



AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Gramatyki, wyprowadzenia, rozbiór

Języki formalne i automaty

Dr inż. Janusz Majewski
Katedra Informatyki

Gramatyka

Gramatyką G nazywamy czwórkę uporządkowaną

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

gdzie:

V – zbiór symboli nieterminalnych,

Σ – zbiór symboli terminalnych,

P – zbiór produkcji, z których każda ma postać
 $\alpha \rightarrow \beta$

$S \in V$ – wyróżniony symbol początkowy
(nieterminal)

przy czym

$$P \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^*$$

$$P = \{ \alpha \rightarrow \beta \mid \alpha \in (V \cup \Sigma)^+, \beta \in (V \cup \Sigma)^* \}$$

Przykład – palindromy (1)

Palindromem nazywamy łańcuch, który czytany od przodu oraz czytany wspak brzmi tak samo, czyli

$$x^R = x$$

Definicja rekurencyjna palindromu:

- 1) ε jest palindromem,
- 2) jeśli a jest symbolem ($a \in \Sigma$), to łańcuch zbudowany z pojedynczego symbolu a jest palindromem,
- 3) jeśli x jest palindromem, zaś a jest symbolem ($a \in \Sigma$), to łańcuch axa jest palindromem,
- 4) nic innego nie jest palindromem poza tym, co wynika z punktów (1), (2) i (3).

Przykład – palindromy (2)

Niech $\Sigma = \{a, b\}$

Definicja rekurencyjna palindromu:

1) ε jest palindromem,

$$S \rightarrow \varepsilon$$

2) jeśli a jest symbolem ($a \in \Sigma$), to łańcuch zbudowany z pojedynczego symbolu a jest palindromem,

$$S \rightarrow a \quad S \rightarrow b$$

3) jeśli S jest palindromem, zaś a jest symbolem ($a \in \Sigma$), to łańcuch aSa jest palindromem.

$$S \rightarrow aSa \quad S \rightarrow bSb$$

Przykład – palindromy (3)

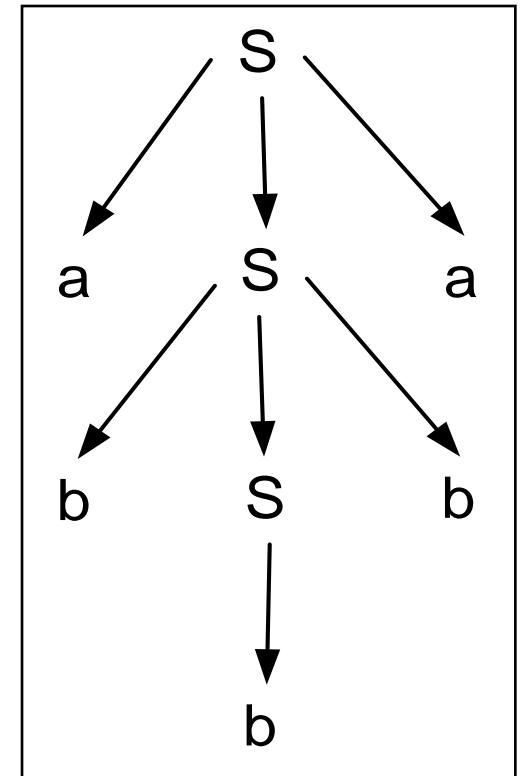
$G = \langle V = \{S\}, \Sigma = \{a, b\}, P = \{ S \rightarrow \varepsilon; S \rightarrow a; S \rightarrow b; S \rightarrow aSa; S \rightarrow bSb \}, S = S \rangle$

Krótszy zapis: $S \rightarrow \varepsilon \mid a \mid b \mid aSa \mid bSb$

Przykładowe wyprowadzenie słowa: abbba

$S \Rightarrow S \xrightarrow{S \rightarrow aSa} aSa \Rightarrow S \xrightarrow{S \rightarrow bSb} abSba \Rightarrow S \xrightarrow{S \rightarrow b} abbba$

Drzewo wyprowadzenia:



Przykład

$S \rightarrow aSBa \mid aba$
 $aB \rightarrow Ba$
 $bB \rightarrow bb$

język: $\{ a^n b^n a^n \mid n \geq 0 \}$

S
aba

S
a**S**Ba
aaba**B**a
aab**B**aa
aabbaa

S
a**S**Ba
aa**S**BaBa
aaaba**B**aBa
aaab**B**aaBa
aaabbaa**B**a
aaabba**B**aa
aaabb**B**aaa
aaabbbaaa



