# *LFA*: C3 – LIMBAJE REGULATE

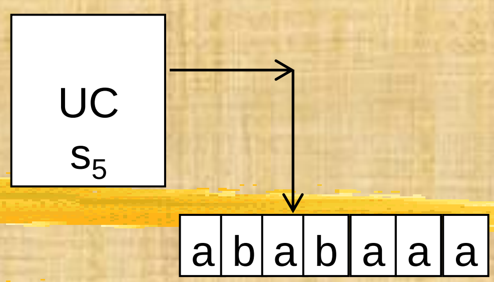
1. Automate finite deterministe
2. Operatii de inchidere
3. Automate finite nedeterministe
4. Expresii regulate
5. Lema de pompare
6. Probleme de decizie.

# *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



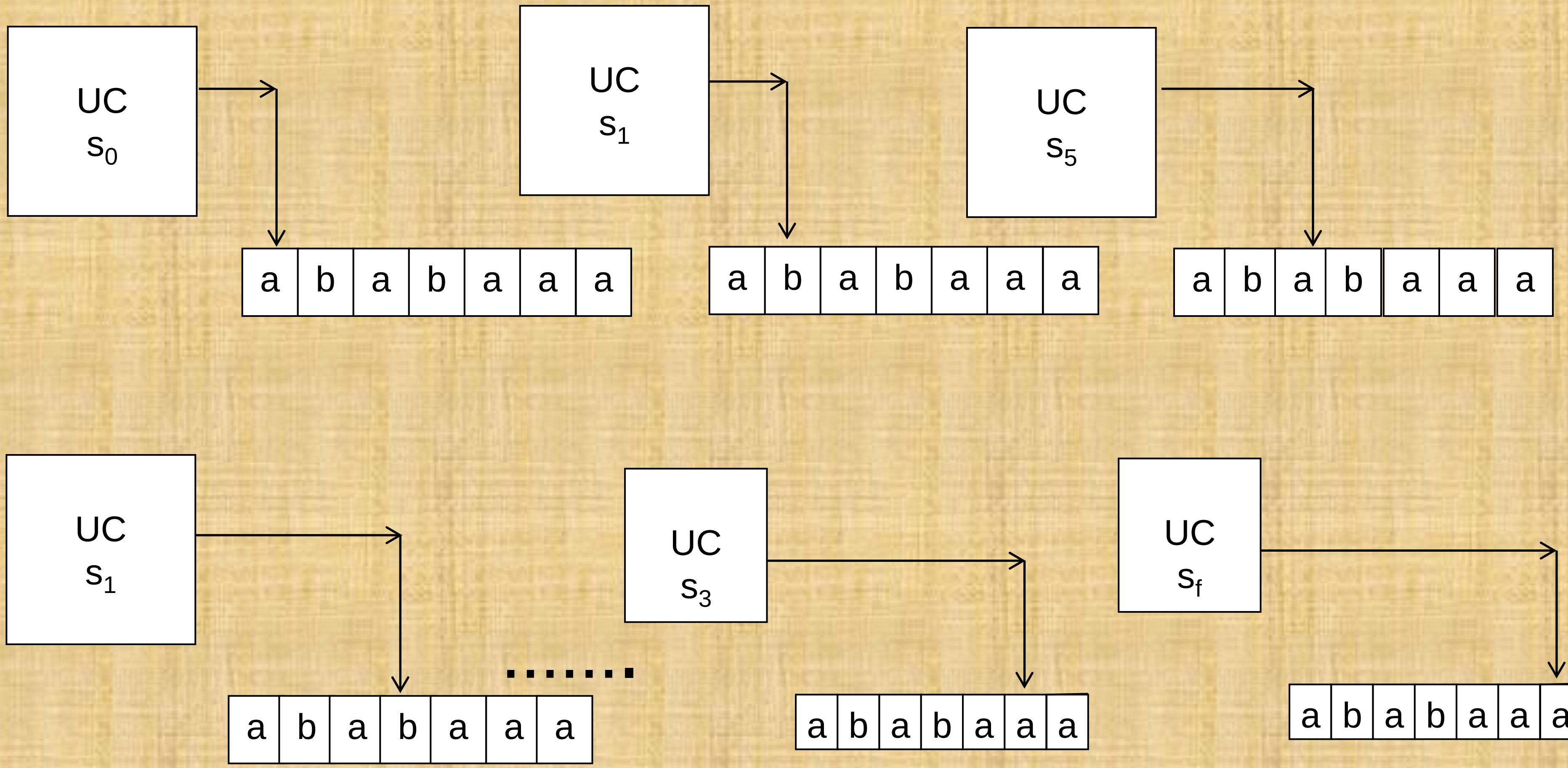

## Automat

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 – LIMBAJE REGULATE

## Automat

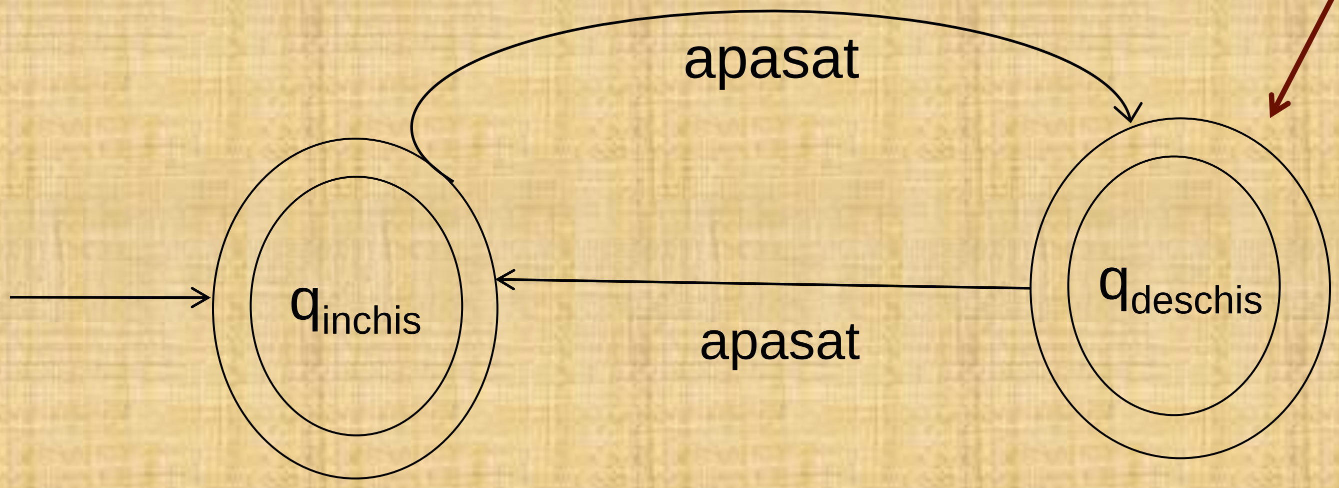


# *LFA*: C3 – LIMBA

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

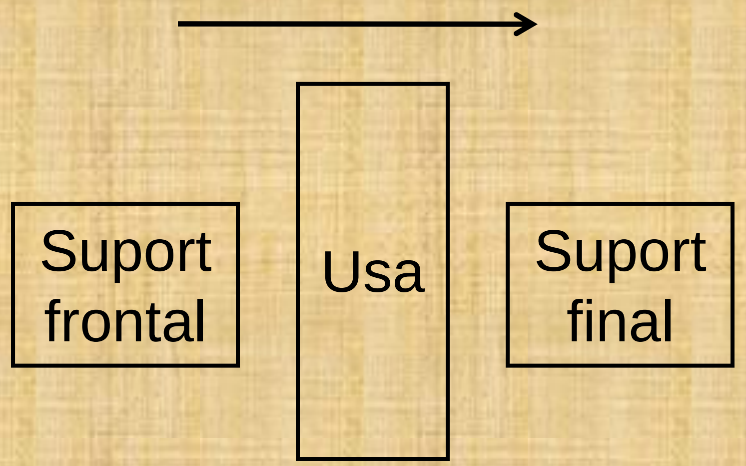



Ascensoare, termostate, masini de spalat etc.

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

Exemplu 2:

AF pt o usa automata pentru acces auto:

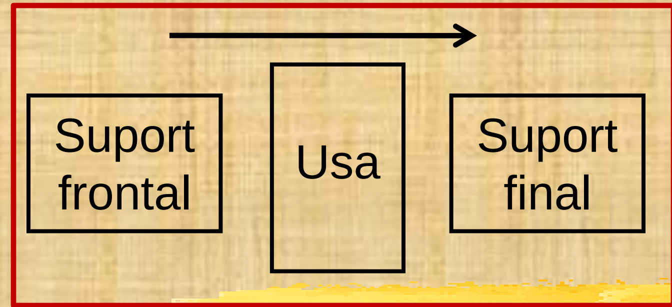



3 descrieri posibile

i. Descrierea in limbajul natural;

ii. Descrierea formala;

iii. Descrierea cu ajutorul diagramei de stare.

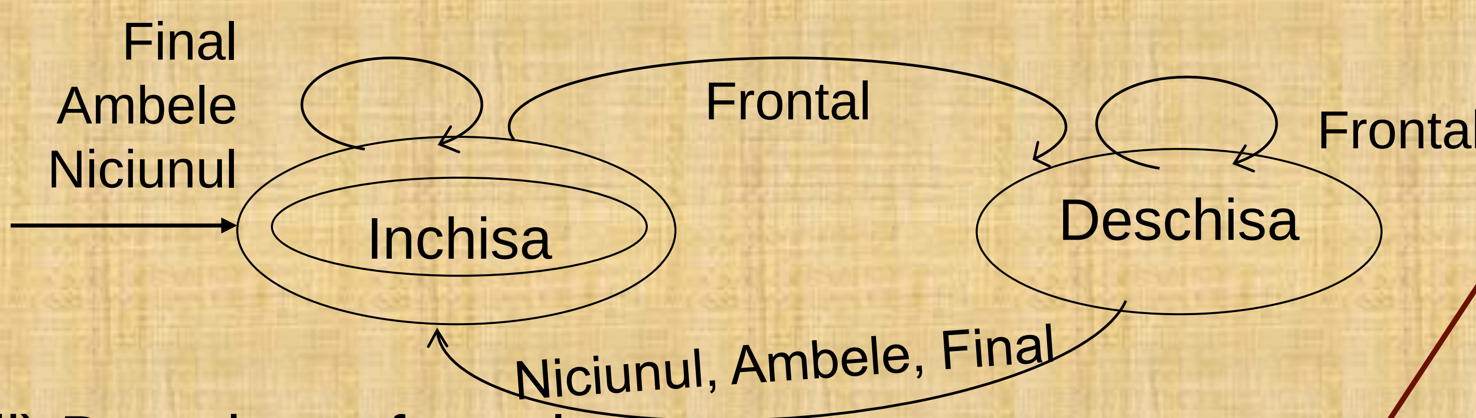



# *LFA*: C3 – Limbaje regulate

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

(i) Descrierea in limbajul natural:

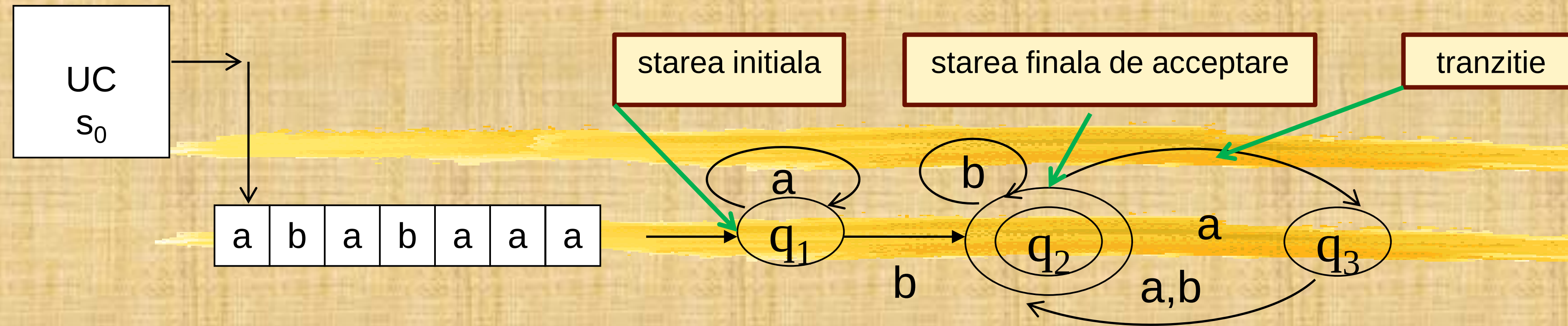
(ii) Descrierea cu ajutorul diagramei de stare




(iii) Descrierea formala

| | *Pe suportul frontal* | *Pe suportul final* | *Pe ambele suporturi* | *Pe niciun support* |
|---|---|---|---|---|
| *Inchis* | Deschis | Inchis | Inchis | Inchis |
| *Deschis* | Deschis | Inchis | Inchis | Inchis |

# *LFA*: C3 – LIMBAJE REGULATE




<u>Observatie 3: Principiul de lucru</u>

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

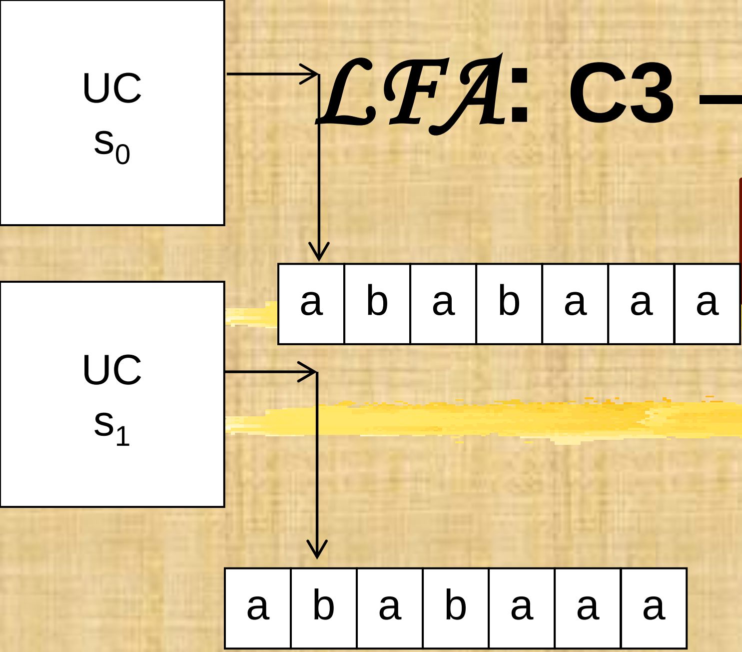
Date de intrare:

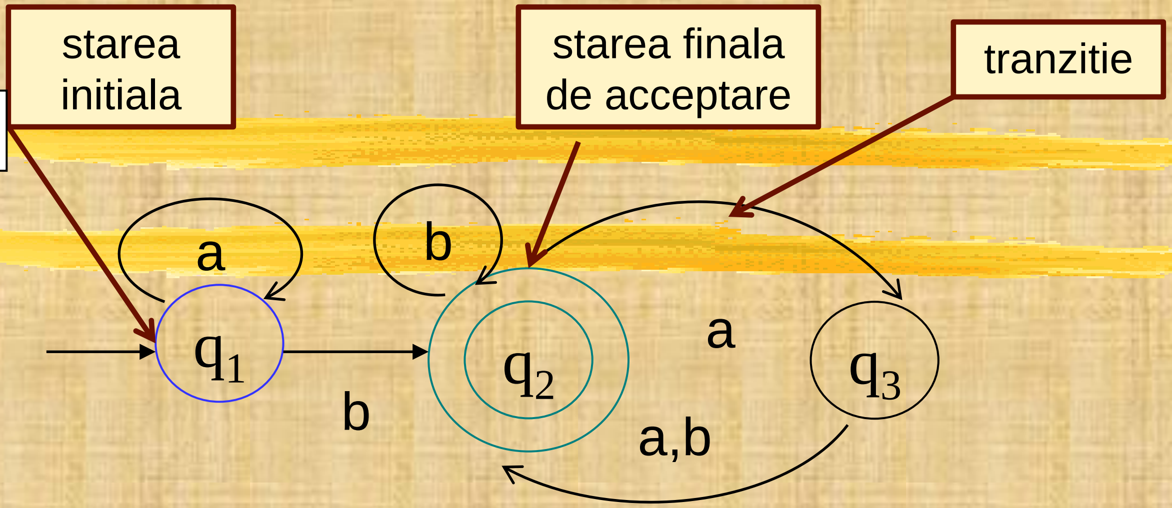
- ✓ o secventa FINITA de simboluri din alfabet, care sunt ”citite” <u>unul cate unul</u>;

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 (<u>unic determinata</u>)
- ✓ 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.

# *LFA*: C3 – LIMBAJE REGULATE




## Observatie 3 (cont.)

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

- ✓ starea curenta
- ✓ simbolul curent "citit"

**Cand** se termina calculul?

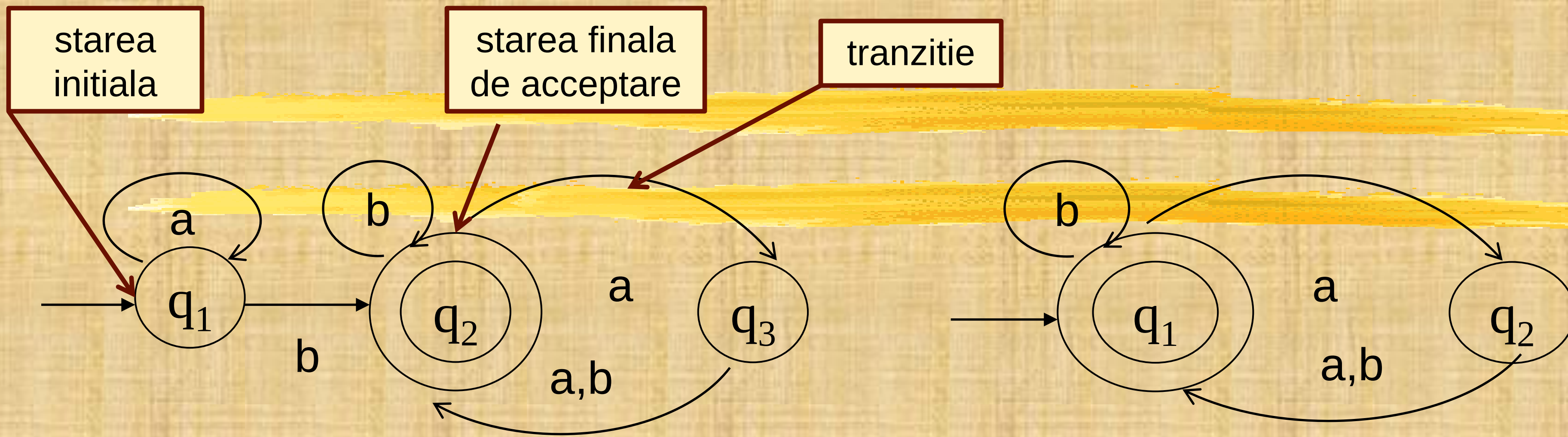
- ✓ cand au fost citite toate simbolurile din secventa de intrare

**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**;

## Observatie 4: Conventii de reprezentare. →

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




**($A_1$)** ≢ **($A_2$)**

√ 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 = ($Q$, $\Sigma$, $\delta$, $s$, $F$), unde:

- $Q$ = multime finita, nevida (stari),
- $\Sigma$ = 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 }

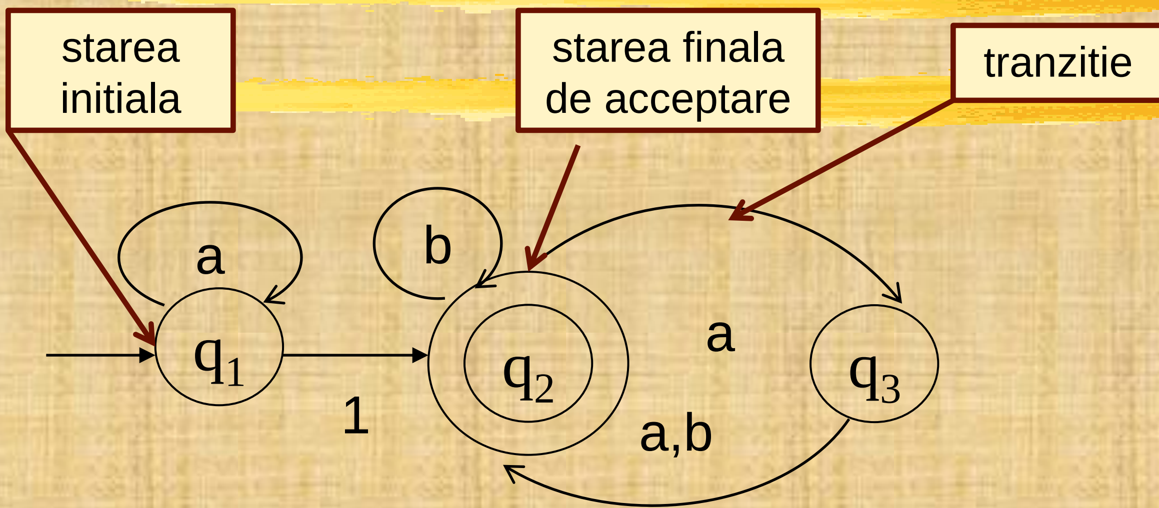
Observatie 7

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

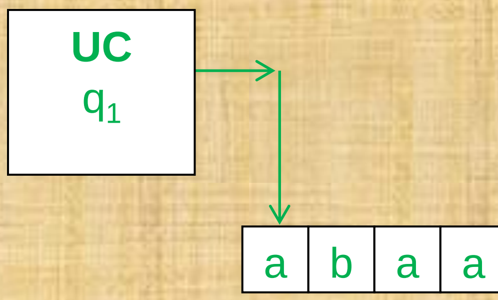
$\delta : Q \times \Sigma^* \rightarrow Q$ : $\delta(s, \varepsilon) = s$

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

# $\mathcal{LFA}$: 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>

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

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

$s = q_1$;

$F = \{q_2\}$

$\delta$ :

| | a | b |
|---|---|---|
| **$q_1$** | **$q_1$** | **$q_2$** |
| **$q_2$** | **$q_3$** | **$q_2$** |
| **$q_3$** | **$q_2$** | **$q_2$** |

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

## Exemplu 8

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

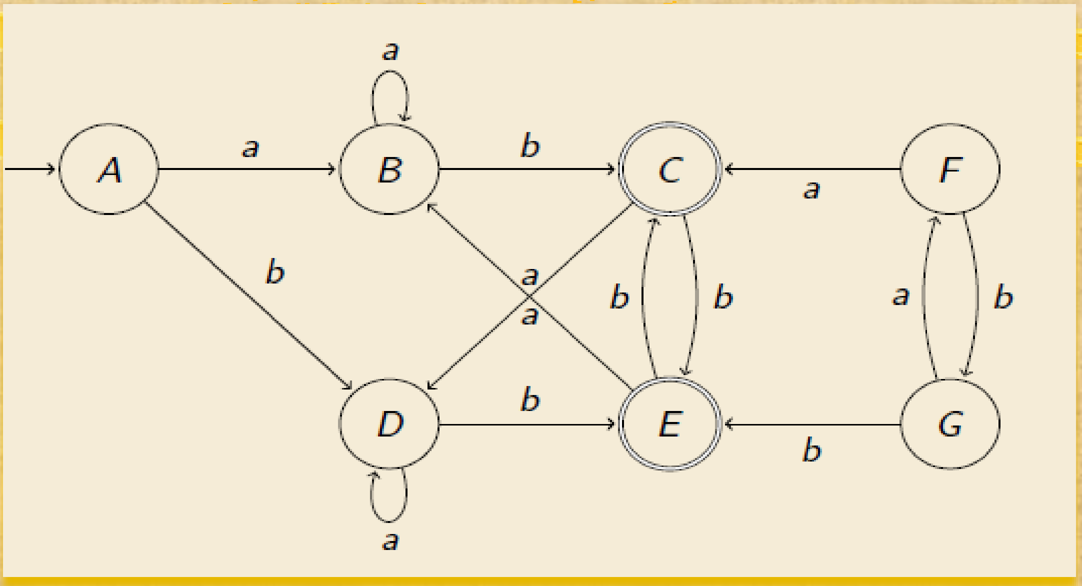
$\Sigma = \{a,b\}$;

$s = A$;

$F = \{C,E\}$

$\delta$ :

| | 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, …..

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

Definitie 9

L(A) = limbajul recunoscut de AFD A

= { w $\in\Sigma^*$| $\delta$(s,w)=q$\in$F }

= multimea secventelor peste $\Sigma$ care aduc A intr-o stare finala

Observatie 10: acceptare vs. recunoastere

Fie AFD $A_3$= (Q, $\Sigma$, $\delta$, s, $\varnothing$)

=> L($A_3$) = $\varnothing$

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

0101110010000011110001111100000111000111111100000111100001:

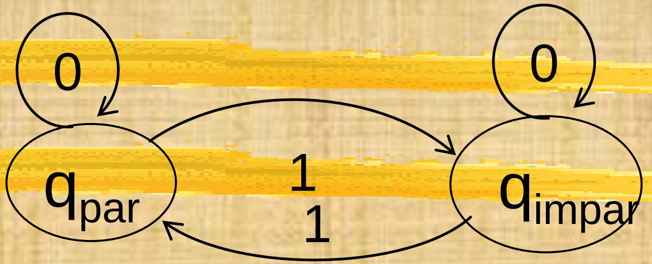
*Pas 1: stabilim informatia de memorat:*

- 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:*

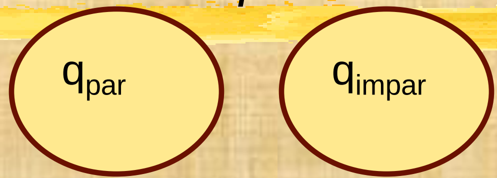
- numar par de simboluri 1, pana acum;
- numar impar de simboluri 1, pana acum. ->

# *LFA* : C3 – LIMBAJE REGULATE




*Pas 3: asignam fiecarei posibilitati cate o stare:*

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

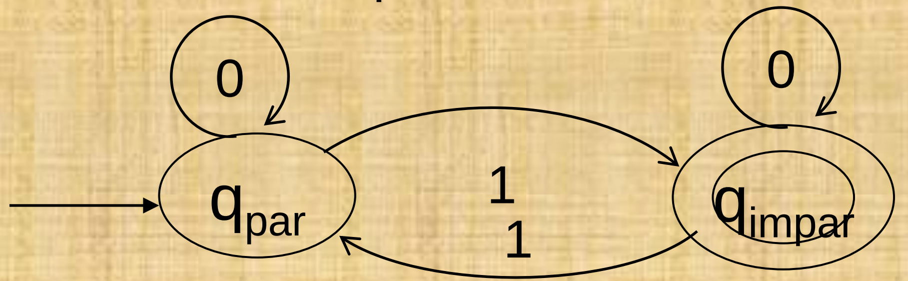



*Pas 4: definim tranzitiile, examinand modul in care se trece de la o posibilitate la alta la citirea fiecarui tip de simbol din $\Sigma$:*

- 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

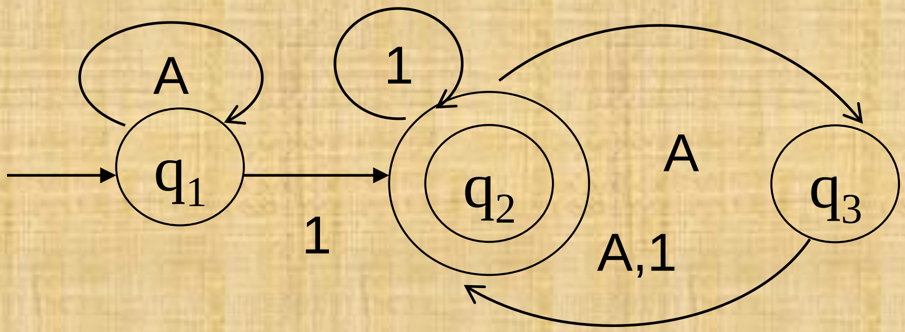
## Definitie 12: Calculul efectuat de un AFD

Fie A = (Q, $\Sigma$, $\delta$, 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 ABABAA pentru ca exista secventa de stari

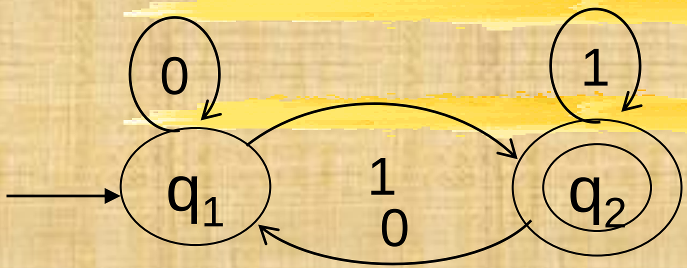
$q_B$, $q_1$, $q_2$, $q_3$, $q_2$, $q_3$, $q_2$ , care indeplineste toate cele 3 conditii:

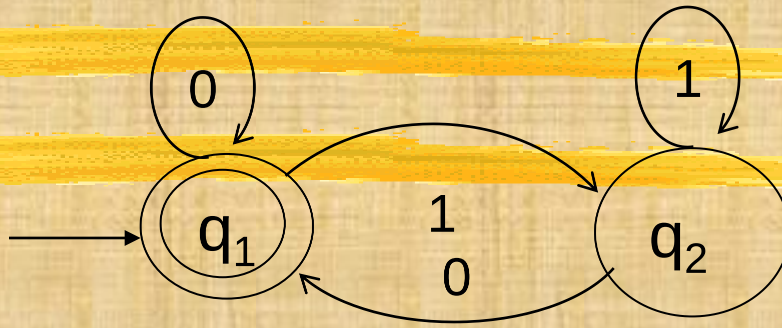
$\delta(q_1, A) = q_1$, $\delta(q_1, B) = q_2$, $\delta(q_2, A) = q_3$, $\delta(q_3, B) = q_2$, $\delta(q_2, A) = q_3$, $\delta(q_3, A) = q_2$

## 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, ….. =>

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

| $\delta$ | 0 | 1 |
|---|---|---|
| $q_1$ | $q_1$ | $q_2$ |
| $q_2$ | $q_1$ | $q_2$ |

Fie acum:

√ 0000, 1010, …..

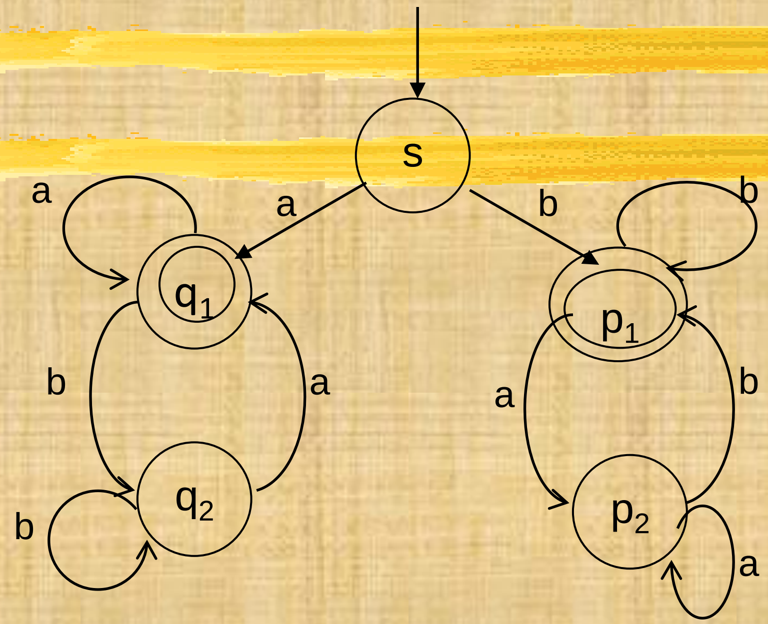
⊗ 10101, 0001, …… =>

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

| $\delta$ | 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….

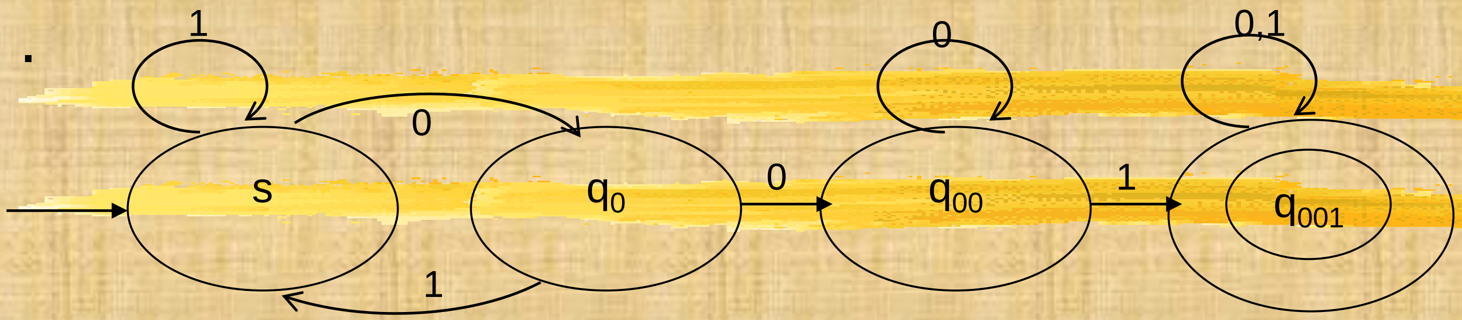
=> $a^n$, $a^nb^ma^k$, $a^nb^ma^kb^ua^v$ =>$a^{n1}b^{m1}a^{k1}a^{n2}b^{m2}a^{k2}…a^{nx}b^{mx}a^{kx}$

=> $L_3$= {w∈{a,b}* | w incepe şi se termina cu a} ∪

∪ {w∈{a,b}* | w incepe şi se termina cu b} .

# $\mathcal{LFA}$ : 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

<u>Definitie 16</u>

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

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

<u>Observatii 17</u>

- 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 $\varepsilon$ (n>0 sau n=0!);

Exemplu 18

Fie $\Sigma$ = {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* = {$\varepsilon$, fix, mobil, fixfix, fixmobil, mobilfix, mobilmobil, fixfixfix, fixfixmobil, fixmobilfix, fixmobilmobil, fixfixfixfix, …. }.

## 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 $q_2 \in Q_2\} = Q_1 \times Q_2$

$\Sigma = \Sigma_1 \cup \Sigma_2$

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

$s = (s_1, s_2)$

$F = \{(q_1,q_2) \mid q_1 \in F_1$ 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 $w = w_1w_2 \ldots w_n$ : $\forall\ 1 \le i \le n$: $w_i \in \Sigma$

Atunci, A accepta w ddaca $\exists\ r_o, r_1, \ldots, r_n \in Q$ astfel 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 $w = w_1w_2 \ldots w_n \in L(A_1)$ =>

$\exists\ r_o, r_1, \ldots, r_n \in Q_1$ 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 $w = w_1w_2 \ldots w_n \in L(A_2)$: analog;

Reciproc: fie $w = w_1w_2 \ldots w_n \in L(A)$ =>

$\exists\ (r_1, q_1), (r_2, q_2), \ldots, (r_n, q_n) \in Q = Q_1 \times Q_2$ 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 $L_1 / L_2 = \{w \in \Sigma^* \mid \exists y \in L_2 : wy \in L_1\} \in \mathcal{L}_3$

*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)).