

# **Gramatici independente de context (GIC) (CFG – context free grammars)**

derivari, arbori de derivare,  
tipuri de gramatici independente de context,  
gramatici echivalente - constructii

# Ne reamintim: Gramatica

O gramatica este un cvadruplu  $\mathbf{G} = (\mathbf{N}, \Sigma, \mathbf{P}, \mathbf{S})$

- $\mathbf{N}$  este un alfabet de simboluri ***neterminale***
- $\Sigma$  este un alfabet de simboluri ***terminale***
- $\mathbf{N} \cap \Sigma = \emptyset$
- $\mathbf{P} \subseteq (\mathbf{N} \cup \Sigma)^* \mathbf{N} (\mathbf{N} \cup \Sigma)^* \times (\mathbf{N} \cup \Sigma)^*$   
     $\mathbf{P}$  multime finită                      (multimea regulilor de productie)
- $\mathbf{S} \in \mathbf{N}$                                       (simbolul de start - simbolul initial)

Notatie:

$(\alpha, \beta) \in \mathbf{P}$  se noteaza:  $\alpha \rightarrow \beta$

( $\alpha$  se înlocuieste cu  $\beta$ )

# Ne reamintim: clasificarea Chomsky

- Gramatici de tip 0:  
nici o restricție (*suplimentară*) referitoare la forma regulilor de producție
  - Gramaticile de tip 1 (gramatici monotone)
    - $\forall \alpha \rightarrow \beta \in P: |\alpha| \leq |\beta|$
    - caz special:  $S \rightarrow \epsilon$  poate  $\in P$ . În acest caz  $S$  nu apare în membrul drept al nici unei reguli de producție.
- 

- **Gramatici independente de context:**

reg. producție sunt de forma  $A \rightarrow \alpha$ ,  $A \in N$ ,  $\alpha \in (N \cup \Sigma)^*$   
(gramatici de tip 2)

---

- Gramaticile de tip 3:  
reg. prod. sunt de forma
  - $A \rightarrow aB$
  - $A \rightarrow b$unde  $A, B \in N$  și  $a, b \in \Sigma$   
caz special:  $S \rightarrow \epsilon$  poate  $\in P$ . În acest caz  $S$  nu apare în membrul drept al nici unei reguli de producție.

# derivari de stanga/ dreapta

- *derivare de stânga*  $\Rightarrow_{st}$   
o derivare directă în care se înlocuiește cel mai din stânga neterminal
- *derivare de dreapta*  $\Rightarrow_{dr}$   
o derivare directă în care se înlocuiește cel mai din dreapta neterminal

# Analiza sintactica

- *analiză sintactică* pt. cuvântul  $w$   
succesiunea de derivări directe:
  - $S \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \dots \Rightarrow \alpha_n = w$   
altfel spus: reprezintă o derivare pentru cuvântul  $w$
- *analiză sintactică descendentă*  
dacă această succesiune de derivări directe se obține pornind de la  $S$  și terminând cu  $w$
- *analiză sintactică ascendentă*  
dacă această succesiune de derivări directe se obține pornind de la  $w$  și terminând cu  $S$

# Arbore de derivare

- **Fie  $G = (N, \Sigma, P, S)$**  o gramatică independentă de context. Numim *arbore de derivare* sau *arbore de analiză sintactică* un arbore cu radacina, ordonat, cu următoarele proprietati:
  1. Orice nod interior - o eticheta din  $N$ ;
  2. Orice nod frunza - o *etichetă* din  $\Sigma \cup \{\epsilon\}$
  3. Eticheta rădăcinii este  $S$ ;
  4. Dacă un nod are eticheta  $A$  iar nodurile succesoare acestuia, în ordine de la stânga la dreapta sunt etichetate cu  $X_1, X_2, \dots, X_n$  atunci  $A \rightarrow X_1 X_2 \dots X_n$  trebuie să fie o producție din  $P$ .