Język generowany przez gramatykę

Gramatyka jest jednym ze sposobów definiowania języka formalnego. Mając daną gramatykę G oznaczamy przez $L(G)$ zbiór wszystkich słów x , które mogą być w tej gramatyce wyprowadzone z symbolu początkowego S . Zbiór ten nazywamy językiem generowanym przez daną gramatykę.

$$L(G) = \{ x \in \Sigma^* \mid S \Rightarrow_G^* x \}$$

Automat skończony vs. gramatyka

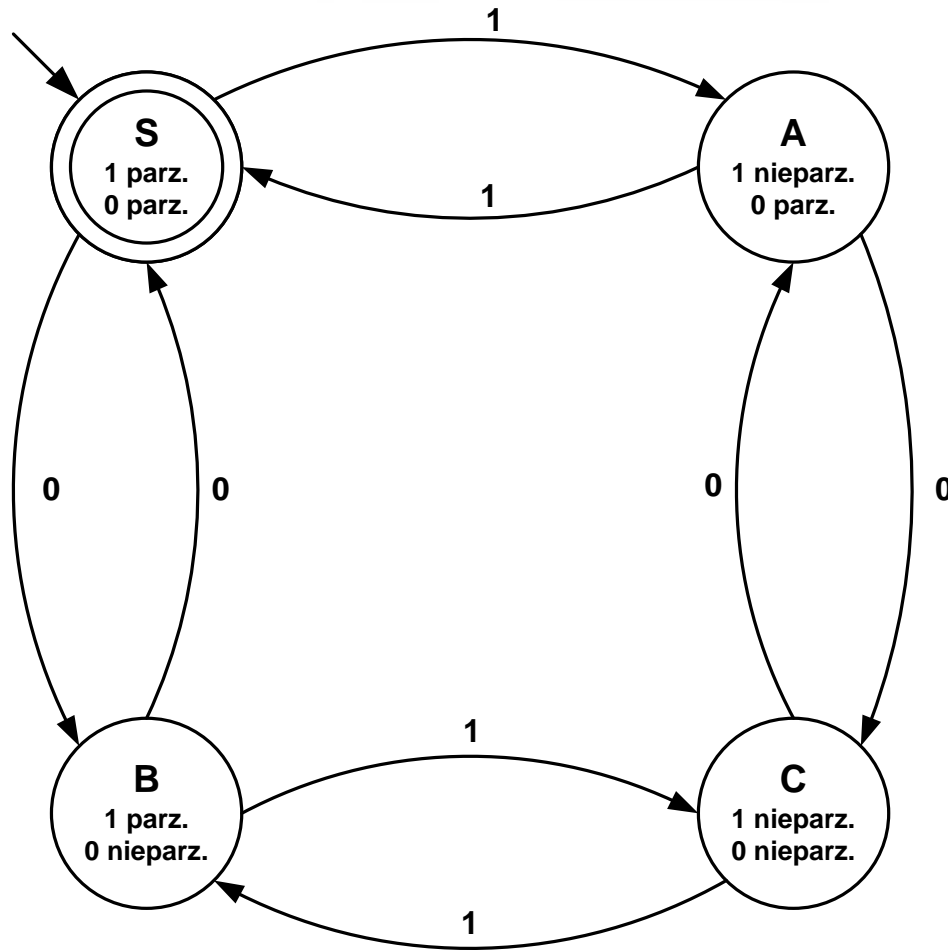
Rozważamy język nad alfabetem binarnym $\Sigma = \{0, 1\}$ składający się z łańcuchów zero-jedynkowych o tej własności, że liczba zer w każdym łańcuchu jest parzysta i liczba jedynek w każdym łańcuchu też jest parzysta. Wszystkie łańcuchy binarne możemy podzielić na cztery grupy:

- S – łańcuchy z parzystą liczbą jedynek i parzystą liczbą zer,
- A – łańcuchy z parzystą liczbą jedynek i nieparzystą liczbą zer,
- B – łańcuchy z nieparzystą liczbą jedynek i parzystą liczbą zer,
- C – łańcuchy z nieparzystą liczbą jedynek i nieparzystą liczbą zer.

