate fi atinsa din R cu ajutorul a 1 sau m. multe tranzitii etichet. cu ε}

⇒ δ'(R,a) = {q ∈Q | ∃r∈R: q∈Vid(δ(r,a))}=$\cup_{r\in R}$Vid(δ(r,a))

q_0'=Vid({q_0})

Aplicatie 33

Fie AFN de mai sus (care accepta secvente de forma ε, a, baba, baa etc. (şi nu accepta b, bb, babba etc.) => NA = ({1,2,3}, {a,b}, δ, 1, {1});

construim AFD A, echivalent, cf. Teoremei 23:

Q' = {∅, {1}, {2}, {3}, {1,2}, {1,3}, {2,3}, {1,2,3}}; Σ' = {a,b}

s' = {1} ∪ Vid({1})= {1} ∪ {3} = {1, 3}

F' = submultimile lui Q care contin cel putin o stare de acceptare =

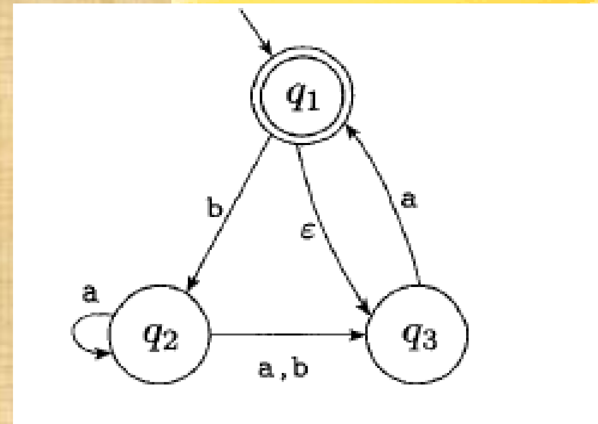
= {{1}, {1,2}, {1,3}, {1,2,3}}

δ' : δ'(∅,a)= ∅, δ'(∅,b)= ∅

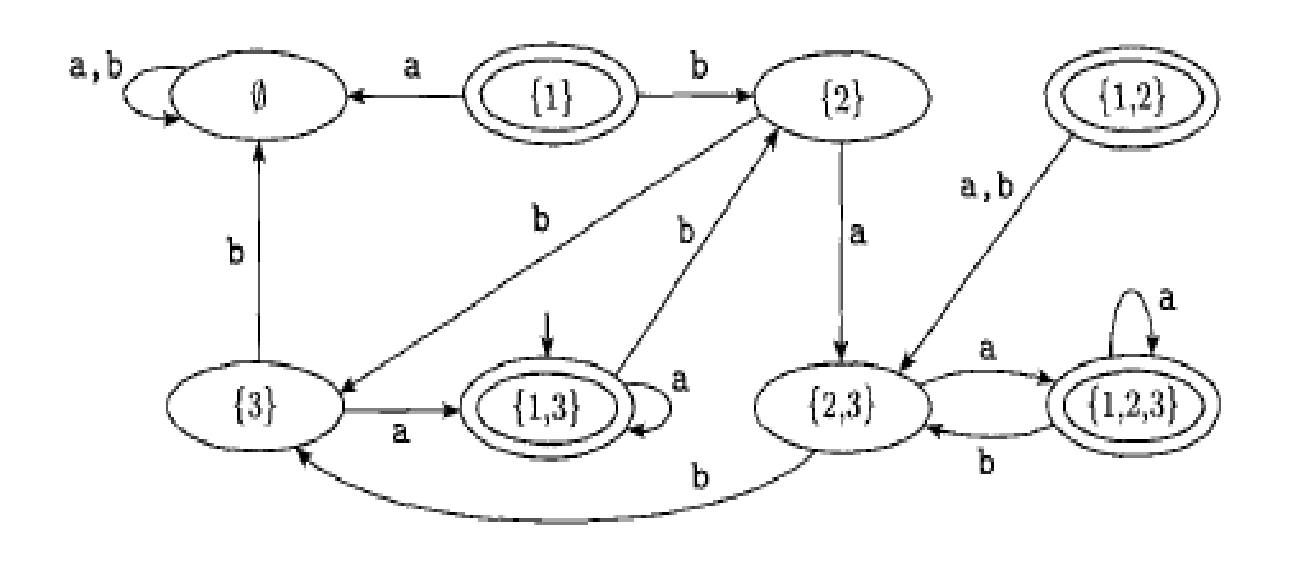
δ'({1},a)=∅, δ'({1},b)={2}, δ'({2},a)={2,3}, δ'({2},b)={3}, δ'({3},a)={1,3},

δ'({3},b)=∅,

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



<=>



$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Teorema 34

$\mathcal{L}_3$ e inchisa la reuniune

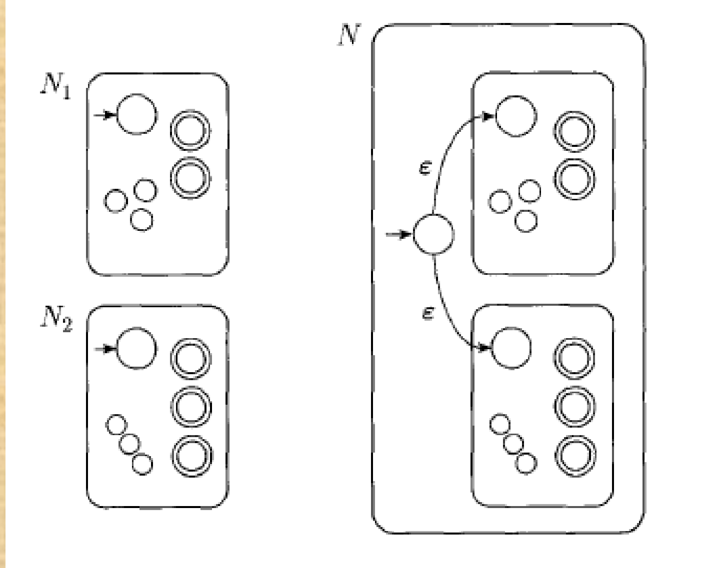
Demonstratie

Fie $N_1=(Q_1, \Sigma_1, \delta_1, s_1, F_1)$, $L(N_1) = L_1$ şi
$N_2=(Q_2, \Sigma_2, \delta_2, s_2, F_2)$, $L(N_2) = L_2$

Construim N care recunoaste $L_1 \cup L_2$ folosind aceeasi idee ca in dem. ant. dar cu AFN:

Avantaj: noul AFN, N, poate ghici care dintre N_1 sau N_2 poate accepta cuvantul de intrare astfel:

N are o noua stare initiala din care ajunge in s_1 sau s_2 cu ajutorul unor tranzitii etichetate cu ε.



