$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

   echivalenta cu gramatica independenta de context.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

∃ limbaje suficient de simple pe care
un AFD / o expresie regulata / o gramatica regulata
nu le pot descrie (recunoaste):
ex.: limbajul $\{0^n1^n \mid n \geq 0\}$
motivul: memoria **finita** a unui *AFD* nu poate memora numere **n** foarte mari;

∃ mecanisme mai puternice:

- ✓ gramaticile independente de context (*GIC*)
- ✓ automatele pushdown (*APD*).

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Avantajul *GIC:* pot descrie structuri recursive =>

variate domenii de aplicabilitate a *GIC*:

1. studiul limbilor naturale:
   - ✓ relatiile dintre termeni precum: substantiv, verb, adjectiv, prepozitie,
   - ✓ relatiile dintre expresiile substantivale, verbale etc.

     sunt – in mod natural – de tip recursiv: o expresie verbala poate contine o expresie substantivala (de ex.) care, la randul ei, poate contine o expresie verbala sau adjectivala etc.
2. specificarea si compilarea limbajelor de programare,
   - ❑ sintaxa unui limbaj de programare,
     - ✓ sintaxa unui lb.de programare poate fi invatata si pornind de la gramatica sa,
     - ✓ proiectarea compilatoarelor si interpretoarelor de limbaje de programare

       incepe deseori cu construirea unei GIC pt acel limbaj,
   - ❑ parsere:
     - ✓ o posibila reprezentare a semnificatiei unui program extrasa de *parser* chiar inainte de compilarea codului /executarea instructiunii interpretate,

       se poate realiza cu ajutorul arborelui de derivare al codului, obtinut cu *GIC* a limbajului de programare respectiv,
     - ✓ exista numeroase metodologii care permit construirea – uneori automat – a unui parser direct din *GIC* a limbajului de programare respectiv.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Reamintim:

Definitia 1

Gramatica = (**V, Σ, S, P**) unde:

V = multime finita, nevida, ale carei elemente se numesc variabile sau simboluri neterminale (alta notatie V_N);

Σ = multime finita, nevida, numita alfabet de intrare, ale carei elemente se numesc [simboluri] terminale (alta notatie V_T); $V \cap \Sigma = \emptyset$;

S∈V se numeste simbolul de start (axioma) gramaticii;

$\mathbf{P \subseteq (V \cup \Sigma)^* V (V \cup \Sigma)^* \times (V \cup \Sigma)^*}$ = multime finita, nevida (productii) = $\{\alpha \rightarrow \beta \mid \alpha \in (V \cup \Sigma)^* \circ V \circ (V \cup \Sigma)^*;\ \beta \in (V \cup \Sigma)^*\}$

OBS. : ($(\alpha,\beta) \in P \equiv \alpha \rightarrow \beta$: α se inlocuieste cu β);

Definitii 2

Se numeste **substitutie = derivare directa** = aplicarea unei productii i.e.:

daca $\alpha \rightarrow \beta \in P$ şi $\delta, \gamma \in (V \cup \Sigma)^*$, atunci $\delta\alpha\gamma \Rightarrow \delta\beta\gamma$

Se numeste **derivare** = aplicarea consecutiva a mai multor productii =

daca $\alpha_1 \Rightarrow \alpha_2, \alpha_2 \Rightarrow \alpha_3, \ldots, \alpha_{n-1} \Rightarrow \alpha_n$ atunci $\alpha_1 \Rightarrow^* \alpha_n$.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Reamintim:

Definitia 3

Gramatica independenta de context = **GIC** = (**V**,**Σ**, **S**, **P**) unde:

$\forall p \in P$: p este de forma $A \rightarrow \alpha$ unde: $A \in V$, $\alpha \in (V \cup \Sigma)^*$

Definitia 4

Limbaj independent de context = LIC =

$L(G)=\{ w \in \Sigma^* \mid \exists\ S \ {}_G\!\Rightarrow^* w$ si $G=GIC\}$

Notatii 5

Multimea gramaticilor independente de context: $\mathcal{G}_2=\{ G \mid G = GIC\}$,

Multimea limbajelor independente de context: $\mathcal{L}_2=\{ L \mid L = LIC\}$.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

<u>Exemple 6</u>

G=({S}, {a,b}, S, {S→aSb, S→ε})$\in \mathcal{G}_2$ =>

L={$a^n b^n$ / n$\in \mathcal{N}$}$\in \mathcal{L}_2$

G=({S}, {a,b}, S, {S→SS, S→aSb, S→ε})$\in \mathcal{G}_2$ =>

L={ε, ab, $a^n b^n$, $a^n bab^n$, $a^n (ba)^k b^n$, … }$\in \mathcal{L}_2$.

G = ({S,A,B}, {a,b}, S, {S→aB, S→bA, A → aS, B → bS, A → bAA, B → aBB, A → a, B → b})$\in \mathcal{G}_2$ =>

L(G) = { w ∈ {a, b}* / $\#_a(w) = \#_b(w)$ }.

Definitie 7

Arbori de derivare = arbori de parsare =

= o metoda de reprezentare vizuala a derivarilor dintr-o gramatica independenta de context $G=(V,\Sigma,S,P)$

= un arbore in care

1. fiecare nod este etichetat cu un simbol din $V\cup\Sigma$;
2. radacina este etichetata cu S;
3. daca un nod n, etichetat cu A, are cel putin un nod descendent, atunci A trebuie sa fie in V;
4. daca nodul n şi descendentii sai directi $n_1, n_2, \ldots, n_k$ sunt etichetati respectiv cu $A, A_1, A_2, \ldots, A_k \in V\cup\Sigma$ atunci P trebuie sa contina productia

$$A \rightarrow A_1A_2\ldots A_k \ ;$$

Observatie 8

Etichetele nodurilor terminale formeaza un cuvant din $L(G)$, numit şi <u>rezultat al derivarii</u> .

𝓛𝓕𝓐: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Exemple 9

1. Fie $G_1 = (\{S,C\}, \{0,1,\#\}, S, \{S \rightarrow 0S1,\ S \rightarrow C,\ C \rightarrow \#\})$

$\Rightarrow$ Reprezentarea derivarilor:

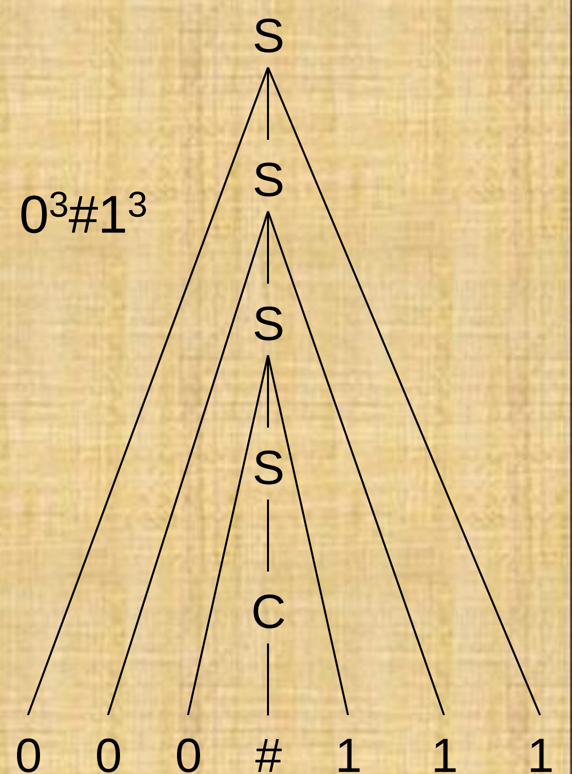
linear:

$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 000S111 \Rightarrow 0^3S1^3 \Rightarrow 0^3C1^3 => 0^3\#1^3$

sintetic:

$S \ {}_G\!\Rightarrow^* 0^3\#1^3$

arbore de derivare:



LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

2. Fie $G_2 = (\{S,A\}, \{a,b\}, S, \{S \rightarrow aAS,\ S \rightarrow a,\ A \rightarrow SS,\ A \rightarrow SbA,\ A \rightarrow ba\})$

=> Reprezentarea derivarilor:

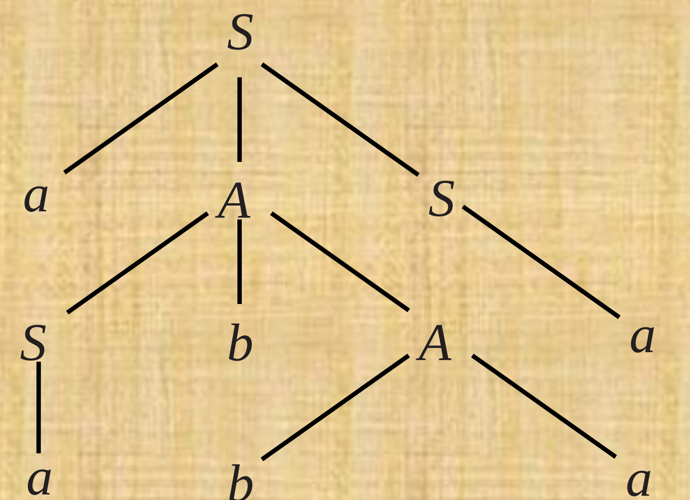
linear:

$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow aabbaS \Rightarrow aabbaa$

sintetic:

$S \ {}_G\!\Rightarrow^* a^2b^2a^2$

arbore de derivare:



$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

3. Fie $G_3 = (\{S\}, \{a,b\}, S, \{S\rightarrow SS,\ S\rightarrow aSb,\ S\rightarrow\varepsilon\})$

=> Reprezentarea derivarilor:

linear:

