



AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Przekształcenia gramatyk bezkontekstowych

Teoria automatów i języków formalnych

Dr inż. Janusz Majewski
Katedra Informatyki



Symbole nadmiarowe, gramatyka bez ε -produkcji

Niech będzie dana gramatyka bezkontekstowa $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$

Symbol $X \in (V \cup \Sigma)$ nazywamy nieużytecznym w $G \in \mathcal{G}_{BK}$ jeśli nie można w tej gramatyce przeprowadzić wyprowadzenia:

$$S \Rightarrow_G^* wXy \Rightarrow_G^* wxy$$

przy czym $w, x, y \in \Sigma^*$ (czyli z X nie można wyprowadzić żadnego słowa w symbolach terminalnych).

Symbol $X \in (V \cup \Sigma)$ nazywamy nieosiągalnym w $G \in \mathcal{G}_{BK}$, jeśli X nie pojawia się w żadnej formie zdaniowej tej gramatyki (czyli nie można go wyprowadzić z S).

Gramatyka $G \in \mathcal{G}_{BK}$ jest gramatyką bez ε -produkcji, gdy:

- (i) P nie zawiera produkcji postaci $A \rightarrow \varepsilon$, $A \in V$, $A \neq S$
- (ii) jeśli $(S \rightarrow \varepsilon) \in P$ to symbol S nie występuje w prawych stronach pozostałych produkcji



Możliwość przekształcania gramatyk

Twierdzenie

Każdą gramatykę bezkontekstową

$G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ można przekształcić do postaci równoważnej $G' = \langle V', \Sigma', P', S \rangle \in \mathcal{G}_{BK}$ (to znaczy takiej, że $L(G') = L(G)$) nie zawierającej symboli nieosiągalnych, symboli nieużytecznych oraz będącej gramatyką bez ϵ -produkcji (w sensie powyższych definicji).

Sposoby wykonania odpowiednich przekształceń podają dalej podane algorytmy. Słuchacze zechcą udowodnić poprawność tych algorytmów.



Algorytm usuwania symboli nieosiągalnych (1)

wejście: $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$

wyjście: $G' = \langle V', \Sigma', P', S \rangle \in \mathcal{G}_{BK}$ taka, że:

(i) $L(G') = L(G)$

(ii) $(\forall X \in (V' \cup \Sigma')) (\exists \alpha, \beta \in (V' \cup \Sigma')^*) (S \Rightarrow_{G'}^* \alpha X \beta)$

Metoda:

$W_0 := \{S\};$

$i := 0;$

repeat

$i := i + 1;$

$W_i := W_{i-1} \cup \{X \in (V \cup \Sigma) \mid (A \rightarrow \alpha X \beta) \in P \wedge A \in W_{i-1} \wedge \alpha, \beta \in (V \cup \Sigma)^*\};$

until $W_i = W_{i-1}$;/* W_i zawiera te symbole terminalne i nieterminalne które:

- występują w produkcjach z P
- dają się wyprowadzić z S */



Algorytm usuwania symboli nieosiągalnych (2)

$V' := W_i \cap V;$

$\Sigma' := W_i \cap \Sigma;$

$P' := \{(u \rightarrow v) \in P \mid u, v \in W_i^*\}$

/* P' zawiera te produkcje z P, które zbudowane są z symboli z W_i^* */

$G' := \langle V', \Sigma', P', S \rangle$



Algorytm usuwania symboli nieużytecznych

wejscie: $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$

wyjście: $G' = \langle V', \Sigma', P', S \rangle \in \mathcal{G}_{BK}$ taka, że:

(i) $L(G') = L(G)$

(ii) $(V' \cup \Sigma')$ nie zawiera symboli nieużytecznych

Metoda:

$V_0 := \emptyset;$ /* \emptyset - zbiór pusty */

$i := 0;$

repeat

$i := i + 1;$

$V_i := V_{i-1} \cup \{A \in V \mid (A \rightarrow \alpha) \in P \wedge \alpha \in (V_{i-1} \cup \Sigma)^*\}$

until $V_i = V_{i-1};$ /* V_i zawiera wszystkie nieterminale „użyteczne” */

$G_1 := \langle V \cap V_i, \Sigma, P_1, S \rangle$ gdzie

$P_1 := \{(u \rightarrow v) \in P \mid u, v \in (V_i \cup \Sigma)^*\};$

zastosować do G_1 algorytm usuwania symboli nieosiągalnych uzyskując w wyniku tego gramatykę $G' = \langle V', \Sigma', P', S \rangle$



Przykład usuwania symboli nadmiarowych

Przykład:

- $G = \langle \{S, A, B\}, \{a, b\}, \{ \begin{array}{l} S \rightarrow a \\ S \rightarrow A \\ A \rightarrow AB \\ B \rightarrow b \end{array} \}, S \rangle$
- $G_1 = \langle \{S, B\}, \{a, b\}, \{ \begin{array}{l} S \rightarrow a \\ B \rightarrow b \end{array} \}, S \rangle$
- $G' = \langle \{S\}, \{a\}, \{ S \rightarrow a \}, S \rangle$



Algorytm usuwania ϵ -produkcji (1)

wejscie: $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$

wyjście: $G' = \langle V', \Sigma, P', S' \rangle \in \mathcal{G}_{BK}$ taka, że:

- (i) $L(G) = L(G')$;
- (ii) G' jest gramatyką bez ϵ -produkcji

Metoda:

$V_0 := \emptyset$;

$i := 0$;

repeat

$i := i + 1$;

$V_i := V_{i-1} \cup \{ A \in V \mid (A \rightarrow x) \in P \wedge x \in V_{i-1}^* \}$;

until $V_i = V_{i-1}$;

$/* V_i = \{ A \mid A \in V \wedge A \Rightarrow_G^+ \epsilon \} */$



Algorytm usuwania ϵ -produkcji (2)

```
P' := P ;  
for (A → x) ∈ P do  
  begin  
    przedstawiamy „x” w postaci  $\alpha_0 B_1 \alpha_1 B_2 \alpha_2 \dots B_k \alpha_k$ ,  $k \geq 0$ ,  
    gdzie  $B_j \in V_i$  ( $1 \leq j \leq k$ ),  $\alpha_j \in ((V \cup \Sigma) - V_i)^*$  ( $0 \leq j \leq k$ );  
    P' := P' ∪ {(A →  $\alpha_0 X_1 \alpha_1 X_2 \alpha_2 \dots X_k \alpha_k$ ) | ( $X_j = B_j \vee X_j = \epsilon$ )};  
    if (A →  $\epsilon$ ) ∈ P' then P' := P' - {(A →  $\epsilon$ )};  
  end;
```



Algorytm usuwania ϵ -produkcji (3)

```
if S ∈ Vi then  
  begin  
    V' := V ∪ {S'};  
    P' := P' ∪ {(S' → S), (S' →  $\epsilon$ )};  
    S' := S';  
  end  
else  
  begin  
    V' := V;  
    S' := S;  
  end;
```



Przykład usuwania ε -produkcji

Przykład:

$S \rightarrow aSbS$

$S \rightarrow bSaS$

$S \rightarrow \varepsilon$

Po usunięciu $S \rightarrow \varepsilon$ otrzymujemy

$S' \rightarrow S$

$S' \rightarrow \varepsilon$

$S \rightarrow aSbS$, $S \rightarrow aSb$, $S \rightarrow abS$, $S \rightarrow ab$

$S \rightarrow bSaS$, $S \rightarrow bSa$, $S \rightarrow baS$, $S \rightarrow ba$



Produkcje łańcuchowe

Produkcje łańcuchowe, to produkcje postaci: $A \rightarrow B$; gdzie $A, B \in V$

Gramatyka G zawiera cykle, wówczas gdy istnieją w gramatyce wprowadzenia postaci: $A \Rightarrow_G^+ A$; $A \in V$

Cykle typu $A \Rightarrow^+ A$ powstają poprzez istnienie w gramatyce produkcji łańcuchowych, np. $A \rightarrow B, B \rightarrow C, C \rightarrow A$, więc usuwając produkcje łańcuchowe – usuwamy również automatycznie wszystkie cykle.