Analizujemy łańcuch zero-jedynkowy symbol po symbolu od lewej strony. Przed rozpoczęciem analizy jesteśmy w grupie S (łańcuch pusty zawiera zero jedynek i tyleż zer, więc liczba jedynek i liczba zer w tym łańcuchu są parzyste). Jeśli pierwszym symbolem jest jedynka – przechodzimy do grupy A (wtedy liczba jedynek jest nieparzysta, a liczba zer jest dalej parzysta), zaś jeśli pierwszym symbolem jest zero – przechodzimy do grupy B (wtedy liczba zer jest nieparzysta, a liczba jedynek jest dalej parzysta). Zapiszmy to tak:

S → 1A | 0B

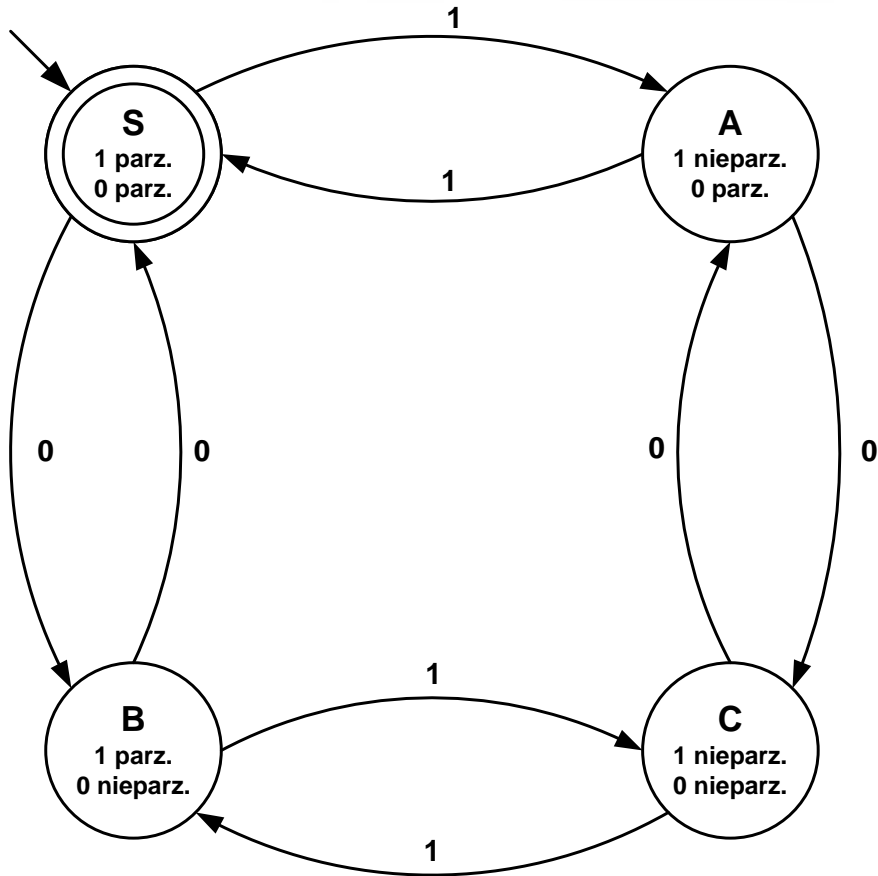
Podobnie czynimy z dalszymi symbolami...



Wreszcie, gdy przeczytaliśmy cały łańcuch, sprawdzamy, czy zatrzymaliśmy się w grupie S. Jeśli tak – badany łańcuch spełnia nałożony nań warunek parzystej liczby jedynek i parzystej liczby zer. Wówczas należy wyeliminować z wyprowadzenia symbol S (odpowiedni zapis: $\mathbf{S} \rightarrow \varepsilon$). Ostatecznie gramatyka naszego języka ma postać:

$\mathbf{S} \rightarrow$	$\mathbf{1A}$	$ $	$\mathbf{0B}$	$ $	ε
$\mathbf{A} \rightarrow$	$\mathbf{1S}$	$ $	$\mathbf{0C}$		
$\mathbf{B} \rightarrow$	$\mathbf{1C}$	$ $	$\mathbf{0S}$		
$\mathbf{C} \rightarrow$	$\mathbf{1B}$	$ $	$\mathbf{0A}$		

Automat skończony vs. gramatyka c.d.