$S\Rightarrow SS\Rightarrow aSbS\Rightarrow aSbaSb\Rightarrow aaSbbaSb\Rightarrow aaSbbab => aabbab$

$S\Rightarrow SS\Rightarrow (S)S\Rightarrow (S)(S)\Rightarrow ((S))(S)\Rightarrow ((S))() => (())()$

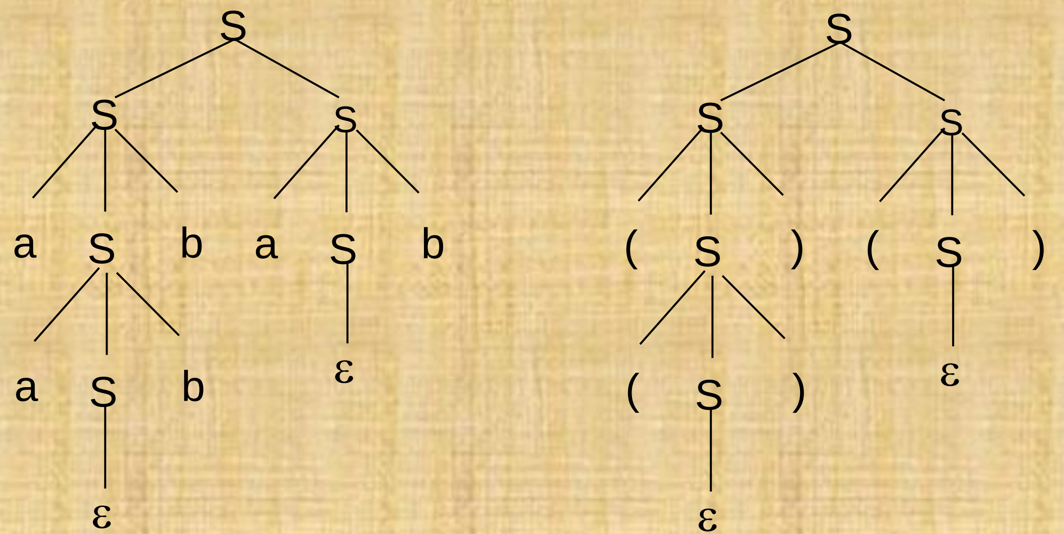
sintetic:

$S \;_G\Rightarrow^* a^2b^2ab \quad S \;_G\Rightarrow^* abab \quad S \;_G\Rightarrow^* a^3b^3 \quad S \;_G\Rightarrow^* a^2bab^2$

OBS.: daca $a \leftarrow ($ $b \leftarrow)$ => $S \;_G\Rightarrow^* (())()$

=> L_3 = limbajul parantezelor corect imbricate

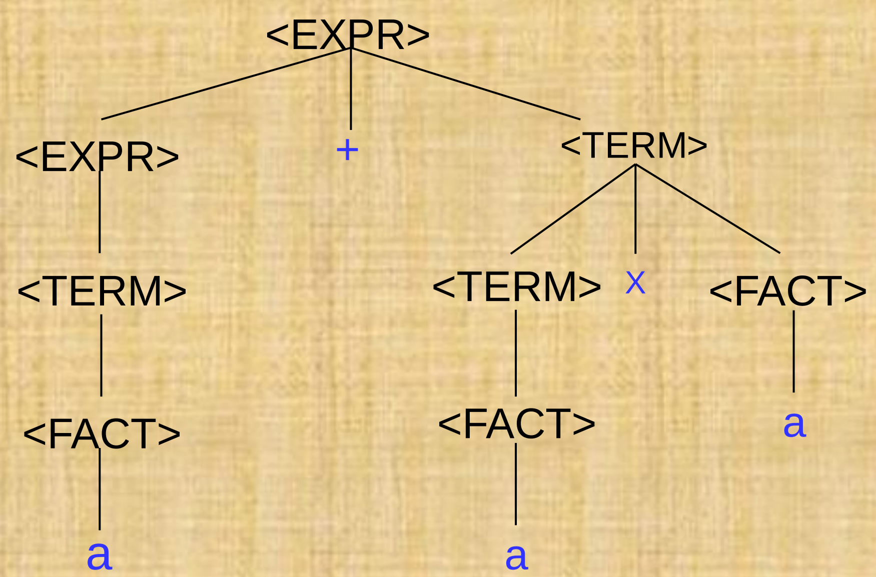
arborii de derivare.



LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

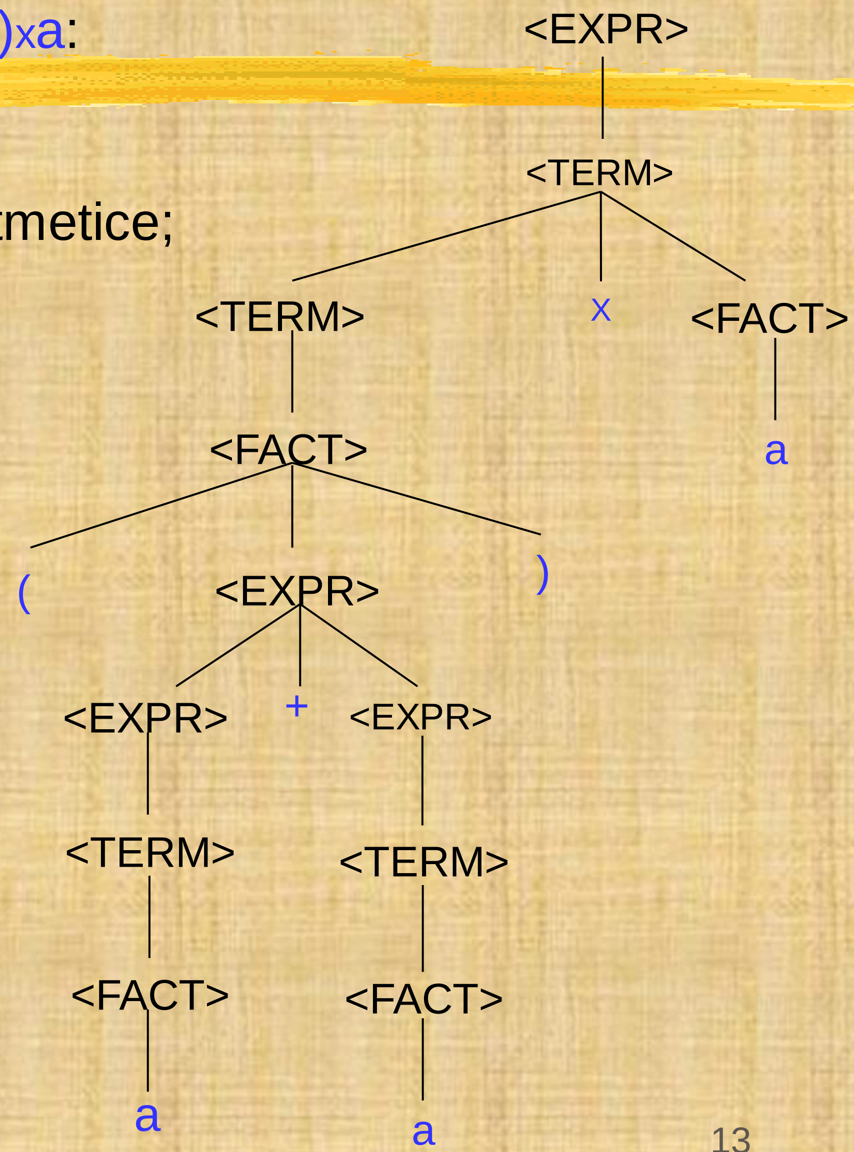
4. Fie G_4 = ({<EXPR>,<TERM>, <FACT>} , {a, +, x, (,)}, <EXPR> ,
{<EXPR>→<EXPR>+<TERM>, <EXPR>→ <TERM>,
<TERM>→<TERM>x<FACT>, <TERM>→<FACT>,
< FACT>→(< EXPR>), <FACT>→a})

=> arbore de derivare pentru cuvintul a+axa .



LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

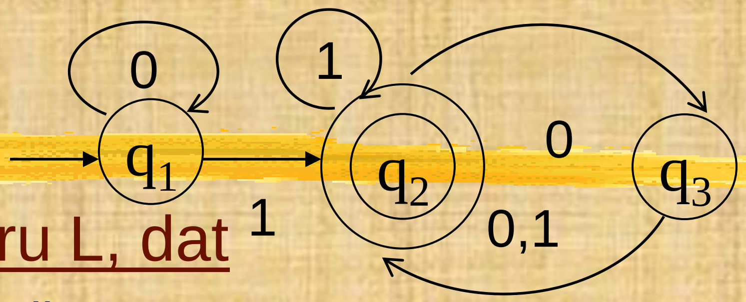
=> arbore de derivare pentru cuvintul (a+a)xa:



<u>Observatie:</u>

G_4 descrie acel fragment din limbajele de programare care trateaza expresiile aritmetice;

arborele de parsare grupeaza operatorii in conformitate cu regulile de precedenta (respectand inclusiv rolul parantezelor).



Observatie 10: Proiectarea G.I.C. pentru L, dat

Fie L = L(G); G$\in \mathcal{G}_2$; Distingem cel putin 4 situatii:

1. L $\in \mathcal{L}_3$; =>
   (i) construim AFD A=(Q, Σ, δ, s, F): L=L(A);
   (ii) convertim A=(Q, Σ, δ, s, F) in G=(V, Σ, S, P)$\in \mathcal{G}_2$ astfel:

   Q **~>** V
   s **~>** S
   $\delta(q_i,a)=q_k$ **~>** $B_i \to aB_k$
   $\forall\ q_i \in F$ **~>** se adauga $B_i \to \varepsilon \in P$

   (iii) verificam direct ca L(A)=L(G)

   ex.: L= $\{w\in\{0,1\}^* \mid w=\alpha 1(00)^n,\ \alpha\in\{0,1\}^*\}$
   (i) A=($\{q_1,q_2,q_3\}$, {0,1}, δ, q_1, $\{q_2\}$)
   (ii) G=({S,A,B}, {0,1}, S,
   {S →0S, S →1A, A →0B, A →1A , B →0B, B →1B, A →ε})
   (iii) 1, 01, 11, 0101, 010100, 01001, ……

	0	1
q_1	**q_1**	**q_2**
q_2	**q_3**	**q_2**
q_3	**q_2**	**q_2**

Observatie 10: Proiectarea G.I.C. pentru L, dat (cont.)

2. putem descompune L in limbaje mai simple: $L_1, L_2, \ldots$ =>
   (i) construim gramaticile $G_1, G_2, \ldots$: $L_i=L(G_i)$;
   (ii) le asamblam şi adaugam:
   - un nou simbol de start S şi
   - productiile $S \rightarrow S_i$

   ex.: $L=L_1 \cup L_2$ (ex.: $L=\{a^n b^n | n \in \mathcal{N}\} \cup \{b^n a^n | n \in \mathcal{N}\}$)

   => $L_1=\{a^n b^n | n \in \mathcal{N}\}$ şi $L_2= \{b^n a^n | n \in \mathcal{N}\}$

   => construim $G_i=(V_i,\Sigma,S_i,P_i)$ i=1,2 unde

   $V_i=\{S_i\}$, $\Sigma=\{a,b\}$ şi $P_1=\{S_1 \rightarrow aS_1b \mid \varepsilon\}$, $P_2=\{S_2 \rightarrow bS_2a \mid \varepsilon\}$

   adaugam S şi productiile $S \rightarrow S_1$ şi $S \rightarrow S_2$

   => $G=(\{S,S_1,S_2\}, \{a,b\}, S, \{S \rightarrow S_1 \mid S_2, S_1 \rightarrow aS_1b \mid \varepsilon, S_2 \rightarrow bS_2a \mid \varepsilon\})$.

Observatie 10: Proiectarea G.I.C. pentru L, dat (cont.)

3. descoperim simetriile / dependentele dintre subcuvintele care formeaza cuvintele din L =>

   le transpunem in simetrii / dependente intre [ne]terminalele care apar in m.dr. al productiilor din P

   ex.: $L=\{a^n b^{3n} | n \in \mathcal{N}\}$

   => construim $G=(V,\Sigma,S,P)$ unde

   $V=\{S, A\}$, $\Sigma=\{a,b\}$ şi $P=\{S \to aSbbb \mid \varepsilon\}$

Exercitiu

Fie L = limbajul parantezelor corect imbricate; sa se defineasca G a.i. L=L(G), aplicand observatiile metodologice (2) şi apoi (3).

Observatie 10: Proiectarea G.I.C. pentru L, dat (cont.)

4. descoperim structurile recursive care apar in cuvintele din L =>

(i) fie *rec* acea structura recursiva şi R variabila care o genereaza prin productia $R \to$ *rec* ,

(ii) plasam variabila R in m.dr. al productiilor care genereaza *rec* in pozitia in care apare repetitia ;

ex.: limbajul expresiilor aritmetice: intr-o expresie aritmetica, orice aparitie a unei constante poate fi inlocuita cu o noua expresie

$\Rightarrow$ a se vedea ultimele 2 productii din

G_4 = ({<EXPR>, <TERM>, <FACT>}, {a, +, x, (,)}, <EXPR>,

{<EXPR>→<EXPR>+<TERM>, <EXPR>→<TERM>,

<TERM>→<TERM>x<FACT>, <TERM>→<FACT>,

< FACT>→(< EXPR>), <FACT>→a}) .

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

   echivalenta cu gramatica independenta de context.

Observatie 11

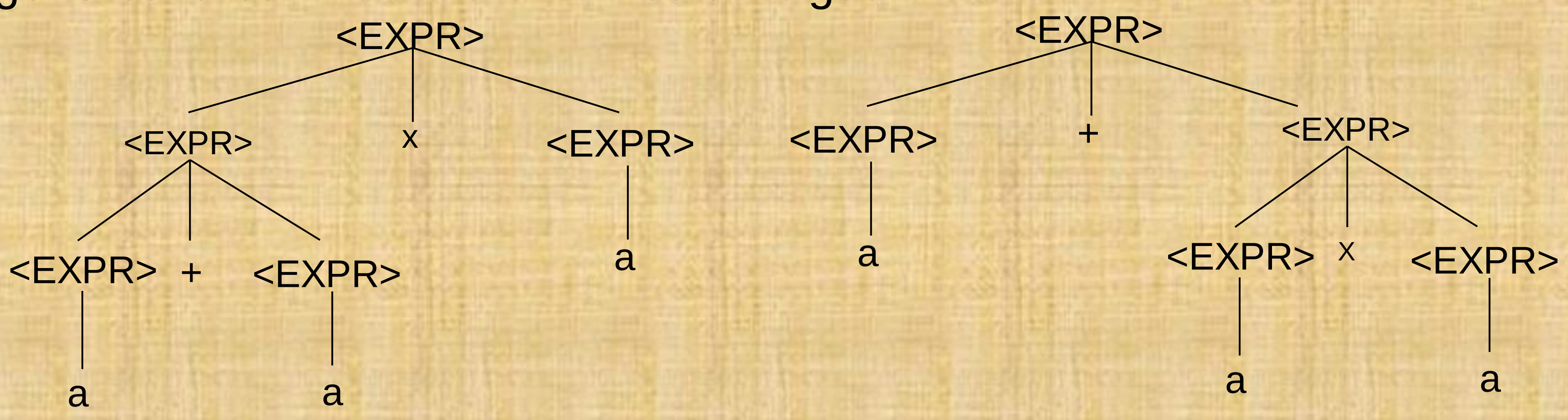
$\exists$ G $\in \mathcal{G}_2$ şi $\exists$ w$\in$L(G) pe care G le poate genera in mai multe moduri =>
cuvantul w va admite mai multi arbori de derivare (parsare) =>
cuvantul w va avea mai multe intelesuri:

=> in limbile naturale: avantaj,
in limbajele de programare: mare dezavantaj;

Exemplu 12

Fie G_5 = ({<EXPR>}, {a, +, x, (,)}, <EXPR>, {<EXPR>$\rightarrow$(<EXPR>), <EXPR>$\rightarrow$a, <EXPR>$\rightarrow$<EXPR>+<EXPR>, <EXPR>$\rightarrow$<EXPR>x<EXPR>})

G_5 genereaza cuvintul a+axa in mod ambiguu:



motivul: G_4 modeleaza şi regulile de precedenta a operatorilor, G_5: NU!
=> in G_4 orice cuvant are 1! arbore de parsare, in G_5: NU!

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Observatie 13

Notiunea de ambiguitate este legata de notiunea de arbore de derivare :

daca $w \in L(G)$, $G \in \mathcal{G}_2$, admit mai multe derivari dar 1! arbore de parsare => G ***NU*** este ambigua

motivul: 2 derivari ale w in G pot diferi prin ordinea in care sunt alese neterminalele care se substituie, ceea ce nu altereaza structura derivarii

=> s-a introdus notiunea de derivare extrem stanga pt a evidentia structura derivarii (independenta de alegerea netereminalelor);

Definitie 14

Fie $S \ {}_G\!\Rightarrow^* w$ o derivare a cuvantului $w \in L(G)$, $G \in \mathcal{G}_2$;

aceasta derivare se numeste **derivare extrem stanga** daca la fiecare pas de derivare neterminalul substituit este cel mai din stanga neterminal

Ex.: $G_2 = (\{S,A\}, \{a,b\}, S, \{S \to aAS, S \to a, A \to SS, A \to SbA, A \to ba\})$

$$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow aabbaS \Rightarrow aabbaa$$

Cex.: …….

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Definitie 15

Fie $S \ {}_G\!\Rightarrow^* w$ o derivare a cuvantului $w \in L(G)$, $G \in \mathcal{G}_2$;

Spunem ca **w este ambiguu derivat in G** ddaca w admite cel putin 2 derivari extrem stangi diferite.

Gramatica **G se numeste ambigua** ddaca genereaza ambiguu cel putin un cuvant din L(G)

Observatie 16

$\exists\ G \in \mathcal{G}_2$ ambigua a.i. $\exists\ G' \in \mathcal{G}_2$, neambigua, $L(G') = L(G)$.

Definitie 17

$L \in \mathcal{L}_2$ se numeste **inerent ambiguu** ddaca

$\forall G \in \mathcal{G}_2$, $L = L(G)$: G=ambigua.

Exemplu 18

$L = \{a^k b^i c^n \mid i=k \text{ sau } i=n\}$ este inerent ambiguu.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

   echivalenta cu gramatica independenta de context.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Definitia 19

G $\in \mathcal{G}_2$ se afla in **forma normala GREIBACH (FNG)** $\Leftrightarrow$

$\forall$ p $\in$ P, p : **A → aB**,

unde: **A** $\in$ **V,**

a $\in$ **Σ,**

B $\in$ **(V** $\cup$**Σ)*;**

Teorema 20

$\forall$ L=L(G)$\in\mathcal{L}_2$., $\varepsilon \notin$L => $\exists$ G' in FNG a.i. L=L(G').

Definitia 21

$G \in \mathcal{G}_2$ se afla in **forma normala CHOMSKY (FNC)** $\Leftrightarrow$

$\forall$ p $\in$ P, p : **$A \rightarrow BC$** sau **$A \rightarrow a$,**

unde: **$A,B,C \in V$, $B \neq S \neq C$,**

$a \in \Sigma$,

in plus, **$S \rightarrow \varepsilon \in P$;**

Lema 22 (demonstratia: EXERCITIU)

Fie $G \in \mathcal{G}_2$ aflata in FNC si $L = L(G) \Rightarrow$

$\forall w \in L, |w|=n$: $\forall S \,_G\Rightarrow^* w$: $|S \,_G\Rightarrow^* w| = 2n-1$, $n \in N$.

Teorema 23

Fie L$\in \mathcal{L}_2$ şi G$\in \mathcal{G}_2$ a.i. L=L(G) => $\exists$ G' in FNC a.i. L(G')=L

Ideea demonstratiei

Convertim pe rand p$\in$P (in ordinea crescatoare a nr de neterminale din m.dr.) aplicand un fel de tranzitivitate a productiilor;

Demonstratie

Fie G = (V, Σ, S, P)$\in \mathcal{G}_2$, L = L(G);

construim G'= (V',Σ', S', P') a.i. L(G') = L(G) şi G' in FNC;

Evident: Σ'=Σ, dar V', S', P' se vor modifica astfel:

(1) adaugam un nou simbol de start **S'** si

o noua productie: **S' → S**

=> S' nu va aparea in m.dr. vreunei p' $\in$ P' .

(2) **eliminam** ε-productiile **A→ ε, A ≠ S;**

apoi, pt fiecare aparitie a simbolul A in m.dr. al unei productii

adaugam o productie similara, in care simbolul A respectiv a disparut

<u>exemplul 1:</u>

pt. productia R → uAvAw, u,v,w ∈ (V ∪ Σ)*, adaugam:

R → uvAw,

R → uAvw,

R → uvw,

<u>exemplul 2:</u>

pt. productia R → uAv, u,v ∈ (V ∪ Σ)*, adaugam

R → uv

<u>exemplul 3:</u>

pt productia R → A adaugam

R → ε (doar daca R → ε NU a fost deja eliminata);

acest pas se repeta pana cand toate <u>ε-productiile</u> care nu il implica pe S au fost eliminate.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

(3) **eliminam** "productiile unitare" **A → B,**

apoi, pt fiecare productie **B → u,** **$u \in (V \cup \Sigma)^*$,**

adaugam productia **A → u,**

doar daca A → u NU este o "productie unitara" care a fost deja eliminata;

acest pas se repeta pana cand toate "productiile unitare" au fost eliminate;

(4) pentru productiile "duble" de tipul **A → u_1u_2**, unde fie $u_1 \in \Sigma$ fie $u_2 \in \Sigma$

se inlocuieste terminalul cu neterminalul U ,

se adauga neterminalul U la V' şi

se adauga productia **U → u_1**, $u_1 \in \Sigma$ (respectiv **U → u_2**, $u_2 \in \Sigma$) la P;

acest pas se repeta pana cand toate "productiile duble" au fost prelucrate;

<u>exemplul 4:</u>

productia R → Ab este inlocuita cu R → AU şi se adauga U → b .

(5) restul productiilor, adica cele de tipul:

$A \rightarrow u_1u_2u_3\ldots u_n$, $n \geq 3$ si $\forall\ 1 \leq i \leq n$: $u_i \in V$ sau $u_i \in \Sigma$,

se inlocuiesc, fiecare, cu setul de productii

$A \rightarrow u_1A_1$,

$A_1 \rightarrow u_2A_2$,

$A_2 \rightarrow u_3A_3$,

…………..

$A_{n-2} \rightarrow u_{n-1}u_n$,

unde $A_1, A_2, \ldots, A_{n-2}$ sunt noi variabile, adaugate lui V'

q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Exemplu 24

Fie G $\in \mathcal{G}_2$ cu productiile:

$S \to ASA \mid aB, \quad A \to B \mid S, \quad B \to b \mid \varepsilon$ =>

? G' $\in \mathcal{G}_2$, FNC echivalenta

(1) $V' = V \cup \{S'\}$ si $P' = P \cup \{S' \to S\}$ =>

$P' = \{S' \to S,\ S \to ASA \mid aB,\ A \to B \mid S, B \to b \mid \varepsilon\}$

(2) eliminam $B \to \varepsilon$ şi apoi adaugam $S \to a$, $A \to \varepsilon$ =>

$P' = \{S' \to S,\ S \to ASA \mid aB \mid a,\ A \to B \mid S \mid \varepsilon,$
$B \to b\}$

obs. ca a aparut o noua ε-productie: $A \to \varepsilon$, deci trebuie eliminata

dupa care aduagam $S \to AS \mid SA \mid S$ =>

$P' = \{S' \to S,\ S \to ASA \mid AS \mid SA \mid S \mid aB \mid a,$
$A \to B \mid S,\ B \to b\}$.

1. **$V' = V \cup \{S'\}$ si $P' = P \cup \{S' \to S\}$**
2. **eliminam ε-productiile $A \to \varepsilon$, $A \neq S$**; apoi, pt $\forall$ aparitie a smb. A in m.dr. al unei productii **adaugam o productie** similara, in care simbolul A respectiv a disparut;
3. **eliminam "productiile unitare $A \to B$;** apoi, pt. $\forall$ productie $B \to u$, $u \in (V \cup \Sigma)^*$, adaugam productia $A \to u$
4. **pentru productiile "duble": $A \to u_1u_2$, unde $u_1 \in \Sigma$ sau $u_2 \in \Sigma$:** se inlocuieste $u \in \Sigma$ cu $U \in V'$ şi se adauga productia $U \to u$ la P;
5. restul productiilor, adica : **$A \to u_1u_2u_3\ldots u_n$, $n \geq 3$, $\forall\ 1 \leq i \leq n$: $u_i \in V$ sau $u_i \in \Sigma$, se inlocuiesc**, fiecare, cu setul de productii $A \to u_1A_1$, $A_1 \to u_2A_2$, $A_2 \to u_3A_3$, …. $A_{n-2} \to u_{n-1}u_n$, unde $A_1, A_2, \ldots, A_{n-2}$ sunt noi variabile, adaugate lui V'.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Exemplu (cont.)

(3) tb. eliminate $S \to S$ si $S' \to S$ cu adaugarile impuse de S' =>

$P'=\{S' \to ASA|AS|SA|aB|a,$
$S \to ASA|AS|SA|aB|a,\ A \to B|S,\ B \to b\}$

(3) tb. eliminate $A \to B$ şi $A \to S$ =>

$P'=\{S' \to ASA|AS|SA|aB|a,$
$S \to ASA|AS|SA|aB|a,\ B \to b\}$

şi apoi aduagarile impuse:

$P'=\{S' \to ASA|AS|SA|aB|a,$
$S \to ASA|AS|SA|aB|a,$
$A \to S|b|ASA|AS|SA|aB|a,\ B \to b\}$

obs. ca avem o noua productie unitara $A \to S$ care tb. eliminata (şi care nu aduce noi productii):

$P'=\{S' \to ASA|AS|SA|aB|a,$
$S \to ASA|AS|SA|aB|a,$
$A \to b|ASA|AS|SA|aB|a,\ B \to b\}$.

1. $V'=V \cup \{S'\}$ si $P'=P \cup \{S' \to S\}$

2. eliminam ε-productiile $A \to \varepsilon$, $A \neq S$; apoi, pt $\forall$ aparitie a smb. A in m.dr. al unei productii **adaugam o productie** similara, in care simbolul A respectiv a disparut;

3. eliminam "productiile unitare $A \to B$; apoi, pt. $\forall$ productie $B \to u$, $u \in (V \cup \Sigma)^*$, adaugam productia $A \to u$

4. pentru productiile "duble":
$A \to u_1 u_2$, unde $u_1 \in \Sigma$ sau $u_2 \in \Sigma$:
se inlocuieste $u \in \Sigma$ cu $U \in V'$ şi
se adauga productia $U \to u$ la P;

5. restul productiilor, adica :
$A \to u_1 u_2 u_3 \ldots u_n$, $n \geq 3$, $\forall\ 1 \leq i \leq n$: $u_i \in V$ sau $u_i \in \Sigma$, se inlocuiesc, fiecare, cu setul de productii
$A \to u_1 A_1,\ A_1 \to u_2 A_2,\ A_2 \to u_3 A_3, \ldots.$
$A_{n-2} \to u_{n-1} u_n$, unde $A_1, A_2, \ldots, A_{n-2}$ sunt noi variabile, adaugate lui V'.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Exemplu (cont.)

(4) tratam productiile duble $S' \to aB$, $S \to aB$ şi $A \to aB$ =>

$V' = V' \cup \{U\}$ şi

$P' = \{S' \to ASA|AS|SA|UB|a,$

$S \to ASA|AS|SA|UB|a,$

$A \to b|ASA|AS|SA|UB|a, B \to b, U \to a\}$

(5) inlocuim productiile $S' \to ASA$, $S \to ASA$, $A \to ASA$ respectiv cu

$S' \to ASA$: $S' \to AX$, $X \to SA$

$S \to ASA$: $S \to AX$, $X \to SA$

$A \to ASA$: $A \to AX$, $X \to SA$ şi $V' = V' \cup \{X\}$ =>

$P' = \{S' \to AX|AS|SA|UB|a,$

$S \to AX|AS|SA|UB|a,$

$A \to AX|AS|SA|UB|a|b, B \to b, U \to a,$

$X \to SA\}$ =>

⇒ $G' = (\{S', S, A, B, U, X\}, \{a, b\}, S', \{B \to b, U \to a, X \to SA, S' \to AX|AS|SA|UB|a, S \to AX|AS|SA|UB|a, A \to AX|AS|SA|UB|a|b\})$

1. $V' = V \cup \{S'\}$ si $P' = P \cup \{S' \to S\}$

2. eliminam ε-productiile $A \to \varepsilon$, $A \neq S$; apoi, pt $\forall$ aparitie a smb. A in m.dr. al unei productii **adaugam o productie** similara, in care simbolul A respectiv a disparut;

3. eliminam "productiile unitare $A \to B$; apoi, pt. $\forall$ productie $B \to u$, $u \in (V \cup \Sigma)^*$, adaugam productia $A \to u$

4. pentru productiile "duble":

$A \to u_1u_2$, unde $u_1 \in \Sigma$ sau $u_2 \in \Sigma$:

se inlocuieste $u \in \Sigma$ cu $U \in V'$ şi

se adauga productia $U \to u$ la P;

5. restul productiilor, adica :

$A \to u_1u_2u_3\ldots u_n$, $n \geq 3$, $\forall\ 1 \leq i \leq n$: $u_i \in V$ sau $u_i \in \Sigma$, se inlocuiesc, fiecare, cu setul de productii

$A \to u_1A_1, A_1 \to u_2A_2, A_2 \to u_3A_3, \ldots.$

$A_{n-2} \to u_{n-1}u_n$, unde $A_1, A_2, \ldots, A_{n-2}$ sunt noi variabile, adaugate lui V'.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

echivalenta cu gramatica independenta de context.

Lema de pompare pentru L.I.C.

Fie $L \subseteq \Sigma^*$, $L \in \mathcal{L}_2$ =>

$\exists$ $p \in N$ (constanta=lungimea de pompare) a.i. $\forall s \in L$, $|s| \geq p \rightarrow$ $\exists$ $u,v,x,y,z \in \Sigma^*$ cu proprietatile:

(i) $s = uvxyz$,

(ii) $\forall i \geq o$: $uv^ixy^iz \in L$,

(iii) $|vy| > 0$,

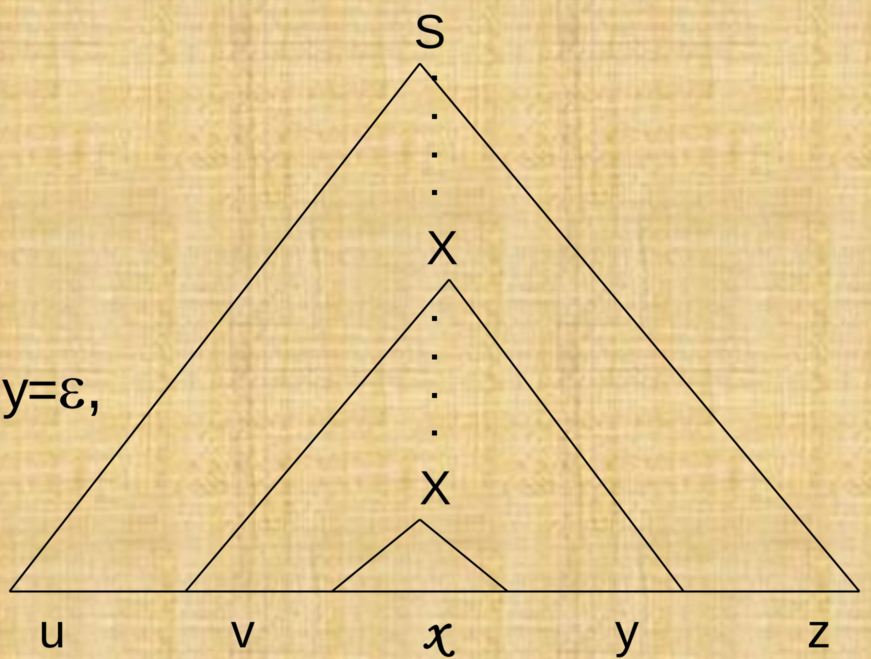
(iv) $|vxy| \leq p$

Obs.:

(iii) elimina cazurile in care fie $v=\varepsilon$ fie $y=\varepsilon$, in care lema e trivial adevarata;

(iv) e utila in a demonstra ca $L \notin \mathcal{L}_2$

Ideea demonstratiei



Demonstratie

Fie $G \in \mathcal{G}_2$ şi $L=L(G)$;

fie $b = \max \{|w| \mid A \to w \in P\}$

(nr. max. de simb., terminale şi neterm., inclusiv repetitii, care pot aparea in m.stg al p, $\forall p \in P$);

=> $\forall\ w \in L(G)$: orice nod din arborele de parsare al w va avea max. b descendenti directi;

=> $\exists$ max. b frunze care se afla la distanta de 1 arc de radacina arborelui (etichetata cu S),

$\exists$ max. b^2 frunze care se afla la distanta de 2 arce de radacina, **……..**

$\exists$ max. b^k frunze care se afla la distanta de k arce de radacina,

=> daca H = inaltimea arborelui şi $H \leq k$, atunci $|w| \leq b^k$;

reciproc: daca $|w| \geq b^k+1$, atunci $\forall$ dintre arborii sai de parsare va avea o inaltime $H \geq k+1$;

=> definim ct. de pompare: **$p = b^{card(V)+1}$**, unde $b=\max\{|w| \mid A \to w \in P\}$ =>

=> $\forall\ w \in L(G)$: daca $|w| \geq k$ atunci inaltimea arborelui sau de parsare va fi: $H \geq card(V)+1$

deorece $b^{card(V)+1} \geq b^{card(V)}+1$; ***(1)***

Obs. ca, daca n =card (V) , putem pp. $n \geq 2$:

n=2, daca G e in FNC

n>2, altfel.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Demonstratie (cont.)

Cum putem “pompa” w?

fie τ arborele de parsare al w, care are cel mai mic nr. de noduri,

cf. (1). $H_\tau \geq |V|+1$; atunci

=> $\exists$ un drum in H_τ de la radacina la o frunza de lg. $l \leq |V|+1$

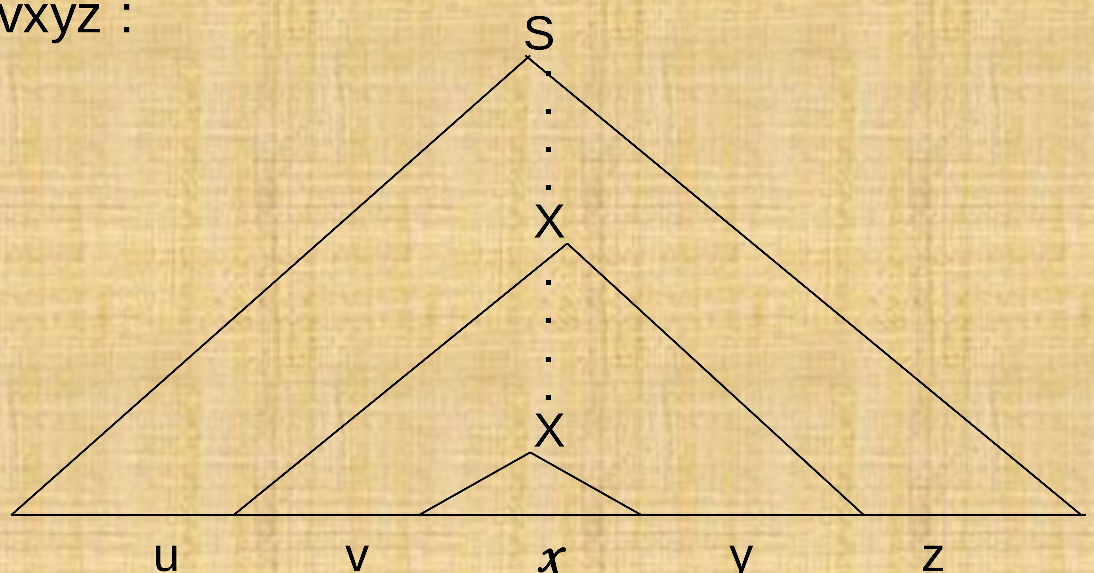
=> acest drum are cel putin $|V|+2$ noduri, dintre care 1! nod e etichetat cu un terminal

=> acest drum are cel putin $|V|+1$ noduri etichetate cu neterm.

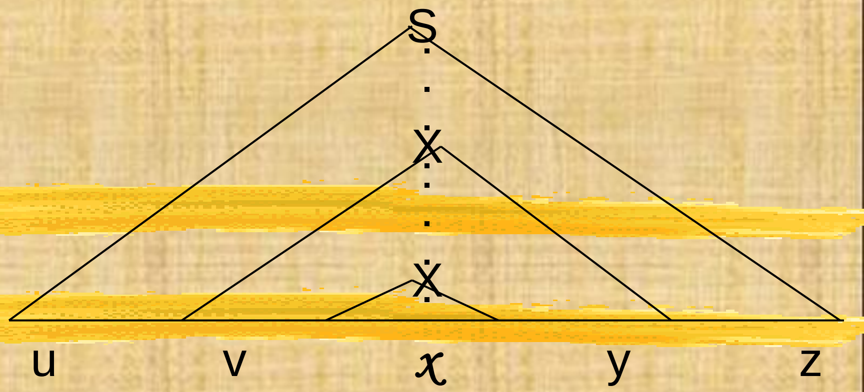
=> cel putin un neterminal se repeta de-a lungul acestui drum

alegem neterminalul care se repeta cel mai aproape de frunza;

Descompunem w=uvxyz :



LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT



Demonstratie (cont.)

<u>verificam cond.(ii) :</u> $\forall i \geq o$: $uv^ixy^iz \in L$,

fiecare aparitie a lui X "anunta" un subarbore care genereaza un subcuv. al cuv. w:

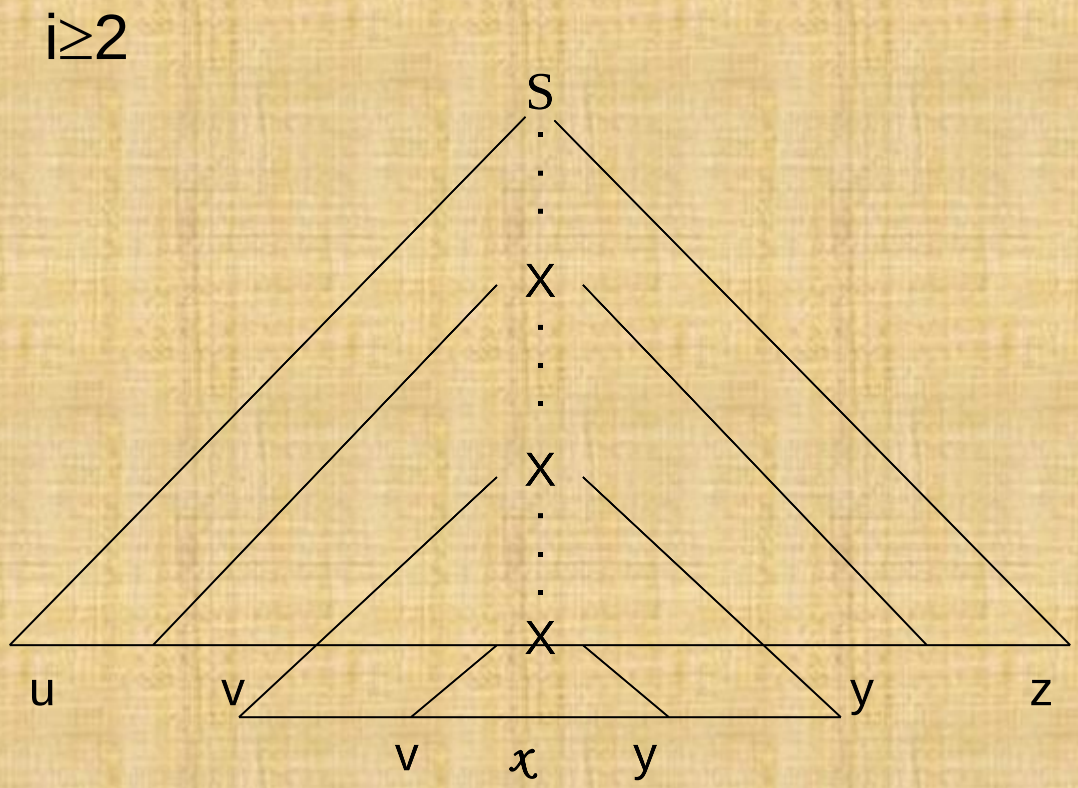
- "prima" aparitie reprezinta radacina subarborelui celui mai mare, cel care genereaza vxy;
- "ultima" aparitie reprezinta radacina subarborelui celui mai mic, cel care genereaza x;

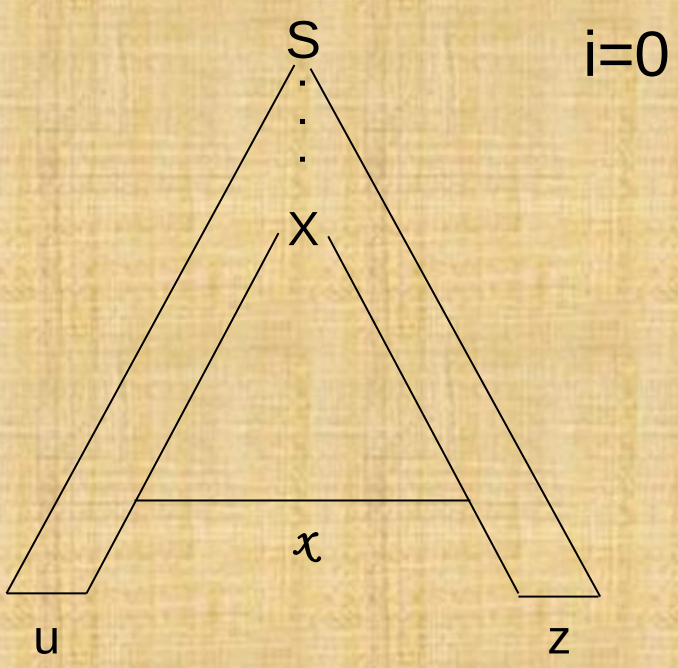
intrucat cei 2 subarbori au aceeasi radacina (etichetata X), i.e. sunt generati de acelasi neterminal $\rightarrow$ putem substitui pe unul cu celalalt obtinand tot un arbore de derivare corect

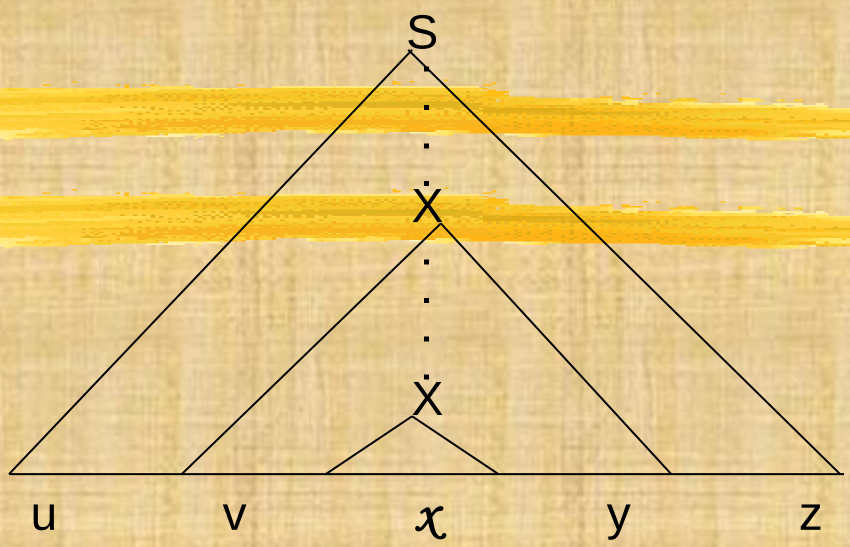
=> daca inlocuim, repetat:

- cel mai mic arbore cu cel mai mare, obtinem arborii de parsare ai cuvintelor uv^ixy^iz, $i>1$;
- cel mai mare arbore cu cel mai mic, obtinem arborele de parsare al cuvantului uxz (=> cond. (ii));

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT







Demonstratie (cont.)

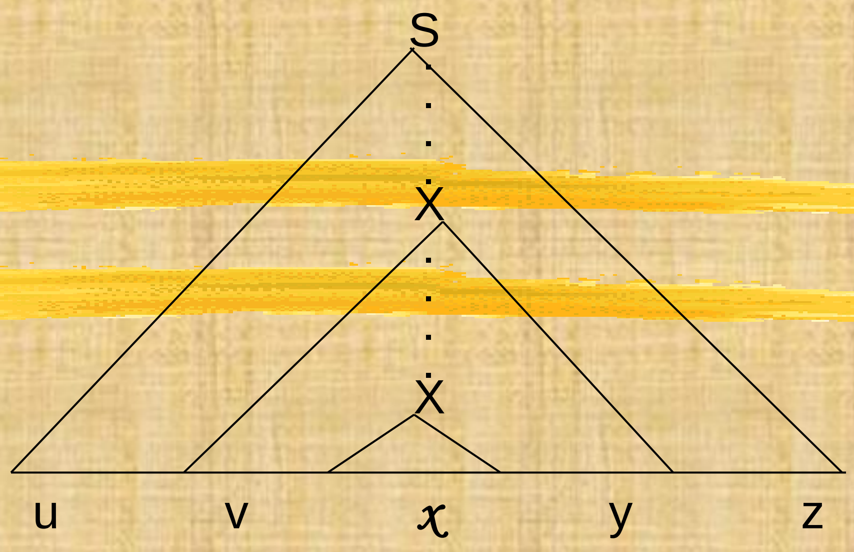
verificam cond.(iii) : $|vy| > 0$,

ppa $v=\varepsilon=y$

=> arborele de parsare obtinut prin inlocuirea celui mai mare subarbore de radacina X cu cel mai mic ar avea mai putine noduri decat τ şi totusi ar genera cuvantul w

=> contradictie cu alegerea lui τ ca fiind arborele de parsare al lui w cu cel mai mic nr. de noduri.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT



Demonstratie (cont.)

<u>verificam cond.(iv) : </u>(iv) $|vxy| \le p$

stim ca in τ, prima aparitie a lui X genereaza cel mai mare subarbore al lui w, şi anume subarborele asociat lui vxy;

dar X a fost ales a.i. ambele sale aparitii sa fie cat mai aproape de frunza

=> aceste aparitii se afla printre ultimele |V|+1 noduri de pe drum ***(2)***

in plus, drumul a fost ales ca fiind cel m. lung de la radacina la frunza ***(3)***

din (2) şi (3) rezulta ca inaltimea H a subarborelui de radacina X care genereaza vxy este $H \le |V|+2$ =>

acest arbore poate genera un cuvant c, $|c| \le b^{|V|+2} = p$. q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

   echivalenta cu gramatica independenta de context.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Definitii 26

$\forall L_1, L_2 \subseteq \Sigma^*$:

$L_1 \cup L_2 = \{w \in \Sigma^* \mid w \in L_1 \text{ sau } w \in L_2\}$,

$L_1 \cap L_2 = \{w \in \Sigma^* \mid w \in L_1 \text{ si } w \in L_2\}$,

$L_1 \setminus L_2 = \{w \in \Sigma^* \mid w \in L_1 \text{ si } w \notin L_2\}$,

$L_1 \circ L_2 = \{w_1 w_2 \in \Sigma^* \mid w \in L_1 \text{ si } w \in L_2\}$,

$mi(L) = \{mi(w) \mid w \in L\}$.

$$\overline{L} = \{w \in \Sigma^* \mid w \notin L\} = \Sigma^* \setminus L,$$

$$L^n : \begin{cases} L^0 = \varepsilon \ \ si \\ L^{n+1} = L \cdot L^n = L \cdot L^n, \forall n \in N, \end{cases}$$

$$\mathrm{L} \cdot \varnothing = \varnothing \cdot L = \varnothing,$$

$$\mathrm{L} \cdot \{\varepsilon\} = \{\varepsilon\} \cdot L = L,$$

$$L^* = \bigcup_{i=0}^{\infty} L^i, \quad L^+ = \bigcup_{i=1}^{\infty} L^i.$$

Lema 27

$\mathcal{L}_2$ este inchisa la reuniune, concatenare si operatia *.

Demonstratie

Fie $L_1, L_2 \in \mathcal{L}_2$, $L_i = L(G_i)$, unde $G_i = (V_i, \Sigma_i, S_i, P_i)$, $\forall$ i=1,2:

$L_1 \cup L_2$ este generat de $G' = (V', \Sigma', S', P')$, unde:

$V' = V_1 \cup V_2 \cup \{S_0\}$,

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

$S' = S_0$,

$P' = P_1 \cup P_2 \cup \{S_0 \rightarrow S_1, S_0 \rightarrow S_2\}$.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

$L_1 \circ L_2$ este generat de $G'=(V', \Sigma', S', P')$, unde:

$V'=V_1 \cup V_2 \cup \{S_0\}$,

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

$S'=S_0$,

$P'=P_1 \cup P_2 \cup \{S_0 \rightarrow S_1S_2\}$;

L^* este generat de $G'=(V', \Sigma', S', P')$, unde:

$V'=V \cup \{S'\}$,

$\Sigma'= \Sigma$,

S'=simbolul de start,

$P'=P \cup \{S' \rightarrow S'S\}$;

Evident, toate cele 3 gramatici sunt independente de context q.e.d.

Lema 28

$\mathcal{L}_2$ este inchisa la operatia mirror

Demonstratie

(i) Fie $L \in \mathcal{L}_2$, $L=L(G)$, unde $G=(V, \Sigma, S, P)$;

putem pp ca G este in FNC

=> construim $G'=(V, \Sigma, S, P')$, unde

$P' = \{A \rightarrow CB \mid \exists\ A \rightarrow BC \text{ in } G\} \cup \{A \rightarrow a \mid \forall\ A \rightarrow a \text{ in } G\}$

$L(G')=mi(L)$, i.e. : $S_G \Rightarrow^* w \Leftrightarrow S_{G'} \Rightarrow^* mi(w)$, adica:

$S \Rightarrow x_1 \Rightarrow x_2 \Rightarrow \ldots \Rightarrow x_n$ este o derivare in G $\Leftrightarrow$

$S \Rightarrow mi(x_1) \Rightarrow mi(x_2) \Rightarrow \ldots \Rightarrow mi(x_n)$ este o derivare in G'

demonstratie prin inductie dupa n= nr pasi ai derivarii lui x in G **.**

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

<u>n=1</u>

$\Rightarrow x = \varepsilon$ sau $x \in \Sigma$ => evident

(se aplica productia $x \rightarrow \varepsilon$ sau $x \rightarrow a$ dar $a = mi(a)$ deci avem si productia corespunzatoare in G'.)

<u>n≥1</u>

Pp. afirmatia adevarata pt derivarea cuvantului **x** in n ≥ 1 pasi si demonstram pt o derivare cu n+1 pasi:

Fie $S_G \Rightarrow^* x$, respectiv $S_{G'} \Rightarrow^* mi(x)$ in n+1 pasi =>

1) Prima productie este obligatoriu de forma $S \rightarrow AB$, respectiv $S \rightarrow BA$

2) $\exists\, y \in \Sigma^*$ a.i. $x = yB \Rightarrow mi(x) = mi(yB) = Bmi(y)$

=> lg derivarii lui y in G (şi a lui *mi(y)* in G') este n

$\Downarrow$ cf. ip.ind. pt y

$\exists$ o derivare pt y in G iff $\exists$ o derivare pt *mi(y)* in G'.

$\Downarrow$ compunere cu $S \rightarrow AB$ ($S \rightarrow BA$)

$\exists\ S_G \Rightarrow^* x \Leftrightarrow S_{G'} \Rightarrow^* mi(x)$. q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Lema 29

$\mathcal{L}_2$ este inchisa la substituie şi omomorfism

Demonstratie

(i) Fie $\Sigma=\{a_1,a_2,\ldots,a_n\}$ şi Ψ 2 alfabete oarecare;

fie $L\subseteq\Sigma^*$, $L\in\mathcal{L}_2$, $L=L(G)$, unde $G=(V, \Sigma, S, P)\in\mathcal{G}_2$ şi

fie $s : \Sigma \to \mathcal{P}(\Psi^*)$ o substitutie a.i. $s(a_i) = L(G_i)$, unde $G_i=(V_i, \Sigma_i, S_i, P_i)\in\mathcal{G}_2$,

trebuie sa construim $G'\in\mathcal{G}_2$ a.i. $L(G') = s(L) \subseteq \Psi^*$

putem pp ca: $V\cap V_1\cap V_2\cap\ldots\cap V_n=\emptyset$

$G, G_1, G_2, \ldots, G_n$ sunt in FNC;

construim $G' = (V', \Psi, S', P')$ astfel:

$V' = V\cup V_1\cup V_2\cup\ldots\cup V_n$

$\Psi = \Sigma_1\cup\Sigma_2\cup\ldots\cup\Sigma_n$

$S' = S$

$P' = P_0\cup P_1\cup P_2\cup\ldots\cup P_n$, unde productiile din P_0 se obtin din productiile din P inlocuind fiecare aparitie a unui terminal a_i cu axioma S_i, $1\leq i\leq n$. q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Lema 30

$\mathcal{L}_2$ NU este inchisa la intersectie si complementara

Demonstratie

(i) Ppa. ca $\mathcal{L}_2$ este inchisa la $\cap$.

Fie $G_1=(\{S,T\},\{a,b,c\},\{S\rightarrow Sc \mid Tc,\ T\rightarrow aTb \mid ab\},S)$

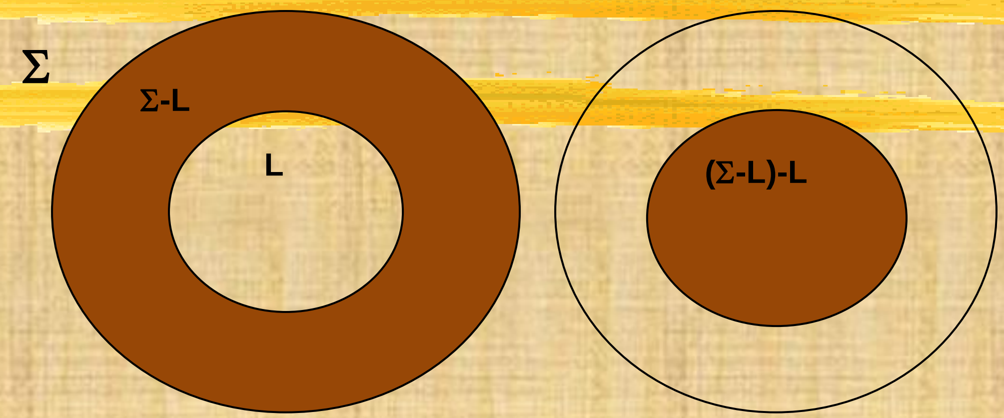
$G_2=(\{S,T\},\{a,b,c\},\{S\rightarrow aS \mid aT,\ T\rightarrow bTc \mid bc\},S)$

=> $L(G_1)=\{a^nb^nc^m \mid n,m\geq 1\} \in \mathcal{L}_2$

$L(G_2)=\{a^mb^nc^n \mid n,m\geq 1\} \in \mathcal{L}_2$

Observam ca $L(G_1)\cap L(G_2)=\{a^nb^nc^n \mid n\geq 1\} \notin \mathcal{L}_2$ (cf. Lema pompare).

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT



(ii) Ppa ca $\mathcal{L}_2$ este inchisa la complementara;

fie $L=\{a^k b^j c^n \mid k,j,n\geq 1,\ k\neq j\neq n\}$; $L\in\mathcal{L}_3\subset\mathcal{L}_2$: $L=L(G)$, unde

$G=(\{S,A,B,C\}, \{a,b,c\}, S, \{S\to aA,\ A\to aA|bB,\ B\to bB|cC,\ C\to cC|\varepsilon\})$

$\Rightarrow \mathcal{C}(L) = \{a^k b^j c^n \mid k,j,n\geq 1,\ k=j=n\} = \{a^n b^n c^n \mid n\geq 1\}\notin\mathcal{L}_2$

Alta dem: fie $L\in\mathcal{L}_2 \Rightarrow L=\Sigma \cap L$ dar $\Sigma \cap L= \mathcal{C}(\mathcal{C}(L))$.

ppa ca $\mathcal{L}_2$ este inchisa la complementara => $\mathcal{L}_2$ este inchisa la intersectie

=> contradictie cu (i)   q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

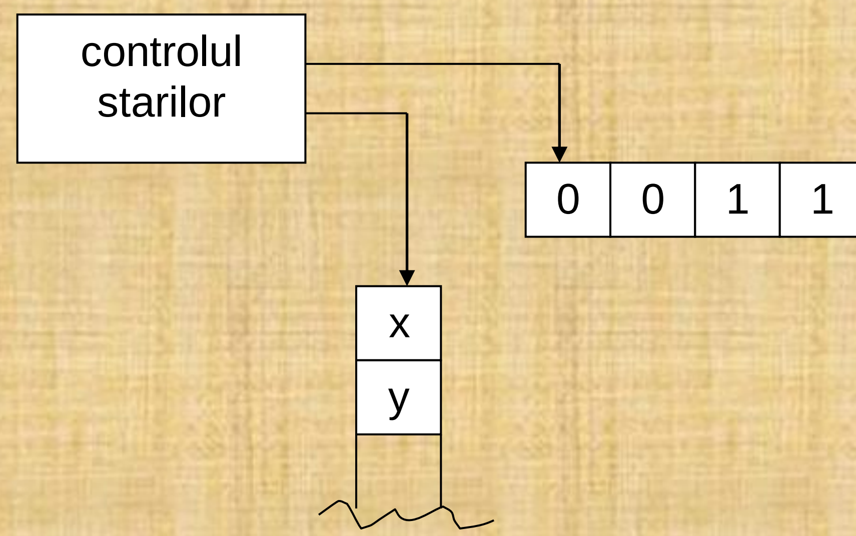
echivalenta cu gramatica independenta de context.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

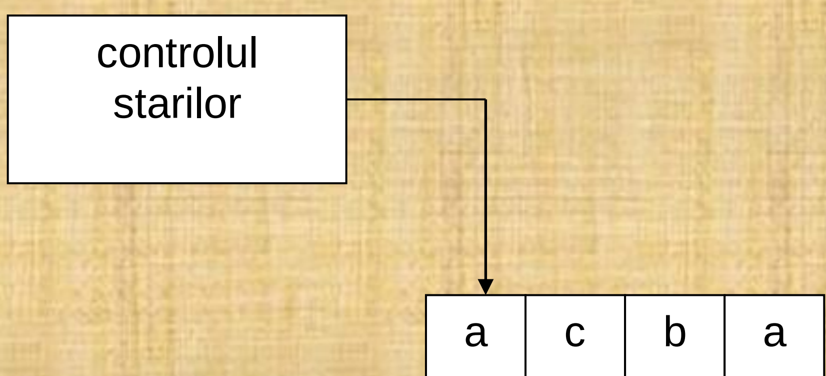
APD reprezinta un nou model de calculabilitate

Stiva

APD



AF



$\{0^n1^n \mid n \in N\}$.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Observatii 32: Deosebiri intre APD si AFD:

(i) La APD trebuie sa luam in considerare:

- ✓ setul de stari,
- ✓ banda de intrare,
- ✓ stiva;

Simbolurile scrise in stiva pot fi preluate din acelasi alfabet / din alt alfabet decat cel al benzii de intrare

=> in definitia APD vom avea 2 alfabete: Σ si Γ,

urmatoarea actiune a APD este determinata de:

- ✓ starea crt. a APD,
- ✓ simbolul citit de pe banda de intrare
- ✓ simbolul aflat in varful stivei,

(ii) APD poate intalni si pe banda de intrare si in stiva smb. vid, ε =>

$dom(\delta) = Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\})$

(iii) Urmatoarea actiune a APD poate consta in trecerea intr-o noua stare si EVENTUAL scrierea unui simbol in stiva;

In plus, modelul APD fiind intrinsec nedeterminist,

APD poate trece in diferite stari => $codom(\delta) = \mathcal{P}(Q \times (\Gamma \cup \{\varepsilon\}))$.

Definitie 33

Automat pushdown = APD=(**Q, Σ, Γ, δ, q_0, F**), 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, ($\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$);

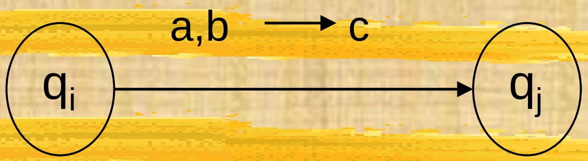
Γ= multime finita, nevida, numita alfabetul stivei, ($\Gamma_\varepsilon = \Gamma \cup \{\varepsilon\}$);

$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$, numita functia de tranzitie;

$q_0 \in Q$, numita starea initiala;

$F \subseteq Q$ numita multimea starilor finale.

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT



Notatie 34

⇔ APD, aflat in starea qi,

✓ citeste smb. a de pe banda de intrare,

✓ extrage smb. b din stiva,

✓ depune smb. c in stiva;

- Daca $a = \varepsilon$ ⇒ APD face tranzitia şi fara sa citeasca nimic de pe banda de intrare,
- Daca $b = \varepsilon$ ⇒ APD face tranzitia şi fara sa extraga nimic din stiva,
- Daca $c = \varepsilon$ ⇒ APD face tranzitia şi fara sa depuna nimic din stiva.

$$q_i \xrightarrow{a,b \to c} q_j$$

Observatie 35

(i) Definitia formala a APD NU contine nici un mecanism explicit de testare a vidarii stivei $\Rightarrow$

METODA STANDARD:

- ✓ se utilizeaza un smb. special din Γ, \$, care este depus in stiva de la inceput,
- ✓ cand acest smb. este intalnit (cand s-a ajuns in varful stivei) inseamna ca stiva s-a golit;

(ii) Analog, definitia formala a APD NU contine nici un mecanism explicit de testare a terminarii secventei de intrare $\Rightarrow$

METODA STANDARD:

- ✓ trecerea intr-o stare finala.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

Teorema 36

Fie L $\subseteq\Sigma^*$ => ($\exists$ G$\in$ $\mathcal{G}_2$: L=L(G) $\Leftrightarrow$ $\exists$ A $\in$ APD: L=L(A))

Demonstratie "$\Rightarrow$"

Fie L $\subseteq\Sigma^*$ şi fie G=(V, Σ, S, P)$\in\mathcal{G}_2$ ai. L(G)=L;

putem defini un APD R=(Q, Σ_ε,Γ_ε, δ, q_0, F) cu ajutorul lui G astfel:

Q = {q_{start}, q_{loop}, q_{accept}} $\cup$ E:

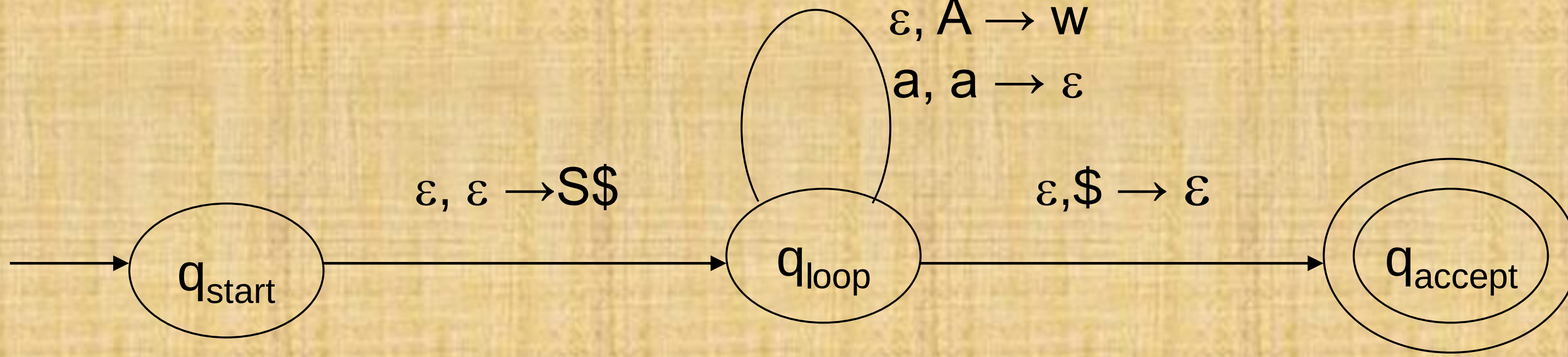
E = multimea starilor auxiliare necesare implementarii depunerii in stiva a secventelor intermediare din derivarea S $\Rightarrow^*$ w, w $\in$ L;

Σ_ε, Γ_ε depind de limbajul L considerat;

q_0= q_{start};

F = {q_{accept}};

LFA: C4 – LIMBAJE INDEPENDENTE DE CONTEXT



$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ definita prin:

$\delta(q_{start},\varepsilon,\varepsilon)=\{(q_{loop},S\$)\}$

$\delta(q_{loop},\varepsilon,A)=\{(q_{loop},w) \mid \exists\ A \rightarrow w \in P\ \}$

$\delta(q_{loop},a,a)=\{(q_{loop}, \varepsilon)\}$

$\delta(q_{loop},\varepsilon,\$)=\{(q_{accept}, \varepsilon)\}$.

q.e.d.

$\mathcal{LFA}$: C4 – LIMBAJE INDEPENDENTE DE CONTEXT

1. Definitii
   - gramatici şi limbaje independente de context
   - arbori de derivare
   - proiectarea g.i.c.
2. Ambiguitate
3. Forme normale
   - definitii
   - exemple
   - aducerea la forma normala Chomsky
4. Lema de pompare
5. Operatii de inchidere
   - $\mathcal{L}_2$ este inchisa la reuniune, concatenare, operatia *, omomorfism
   - $\mathcal{L}_2$ nu este inchisa la intersectie si complementara
6. Automatul pushdown;

echivalenta cu gramatica independenta de context.