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 **$G = (N, \Sigma, P, S)$**

- **N** este un alfabet de simboluri ***neterminale***
- **Σ** este un alfabet de simboluri ***terminale***
- **$N \cap \Sigma = \phi$**
- **$P \subseteq (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$**

  P multime finitã (multimea regulilor de productie)
- **$S \in N$** (simbolul de start - simbolul initial)

Notatie:

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

(α se înlocuieste cu β)

Ne reamintim: clasificarea Chomsky

- Gramatici de tip 0:
  nici o restrictie *(suplimentara)* referitoare la forma regulilor de productie
- Gramaticile de tip 1 (gramatici monotone)
  - $\forall\ \alpha \to \beta \in P$: $|\alpha| <= |\beta|$
  - caz special: $\mathbf{S} \to \boldsymbol{\varepsilon}$ poate $\in$ P. In acest caz S nu apare în membrul drept al nici unei reguli de productie.

- **Gramatici independente de context:**

  reg. productie sunt de forma $A \to \alpha$, $A \in N$, $\alpha \in (N \cup \Sigma)^*$

  (gramatici de tip 2)

- Gramaticile de tip 3:
  reg. prod. sunt de forma
  - $A \to aB$
  - $A \to b$ unde $A,B \in N$ si $a,b \in \Sigma$

  caz special: $\mathbf{S} \to \boldsymbol{\varepsilon}$ poate $\in$. P In acest caz S nu apare în membrul drept al nici unei reguli de productie.

derivari de stanga/ dreapta

- ***derivare de stânga*** $=>_{st}$

  o derivare directã in care se înlocuieste cel mai din stânga neterminal
- ***derivare de dreapta*** $=>_{dr}$

  o derivare directã in care se înlocuieste cel mai din dreapta neterminal

Analiza sintactica

- ***analizã sintacticã*** pt. cuvantul w

  succesiunea de derivãri directe:

  - $\mathbf{S \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_n = w}$

  altfel spus: reprezintã o derivare pentru cuvântul **w**

- ***analizã sintacticã descendentã***

  dacã aceastã succesiune de derivãri directe se obtine pornind de la **S** si terminand cu w

- ***analizã sintacticã ascendentã***

  dacã aceastã succesiune de derivãri directe se obtine pornind de la w si terminand cu **S**

Arbore de derivare

- **Fie G = (N, Σ, P, S)** o gramaticã independentã de context. Numim ***arbore de derivare*** sau ***arbore de analizã sintacticã*** un arbore cu radacina, ordonat,

  cu urmatoarele proprietati:

1. Orice nod interior - o eticheta din ***N*;**
2. Orice nod frunza - o ***etichetã*** din Σ U {ε}
3. Eticheta rãdãcinii este **S;**
4. Dacã un nod are eticheta A iar nodurile succesoare acestuia, în ordine de la stânga la dreapta sunt etichetate cu $\mathbf{X_1, X_2, \ldots X_n}$ atunci $\mathbf{A \rightarrow X_1X_2 \ldots X_n}$ trebuie sã fie o productie din **P.**

Arbore de derivare

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

<u>Teoremã</u>.

Fie **G = (N, Σ, P, S)** o gramaticã independentã de context. Un cuvânt **w** peste alfabetul **Σ**, deci din Σ^*, apartine limbajului generat de **G**, adicã **w** $\in$ **L(G),** dacã si numai dacã **w** este frontul unui arbore de analizã sintacticã.

Gramatica ambigua

O gramaticã $\mathbf{G = (N, \Sigma, P, S)}$ independentã de context este ***ambiguã***

dacã si numai dacã existã cel putin un cuvânt w care admite doi arbori de derivare distincti; în caz contrar gramatica este ***neambiguã***.

- ⇔ ∃ 2 analize sintactice care folosesc numai derivari de stanga, diferite
- ⇔ ∃ 2 analize sintactice care folosesc numai derivari de dreapta, diferite

Descrieri echivalente. Forme normale

Ne reamintim :

O gramatica G_1

este echivalenta cu gramatica G_2

ddaca $L(G_1)=L(G_2)$

ε-productii si gram. ε-independente

- ***ε-productie*** : o productie de forma $A \to \varepsilon$
- Gramatica $G = (N, \Sigma, P, S)$ este **ε-independentã** daca:

  a) dacã $\varepsilon \notin L(G)$ atunci G nu are ε-productii

  b) dacã $\varepsilon \in L(G)$ atunci avem o singurã productie $S \to \varepsilon$ iar celelalte productii nu-l contin în membrul drept pe S

- <u>Teorema</u>

  $\forall$ **G= (N, Σ, P, S)**

  $\exists$ **G' = (N', Σ', P', S)** echivalentã, ε-independentã

Redenumiri. Cicluri.

- ***redenumire:*** reg.prod. de forma $\mathbf{A \rightarrow B}$
- ***Gramatica fãrã redenumiri***: fara r.p. de redenumire

  Teorema:

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã redenumiri
- ***ciclu***: o * derivare de forma **A =>* B**
- ***Gramatica fãrã cicluri***: nu se pot obtine cicluri (la derivare)

  Teorema:

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã cicluri

Recursivitate

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

Recursivitate

- **neterminal recursiv la stanga**:
  $A \in N$ daca $\exists$ o derivare de forma: $A \stackrel{+}{\Rightarrow} A\alpha$
- **neterminal recursiv la dreapta**:
  $A \in N$ daca $\exists$ o derivare de forma: $A \stackrel{+}{\Rightarrow} \alpha A$
- **neterminal recursiv**:
  $A \in N$ daca $\exists$ o derivare de forma: $A \stackrel{+}{\Rightarrow} \alpha A \beta$
- **gramatica recursiva la stanga**:
  are cel putin un neterminal recursiv la stanga
- **gramatica recursiva la dreapta**: …

Forma normalã Chomsky

O gramaticã independentã de context **G = (N, Σ , P, S)** este în ***forma normalã Chomsky*** (***FNC***)

dacã orice regula de productie din **P** este de una din formele:

a) $A \rightarrow BC$ $\quad$ $A,B,C \in N$;

b) $A \rightarrow a$ $\quad$ $a \in \Sigma$, $A \in N$;

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

<u>Teoremã</u>. Oricare ar fi **G=(N, S, P, S)** o gramaticã independentã de context, întotdeauna existã o gramaticã în forma normalã Chomsky **G'**, astfel încât L(G) = L(G').

Forma normalã Greibach

O gramaticã **G = (N, Σ, P, S)**

este în ***forma normalã Greibach*** (***FNG***)

dacã **P** are productii numai de forma:

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

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.

<u>Teoremã</u>. Oricare ar fi **G = (N, Σ, P, S)** o gramaticã independentã de context, întotdeauna existã o gramaticã în forma normalã Greibach, astfel încât **L(G)=L(G')**.

Simplificarea GIC

- ***simbol neproductiv***

  Un simbol $A \in N$ este ***neproductiv*** dacã nu existã nici o derivare de forma $A \overset{*}{=}> \mathbf{x}$ $\quad (\mathbf{x} \in \Sigma^*)$

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

- <u>Teorema</u>

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã, fãrã simboluri neproductive

Simplificarea GIC

(transformari echivalente)

- ***simbol inaccesibil***

  Un simbol $X \in N \cup \Sigma$ este ***simbol inaccesibil*** dacã nu existã nici o * derivare: $S \Rightarrow^{*} \alpha X \beta \quad (\alpha, \beta \in (N \cup \Sigma)^{*})$

- în caz contrar simbolul este ***accesibil***
- <u>Teorema</u>

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã, fãrã simboluri inaccesibile

Determinarea simbolurilor productive

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

$\approx$ algoritm AF determ. stari productive

1. $i := 0$; $V_0 := \Phi$
2. **Repeta**

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

   $i := i+1$

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

   $\{V_i$ – multimea simbolurilor productive$\}$

Determinarea simbolurilor accesibile

$\approx$ algoritm AF determ. stari accesibile

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

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

$i:=i+1$

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

$\{V_i$ – multimea simbolurilor neterminale accesibile$\}$

* analog pentru simboluri terminale accesibile

Simplificarea GIC

- Un simbol este **neutilizabil** dacã el este fie inaccesibil, fie neproductiv
- <u>Teorema</u>

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã simboluri neutilizabile

Observatii:

Fie **G= (N, Σ, 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

De multe ori, vom da o gramatica independenta de context prin regulile ei de productie. Vom considera ca pentru fiecare neterminal al gramaticii, exista cel putin o regula de productie cu acel neterminal in membrul stang. Astfel, vom identifica neterminalele, iar restul simbolurilor care apar in regulile de productie sunt terminale. Facem conventia ca prima regula de productie are in membrul stang simbolul de start al gramaticii.

Eliminarea ε-productiilor

1. Construim multimea N_ε care are ca elemente acele neterminale care prin derivare conduc la ε adicã :

- $N_\varepsilon = \{A \mid A \in N, A \Rightarrow^* \varepsilon\}$

    alg. ≈ 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_ε

1. $i:=0$;

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

2. **Repeta**

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

   $i:=i+1$

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

determinam noile reguli de productie

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

Fie r.p. $A \to \alpha_0 B_1 \alpha_1 B_2 \alpha_2 \ldots B_k \alpha_k$

unde: $B_i \in N_\varepsilon$

α_j nu contine simb. din N_ε

Se inlocuieste cu:

$$A \to \alpha_0 X_1 \alpha_1 X_2 \alpha_2 \ldots X_k \alpha_k$$

unde $X_i = \begin{cases} B_i \\ \varepsilon \end{cases}$ este unul dintre B_i sau ε (se fac toate inlocuirile posibile)

Ce lipseste ???

determinam noile reguli de productie

- continuare

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

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

(gram. ε-independenta)

- adaugam un nou simbol de start S' si productiile

$S' \rightarrow \varepsilon \mid S$

Redenumiri. Cicluri.

- ***redenumire:*** reg.prod. de forma $\mathbf{A \rightarrow B}$
- ***Gramatica fãrã redenumiri***: fara r.p. de redenumire

  Teorema:

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã redenumiri
- ***ciclu***: o * derivare de forma $\mathbf{A \Rightarrow^* B}$
- ***Gramatica fãrã cicluri***: nu se pot obtine cicluri (la derivare)

  Teorema:

  $\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã cicluri

Eliminarea redenumirilor

PP. G – ε-independenta (daca nu , luam gr.echiv. ε-ind.)

Pentru fiecare **A$\in$ N**

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

- construieste multimile $N_A = \{B \mid A \stackrel{*}{\Rightarrow} B\}$;

  (≈ det. simb. accesibile)
- determinam noile reguli de productie

Construieste $N_A = \{D \mid A \overset{*}{\Rightarrow} D\}$

1. $i := 0$;

   $V_0 := \{A\}$

2. **Repeta**

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

   $i := i+1$

   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** α 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, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã cicluri

Daca G – ε-independenta si fara redenumiri atunci este fara cicluri

Gramatica proprie:

este o gramatica
- fara simb. neutilizabile
- **ε-independenta**
- fara cicluri

Teorema:

$\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', 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. recursiva la stanga: $A \rightarrow A\alpha$

Teorema:

$\forall$ **G= (N, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã 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

  $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_r$ (r.p. recursive)

  $A \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s$ (r.p. ne-recursive)

- r.p. se transforma astfel:

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

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

  (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-> Sa

S->a

Recursivitate

- **neterminal recursiv la stanga**:

  $A \in N$ daca $\exists$ o derivare de forma: $A =^{+}> A\alpha$
- **neterminal recursiv la dreapta**:

  $A \in N$ daca $\exists$ o derivare de forma: $A =^{+}> \alpha A$
- **neterminal recursiv**:

  $A \in N$ daca $\exists$ o derivare de forma: $A =^{+}> \alpha A \beta$
- **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, Σ, P, S)** $\exists$ **G' = (N', Σ', P', S)** echivalentã fãrã neterminale recursive la stanga

- PP. G – gr. proprie

  (daca nu este, det. gr. proprie echiv. si lucram cu ea)

- impunem o ordine asupra neterminalelor

  $N = \{A_1, A_2, \ldots, A_n\}$

  si apoi modific r.p. a.i. sa nu existe $A_i \rightarrow A_j\alpha$ cu j<=i

  de aici => 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 $\boldsymbol{A_i} \rightarrow A_j\alpha$ *cu j<=i astfel*:

***repeta**

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

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

cu toti β cu proprietatea $A_j \rightarrow \beta \in P_{inloc}$

sf.pentru

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

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

***pana cand** toate r.p. cu A_i in m.s. respecta: $\nexists$ 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)

$A \rightarrow a \mid aB$

$B \rightarrow AC \mid b$

$C \rightarrow BA \mid c$