# Arbore de derivare

- **frontiera (frontul):** nodurile terminale, în ordine de la stânga la dreapta
- etichetele lor formeaza o secventa peste  $\Sigma^*$
- obs: denumirea de frontiera (front) se foloseste si pentru a denumi succesiunea etichetelor nodurilor terminale

## Teoremă.

Fie  $G = (N, \Sigma, P, S)$  o gramatică independentă de context. Un cuvânt  $w$  peste alfabetul  $\Sigma$ , deci din  $\Sigma^*$ , apartine limbajului generat de  $G$ , adică  $w \in L(G)$ , dacă și numai dacă  $w$  este frontul unui arbore de analiză sintactică.

# Gramatica ambigua

O gramatică  $G = (N, \Sigma, P, S)$  independentă de context este *ambiguă*

dacă și numai dacă există cel puțin un cuvânt  $w$  care admite doi arbori de derivare distincti; în caz contrar gramatica este *neambiguă*.

- $\Leftrightarrow \exists$  2 analize sintactice care folosesc numai derivări de stanga, diferite
- $\Leftrightarrow \exists$  2 analize sintactice care folosesc numai derivări de dreapta, diferite

# Descreri echivalente. Forme normale

**Ne reamintim :**

O gramatica  $G_1$   
este echivalenta cu gramatica  $G_2$   
ddaca  $L(G_1)=L(G_2)$

g.i.c.

# $\varepsilon$ -productii si gram. $\varepsilon$ -independente

- **$\varepsilon$ -productie** : o productie de forma  $A \rightarrow \varepsilon$
- Gramatica  $G = (N, \Sigma, P, S)$  este  **$\varepsilon$ -independentă** daca:
  - a) dacă  $\varepsilon \notin L(G)$  atunci  $G$  nu are  $\varepsilon$ -productii
  - b) dacă  $\varepsilon \in L(G)$  atunci avem o singură productie  $S \rightarrow \varepsilon$  iar celelalte productii nu-l contin în membrul drept pe  $S$
- Teorema
  - $\forall G = (N, \Sigma, P, S)$
  - $\exists G' = (N', \Sigma', P', S)$  echivalentă,  $\varepsilon$ -independentă

# Redenumiri. Cicluri.

- ***redenumire***: reg.prod. de forma  $A \rightarrow B$
- ***Gramatica fără redenumiri***: fara r.p. de redenumire

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă fără redenumiri

- ***ciclu***: o  $*$  derivare de forma  $A \Rightarrow^* B$
- ***Gramatica fără cicluri***: nu se pot obtine cicluri (la derivare)

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă fără cicluri

# Recursivitate

- reg.prod. recursiva la stanga:
  - o reg.prod. de forma:  $A \rightarrow A\alpha$
- reg.prod. recursiva la dreapta:
  - o reg.prod. de forma:  $A \rightarrow \alpha A$
- reg.prod. recursiva:
  - o reg.prod. de forma:  $A \rightarrow \alpha A\beta$

# Recursivitate

- **neterminal recursiv la stanga:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ A\alpha$

- **neterminal recursiv la dreapta:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ \alpha A$

- **neterminal recursiv:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ \alpha A \beta$

- **gramatica recursiva la stanga:**

are cel putin un neterminal recursiv la stanga

- **gramatica recursiva la dreapta:** ...

# Forma normală Chomsky

O gramatică independentă de context  $G = (N, \Sigma, P, S)$  este în *forma normală Chomsky (FNC)*

dacă orice regula de producție din  $P$  este de una din formele:

- a)  $A \rightarrow BC$                        $A, B, C \in N$ ;
- b)  $A \rightarrow a$                          $a \in \Sigma, A \in N$ ;

Si un caz special:  $S \rightarrow \varepsilon$  poate  $\in P$ . In acest caz  $S$  nu apare în membrul drept al nici unei reguli de producție.

**Teoremă.** Oricare ar fi  $G=(N, S, P, S)$  o gramatică independentă de context, întotdeauna există o gramatică în forma normală Chomsky  $G'$ , astfel încât  $L(G) = L(G')$ .

# Forma normală Greibach

O gramatică  $G = (N, \Sigma, P, S)$   
este în *forma normală Greibach (FNG)*  
dacă  $P$  are productii numai de forma:

$$A \rightarrow a\alpha, \quad A \in N, a \in \Sigma, \alpha \in N^*;$$

Si un caz special:  $S \rightarrow \varepsilon$  poate  $\in P$ . In acest caz  $S$  nu apare în membrul drept al nici unei reguli de productie.