$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

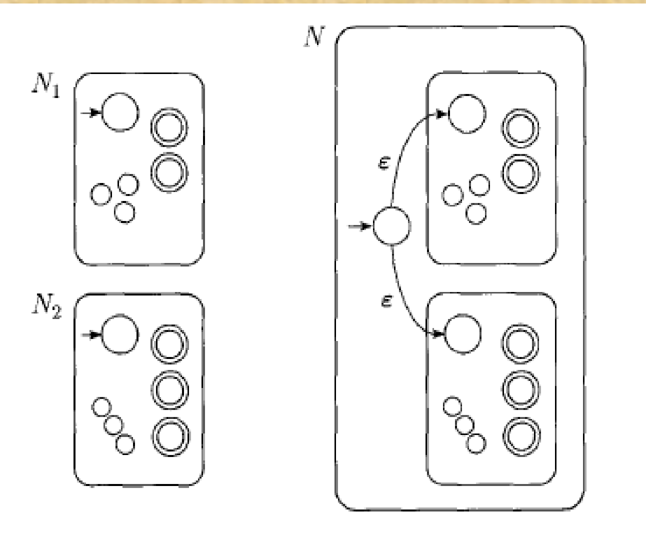
Formal:

Construim N = (Q, Σ, δ, s, F) care va recunoaste $L_1 \cup L_2$ astfel:

$Q = \{s\} \cup Q_1 \cup Q_2$

s = s

$F = F_1 \cup F_2$ (pt ca N accepta cand fie N_1 accepta, fie N_2 accepta, fie ambele)



$$\delta(q,a) = \begin{cases} \delta_1(q,a), & q \in Q_1 \\ \delta_2(q,a), & q \in Q_2 \\ \{s_1, s_2\}, & q = s, a = \varepsilon \\ \Theta, & q = s, a \neq \varepsilon \end{cases}$$

$L(N)=L(N_1) \cup L(N_2)$ (dubla incluziune).

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Teorema 35

$\mathcal{L}_3$ e inchisa la concatenare

Demonstratie

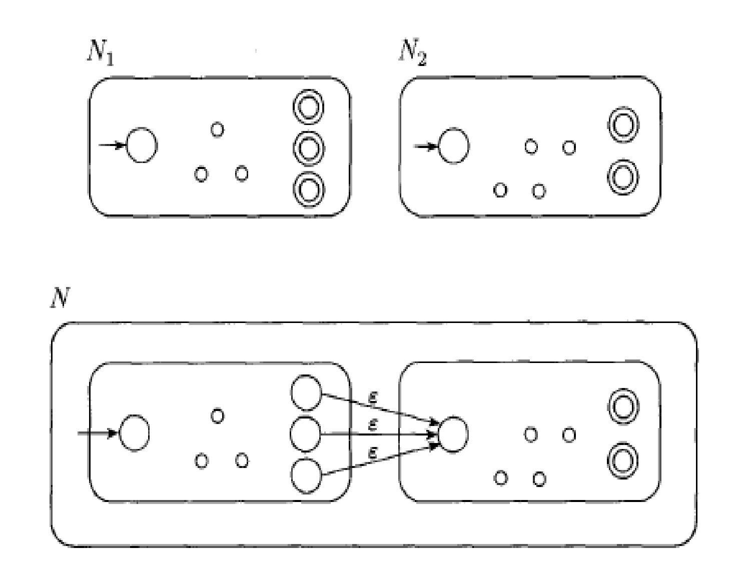
Fie $N_1=(Q_1, \Sigma_1, \delta_1, s_1, F_1)$, $L(N_1) = L_1$

şi $N_2=(Q_2, \Sigma_2, \delta_2, s_2, F_2)$, $L(N_2) = L_2$

Construim N care recunoaste $L_1 \circ L_2$ folosind aceeasi idee ca in dem. ant.

Diferenta: noul AFN, N, poate ghici unde se termina primul cuvant şi poate trece din orice stare finala a lui N_1 in starea initiala a lui N_2 printr-o tranzitie etichetata cu ε.

Starile finale ale lui N sunt numai starile finale ale lui N_2.



$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

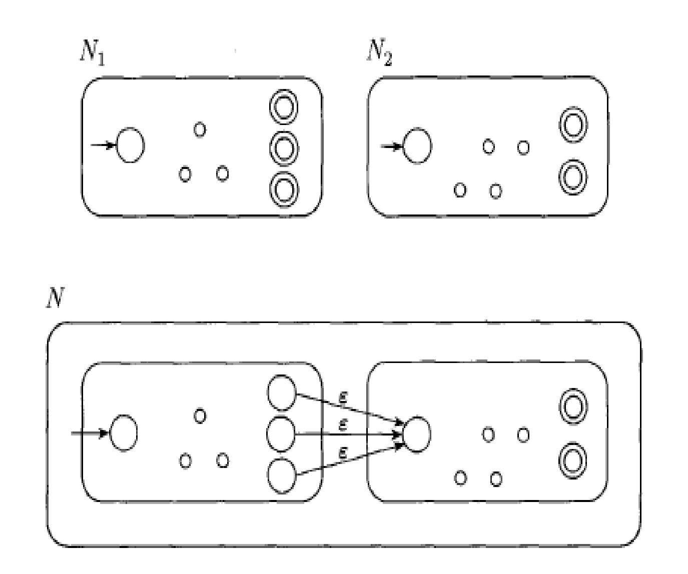
Formal:

Construim N = (Q, Σ, δ, s, F) care va recunoaste $L_1 \circ L_2$ astfel:

$Q = Q_1 \cup Q_2$

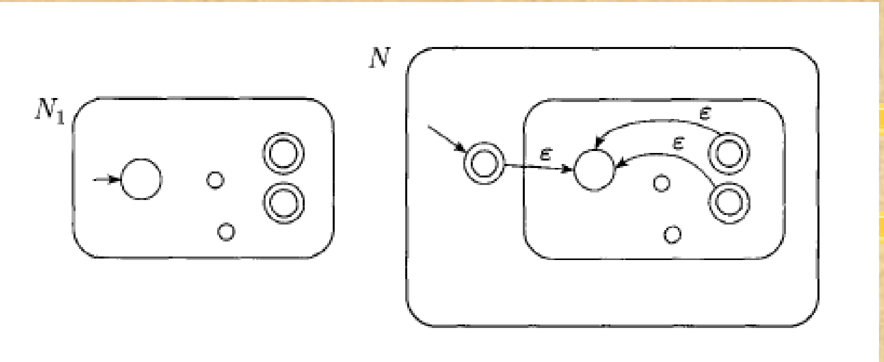
$s = s_1$

$F = F_2$ (pt ca N accepta doar cand N_2 accepta dupa ce N_1 a aceptat la randul sau)



$$\delta(q,a) = \begin{cases} \delta_1(q,a), & q \in Q_1 \setminus F_1 \\ \delta_1(q,a), & q \in F_1, a \neq \varepsilon \\ \delta_1(q,a) \bigcup \{s_2\}, & q \in F_1, a = \varepsilon \\ \delta_2(q,a), & q \in Q_2 \end{cases}.$$

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



Teorema 36

$\mathcal{L}_3$ e inchisa la operatia star

Demonstratie

Fie $N_1=(Q_1, \Sigma_1, \delta_1, s_1, F_1)$, $L(N_1) = L_;$ construim $N=(Q, \Sigma, \delta, s, F)$, $L(N) = L^*$;

Folosim aceeasi idee ca in cazul reuniunii şi concatenarii:

N va recunoaste secventa de intrare doar cand o va putea descompune in mai multe subsecvente (identice) pe care N_1 le va recunoaste (pe fiecare in parte)

N are aceleasi elemente ca N_1 dar contine in plus tranzitii etichetate cu ε care ii permit sa se intoarca din orice stare finala in starea initiala =>

cand N incheie calculul pentru o subsecventa pe care N_1 o accepta, N are optiunea de a reveni la starea initiala pentru a citi o noua subsecventa acceptabila de catre N_1;

Dificultate specifica: N trebuie sa accepte ε (L^* contine intotdeauna ε):

- adaugam o noua stare intiala, s, pentru N
- o defininim şi ca stare finala
- etichetam tranzitia dintre s şi s_1 cu ε (pentru a nu introduce secv. noi in L(N)).

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Formal:

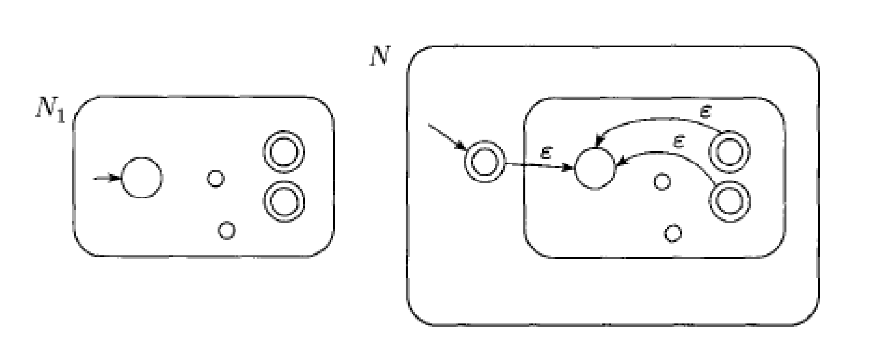
Construim N = (Q, Σ, δ, s, F) care va recunoaste L* astfel:

Q = $Q_1 \cup \{s\}$

s = s_1

F = $F_1 \cup \{s\}$ (pt ca N "continua" sa accepte subcuvinte doar dupa ce N_1 a aceptat la randul sau subcuvantul)

$$\delta(q,a) = \begin{cases} \delta_1(q,a), & q \in Q_1 \setminus F_1 \\ \delta_1(q,a), & q \in F_1, a \neq \varepsilon \\ \delta_1(q,a) \bigcup \{s_1\}, & q \in F_1, a = \varepsilon \\ \{s_1\}, & q = s, a = \varepsilon \\ \Theta, & q = s, a \neq \varepsilon \end{cases}.$$



LFA: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 37

Expresie regulata peste un alfabet Σ = **R** =

= o secventa $R \in \Sigma^*$ care satisface una dintre urmatoarele conditii:

1. $R=a$, $\forall\ a \in \Sigma$ (R reprezinta lb. $\{a\} \subseteq \Sigma^*$);
2. $R=\varepsilon$ (R reprezinta lb. $\{\varepsilon\} \subseteq \Sigma^*$);
3. $R=\varnothing$ (R reprezinta limbajul vid);
4. daca R_1 si R_2 sunt expresii regulate $\Rightarrow$
   - $(R_1 \cup R_2)$ este o expresie regulata,
   - $(R_1 \circ R_2)$ este o expresie regulata,
   - $(R_1{}^*)$ este o expresie regulata;

Notatii 38

L(R) = limbajul generat de expresia regulata R;

$\mathcal{R}$ = { R | R este o expresie regulata }.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Observatii 39

Fie $R \in \mathcal{R}$: o expresie regulata; atunci:

