**LIMBAJE FORMALE ŞI AUTOMATE**

**Lista de subiecte (Teorie)**

**1. Multimea limbajelor nu este numarabilă (alfabet, limbaj, reprezentare, relaţia de ordine, secvenţa caracteristică, enuntul teoremei, demonstratia)**

**Alfabet** **(Σ)** – Este orice mulţime finită şi nevidă de elemente numite şi simboluri, este notată cu **Σ**

Exemple:

Σbool = {0,1}

Σlatin = {a,b,c,…,z}

ΣADN = {A,C,G,T}

Σ\* = mulţimea tuturor elementelor peste alfabetul **Σ** incluzând (\*) elementul vid ( ε )

Σ + = Σ\* - {ε} = mulţimea tuturor elementelor peste alfabetul **Σ** excluzând (+) elementul vid ( ε )

**Un cuvânt ( w )** peste un alfabet este notat cu w, are lungimea |w|

|ε| = 0 ε este cuvântul vid, nu are valoare

Reversul unui cuvânt (w) este wR / w = capac, wR = capac

#t(w) = numărul de apariţii ale simbolului t în cuvântul w

#t(test) = 2

**Funcţia lui Parikh**(ψ)ψ Σ (w) = (#s1|w|, #s2|w|, …, #sk |w|)

Fie Σ = {a, b, …,z}; atunci ψ Σ : Σ\*→N 27 , ψ Σ (Constantinopol) = (1,0,1,0,0,0,0,0,1,0,0,1,0,3,3,1,0,1,2,0,0,0,0,0)

**prefix**w∈{ε, î, în, înt, într, întru, întruc, întrucâ, întrucâtv, întrucâtva}

**sufix**w∈{ε, a, va, tva, âtva, câtva, ucâtva, rucâtva, trucâtva, ntrucâtva, întrucâtva}

**subcuvânt**w∈{ε, î, n, t, r, u, c, â, v, în, nt, tr, …, întru, …, câtva, …, întrucâtva}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Limbaj** **(L)** – Este format din concatenarea elementelor, formând cuvinte

**Reprezentare** – Limbajele pot fi reprezentate prin expresii regulate, automate sau gramatici

**Relaţia de ordine** – ordinea canonică este influenţată de lungimea cuvintelor limbajului

**Secvenţa caracteristică** – **λL**

Fie ∑ = {a,b} şi L = {a, ab, abb}

∑\*= {ε,a,b,aa,ab,ba,bb,aaa,aab,aba,abb,baa,..., aaaa, aaab, aaba,...} L = { , a, , , ab, , , , , , abb }

λL= 0 1 0 0 1 0 0 0 0 0 1 0 0 0 ,...= 0100100000100...

1, dacă a,b ∈ L

0, dacă a,b ∉L

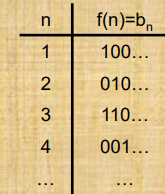
**Enunţul teoremei** – Mulţimea limbajelor definite peste un alfabet dat nu este numărabilă

**Mulţimea L = { L⊆∑\* | L =limbaj } este nenumărabilă**

**Demonstraţia**:

1. **mulţimea B a secvenţelor binare infinite este nenumărabilă**

Folosim metoda diagonalizarii şi a p.p.a. ppa ∃ f:N -> B, bijectiva a.i. f(n)=bn∈B ->



putem construi o secventa binara b astfel: a n a cifra binara din b este:

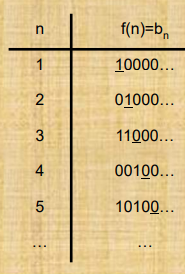
0, daca a n-a cifra binara din f(n) este 1,

1, daca a n-a cifra binara din f(n) este 0

=> b=00111......

=> b≠f(n), ∀n∈N :

=> B este nenumarabila.



1. **mulţimea L = {L⊆∑\* | L = limbaj} este nenumărabilă**

E suficient sa gasim f: L -> B, bijectivă ori, ∃ f: L -> B: f(L)= λL şi, evident, f = bijectiva; cf. (i) B = nenumărabilă => L nenumărabilă.

Prin urmare se poate demonstra cu:

**F:L -> B**: f(n)=bn bijectivă ( prin diagonalizare şi p.p.a ) => B - nenumărabilă

**F:L -> B**: f(L) = λL (secvenţa caracteristică) => L - nenumărabilă

**2. Ierarhia Chomsky (definiţia fiecărui tip de gramatică, exemplificare, justificarea incluziunilor nestricte, respectiv stricte; o demonstratie)**

**Definiţia fiecărui tip de gramatică:**

**Tipul 0 ( Gramatica fără restricţii ) – fără restricţii asupra regulilor de producţie**

**Tipul 1 ( Gramatica dependente de context ) – Reguli de forma αAβ→αγβ unde A este neterminal, α,β pot fi șiruri de terminali și/sau neterminali, și γ este un șir nenul.**

**Tipul 2 ( Gramatica independentă de context ) - Reguli de forma A→γ unde A este neterminal și γ este un șir de terminali și/sau neterminali.**

**Tipul 3 ( Gramatica regulată ) - Reguli de forma A→aB sau A→aA unde A, B sunt neterminali și a este un terminal. / exemple: gramatici lineare, la dreapta/stânga, regulate**

**Exemplificare:**

1. **Tipul 0 ( gramatica neregulata ) – nu are restrictii in ceea ce priveste regulile de productie de forma:** L = { 02n | n apartine N}

α - > β

Exemplu: G = ( {0}, {S,L,Z,R},S,P ) => L = 02n

1. **Tipul 1 ( gramatica dependenta de context ) –** L = { anbncn| n≥1 }

αAβ -> αγβ

A – neterminal

α,β – siruri de neterminali si/ sau terminali

γ – este sir nenul

Exemplu:

P = {S->aSBc, S->abc, cB->Bc, bB->bb}

G = ({a,b,c}, {S,B},S,P) => L = { anbncn| n≥1 }

1. **Tipul 2 ( gramatica independenta de context ) – pot descrie sintaxa oricarui limbaj de programare > utili pentru proiectarea algoritmilor de parsare -** L = { anbn| n∈N }

A -> α sau S -> aSb

A – neterminal

α -> sir de terminal sau neterminal

Exemplu: G= ({a,b}, {S},S,{S -> aSb , S -> ε} ) => L = { anbn| n∈N }

1. **Tipul 3 ( gramatica regulata ) -** L = {1}{0,1}\*

A→aB / A→Ba, A→a

A,B – neterminali

a - terminal

Exemplu**:** G= ({0,1}, {S,A} ,S,{S->1A, A->0A, A->1A, A->ε } => 1001011 => L = {1}{0,1}\*

**Justificarea incluziunilor nestricte, respectiv stricte:**

**Incluziunile stricte:** Limbajele regulate sunt incluse strict în limbajele independente de context, care la rândul lor sunt incluse strict în limbajele sensibile la context, iar acestea sunt incluse strict în limbajele nelimitate.

L0 - multimea limbajelor generate de gramatici de tip 0

L1 - multimea limbajelor generate de gramatici de tip 1

L2 - multimea limbajelor generate de gramatici de tip 2

L3 - multimea limbajalor generate de gramatici de tip 3

Atunci: L0 ⊃ L1 ⊃ L2 ⊃ L3

Incluziunile nestricte: forma productiilor;

Incluziunile stricte: contraexemple.

**Demonstratie:**

Demonstrăm că limbajele regulate sunt strict incluse în limbajele independente de context: Un limbaj regulat poate fi generat de o gramatică regulată și orice limbaj generat de o gramatică regulată poate fi generat de o gramatică independentă de context, dar inversul nu este valabil (de exemplu, limbajul {a^n b^n |n≥0} este independent de context dar nu este regulat).

**3. Descrierea limbajelor regulate (automate finite, gramatici, expresii regulate: definiţie, exemple, reprezentare, teoreme de echivalenţă; o demonstraţie)**

* **automate finite (DFA)** – exemplu

Considerăm un automat finit determinist (DFA) care recunoaște limbajul L={w∈{a,b}∗∣w conține ab}.

DFA-ul poate fi definit ca (Q,Σ,δ,q0,F) unde:

* Q={q0,q1,q2}
* Σ={a,b}
* q0​ este starea inițială
* F={q2} este setul de stări finale
* Funcția de tranziție δ este definită astfel:
  + δ(q0,a)=q1 ​
  + δ(q0,b)=q0
  + δ(q1,a)=q1
  + δ(q1,b)=q2
  + δ(q2,a)=q2 ​
  + δ(q2,b)=q2 ​
* **gramatici regulate** – exemplu ( tip 4 Chomsky = L = ( {1}{0,1}\* ) / generat de gramatica G = ( {0,1}, {S,A}, S,P) / P = {S->1A, A->A0, A->1A, A->ε}
* **expresie regulata – E** = (a|b)\*ab(a|b)\* - descriu cuvintele care contin secventa „ab” undeva in interiorul lor

**4. Descrierea limbajelor independente de context (automate pushdown, gramatici: definiţie, exemple, reprezentare, forme normale, teoreme de echivalenţă; o demonstraţie)**

* **automat pushdown (PDA)**

Considerăm PDA-ul care recunoaște limbajul L={anbn ∣ n≥0}.

PDA-ul poate fi definit ca (Q,Σ,Γ,δ,q0,Z0,F) unde:

* Q={q0,q1,q2}
* Σ={a,b}
* Γ={Z0,A}
* q0 este starea initiala
* Z0​ este simbolul inițial al stivei
* F={q2} este setul de stări finale
* Funcția de tranziție δ definită astfel:
  + δ(q0,a,Z0)=(q0,AZ0}
  + δ(q0,a,A)=(q0,AA)
  + δ(q0,b,A)=(q1,ϵ)
  + δ(q1,b,A)=(q1,ϵ)
  + δ(q1,ϵ,Z0)=(q2,Z0)
* gramatica independenta de context ( tip 2 Chomsky ) = L ={anbn | n apartine N}
* forma normala Chomsky – productie de forma A -> BC sau A -> a
* forma normala Greibach – productie de forma A->ab / b – nenul
* teorema de echivalenta - orice limbaj care poate fi recunoscut de un PDA poate fi generat de o gramatică independentă de context, și invers.
  + - PDA – recunoaste limbaj tip 2 => Gramatica tip 2 genereaza limbaj tip 2 si invers

**5. Operaţii la care clasa limbajelor regulate este, respectiv nu este închisă (pt fiecare operaţie: definiţia, teorema; o demonstraţie)**

Operatii:

- uniune

- concatenare

- kleen star

- intersectia

- complementul

Nu este inchis:

- reversul

- diferenta simetrica

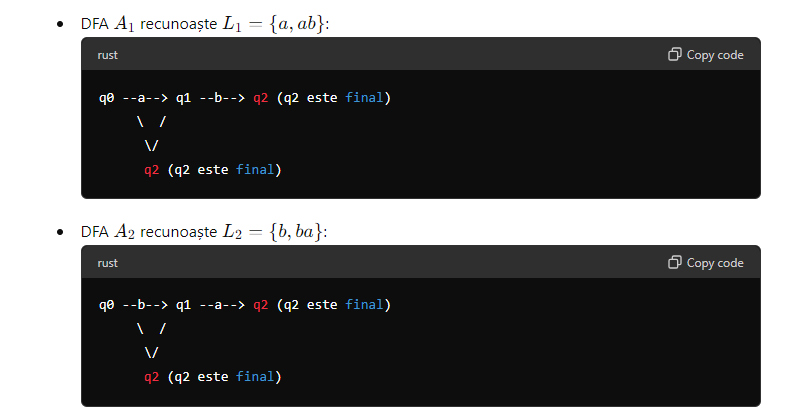
**Demonstratie – operatia de uniune**

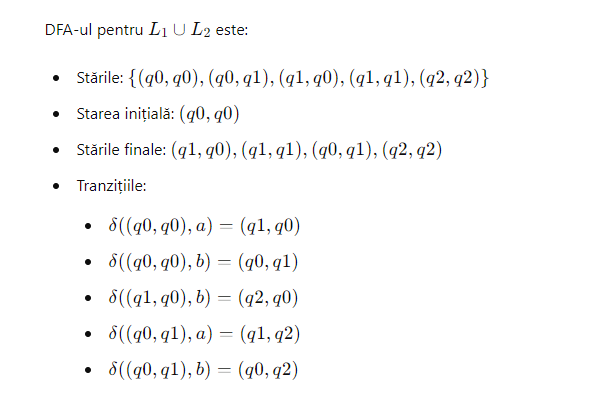
Fie două DFA-uri A1​ și A2​ care recunosc limbajele L1 si L2​. Să construim un DFA care recunoaște limbajul L1∪L2​.

Fie A1=(Q1,Σ,δ1,q01,F1) ) și A2=(Q2,Σ,δ2,q02,F2)

Construim DFA-ul AAA:

* Stările lui A sunt perechi (q1,q2) unde q1∈Q1 și q2∈Q2​.
* Starea inițială este (q01,q02).
* Funcția de tranziție este definită astfel: δ((q1,q2),a)=(δ1(q1,a),δ2(q2,a))
* Stările finale sunt perechi (q1,q2) unde q1∈F1 sau q2∈​F2​.





**6. Operaţii la care clasa limbajelor independente de context este, respectiv nu este închisă (pt fiecare operaţie: definiţia, teorema; o demonstraţie)**

Operatii:

- uniune

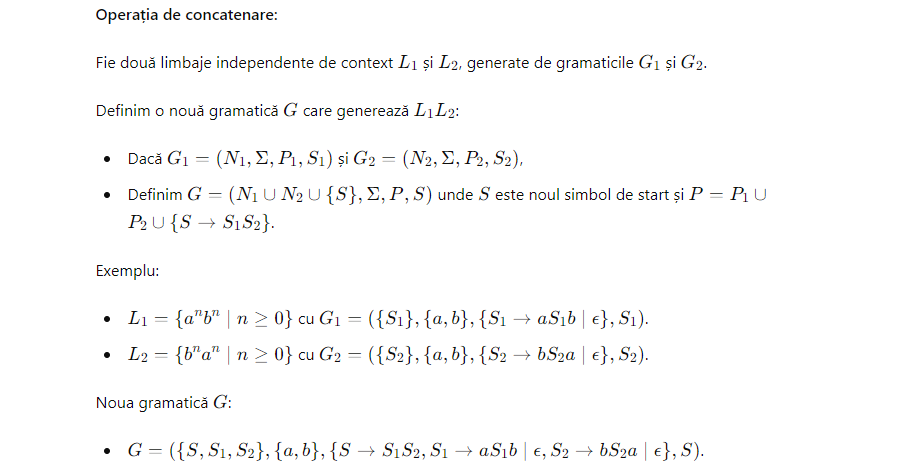
- concatenare

- kleen star

Nu este inchis:

- intersectia

- complementul



**7. Lema de pompare pentru limbajele regulate; aplicaţii**

**8. Lema de pompare pentru limbajele independente de context; aplicaţii**

**9. Probleme de decizie în clasa limbajelor regulate (enunt; o demonstraţie)**

**10. Probleme de decizie în clasa limbajelor independente de context (enunt; o demonstraţie)**