**Teoremă.** Oricare ar fi  $G = (N, \Sigma, P, S)$  o gramatică independentă de context, întotdeauna există o gramatică în forma normală Greibach, astfel încât  $L(G)=L(G')$ .

# Simplificarea GIC

- *simbol neproductiv*

Un simbol  $A \in N$  este *neproductiv* dacă nu există nici o derivare de forma  $A \xRightarrow{*} \mathbf{x} \quad (\mathbf{x} \in \Sigma^*)$

- în caz contrar  $A$  este *simbol productiv*

- Teorema

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă, fără simboluri neproductive

# Simplificarea GIC

(transformari echivalente)

- *simbol inaccesibil*

Un simbol  $X \in N \cup \Sigma$  este *simbol inaccesibil* dacă nu există nici o  $*$  derivare:  $S \Rightarrow^* \alpha X \beta$  ( $\alpha, \beta \in (N \cup \Sigma)^*$ )

- în caz contrar simbolul este *accesibil*

- Teorema

$\forall \mathbf{G} = (N, \Sigma, P, S) \exists \mathbf{G}' = (N', \Sigma', P', S)$  echivalentă, fără simboluri inaccesibile

# Determinarea simbolurilor productive

$\approx$  algoritm AF determ. stari productive

$A \rightarrow BC$
$B \rightarrow bB$
$C \rightarrow c$

1.  $i := 0$  ;  $V_0 := \Phi$

2. **Repeta**

$V_{i+1} := V_i \cup \{A \in N \mid \exists A \rightarrow \alpha \in P, \alpha \in (V_i \cup \Sigma)^*\}$

$i := i + 1$

**pana cand**  $V_i = V_{i-1}$

$\{V_i - \text{multimea simbolurilor productive}\}$

# Determinarea simbolurilor accesibile

$\approx$  algoritm AF determ. stari accesibile

1.  $i:=0$  ;  $V_0:=\{S\}$

2. **Repeta**

$V_{i+1} := V_i \cup \{ B \in N \mid \exists A \in V_i, \alpha, \beta \in (N \cup \Sigma)^* \text{ a.i. } A \rightarrow \alpha B \beta \in P \}$

$i:=i+1$

**pana cand**  $V_i = V_{i-1}$

$\{ V_i - \text{multimea simbolurilor neterminale accesibile} \}$

\* analog pentru simboluri terminale accesibile

# Simplificarea GIC

- Un simbol este **neutilizabil** dacă el este fie inaccesibil, fie neproductiv
- Teorema  
 $\forall \mathbf{G} = (\mathbf{N}, \Sigma, \mathbf{P}, \mathbf{S}) \exists \mathbf{G}' = (\mathbf{N}', \Sigma', \mathbf{P}', \mathbf{S})$  echivalentă fără simboluri neutilizabile



## Observatii:

Fie  $G = (N, \Sigma, P, S)$  o gramatica independenta de context:

- Fie un simbol neterminal  $A$  al gramaticii  $G$ .

Daca nu exista o regula de productie  $A \rightarrow \alpha$  in  $P$   
atunci  $A$  este neproductiv

- Fie un simbol terminal  $a$  al gramaticii  $G$ .

Daca nu exista o regula de productie de forma  
 $B \rightarrow \alpha a \beta$  in  $P$   
atunci  $a$  este inaccesibil

# *Eliminarea $\varepsilon$ -productiilor*

1. Construim multimea  $N_\varepsilon$  care are ca elemente acele neterminale care prin derivare conduc la  $\varepsilon$  adică :
  - $N_\varepsilon = \{A \mid A \in N, A \Rightarrow^* \varepsilon\}$   
alg.  $\approx$  determinarea simb. productive
2. Determinam noile reguli de productie
  - astfel incat productiile de forma  $A \rightarrow \varepsilon$  se elimina
  - dar, daca  $\varepsilon \in L(G)$ , atunci  $\exists S \rightarrow \varepsilon$  si  $S$  nu apare în membrul drept al nici unei productii