1. $R \cup \emptyset = R$

   (i.e. adaugarea limbajului vid altui limbaj nu il modifica pe acesta);

2. $R \circ \varepsilon = R$

   (i.e. concatenarea cuvantului vid la oricare cuvant cu nu il modifica pe acesta);

3. $R \cup \varepsilon \neq R$ (fie R=0 => L(R)={0} dar $L(R \cup \varepsilon)=\{\varepsilon,0\}$);

4. $R \circ \emptyset \neq R$ (fie R=0 => L(R)={0} iar $L(R \circ \emptyset) = \emptyset$);

5. Precedenta operatorilor regulati: **$* > {}^{o} > \cup$** .

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Exemple 40: Expresii regulate peste alfabetul $\Sigma=\{a,b\}$

1. $ab\cup ba = \{ab,ba\}$
2. $a\cup\varepsilon = \{\varepsilon,a\}$
3. $(a\cup\varepsilon)\,(b\cup\varepsilon) = \{\varepsilon,a,b,ab\}$
4. $(a\cup\varepsilon)b^* = ab^*\cup b^*$
5. $b^*\emptyset = \emptyset$
6. $\emptyset^* = \{\varepsilon\}$
7. $a^*ba^* = \{ w \mid \#|w|_b=1 \}$
8. $\Sigma^*b\Sigma^* = \{ w \mid \#|w|_b \geq 1 \}$
9. $(a\cup b)a^* = \{w \mid w$ consta numai din smb. a, precedate eventual de 1! b$\}$
10. $\Sigma^*aab\Sigma^* = \{ w \mid w$ contine subcuvantul *aab* $\}$
11. $(ab^+)^* = \{ w \mid$ fiecare smb. *a* din *w* este urmat de cel putin un smb. *b* $\}$
12. $a\Sigma^*a\cup b\Sigma^*b\cup a\cup b = \{ w \mid w$ incepe şi se termina cu acelasi simbol$\}$
13. $(\Sigma\Sigma)^* = \{ w \mid |w| =2k, k\in \mathcal{N}\}$
14. $(\Sigma\Sigma\Sigma)^* = \{ w \mid |w| =3k, k\in \mathcal{N}\}$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Observatie 41: Aplicatii ale expresiilor regulate

1. descrierea pattern-urilor:
   - utilitare: AWK sau GREP din UNIX;
   - limbaje de programare moderne: PERL;
   - editoarele de texte

ofera mecanisme de descriere a patternurilor folosind expresii regulate pentru cautari de secvente care satisfac anumite conditii;

2. proiectarea analizoarelor lexicale (parte a compilatoarelor pentru limbajele de programe; efectueaza analiza lexicala a programului-sursa ca prima faza a traducerii acestuia in program-obiect):

   expresiile regulate permit descrierea sintaxei identificatorilor (nume de variabile, constante etc.) ca in ex.:

   o constanta numerica, formata dintr-o parte intreaga şi eventual dintr-o parte fractionara şi/sau un semn, poate fi descrisa ca un cuvant din limbajul

   $(+ \cup - \cup \varepsilon)\ (C^+ \cup C^+ . C^* \cup C^* . C^+)$ peste alfabetul $C = \{ 0,1,2,\ldots,9 \}$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Definitie 42

Automat finit nedeterminist generalizat = AFNG =

$(Q, \Sigma, \delta, q_{start}, q_{accept})$, unde:

Q = multime finita, nevida, ale carei elemente se numesc stari;

Σ = multime finita, nevida, numita alfabet de intrare, ale carei elemente se numesc simboluri;

$q_{start} \in Q$, numita starea initiala;

$q_{accept} \in Q$, numita starea finala;

$\delta : (Q \setminus \{q_{accept}\}) \times (Q \setminus \{q_{start}\}) \to \mathcal{R}$, numita functia de tranzitie.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Teorema 43

$\forall$ L$\subseteq\Sigma^*$, L$\in\mathcal{L}_3$: $\Leftrightarrow$ $\exists$ o expresie regulata R peste Σ care descrie L.

Demonstratie

“$\Rightarrow$” (informal)

Fie L$\subseteq\Sigma^*$ un limbaj regulat => $\exists$ A$\in\mathcal{A}$ a.i. L=L(A)

Exista un algoritm de convertire a unui AFD intr-o expresie regulata:

1. se converteste AFD intr-un AFNG,
2. se converteste AFNG intr-o expresie regulata;

“$\Leftarrow$” (formal)

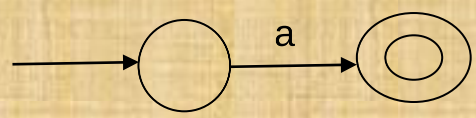
Fie L$\subseteq\Sigma^*$ un limbaj si fie R o expresie regulata peste Σ a.i. L(R)=L;

E suficient sa demonstram cum se transforma o expresie regulata intr-un AFN (examinand pe rand cele 6 cazuri din Definitia 37) şi sa aplicam Corolarul 32) ->

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

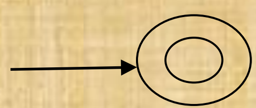
Fie R $\in \mathcal{R}$ => $\exists$ AN$\in \mathcal{AN}$ care o recunoaste, unde AN este:

1. daca R=a, $\forall a \in \Sigma$ => L(R)={a} şi AN care recunoaste L(R) este:



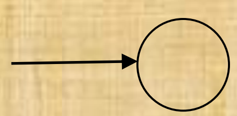
Formal: AN=({s,q}, Σ, δ, s, {q}) unde: δ(s,a)={q}, δ(r,x)=$\varnothing$ daca $r \neq s$ sau $x \neq a$;

2. daca R=ε => L(R)={ε} şi AN care recunoaste L(R) este:



Formal: AN=({s}, Σ, δ, s, {s}) unde: δ(r,x)=$\varnothing$ $\forall$r şi $\forall$ $x \in \Sigma$;

3. daca R= $\varnothing$ => L(R)=$\varnothing$ şi AN care recunoaste L(R) este:



Formal: AN=({s}, Σ, δ, s, $\varnothing$) unde: δ(r,x)=$\varnothing$ $\forall$r şi $\forall$ $x \in \Sigma$.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Fie R $\in \mathcal{R}$ => $\exists$ AN$\in \mathcal{AN}$ care o recunoaste, unde AN este:

4. daca R=$R_1 \cup R_2$ unde L(R_i) este recunoscut de $N_i \in \mathcal{AN}$, i=1,2;
   atunci AN care recunoaste L(R) se construieste din N_1 şi N_2
   ca in Teorema 26 de inchidere a $\mathcal{L}_3$ la $\cup$;
5. daca R=$R_1 \circ R_2$ unde L(R_i) este recunoscut de $N_i \in \mathcal{AN}$, i=1,2;
   atunci AN care recunoaste L(R) se construieste din N_1 şi N_2
   ca in Teorema 27 de inchidere a $\mathcal{L}_3$ la $\circ$;
6. daca R=R_1^* unde L(R_1) este recunoscut de $N_1 \in \mathcal{AN}$;
   atunci AN care recunoaste L(R) se construieste din N_1
   ca in Teorema 28 de inchidere a $\mathcal{L}_3$ la * q.e.d.

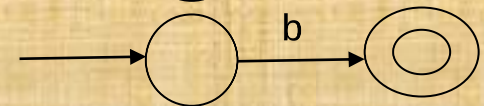
$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Exemplificari 44

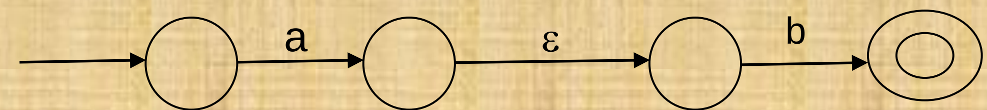
1) ? AFN pentru R=(ab ∪ a)*

a:

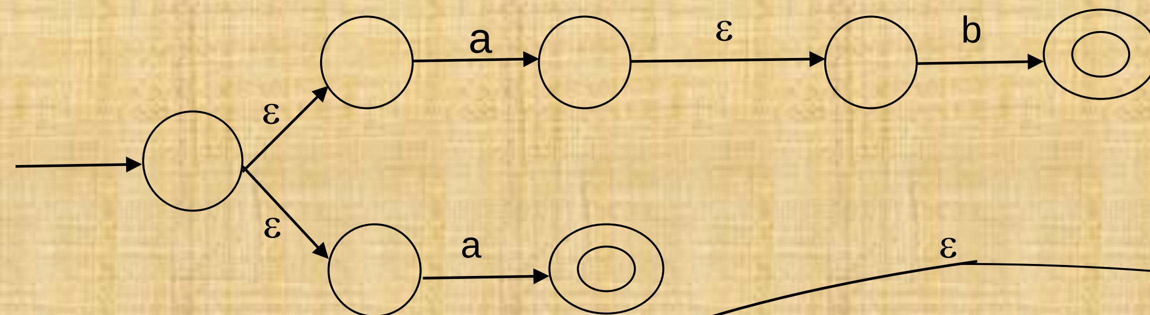
b:



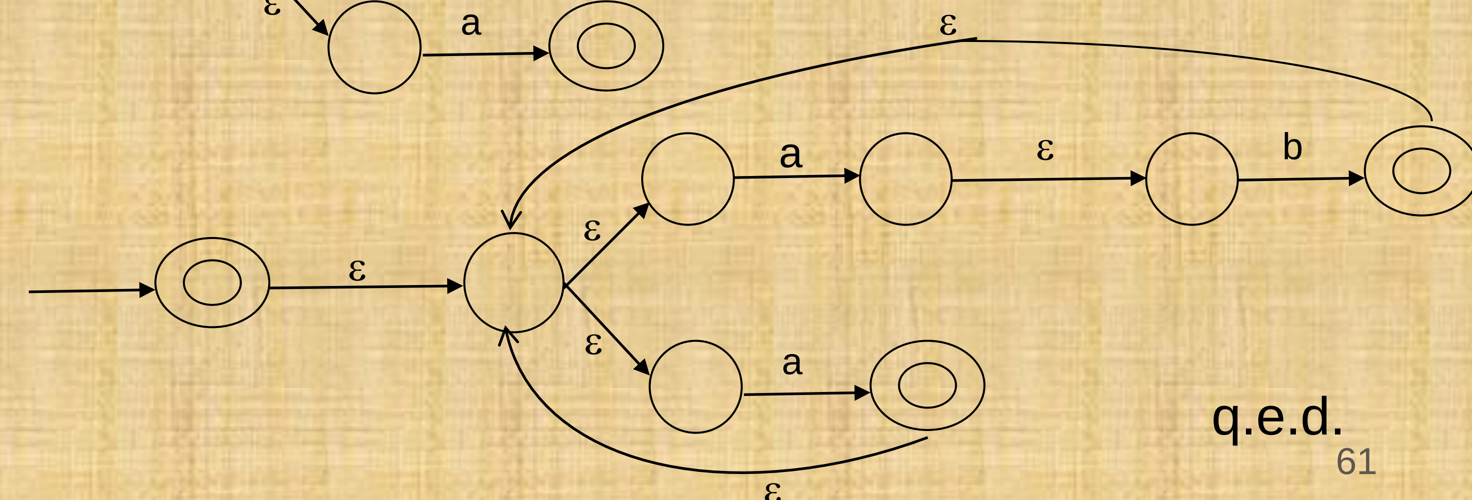
ab:



ab ∪ a



(ab ∪ a)*

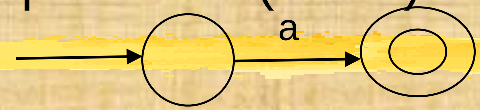


q.e.d.

