

CURSFORME NORMALE RENTRU GRAMATICI TIP 2

Fie $G = (V_N, V_T, S, \delta) \in \mathcal{G}_2$. În clasificarea Chomsky forma generală a regulilor $A \rightarrow P, A \in V_N, P \in \mathcal{P}_G^k$

DEF 1: Spunem că G este în forma normală CHOMSKY dacă regulile au forma

$$\begin{cases} A \rightarrow BC \\ D \rightarrow i \end{cases}, \quad A, B, C, D \in V_N, \quad i \in V_T$$

sau $S \rightarrow \lambda \text{ și } S \notin dr.$

DEF 2: Spunem că G este în forma normală GREIBACH dacă regulile sunt de forma

$$A \rightarrow i p, \quad A \in V_N, \quad i \in V_T, \quad p \in V_H^*$$

sau $S \rightarrow \lambda \text{ și } S \notin dr.$

Obs: • f.n. Chomsky se utilizează pt. demonstrații ale proprietăților lb. de tip 2 și asociarea arborei binare pentru derivări

- f.n. GREIBACH se utilizează pt. algoritmul de pasire a automatului push-down ce recunoaște un limbaj independent de context
- sunt și alte forme utilizate. Forma normală operator, unde în dreapta regulilor nu apar neterminele adiacente, se folosește în descoperirea unui algoritm de analiză sintactică de tip bottom up.

TEOREMĂ: Orice limbaj independent de context L poate fi generat cu o gramatică G în formă normală CHOMSKY (GREIBACH).

Construcția formei normale CHOMSKY

Fie $G = (V_N, V_T, S, \beta) \in \mathcal{G}_2$ și $L = L(G) \in \mathcal{L}_2$.

Puteți presupune că G nu are reguli de stergere
Cu excepția eventual a regulii de completare
 $S \rightarrow \lambda$ și $S \notin \mathcal{F}$

De asemenea presupunem că G nu are redenumiri
și regulile ce contin terminale în dreapta sunt de
formă $A \rightarrow i$, $A \in V_N$, $i \in V_T$.

Rezultă că regulile lui G au una din formele:

$$(1) A \rightarrow BC, \quad A, B, C \in V_N$$

$$(2) D \rightarrow i, \quad D \in V_N, i \in V_T$$

$$(3) X \rightarrow x_1 x_2 \dots x_n, \quad n \geq 2, \quad x_1, x_2, \dots, x_n \in V_N$$

Formele (1), (2) în eventual regula de completare
sunt acceptate la f.n. CHOMSKY

Regulile de formă (3) se întocmesc cu reguli
convenabile, fără să schimbe limbașul generat

Dacă $X \rightarrow x_1 x_2 \dots x_n \in \beta$, $n \geq 2$. atunci introduc
în gramatica echivalente regulile

$$\left\{ \begin{array}{l} X \rightarrow X_1 Z_1 \\ Z_1 \rightarrow X_2 Z_2 \\ \vdots \\ Z_{n-2} \rightarrow X_{n-1} X_n \end{array} \right. , \text{ cu } Z_1, Z_2, \dots, Z_{n-2} \text{ neterminale noi}$$

Evident, dacă

$$X \xrightarrow{\beta} x_1 x_2 \dots x_n$$

atunci (și numai atunci!)

$$x_1 x_2 \dots x_n \xrightarrow{\beta} x_1 x_2 \dots x_{n-2} Z_{n-2} \Rightarrow x_1 x_2 \dots x_n$$

$$X \rightarrow X_1 Z_1 \Rightarrow X_1 X_2 Z_2 \Rightarrow \dots \Rightarrow X_1 X_2 \dots X_{n-2} Z_{n-2} \Rightarrow x_1 x_2 \dots x_n$$

Evident noua gramatică obținută este și f.n. Chomsky
și generează același limbaj L .

Exemplu: Găsiți o gramatică din f.n. Chomsky pentru generarea limbajului

$$L = \{ w \in \{a, b\}^* \mid w \text{ palindrom} \}$$

#

O gramatică ce generează ~~totale~~ ~~toate~~ palindroamele peste $\{a, b\}$ are următoarele

$$S \rightarrow \lambda \mid a \mid b \mid aSa \mid bSb \in \mathcal{G}_2.$$

Aducem gramatica la f.n. CHOMSKY

PAS1: Elimin repurile de stepere

PAS2: Aplic leme "A \rightarrow^* i" "A $\Rightarrow B$ "

PAS3: Elimin "redemuniri" "A $\Rightarrow B$ "