# Determinarea lui $N_\varepsilon$

1.  $i:=0$  ;

$$V_0 := \{A \in N \mid \exists A \rightarrow \varepsilon \in P\}$$

2. **Repeta**

$$V_{i+1} := V_i \cup \{A \in N \mid \exists A \rightarrow \alpha \in P, \alpha \in (V_i)^*\}$$

$$i:=i+1$$

**pana cand**  $V_i = V_{i-1}$

# determinam noile reguli de productie

- productiile de forma  $A \rightarrow \varepsilon$  se elimina
- celelalte r.p. se rescriu astfel incat sa “suplineasca” eliminarea  $\varepsilon$ -productiilor astfel:

Fie r.p.  $A \rightarrow \alpha_0 B_1 \alpha_1 B_2 \alpha_2 \dots B_k \alpha_k$

unde:  $B_i \in N_\varepsilon$

$\alpha_j$  nu contine simb. din  $N_\varepsilon$

Se inlocuieste cu:

$A \rightarrow \alpha_0 X_1 \alpha_1 X_2 \alpha_2 \dots X_k \alpha_k$

unde  $X_i = \begin{cases} B_i \\ \varepsilon \end{cases}$

este unul dintre  $B_i$  sau  $\varepsilon$   
(se fac toate inlocuirile posibile)

**Ce lipseste ???**

# determinam noile reguli de productie

- continuare

Dacă  $\varepsilon \in L(G)$  trebuie sa avem o  $\varepsilon$ -productie

*“atunci avem productia  $S \rightarrow \varepsilon$  si  $S$  nu apare în membrul drept al nici unei productii”*

(gram.  $\varepsilon$ -independenta)

- adaugam un nou simbol de start  $S'$  si productiile  
 $S' \rightarrow \varepsilon \mid S$

# Redenumiri. Cicluri.

- *redenumire*: reg.prod. de forma  $A \rightarrow B$
- *Gramatica fără redenumiri*: fara r.p. de redenumire

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă fără redenumiri

- *ciclu*: o  $*$  derivare de forma  $A \Rightarrow^* B$
- *Gramatica fără cicluri*: nu se pot obtine cicluri (la derivare)

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă fără cicluri

# Eliminarea redenumirilor

**PP. G –  $\varepsilon$ -independenta** (daca nu , luam gr.echiv.  $\varepsilon$ -ind.)

Pentru fiecare  $A \in N$

se elimina redenumirile de forma  $A \rightarrow B$  ( $\forall B \in N$  )

- construiesc multimea  $N_A = \{B \mid A \xrightarrow{*} B\}$ ;  
( $\approx$  det. simb. accesibile)
- determinam noile reguli de productie

# Construieste $N_A = \{D \mid A =^* D\}$

1.  $i := 0$  ;

$V_0 := \{A\}$

**2. Repeta**

$V_{i+1} := V_i \cup \{C \mid (B \rightarrow C) \in P, B \in V_i\}$

$i := i + 1$

**pana cand**  $V_i = V_{i-1}$

$N_A := V_i$

# determinam noile reguli de productie

$A \in N$ :

- **pentru** fiecare  $A \rightarrow \alpha \in P$  **executa**
- **daca**  $\alpha$  e format dintr-un singur neterminal  
**atunci**  
il excludem din mult. noilor reg.prod
- **altfel**  
adaugam:  $B \rightarrow \alpha$  ,  $\forall B \in N$  a.i.  $A \in N_B$
- **sf.daca**
- **sf.pentru**

# Eliminarea redenumirilor

## Exercitiu:

$$E \rightarrow E+T$$

$$E \rightarrow T$$

$$T \rightarrow T * F$$

$$T \rightarrow F$$

$$F \rightarrow (E)$$

$$F \rightarrow a$$

# Gramatica fara cicluri

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă  
fără cicluri

Daca  $G$  –  $\varepsilon$ -independenta si fara redenumiri  
atunci este fara cicluri

# Gramatica proprie:

este o gramatica

- fara simb. neutilizabile
- **$\epsilon$ -independenta**
- fara cicluri

Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  proprie echiv.