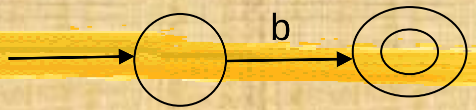
Exemplificari 44

2) ? AFN pentru R=(a ∪ b)*aba

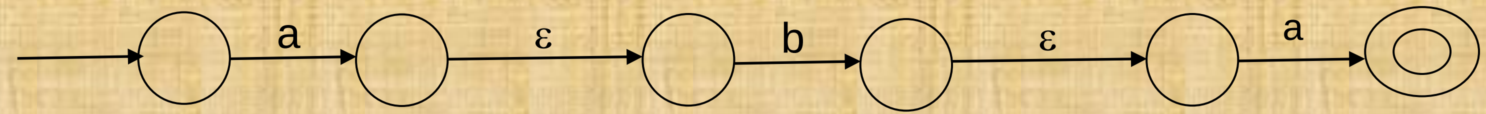
a:



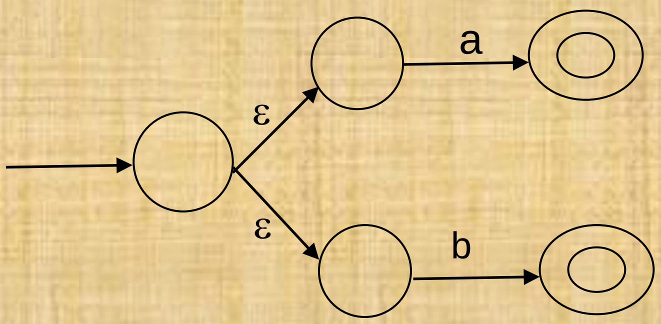
b:



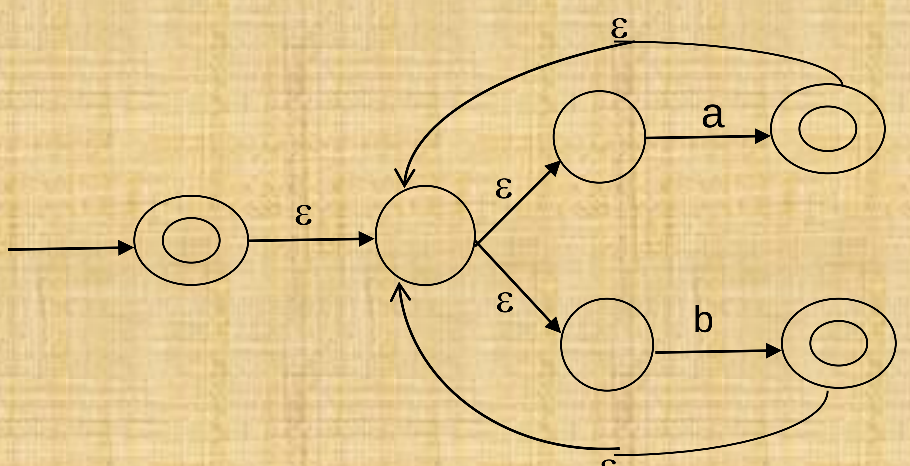
aba:



a∪b:



(a ∪ b)*



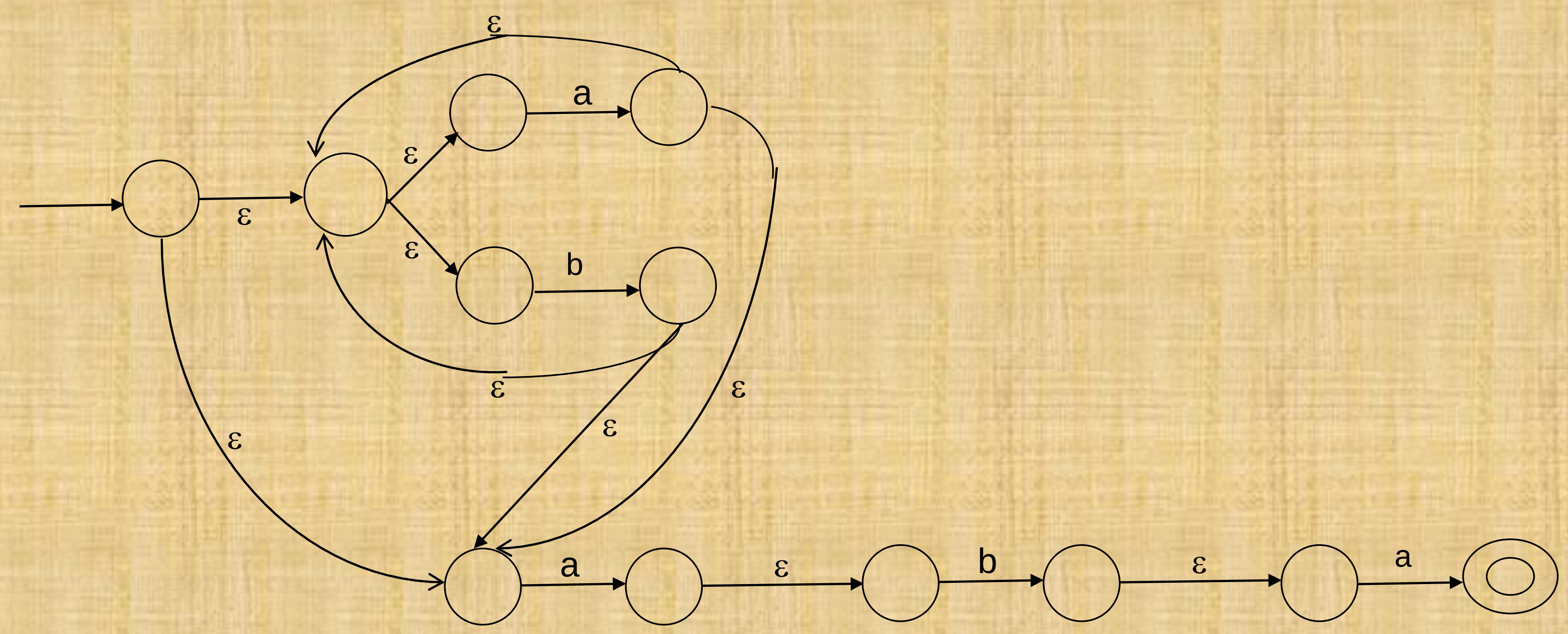
->

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Exemplificari 44

2) ? AFN pentru R=(a ∪ b)*aba (cont.)