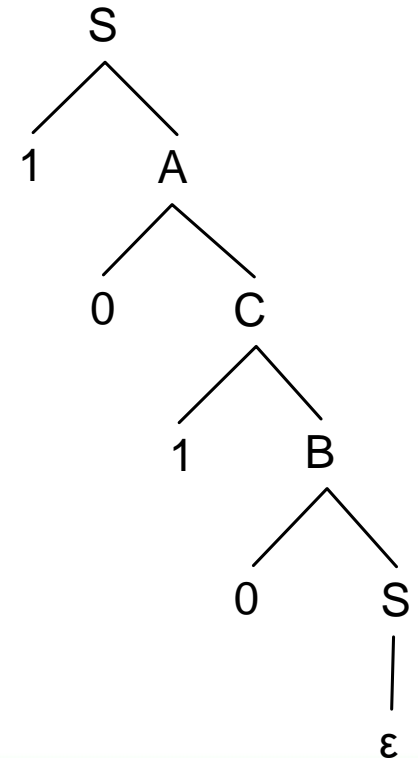
$S \rightarrow 1A \mid 0B \mid \varepsilon$

$A \rightarrow 1S \mid 0C$

$B \rightarrow 1C \mid 0S$

$C \rightarrow 1B \mid 0A$

Drzewo wyprowadzenia dla
słowa: 1010



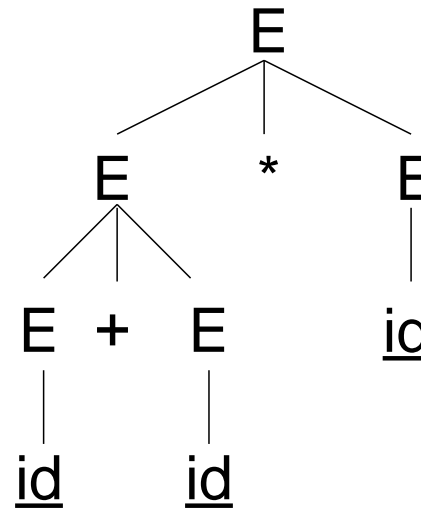
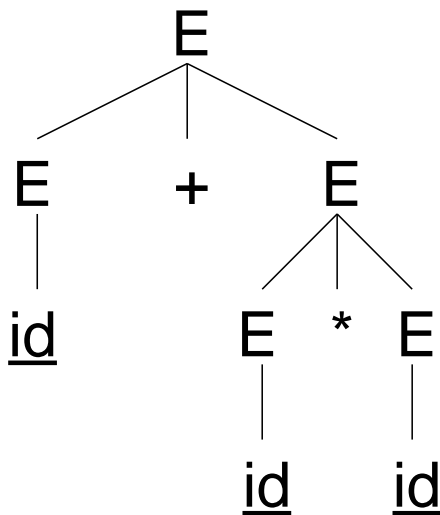
Wyprowadzenie dla słowa: 1010

$S \Rightarrow 1A \Rightarrow 10C \Rightarrow 101B \Rightarrow 1010S \Rightarrow 1010$

Gramatyka wyrażeń – przykład (1)

$G = \langle \{E\}, \{+, *, (,), \underline{id}\}, \{E \rightarrow E + E \mid E * E \mid (E) \mid \underline{id}\}, E \rangle$

Analizowane słowo: id + id * id



Dla pewnego słowa (u nas: id + id * id) udało się zbudować dwa różne drzewa rozbioru syntaktycznego. Taka gramatyka jest niejednoznaczna.

Gramatyka wyrażeń – przykład (2)

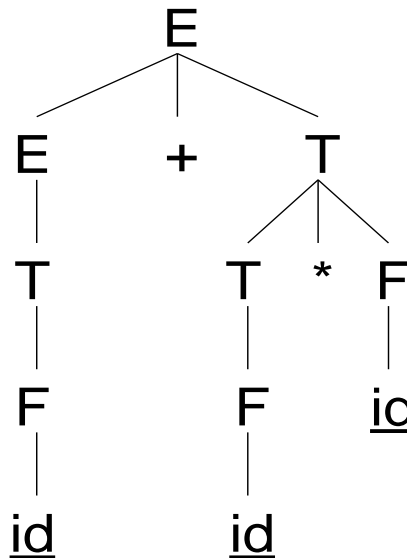
$G = \langle \{E, T, F\}, \{+, *, (,), \underline{id}\}, \{$
 $E \rightarrow E + T \mid T$
 $T \rightarrow T * F \mid F$
 $F \rightarrow (E) \mid \underline{id} \quad \}, E \rangle$

Analizowane słowo: id + id * id

Wyprowadzenie
lewostronne

E
 $E+T$
 $T+T$
 $F+T$
 $\underline{id}+T$
 $\underline{id}+T*F$
 $\underline{id}+F*F$
 $\underline{id}+\underline{id}*F$
 $\underline{id}+\underline{id}*\underline{id}$