PAS4: ScurtEZ repuri neconvenabile

$$\begin{aligned} \text{PAS1: } & \left\{ \begin{array}{l} x_0 \rightarrow \lambda \mid S \\ S \rightarrow a \mid b \mid aSa \mid bSb \mid aa \mid bb \end{array} \right. \end{aligned}$$

$$\begin{aligned} \text{PAS2: } & \left\{ \begin{array}{l} x_0 \rightarrow \lambda \mid S \\ S \rightarrow a \mid b \mid X_a S X_a \mid X_b S X_b \mid X_a X_a \mid X_b X_b \\ X_a \rightarrow a \\ X_b \rightarrow b \end{array} \right. \end{aligned}$$

$$\begin{aligned} \text{PAS3: } & \left\{ \begin{array}{l} x_0 \rightarrow \lambda \mid a \mid b \mid X_a S X_a \mid X_b S X_b \mid X_a X_a \mid X_b X_b \\ S \rightarrow a \mid b \mid X_a S X_a \mid X_b S X_b \mid X_a X_a \mid X_b X_b \\ X_a \rightarrow a \\ X_b \rightarrow b \end{array} \right. \end{aligned}$$

$$\begin{aligned} \text{PAS4: } & \left\{ \begin{array}{l} x_0 \rightarrow \lambda \mid a \mid b \mid X_a Z_1 \mid X_b Z_2 \mid X_a X_a \mid X_b X_b \\ Z_1 \rightarrow S X_a \\ Z_2 \rightarrow S X_b \\ S \rightarrow a \mid b \mid X_a Z_3 \mid X_b Z_4 \mid X_a X_a \mid X_b X_b \\ Z_3 \rightarrow S X_a \\ Z_4 \rightarrow S X_b \\ X_a \rightarrow a ; X_b \rightarrow b \end{array} \right. \end{aligned}$$

LEMA SUBSTITUIRII - permite înlocuirea unei neterminale
în drepta unei reguli (substituția neterminalelor)

Lemă: Fie G o gramatică de tipul 2, $X \rightarrow uTv$ o
regulă a gramaticii și $T \rightarrow \alpha_1 | \dots | \alpha_n$ toate regulile
cu T în stânga.

Atunci G este echivalentă cu o gramatică G'
în care am făcut "substituții"; adică
regula $X \rightarrow uTv$ se înlocuiește cu $X \rightarrow u\alpha_1v | \dots | u\alpha_nv$
(Regulile lui T se păstrează!)

#

Demonstratia formală se poate detalia în cursul
de pe site.

Exemplu: Considerăm gramatica $\begin{cases} A \rightarrow BA | 1C \\ B \rightarrow 011 \\ C \rightarrow 1C | B \end{cases}$

Substituția lui B în $A \rightarrow BA$ rezultă

$G' \quad \begin{cases} A \rightarrow 0A11A1C \\ B \rightarrow 011 \\ C \rightarrow 1C | B \end{cases}$

Substituția lui C în $A \rightarrow 1C$

$G'' \quad \begin{cases} A \rightarrow 0A11A11C11B \\ B \rightarrow 011 \\ C \rightarrow 1C | B \end{cases}$

Substituția nu schimbă $L(G)$.

O derivare ce folosește regula $A \rightarrow BA$

$G; \underbrace{A \Rightarrow BA}_{\text{O derivare ce folosește regula }} \Rightarrow 0A \Rightarrow 01C \Rightarrow 01B \Rightarrow 010$
Se obține și în G' (în mai puțini pași!)

$G'; \underbrace{A \Rightarrow 0A}_{\text{O derivare ce folosește regula }} \Rightarrow 01C \Rightarrow 01B \Rightarrow 010$

GRAMATICI RECURSIVE

DEF. Un simbol neterminat X al unei gramatici de tipul 2 se zice "recursiv" dacă există o rețină cu X în stâng și în dreapta, adică

$$\exists X \rightarrow uXv \in \Sigma^*, \quad u, v \in V_G^*$$

Obs: Dacă $u = \lambda$ at. X este stâng recursiv.
Idem dacă $v = \lambda$ at. X este drept recursiv.

Teorema Orice limbaj independent de context poate fi generat de o gramatică fără recursie stângă.

Fie $G = (V_N, V_T, \Sigma_0, \delta) \in \mathcal{G}_2$ și $L = L(G)$.

Prendem că G conține un singur simbol stâng recursiv X , cu reținile

$$(A) \quad X \rightarrow u_1 | u_2 | \dots | u_n | X v_1 | \dots | X v_m \quad u_i, v_j \in V_G^* \\ (u_i - nu începe cu X, \forall i = \overline{1, n}), \quad i = \overline{1, n}, j = \overline{1, m}$$

Construim $G' = (V_N', V_T, \Sigma_0, \delta') \in \mathcal{G}_2$ unde rețina (A)

se înlocuiește cu

$$(B) \quad \left\{ \begin{array}{l} X \rightarrow u_1 | \dots | u_n | u_1 T | \dots | u_n T \\ T \rightarrow v_1 | \dots | v_m | v_1 T | \dots | v_m T \end{array} \right.$$

cu $T \in V_N'$ un neterminat nou. Restul reținilor se păstrează

Obs: 1) nou T este drept recursiv

2) Orice curănt generat cu reținile (*) până la eliminarea lui X se generață de la dreapta la stânga. Prin reținile (B) se formează același curănt de la stânga la dreapta