# Recursivitate

- reg.prod. recursiva la stanga:
  - o reg.prod. de forma:  $A \rightarrow A\alpha$
- reg.prod. recursiva la dreapta:
  - o reg.prod. de forma:  $A \rightarrow \alpha A$
- reg.prod. recursiva:
  - o reg.prod. de forma:  $A \rightarrow \alpha A\beta$

# Reg. prod. recursive la stanga

- reg.prod. recursive la stanga:  $A \rightarrow A\alpha$

## Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă  
fără reg.prod. recursive la stanga

- **PP. G – gr. proprie**  
(daca nu este, det. gr. proprie echiv. si lucram cu ea)
- Obs.: vom obtine tot o gramatica proprie

# Eliminarea r.p. recursive la stanga

pentru fiecare  $A \in N$ : reg.prod.cu m.s.  $A$

- grupam r.p. in recursive la stng. si nerec. la stanga

$$A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \dots \mid A\alpha_r \quad (\text{r.p. recursive})$$

$$A \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_s \quad (\text{r.p. ne-recursive})$$

- r.p. se transforma astfel:

$$A \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_s \mid \beta_1 A' \mid \beta_2 A' \mid \dots \mid \beta_s A'$$

$$A' \rightarrow \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_r \mid \alpha_1 A' \mid \alpha_2 A' \mid \dots \mid \alpha_r A'$$

(a fost introdus un net. nou:  $A'$ )

# Eliminarea r.p. recursive la stanga

## Observatii:

- Recursivitatea nu se poate elimina.
- Recursivitatea la stanga a fost transformata în recursivitate la dreapta.

Exercitiu

Eliminati recursivitatea la stanga:

$S \rightarrow Sa$

$S \rightarrow a$

# Rekursivitate

- **neterminal recursiv la stanga:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ A\alpha$

- **neterminal recursiv la dreapta:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ \alpha A$

- **neterminal recursiv:**

$A \in N$  daca  $\exists$  o derivare de forma:  $A \Rightarrow^+ \alpha A \beta$

- **gramatica recursiva la stanga:**

are cel putin un neterminal recursiv la stanga

- **gramatica recursiva la dreapta:** ...

# Eliminarea recurs. la stg. a neterm.

## Teorema:

$\forall G = (N, \Sigma, P, S) \exists G' = (N', \Sigma', P', S)$  echivalentă  
fără neterminale recursive la stanga

- PP.  $G$  – gr. proprie  
(daca nu este, det. gr. proprie echiv. si lucram cu ea)
- impunem o ordine asupra neterminalelor  
 $N = \{A_1, A_2, \dots, A_n\}$   
si apoi modific r.p. a.i. sa nu existe  $A_i \rightarrow A_j \alpha$  cu  $j \leq i$   
de aici  $\Rightarrow$  nu va exista recursivitate la stanga

# Eliminarea recurs. la stg. a neterm.

**pentru**  $A_i$  de la  $A_1$  la  $A_n$  **executa**

*//se elimina r.p. de forma  $A_i \rightarrow A_j \alpha$  cu  $j \leq i$  astfel:*

**\*repetă**

**pentru**  $j:=1, i-1$  **executa**

**\***  $A_i \rightarrow A_j \alpha$  ( $j < i$ ) se înlocuiește cu:  $A_i \rightarrow \beta \alpha$

cu toți  $\beta$  cu proprietatea  $A_j \rightarrow \beta \in P_{\text{inloc}}$

**sf.pentru**

**\*** se elimină r.p. de forma  $A_i \rightarrow A_i \alpha$

(se înlocuiesc cf.alg. de elim.r.p.rec.stg)

**\*pană când** toate r.p. cu  $A_i$  în m.s. respectă:  $\nexists j < i : A_i \rightarrow A_j \alpha$

**sf.pentru**

# Eliminarea recurs. la stg. a neterm.

## Exercitii:

(1)

$$A \rightarrow BC \mid a$$

$$B \rightarrow CA \mid b$$

$$C \rightarrow AB \mid c$$

(2)

$$A \rightarrow a \mid aB$$

$$B \rightarrow AC \mid b$$

$$C \rightarrow BA \mid c$$