Drzewo rozbioru
syntaktycznego



Wyprowadzenie
prawostronne

E
 $E+T$
 $E+T*F$
 $E+T*\underline{id}$
 $E+F*\underline{id}$
 $E+\underline{id}*\underline{id}$
 $T+\underline{id}*\underline{id}$
 $F+\underline{id}*\underline{id}$
 $\underline{id}+\underline{id}*\underline{id}$

Jednoznaczność gramatyki

- Dla każdego drzewa rozbioru syntaktycznego istnieje co najmniej jedno wyprowadzenie słowa języka $L(G)$ w gramatyce G .
- Dla każdego wyprowadzenia słowa istnieje odpowiadające mu drzewo rozbioru syntaktycznego. Kilku różnym wyprowadzeniom mogą odpowiadać identyczne drzewa rozbioru syntaktycznego.
- Dwa wyprowadzenia są równoważne, gdy odpowiadające im drzewa rozbioru syntaktycznego są identyczne.
- Słowo języka $L(G)$ jest niejednoznaczne w gramatyce G , jeśli jego wyprowadzenia można opisać przez co najmniej dwa różne drzewa rozbioru syntaktycznego.
- Gramatyka G jest niejednoznaczna, jeśli w języku $L(G)$ **istnieje** co najmniej jedno niejednoznaczne słowo w tej gramatyce. W przeciwnym wypadku gramatyka jest jednoznaczna. W gramatyce jednoznacznej istnieje dokładnie jedno wyprowadzenie lewostronne i dokładnie jedno wyprowadzenie prawostronne (wśród wszystkich równoważnych wyprowadzeń tego samego słowa).

Metody rozbioru gramatycznego

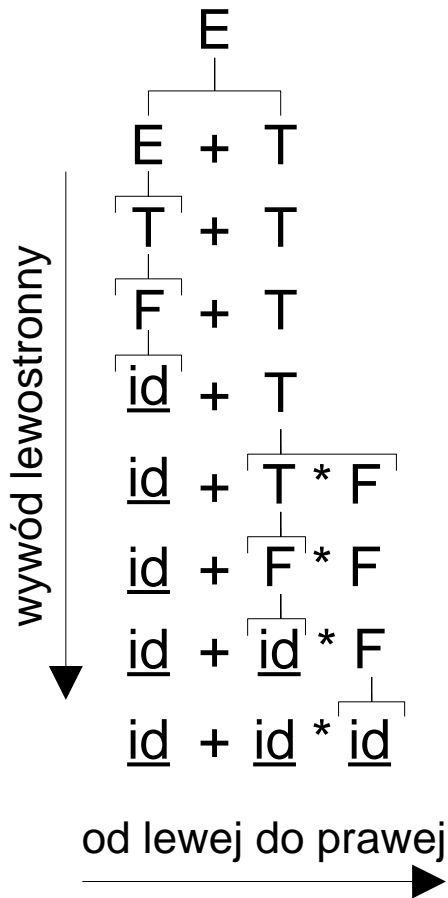
Problem rozbioru gramatycznego jest próbą odpowiedzi na pytanie: „czy dany ciąg symboli terminalnych jest słowem należącym do języka $L(G)$ generowanego przez daną gramatykę?”. Istota postępowania jest następująca: jeśli dla danego słowa można zbudować drzewo rozbioru w danej gramatyce, to słowo to należy do języka generowanego przez gramatykę, jeśli zaś dla badanego słowa nie można zbudować drzewa rozbioru, wówczas słowo nie należy do języka. W zależności od strategii budowania drzewa rozbioru stosujemy jeden z dwóch algorytmów:

- algorytm wyprowadzający „top-down” (budujemy drzewo w kolejności: od korzenia ku liściom, odtwarzając wyprowadzenie lewostronne)
- algorytm redukujący „bottom-up” (budujemy drzewo w odwrotnej kolejności: od liści ku korzeniowi, odtwarzając wyprowadzenie prawostronne)

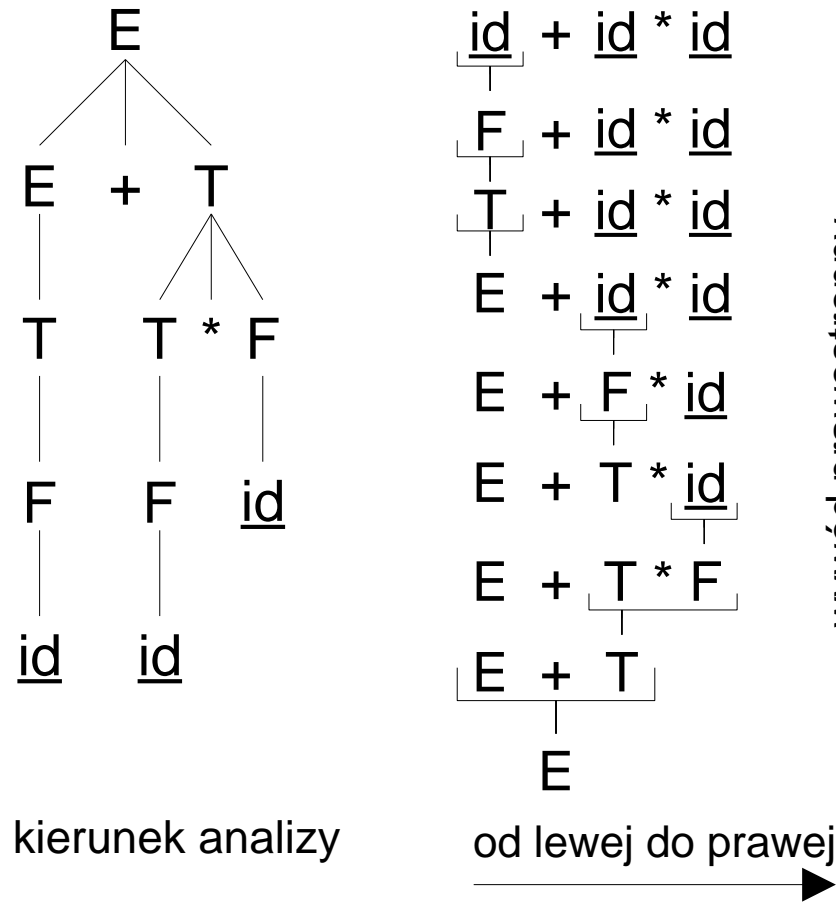
Rozważany ciąg symboli badany jest zawsze od lewej strony, w tym sensie oba algorytmy są lewostronne.

Przykład

Top-down



Bottom-up



Przykład:

Gramatyka:

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

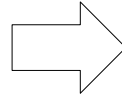
$F \rightarrow (E) \mid \text{id}$

Analizowane
słowo:

id + id * id

Metoda „top-down”

Symbol początkowy
gramatyki



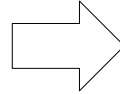
Analizowane słowo

Rozpoczynamy od symbolu wyróżnionego gramatyki. W każdym kroku najbardziej lewy symbol nieterminalny zastępujemy prawą stroną jakiejś produkcji. Jeśli w wyniku kolejnego kroku otrzymamy symbol terminalny, to sprawdzamy, czy jest on identyczny z odpowiednim symbolem analizowanego słowa. Jeśli nie, to możemy „cofnąć się” i spróbować od nowa, korzystając w ponownym wyprowadzeniu z jakiejś innej produkcji (lub innych produkcji). Byłby to algorytm z powrotami (mało efektywny). Wynik: akceptacja słowa lub odrzucenie w wyniku przebadania wszystkich możliwości.

Problem: „jak uniknąć konieczności powrotów” może być pozytywnie i jednoznacznie rozstrzygnięty dla pewnej klasy gramatyk bezkontekstowych zwanych gramatykami $LL(k)$. „Top-down” odtwarza wyprowadzenie lewostronne.

Metoda „bottom-up”

Analizowane słowo



Symbol początkowy
gramatyki

Analizując słowo od lewej strony bierzemy symbol bądź grupę symboli stanowiących prawą stronę produkcji jakiejś produkcji i zastępujemy ją lewą stroną tej produkcji. Postępujemy tak długo, aż dojdziemy do symbolu wyróżnionego gramatyki, co jest równoznaczne z akceptacją słowa. W każdym kroku powinna być zredukowana prawa strona jakiejś produkcji. Także w tej metodzie możliwe jest postępowanie z powrotami.

Problem: „jak znaleźć prawą stronę produkcji nie mając jeszcze drzewa rozbioru syntaktycznego i czym ją zastąpić” może być pozytywnie i jednoznacznie rozstrzygnięty dla pewnych klas gramatyk bezkontekstowych: gramatyk precedencyjnych, czy tzw. gramatyk LR(k). „Bottom-up” odtwarza wyprowadzenie prawostronne.