3) Procedeu de eliminare se repetă pînă toate sub. recursive

EXEMPLU: Fișă gramatica ce generează expresii aritmetice

$$G_1: \left\{ \begin{array}{l} E \rightarrow E+T \mid E-T \mid ET \\ T \rightarrow T * F \mid T / F \mid F \\ F \rightarrow (E) \mid \text{id} \mid \text{număr} \end{array} \right.$$

Observăm producerea curintelor prin aplicarea regulilor; pt. E se face astfel:

$$E \Rightarrow E+T \Rightarrow E-T+T \Rightarrow E-E-T+T = \overleftarrow{T-T-T+T} \Rightarrow P.$$

Eliminarea recursiei pt. E:

$$G'_1: \left\{ \begin{array}{l} E \rightarrow T \mid TT \\ T \rightarrow +T \mid -T \mid +TT \mid -TT \\ T \rightarrow T * F \mid T / F \mid F \\ F \rightarrow (E) \mid \text{id} \mid \text{număr} \end{array} \right.$$

Același curind se obține în G'_1 astfel

$$E \Rightarrow TT \Rightarrow T-TT \Rightarrow T-T-TT = \overrightarrow{T-T-T+T} \Rightarrow P.$$

Repetarea procedeului pt. neterm. T

$$G''_1: \left\{ \begin{array}{l} E \rightarrow T \mid TT \\ T \rightarrow +T \mid -T \mid +TT \mid -TT \\ T \rightarrow F \mid FZ \\ Z \rightarrow *F \mid /F \mid *FZ \mid /FZ \\ F \rightarrow (E) \mid \text{id} \mid \text{număr} \end{array} \right.$$

Construcția formei normale GREIBACH

$A \rightarrow i p$, $A \in V_N$, $i \in V_T$, $p \in V_H^*$

Sau $S \rightarrow \lambda$ și Sefr.

Algoritmul de transformare a unei gramatici

din f.n. CHOMSKY în f.n. GREIBACH

Fie $G = (V_N, V_T, S, \beta)$ din f.n. CHOMSKY. Indexăm
neterminalele astfel $V_N = \{S = X_1, X_2, \dots, X_n\}$

ETAPA 1: Modific regulile ce incep în dreapta un
neterminal a.i. să fie de forma

$$X_i \rightarrow X_k p, \text{ cu } i < k \text{ și } p \in V_H^*$$

Alg: (input G)

PAS 1: Elimin recursia stăupă pt. X_i ;
// neterminalul nou introdus notat T_i

PAS 2: for $i = 2 \text{ to } n$ do

(2.1) for $j = 1 \text{ to } (i-1)$ do

// Substitui X_j în regulile lui X_i

$$X_i \rightarrow X_j p$$

} ; end for

(2.2) Elimin recursia stăupă pt. X_i
// neterminalul nou introdus T_i

} ; end for

Obs: În urma aplicării algoritmului regulile
ce incep cu neterminal în dreapta unei forme

$$X_i \rightarrow X_j p, \quad i < j > p \in V_H^*$$

noile netermale introduse la eliminarea
recursiei sunt T_1, T_2, \dots, T_k , cu $k \leq n$.

Etapa 2 Redenumirea regrulelor

$T_1 \dots T_m \ x_1 \dots x_n$, notez $n+m = h$
 $z_1 z_2 \dots z_m z_{m+1} \dots z_{n+m}$

Obs: În urma redenumirii, regulile au forma

$$z_i \rightarrow z_j p, \quad \{j\} \subset V_H^*$$

$$z \rightarrow aw, \quad w \in V_H^*, a \in V_T$$

$$z_1 \rightarrow \lambda, z_1 \notin dr$$

ETAPA 3 Substituția inversă.

Se începe cu z_{N-1} (deoarece z_N are reguli ce încep cu terminal în dreapta) și substituții z_N în prima poziție dreapta. Rezultă reguli convenabile. Consider z_{N-2} și substituții z_N în dreapta, apoi z_{N-1} .

Algoritm:

```
for j = N-1 to 1 do {
    for k = h to j+1 do {
        // substituții  $z_k$  în regulile
         $z_j \rightarrow z_k p$ 
    } end for
}
```

} end for.

Obs: După etapa 3 regulile sunt în f.n. GREIBACH

- 2) De obicei, f.n. GREIBACH se obține imediat (pt. exemple didactice!) prin examinarea regulilor initiale.

Ex: $S \rightarrow aSb | ab \in \mathcal{G}_2 \quad L(G) = \{a^n b^n | n \geq 1\}$

G' : $\begin{cases} S \rightarrow aSB | aB \in \mathcal{G}_2 \\ B \rightarrow b \end{cases}$ în f.n. GREIBACH