(a ∪ b)*aba



LFA: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Lema de pompare

Fie $L \subseteq \Sigma^*$, $L \in \mathcal{L}_3 \Rightarrow \exists$ **p** $\in \mathcal{N}$ (numita lungimea sau ct.de pompare) a.i.

$\forall$ **w$\in$L: |w|$\geq$p** atunci $\exists$ x,y,z$\in\Sigma^*$ cu proprietatea ca **w=xyz** si:

(1) $\forall\ i \geq 0$: $xy^iz \in L$;

(2) $|y| > 0$;

(3) $|xy| \leq p$.

Observatii 33

- ✓ cond (2) evita solutiile triviale;
- ✓ cond. (3): $x=\varepsilon \vee z=\varepsilon$ dar nu ambele;
- ✓ cond. (3): f. utila in unele demonstratii de neapartenenta;
- ✓ daca $\forall w \in L$: $|w| < p$ (pt. $p \in \mathcal{N}$ ales) => $(\neg\exists)$ $w \in L$: $|w| \geq p$ si atunci cele 3 conditii sunt trivial verificate, lema nemaiavand obiect!! .

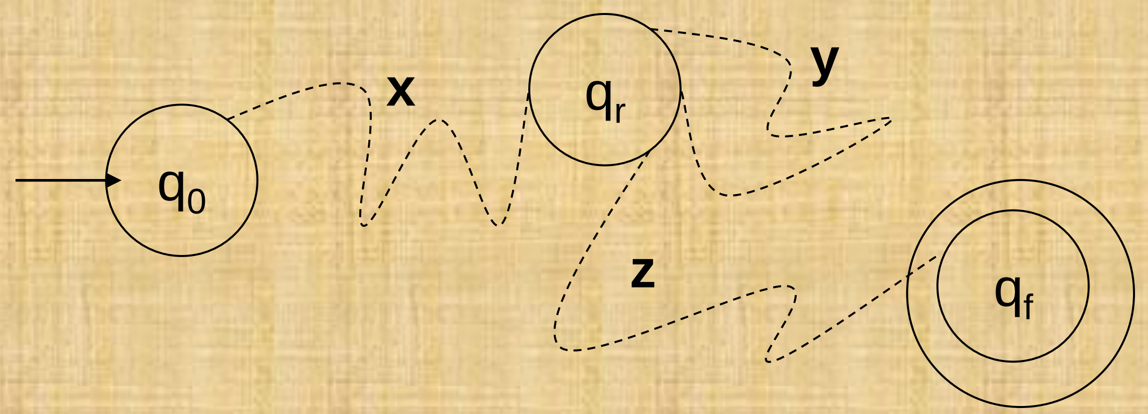
$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Ideea demonstratiei

Luam $p=|Q|$

şi $n=|w|$, $n \geq p \Rightarrow$

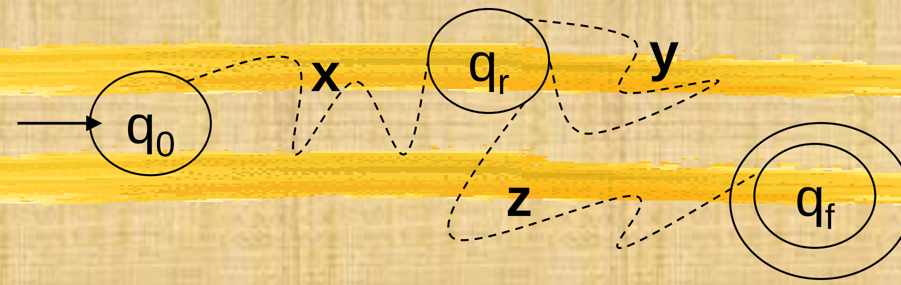
$n+1 > p=|Q| \Rightarrow \exists$ cel putin 1 repetitie: $q_0..q_i..q_k\ldots q_r q_{r+1}\ldots q_r..q_t\ldots q_f$



Verificam conditiile:

- fie $w=xyyz$; analog pt $w=xy^iz$, $\forall$ $i \geq 2 > 0$ şi pt $w=xz$ $\Rightarrow$ (1);
- obs. ca subsecv. y aduce M din q_r inapoi in q_r $\Rightarrow$ (2);
- q_r este prima stare care se repeta iar $n+1>p \Rightarrow$
  repetitia apare in una dintre primele p+1 stari din secv. $\Rightarrow$ (3)

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



Demonstratie

Fie $A = (Q, \Sigma, \delta, s, F) \in \mathcal{A}$, $L(A) = L$

şi $p = |Q|$

Fie $w = w_1 w_2 \dots w_n \in L$, $|w| = n$, $n \geq p$

şi $r_o, r_1, \dots, r_n \in Q$ starile parcurse de A pentru prelucrarea secventei w

=> $r_o = s$; $r_{i+1} = \delta(r_i, w_{i+1})$, $\forall\ 0 \leq i \leq n-1$; $r_n \in F$

Obs. ca numarul de stari este $n+1 \geq p+1$ (am pp. $n \geq p$) =>

cf. principiului cutiei: intre primele p+1 stari exista o stare care se repeta $\Leftrightarrow$

$\Leftrightarrow$ $\exists$ doua stari r_j şi r_k ,$1 \leq j<k \leq p+1$: $r_j = r_k$

=> $k \leq p+1$ => $\exists\ x,y,z \in \Sigma^*$: $x = w_1 w_2 \dots w_{j-1}$,

$y = w_j w_{j+1} \dots w_{k-1}$,

$z = w_k w_{k+1} \dots w_n$. ->

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Demonstratie (cont.)