Twierdzenie

Każdą gramatykę bezkontekstową $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ można przekształcić do postaci równoważnej $G' = \langle V', \Sigma', P', S \rangle \in \mathcal{G}_{BK}$ (to znaczy takiej, że $L(G') = L(G)$) nie zawierającej produkcji łańcuchowych i cykli (w sensie powyższych definicji).



Algorytm usuwania produkcji łańcuchowych i cykli (1)

wejście: $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$; G – bez ϵ -produkcji!

wyjście: $G' = \langle V, \Sigma, P', S \rangle \in \mathcal{G}_{BK}$ taka, że

(i) $L(G) = L(G')$

(ii) G' – nie zawiera produkcji łańcuchowych i cykli

Metoda:

for $A \in V$ do

begin

$V_0 := \{A\};$

$i := 0;$

repeat

$i := i + 1;$

$V_i := V_{i-1} \cup \{ C \in V \mid (B \rightarrow C) \in P \wedge B \in V_{i-1} \}$

until $V_{i-1} = V_i$;

$V_A := V_i$; $/* V_A = \{ B \mid A \Rightarrow_G^* B \} */$

end;



Algorytm usuwania produkcji łańcuchowych i cykli (2)

$P' := \emptyset$; $/* \emptyset$ - zbiór pusty $*/$

for $p \in P$ do

begin

if $p = (B \rightarrow \alpha)$ and $\alpha \notin V$ then $/* p$ nie jest łańcuchowa $*/$

begin

$M_B := \{A \in V \mid B \in V_A\};$ $/* M_B = \{ A \mid A \Rightarrow_G^* B \} */$

for $A \in M_B$ do

$P' := P' \cup \{ A \rightarrow \alpha \};$

end;

end;



Przykład usuwania produkcji łańcuchowych

Przykład: Rozpatrujemy gramatykę:

- (1) $E \rightarrow E + T$
- (2) $E \rightarrow T$
- (3) $T \rightarrow T * F$
- (4) $T \rightarrow F$
- (5) $F \rightarrow (E)$
- (6) $F \rightarrow id$

Wyznaczamy odpowiednie zbiory nieterminali:

$$V_E = \{ E, T, F \}$$

$$V_T = \{ T, F \}$$

$$V_F = \{ F \}$$

$$M_E = \{ E \}$$

$$M_T = \{ E, T \}$$

$$M_F = \{ E, T, F \}$$

Analizujemy kolejno wszystkie produkcje gramatyki przekształcając

$$(1) E \rightarrow E + T$$

(2) wypada, bo jest łańcuchowa

$$(3) E \rightarrow T * F \quad T \rightarrow T * F$$

(4) wypada, bo jest łańcuchowa

$$(5) E \rightarrow (E) \quad T \rightarrow (E) \quad F \rightarrow (E)$$

$$(6) E \rightarrow id \quad T \rightarrow id \quad F \rightarrow id$$



Gramatyka prawidłowa Lewostronna rekursja

Gramatyka $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ nazywa się prawidłową, jeśli:

- (i) nie zawiera cykli,
- (ii) jest gramatyką bez ε -produkcji,
- (iii) nie zawiera nieużytecznych symboli.

$U \in V$ jest symbolem lewostronnie rekursywnym gramatyki G jeżeli $U \Rightarrow^+_G U\beta$; $\beta \in (V \cup \Sigma)^*$

Gramatyka G nazywa się lewostronnie rekursywną \Leftrightarrow
 $(\exists U \in V) (U \Rightarrow^+_G U\beta ; \beta \in (V \cup \Sigma)^*)$



Usuwanie lewostronnej rekursji (1)

Twierdzenie

Dana jest gramatyka $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ oraz symbol nieterminalny $A \in V$.

Zbiór $\{ A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_m \mid \beta_1 \mid \dots \mid \beta_n \} \subseteq P$ jest zbiorem wszystkich produkcji mających po lewej stronie symbol A ; żadne z $\beta_i \in (V \cup \Sigma)^*$ ($i = 1, \dots, n$) nie zaczyna się od symbolu A .

Niech $G' = \langle V \cup \{A'\}, \Sigma, P', S \rangle$ będzie nową gramatyką, w której A' – nowy symbol nieterminalny, zaś zbiór produkcji P' jest określony jak niżej

$$P' = P - \{ A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_m \mid \beta_1 \mid \dots \mid \beta_n \} \cup \{ A \rightarrow \beta_1 A' \mid \dots \mid \beta_n A' \} \\ \cup \{ A' \rightarrow \alpha_1 A' \mid \dots \mid \alpha_m A' \mid \varepsilon \}$$

Wówczas zachodzi $L(G) = L(G')$.

Twierdzenie powyższe podaje sposób usuwania lewej rekursji, ale tylko bezpośredniej.

Twierdzenie

Każdą gramatykę bezkontekstową $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ można przekształcić do postaci równoważnej $G' = \langle V', \Sigma', P', S \rangle \in \mathcal{G}_{BK}$ (to znaczy takiej, że $L(G') = L(G)$) nie zawierającej (bezpośredniej i pośredniej) lewostronnej rekursji.



Przykład usuwania lewostronnej rekursji

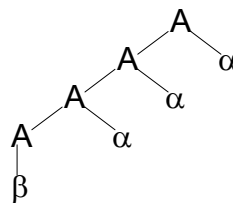
Przykład:

$$A \rightarrow A\alpha$$

$$A \rightarrow \beta$$

Analizowane

słowo : $\beta\alpha\alpha\alpha$



Po modyfikacji:

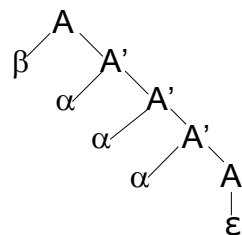
$$A \rightarrow \beta A'$$

$$A' \rightarrow \varepsilon$$

$$A' \rightarrow \alpha A'$$

Analizowane

słowo : $\beta\alpha\alpha\alpha$





Algorytm usuwania lewostronnej rekursji (1)

wejście: $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$,

G - gramatyka prawidłowa!

wyjście: $G' = \langle V', \Sigma, P', S \rangle \in \mathcal{G}_{BK}$ taka, że

(i) G' - nie jest gramatyką lewostronnie rekursywną

(ii) $L(G') = L(G)$

Metoda:

numerujemy symbole nieterminalne, czyli

$V := \{A_1, \dots, A_n\}; n := \#V$

$V' := V$;

$P' := P$;

$i := 1$;



Algorytm usuwania lewostronnej rekursji (2)

repeat

if $\exists(A_i \rightarrow A_i\delta) \in P'$ and $\delta \in (V \cup \Sigma)^*$ then begin

$V' := V' \cup \{A_i'\}$

$P' := P' - \{A_i \rightarrow A_i\alpha_1 \mid \dots \mid A_i\alpha_q \mid \beta_1 \mid \dots \mid \beta_p\}$

$\cup \{A_i \rightarrow \beta_1 A_i' \mid \dots \mid \beta_p A_i'\} \cup \{A_i' \rightarrow \alpha_1 A_i' \mid \dots \mid \alpha_q A_i' \mid \varepsilon\}$ end;

/* q - liczba wszystkich produkcji $A_i \rightarrow A_i \dots$ (lewostr. rekurs.)

żadne z β_1, \dots, β_p nie zaczyna się od A_i

p - liczba wszystkich produkcji $A_i \rightarrow \dots$ bez lewej rekursji */

if $i = n$ then STOP else $i := i + 1$;

for $j := 1$ to $i - 1$ do

$P' := P' - \{A_i \rightarrow A_j\alpha_1 \mid \dots \mid A_j\alpha_x\} \cup \{(A_i \rightarrow \gamma_k\alpha_1) \mid l = 1, \dots, x; k = 1, \dots, m$
 γ_k - prawe strony produkcji: $A_j \rightarrow \gamma_1 \mid \dots \mid \gamma_k \mid \dots \mid \gamma_m\}$;

/* x - liczba wszystkich produkcji typu $A_i \rightarrow A_j\alpha$

m - liczba wszystkich produkcji typu $A_j \rightarrow \gamma_1 \mid \dots \mid \gamma_m$;

w ten sposób prawa strona każdej produkcji $A_i \rightarrow \dots$ zaczyna się albo od symbolu terminalnego, albo od nieterminala A_j o numerze $j > i$ */

until FALSE;

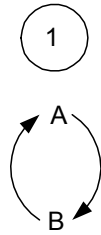


Przykład usuwania lewostronnej rekursji (1)

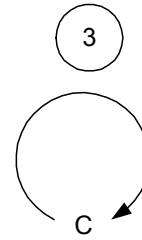
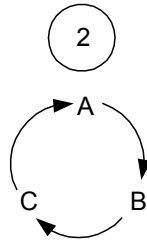
Przykład:

$A \rightarrow BC \mid a$
 $B \rightarrow CA \mid Ab$
 $C \rightarrow AB \mid CC \mid a$

Rodzaje rekursji lewostronnej:



rekursja posrednia



rekursja
bezposrednia



Przykład usuwania lewostronnej rekursji (2)

Przykład:

$A \rightarrow BC \mid a$
 $B \rightarrow CA \mid Ab$
 $C \rightarrow AB \mid CC \mid a$

I	$i = 1$	G bez zmian
II	$i = 2 ; j = 1$	$B \rightarrow CA \mid BCb \mid ab$
I	$i = 2$	$B \rightarrow CAB' \mid abB'$
		$B' \rightarrow CbB' \mid \epsilon$
II	$i = 3 ; j = 1$	$C \rightarrow BCB \mid aB \mid CC \mid a$
	$i = 3 ; j = 2$	$C \rightarrow CAB'CB \mid abB'CB \mid aB \mid CC \mid a$
I	$i = 3$	$C \rightarrow abB'CBC' \mid aBC' \mid aC'$
		$C' \rightarrow AB'CBC' \mid CC' \mid \epsilon$

Ostatecznie:

$A \rightarrow BC \mid a$
 $B \rightarrow CAB' \mid abB'$
 $B' \rightarrow CbB' \mid \epsilon$
 $C \rightarrow abB'CBC' \mid aBC' \mid aC'$
 $C' \rightarrow AB'CBC' \mid CC' \mid \epsilon$



Usuwanie lewostronnej rekursji (2)

Wariant drugi usuwania lewostronnej rekursji – dla potrzeb przekształcania gramatyki bezkontekstowej do [postaci normalnej Greibach](#)

Twierdzenie (wariant drugi)

Dana jest gramatyka $G = \langle V, \Sigma, P, S \rangle \in \mathcal{G}_{BK}$ oraz symbol nieterminalny $A \in V$.

Zbiór $\{ A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_m \mid \beta_1 \mid \dots \mid \beta_n \} \subseteq P$ jest zbiorem wszystkich produkcji mających po lewej stronie symbol A ; żadne z $\beta_i \in (V \cup \Sigma)^*$ ($i = 1, \dots, n$) nie zaczyna się od symbolu A .

Niech $G' = \langle V \cup \{A'\}, \Sigma, P', S \rangle$ będzie nową gramatyką, w której A' – nowy symbol nieterminalny, zaś zbiór produkcji P' jest określony jak niżej

$$P' = P - \{ A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_m \} \cup \{ A \rightarrow \beta_1 A' \mid \dots \mid \beta_n A' \} \\ \cup \{ A' \rightarrow \alpha_1 A' \mid \dots \mid \alpha_m A' \mid \alpha_1 \mid \dots \mid \alpha_m \}$$

Wówczas zachodzi $L(G) = L(G')$.

Twierdzenie podaje drugi sposób usuwania bezpośredniej lewej rekursji (nie powodujący dodawania ϵ -produkcji, ale zwiększający liczbę nowych produkcji wprowadzanych do gramatyki). Cała reszta algorytmu pozostaje niezmieniona – por. <http://kompilatory.agh.edu.pl>