Cum secventa x duce A din starea r_o in starea r_j

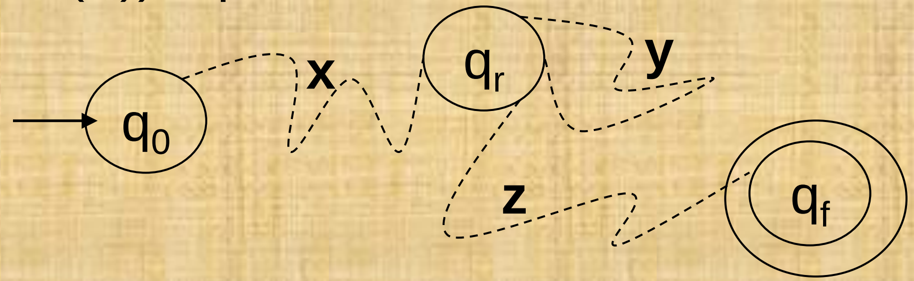
iar z duce A din starea r_k in starea r_n , unde $r_n \in F$ =>

=> A accepta toate secvenetele xy^iz, $\forall i \geq 0$ (=> cond(i));

Cum $1 \leq j<k \leq p+1$ => $j \neq k$ $|y| > 0$ (=> cond(ii)),

=> $|xy| \leq p$ (=> cond(iii)); q.e.d.

<u>Aplicatie 34</u>



Lema de pompare: demonstrarea $L \notin \mathcal{L}_3$:

ppa $L \in \mathcal{L}_3$ => putem aplica Lema:

$\Rightarrow$ exista $p \in \mathcal{N}$ a.i. $\forall$ $w \in L$, $|w| \geq p$, poate fi “pompat”

cautam un contraexmplu i.e.

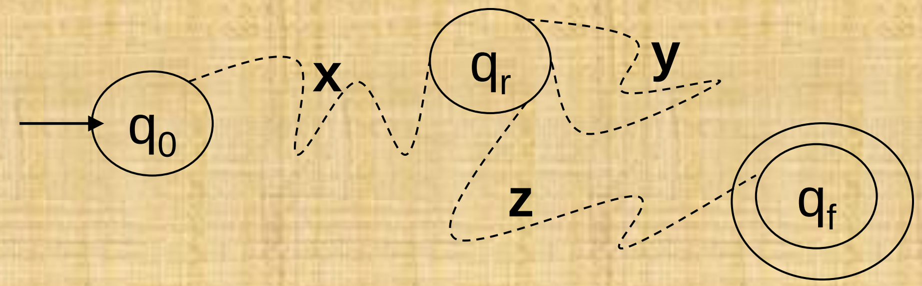
cautam un $w \in L$, $|w| \geq p$, care, oricum ar fi descompus in $x,y,z \in \Sigma^*$:

contrazice cel putin una dintre conditiile (i)-(iii),

(cel mai des: $\exists i \in \mathcal{N}$ (i=0 sau i>0) a.i. $xy^iz \notin L$);

De obicei, alegem acel w care evidentiaza esenta caracterului neregulat al L.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE



<u>Exemplu</u>

$L_1 = \{ a^n b^n \mid n \in \mathcal{N} \} \notin \mathcal{L}_3$

fie $w=a^p b^p$, p=ct de pompare => $|w|=2p>p\geq 1$;

exista 3 descompuneri posibile w=xyz:

$x=a^p$, $y=b^k$, $1\leq k\leq p$, $z=b^{p-k}$ => $xy^2z=a^p b^k b^k b^{p-k}=a^p b^{p+k}$ ✂

$x=a^{p-k}$, $y=a^k$, $1\leq k\leq p$, $z=b^p$ => $xy^2z=a^{p-k}a^k a^k b^p=a^{p+k}b^p$ ✂

$x=a^{p-k}$, $y=a^k b^k$, $1<k<p$, $z=b^{p-k}$ => $xy^2z=a^{p-k}a^k b^k a^k b^k b^{p-k}=a^p b^k a^k b^p$ ✂

𝓛𝓕𝓐: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

Teorema 36

Problema apartenentei, a limbajului vid, a limbajului infinit şi a echivalentei sunt decidabile pentru $\mathcal{L}_3$

Demonstratie

(i) Problema apartenentei:

fie $w \in \Sigma^*$ şi $A \in \mathcal{A}$ oarecare;

$|w| < \infty$ => “rulam” A pe w şi, dupa un nr **finit** de pasi, A ajunge in starea q

daca $q \in F$ atunci $w \in L(A)$, altfel $w \notin L(A)$ q.e.d.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

(ii) Problema limbajului vid:

fie A$\in\mathcal{A}$ oarecare şi fie arborele de derivare care descrie toate derivările posibile executate de A pornind de la starea iniţială; procedam astfel:

P1. marcăm starea iniţială a lui A;

P2. executam P3 până când nu se mai pot marca noi stari:

P3. marcăm orice stare în care intră o săgeată (o tranziţie) care pleacă dintr-o stare deja marcată.

P4. dacă nici una dintre stările finale nu este marcată, atunci A nu acceptă niciun cuvant w$\in\Sigma^*$, deci L(A)= $\emptyset$ q.e.d.

$\mathcal{LFA}$: C3 – LIMBAJE REGULATE

(iii) Problema limbajului infinit:

evidenta prin Lema de pompare q.e.d.

(iv) Problema echivalentei:

Fie A,B$\in\mathcal{A}$; construim C $\in \mathcal{A}$ a.i C accepta numai acele cuvinte w$\in\Sigma^*$ care sunt acceptate fie de A fie de B dar nu de ambele, i.e.:

$$L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$$

Intrucat $\mathcal{L}_3$ este inchisa la reuniune, intersectie si complementara =>L(C)$\in\mathcal{L}_3$

dar L(A)=L(B) $\Leftrightarrow$ L(C)=$\emptyset$

cum problema limbajului vid este decidabila =>

problema echivalentei este decidabila q.e.d.

LFA: C3 – LIMBAJE REGULATE

1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie.