MINUT 2021 (3), s. 245–254
ISSN 2719-3063

Robert BRZESKI[1]

[1]Wydział Automatyki, Elektroniki i Informatyki, Politechnika Śląska, ul. Akademicka 16, 44-100 Gliwice

Gramatyki formalne dla prostych wyrażeń arytmetycznych

Streszczenie. W procesie dydaktycznym dotyczącym gramatyk formalnych, jednym z często używanych przykładów jest definiowanie gramatyki dla wyrażeń algebraicznych. W rozwiązaniach generujących reguły tworzenia słów, uwzględnia się priorytet wykonywania poszczególnych działań algebraicznych. W bieżącym artykule przedstawiono kilka takich rozwiązań, a następnie zaproponowano rozwiązania generujące taki sam język, ale w postaci mniejszej liczby reguł generujących słowa. Zrealizowano to poprzez pominięcie w regułach zdefiniowanej gramatyki informacji o priorytetach działań algebraicznych. W ten sposób przedstawiono możliwość znacznego uproszczenia zdefiniowanej gramatyki, która bardzo często przedstawiana jest studentom, przy okazji omawiania języków i gramatyk formalnych.

Słowa kluczowe: gramatyka formalna, język, alfabet, wyrażenia algebraiczne.

1. Wprowadzenie do gramatyk formalnych

Omawianie gramatyk formalnych rozpocząć należy od opisu języka formalnego [1, 4]. Składa się on ze zbioru słów. Pojedyncze słowo utworzone jest z elementów alfabetu. Poszczególne elementy alfabetu, zwane symbolami, mogą się w danym słowie powtarzać, przy czym liczba wszystkich symboli w słowie jest wartością skończoną. Alfabet to zbiór wszystkich symboli, z jakich można tworzyć słowa. Aby utworzone słowo mogło być zaliczone do zbioru języka, musi być prawidłowe, czyli spełniać określone wymagania. Wymagania te zdefiniowane są w postaci zestawu reguł, które nazywamy gramatyką [2,3]. Reguły tworzące gramatykę mają za zadanie generowanie jedynie słów prawidłowych. Jednocześnie gramatyka musi dać możliwość wygenerowania wszystkich słów prawidłowych, należących do danego języka.

Elementy języka formalnego:

- alfabet - dowolny skończony zbiór symboli, z których będą tworzone słowa języka,
- słowo - skończony ciąg symboli alfabetu,
- język - zbiór wszystkich prawidłowych słów,

Autor korespondencyjny: R. Brzeski (Robert.Brzeski@polsl.pl).
Data wpłynięcia: 01.10.2020.

- gramatyka - zbiór reguł umożliwiających generowanie słów danego języka, jak też rozróżnienie słów poprawnie zbudowanych od zbudowanych niepoprawnie.

Definicja gramatyki formalnej – Gramatyka G składa się z czterech uporządkowanych elementów:

$G =< V, \Sigma, P, \sigma >$ gdzie:

V – skończony zbiór symboli terminalnych (alfabet języka),

Σ – skończony zbiór symboli nieterminalnych (symbole pomocnicze),

P – skończony zbiór reguł definiujących zasady tworzenia słów,

σ – symbol początkowy (zwany symbolem startowym, głową lub aksjomatem gramatyki), jest jednym z symboli nieterminalnych.

Do definiowania zestawu reguł gramatyki możemy korzystać nie tylko z alfabetu języka, ale także z symboli pomocniczych. Po zdefiniowaniu wszystkich reguł, trzeba ustalić, od której reguły zaczyna się generowanie każdego słowa.

Język L generowany przez gramatykę G jest to zbiór wszystkich możliwych słów, które mogą być wygenerowane z symbolu początkowego σ na podstawie zbioru reguł P.

Gramatyka musi więc spełniać m.in. następujące dwa wymagania:

- Musi generować jedynie słowa prawidłowe.
- Musi dać możliwość wygenerowania wszystkich słów należących do danego języka.

Jednocześnie pamiętać należy, że różne definicje gramatyk mogą generować dokładnie ten sam język [5].

2. Zadanie do realizacji

W ramach omawiania tematu gramatyk formalnych, bardzo często, jako przykłady, przedstawia się gramatykę generującą proste wyrażenia algebraiczne. Poszczególne wyrażenia, czyli generowane przez gramatykę słowa, mogą zawierać argumenty, operatory algebraiczne oraz nawiasy.

Elementy prostych wyrażeń algebraicznych:

- argument – pojedyncza litera alfabetu,
- operatory algebraiczne (dodawanie, mnożenie, ewentualnie także: odejmowanie, dzielenie, potęgowanie),
- nawiasy.

Przykłady wyrażeń algebraicznych, czyli wygenerowanych słów, mogą być następujące:

- a,
- $a + b$,
- $a * b + c + d * (e + f * g + h)$,
- $\frac{a * b + c + d}{(\frac{e}{f} - g^h)}$, czyli wyrażenie, które w postaci słowa generowanej gramatyki można zapisać jako: $a * b + c + d/(e/f - \mathrm{g}\hat{}h\ )$.

Jako rozwiązanie należy przedstawić gramatykę, umożliwiającą generowanie takich prostych wyrażeń algebraicznych. Gramatyka ta musi generować jedynie prawidłowe wyrażenia algebraiczne i jednocześnie musi umożliwić wygenerowanie dowolnego prawidłowego wyrażenia algebraicznego.

Przykłady nieprawidłowych wyrażeń algebraicznych, czyli słów, które nie należą do zbioru języka, mogą być następujące:

- $ab + c$,
- $a + ((b)$,
- $a + *b$,
- $a+$.

Tworzona gramatyka musi zawierać, zgodnie z definicją, wszystkie 4 elementy tworzące gramatykę. Rozwiązanie najczęściej przedstawiane jest w postaci gramatyki Chomsky'ego oraz notacji BNF.

3. Typowe rozwiązania

W ramach zazwyczaj przedstawianych rozwiązań, tworzona jest gramatyka uwzględniająca priorytet operatorów algebraicznych.

3.1. Dodawanie, mnożenie, nawiasy – gramatyka Chomsky'ego

Jako pierwsze zostanie przedstawione rozwiązanie dla wyrażeń algebraicznych, zawierających argumenty, nawiasy oraz operatory dodawania i mnożenia. Rozwiązanie uwzględnia wyższy priorytet mnożenia względem dodawania.

$G =< V, \Sigma, P, \sigma >$
$V = \{a, b, ..., z, +, *, (,)\}$
$\Sigma = \{E, M, A\}$
$P = \{$

$E \to E + E$
$E \to M$
$M \to M * M$

$M \to A$
$A \to (E)$
$A \to a$
...
$A \to z$
}
$\sigma = E$

Cała gramatyka G zawiera 4 elementy:

V – zbiór symboli z których można utworzyć słowo (wyrażenie algebraiczne). W skład tego słowa wchodzą nazwy argumentów: od 'a' poprzez 'b' aż do 'z'; możliwe do użycia operatory algebraiczne: dodawanie '+', mnożenie '$*$'; nawias otwierający '(' oraz nawias zamykający ')'.

Σ – zbiór symboli pomocniczych: E, M, A.

P – zbiór reguł poprzez które generowane będą słowa – wyrażenia algebraiczne:

$\boldsymbol{E \to E + E}$
Za pomocą tej reguły można wygenerować operację dodawania w wyrażeniu algebraicznym. Podstawiając za E ponownie całą definicję $E \to E + E$, można utworzyć dowolnie długi ciąg kolejnych operatorów dodawania: wyrażenie $E \to$ wyrażenie E_1+ wyrażenie $E_2 + \cdots +$ wyrażenie E_n, dla $n > 1$.

$\boldsymbol{E \to M}$
Za pomocą tej reguły mamy możliwość podstawienia za symbol E, symbolu pomocniczego M. W ten sposób korzystając jedynie z tej reguły, można utworzyć pojedynczy symbol M lub korzystając wcześniej z reguły $E \to E + E$, można utworzyć dowolnie długi ciąg $M + M + \ldots + M$.

$\boldsymbol{M \to M * M}$
Analogicznie jak dla $E \to E + E$, w tej regule można utworzyć dowolnie długi ciąg kolejnych operatorów mnożenia: wyrażenie $M \to$ wyrażenie M_1* wyrażenie $M_2 * \cdots *$ wyrażenie M_n.

$\boldsymbol{M \to A}$
Za pomocą tej reguły mamy możliwość podstawienia za symbol M symbolu pomocniczego A.

$\boldsymbol{A \to (E)}$
Za pomocą tej reguły mamy możliwość podstawienia za symbol A, symbolu pomocniczego E w nawiasach. W ten sposób można umieścić w wyrażeniu algebraicznym niejako zagnieżdżone, dowolne wyrażenie w nawiasach.

Natomiast korzystając z reguł: $\boldsymbol{A \to a \ldots A \to z}$, można podstawić za symbol A dowolny argument wyrażenia algebraicznego: wyrażenie $A \to a$ lub b lub … lub z.

σ – w ten sposób, za pomocą definicji $\sigma = E$ określamy, że generowanie słowa rozpoczynać się będzie od symbolu pomocniczego E.

W przedstawionych regułach definiujących zasady tworzenia słów, uwzględnione są priorytety, określające kolejność wykonywania działań. Widzimy więc, że wyrażenia w nawiasach są na tym samym poziomie co pojedyncze argumenty. Przy wartościowaniu należałoby podstawić wartości poszczególnych argumentów, wyliczyć wartość wyrażenia w nawiasach i dopiero wtedy realizować kolejne operacje. W definicji reguł kolejnych podstawień jest także operator mnożenia, co pokazuje nam, które działanie algebraiczne należy obecnie wykonać przy wartościowaniu wyrażeń. Na końcu należy dodać poszczególne wartości.

Utworzona gramatyka nie tylko umożliwia wygenerowanie języka (w postaci zbioru wyrażeń algebraicznych), ale jednocześnie pokazuje jak należałoby wartościować uzyskane słowo (wyrażenie algebraiczne). Należy jednak pamiętać, że w ten sposób uzyskujemy jedynie dodatkową informację, na temat pojedynczego słowa, zależności pomiędzy poszczególnymi symbolami alfabetu, a nie zależności pomiędzy poszczególnymi słowami tworzonego języka.

Mając tak zdefiniowaną gramatykę, możemy przy jej użyciu wygenerować dowolne wyrażenie algebraiczne, zawierające argumenty, nawiasy oraz operatory dodawania i mnożenia. Przykładem może być wyrażenie: $a * b + c + d * (e * f + g * h)$.

Poniżej przedstawiono wyprowadzenie/wygenerowanie wyrażenia/słowa z głowy gramatyki σ.

$\sigma = E \rightarrow$
$E + E \rightarrow$
$E + E + E \rightarrow$
$M + M + M \rightarrow$
$M * M + A + M * M \rightarrow$
$A * A + A + A * A \rightarrow$
$A * A + A + A * (E) \rightarrow$
$A * A + A + A * (E + E) \rightarrow$
$A * A + A + A * (M + M) \rightarrow$
$A * A + A + A * (M * M + M * M) \rightarrow$
$A * A + A + A * (A * A + A * A) \rightarrow$
$a * b + c + d * (e * f + g * h)$

Jak widzimy, poszczególne działania algebraiczne są zdefiniowane w ramach kolejnych reguł i ich generowanie należy wykonać w odpowiedniej kolejności, zgodnie z ich priorytetem. Na samym końcu do całej struktury wyrażenia, z już określonymi operatorami, przypisywane są jedynie odpowiednie argumenty.

Dodatkowo kolorem czerwonym pokazano przykład rozwinięcia, zgodnie z regułami P, pojedynczego symbolu Σ, aż do symboli V. Natomiast kolorem niebieskim pokazano ścieżkę kolejnych symboli Σ, z których powstał końcowy symbol V. Takie oznaczenia kolorem czerwonym i niebieskim, będą stosowane także w kolejnych przykładach.

3.2. Dodawanie, mnożenie, nawiasy, odejmowanie, dzielenie, potęgowanie – gramatyka Chomsky'ego

Omówioną w podrozdziale 3.1. gramatykę można rozszerzyć o kolejne działania algebraiczne, np. odejmowanie, dzielenie i potęgowanie.

$G =< V, \Sigma, P, \sigma >$
$V = \{a, b, ..., z, +, *, (,), -, /, \wedge\}$
$\Sigma = \{E, M, A, C\}$
$P = \{$

$$
\begin{array}{l}
\quad E \rightarrow E + E \\
\quad E \rightarrow E - E \\
\quad E \rightarrow M \\
\quad M \rightarrow M * M \\
\quad M \rightarrow M/M \\
\quad M \rightarrow C \\
\quad C \rightarrow C\hat{}C \\
\quad C \rightarrow A \\
\quad A \rightarrow (E) \\
\quad A \rightarrow a \\
\quad ... \\
\quad A \rightarrow z \\
\} \\
\sigma = E
\end{array}
$$

Operacja odejmowania ma taki sam priorytet jak dodawanie, dlatego jest umieszczona na tym samym poziome reguł, jako definicja wyrażenia E. Podobnie dzielenie, umieszczone jest na tym samym poziomie jak mnożenie, jako definicja wyrażenia M. Potęgowanie ma wyższy priorytet niż poprzednie działania, dlatego został utworzony kolejny poziom, jako definicja wyrażenia C. W ten sposób można rozszerzać gramatykę o możliwość generowania wyrażeń zawierających kolejne operacje algebraiczne. W dalszym ciągu brane pod uwagę są priorytety wykonywania poszczególnych operacji.

3.3. Dodawanie, mnożenie, nawiasy, odejmowanie, dzielenie, potęgowanie – notacja BNF

Przedstawioną w podrozdziale 3.2. definicję w postaci gramatyki Chomsky'ego można przekształcić do notacji BNF.

$$
\begin{array}{l}
G =< V, \Sigma, P, \sigma > \\
V = \{a, b, ..., z, (,), +, *, -, /, \wedge\} \\
\Sigma = \{< E >, < M >, < A >, < C >\} \\
P = \{ \\
\quad < E >::=< M > | < E > + < E > | < E > - < E > \\
\quad < M >::=< C > | < M > * < M > | < M > / < M > \\
\quad < C >::=< A > | < C >\hat{}< C > | \\
\quad < A >::= (< E >)|a|b|...|z \\
\} \\
\sigma =< E >
\end{array}
$$

Gramatyka ta generuje na takiej samej zasadzie, ten sam język, jak gramatyka przedstawiona w podrozdziale 3.2. Różnica występuje jedynie w sposobie zapisu samej definicji gramatyki.

4. Proponowane rozwiązania – krótsza definicja gramatyki

Gramatyka w zakresie generacji słów jest prawidłowa, jeżeli spełnia jednocześnie dwa kryteria:

1. Generuje tylko słowa prawidłowe.

2. Umożliwia wygenerowanie wszystkich prawidłowych słów.

Rozwiązania przedstawione w rozdziale 3 oczywiście spełniają te dwa kryteria. Pytanie, które zostaje zadane w ramach tego artykułu, zmierza do zastanowienia się, czy są to rozwiązania optymalne, czy też da się znaleźć rozwiązania prostsze.

Gramatyki przedstawione w rozdziale 3, w ramach swoich reguł, biorą pod uwagę zależności występujące z powodu różnego priorytetu działań algebraicznych. Przy wartościowaniu wyrażeń te zależności są na tyle ważne, że w sposób naturalny zostały uwzględnione w zdefiniowanych gramatykach. W kontekście generowanych słów wnoszą one jednak jedynie informację o zależnościach pomiędzy poszczególnymi symbolami alfabetu. Natomiast w kontekście generowanego języka, taka informacja nie jest nam potrzebna. Dlatego w ramach tego artykułu zaproponowano rozwiązania, które z założenia pomijają informację o wzajemnych priorytetach operacji algebraicznych. Celem jest utworzenie gramatyki generującej w sposób prawidłowy odpowiedni język, czyli gramatykę generującą język złożony jedynie ze słów prawidłowych oraz umożliwiającą wygenerowanie wszystkich prawidłowych słów (należących do tworzonego języka).

Jak się okazuje, takie założenie o nieuwzględnianiu priorytetów działań algebraicznych, umożliwia utworzenie krótszych definicji gramatyk, generujących taki sam język, jak w podrozdziałach 3.1. do 3.3.

4.1. Dodawanie, mnożenie, nawiasy – gramatyka Chomsky'ego – wersja krótsza

W niniejszym podrozdziale została zaprezentowana krótsza definicja gramatyki, generująca dokładnie taki sam język, jak gramatyka zaprezentowana w podrozdziale 3.1.

$$
\begin{array}{l}
G =< V, \Sigma, P, \sigma > \\
V = \{a, b, ..., z, (,), +, *\} \\
\Sigma = \{E\} \\
P = \{ \\
\qquad E \to E + E \\
\qquad E \to E * E \\
\qquad E \to (E) \\
\qquad E \to a \\
\qquad ... \\
\qquad E \to z \\
\quad \} \\
\sigma = E
\end{array}
$$

Zdefiniowane reguły P gramatyki z założenia nie uwzględniają priorytetów działań algebraicznych, dlatego wszystkie reguły można zdefiniować używając tylko jednego symbolu pomocniczego E. Potrzebne działania algebraiczne można generować praktycznie w dowolnej kolejności.

Mając tak zdefiniowaną gramatykę, możemy przy jej użyciu, wygenerować dowolne wyrażenie algebraiczne, zawierające argumenty, nawiasy oraz operatory dodawania i mnożenia. Przykładem może być to samo wyrażenie, co przedstawione w podrozdziale 3.1.: $a * b + c + d * (e * f + g * h)$.

Poniżej przedstawiono wyprowadzenie/wygenerowanie wyrażenia/słowa z głowy gramatyki.

$\sigma = E \to$

$E + E \rightarrow$
$E + E + E \rightarrow$
$E * E + E + E * E \rightarrow$
$E * E + E + E * (E) \rightarrow$
$E * E + E + E * (E + E) \rightarrow$
$E * E + E + E * (E * E + E * E) \rightarrow$
$a * b + c + d * (e * f + g * h)$

4.2. Dodawanie, mnożenie, nawiasy, odejmowanie, dzielenie, potęgowanie – gramatyka Chomsky'ego – wersja krótsza

W niniejszym podrozdziale została zaprezentowana krótsza definicja gramatyki, generująca dokładnie taki sam język jak gramatyka zaprezentowana w podrozdziale 3.2.

$G =< V, \Sigma, P, \sigma >$
$V = \{a, b, ..., z, (,), +, *, -, /, \wedge\}$
$\Sigma = \{E\}$
$P = \{$
$E \rightarrow E + E$
$E \rightarrow E - E$
$E \rightarrow E * E$
$E \rightarrow E/E$
$E \rightarrow \mathrm{E}\hat{}E$
$E \rightarrow (E)$
$E \rightarrow a$
...
$E \rightarrow z$
$\}$
$\sigma = E$

Zdefiniowane reguły P gramatyki, z założenia nie uwzględniają priorytetów działań algebraicznych, dlatego wszystkie reguły, analogicznie jak w gramatyce zdefiniowanej w podrozdziale 4.1., także i w tym przykładzie, można zdefiniować używając tylko jednego symbolu pomocniczego E. Nawet jeżeli dodane zostają kolejne działania algebraiczne, to w dalszym ciągu wystarczy jeden symbol pomocniczy. Stopień zależności pomiędzy poszczególnymi działaniami nie wpływa na możliwość generowania tak uproszczonych reguł (w porównaniu do przykładu z podrozdziału 3.2.). Przy tworzeniu słowa języka, potrzebne działania algebraiczne można generować praktycznie w dowolnej kolejności.

4.3. Dodawanie, mnożenie, nawiasy, odejmowanie, dzielenie, potęgowanie – notacja BNF – wersja krótsza

W niniejszym podrozdziale została zaprezentowana krótsza definicja gramatyki generującej dokładnie taki sam język, jak gramatyka zaprezentowana w podrozdziale 3.2., 3.3. oraz 4.2. Jest to wersja analogiczna jak gramatyka w podrozdziale 4.2., jedynie przekształcona do notacji BNF. Widzimy, że reguły gramatyki dalej można zdefiniować przy użyciu jednego symbolu pomocniczego E.

$G =< V, \Sigma, P, \sigma >$
$V = \{a, b, ..., z, (,), +, *, -, /, \wedge\}$
$\Sigma = \{< E >\}$
$P = \{$
$< E >::=< E > + < E > | < E > - < E > | < E > * < E > | < E > / < E > |$
$< E >\hat{}< E > |(< E >)|a|b|...|z$
$\}$
$\sigma =< E >$

4.4. Dodawanie, mnożenie, nawiasy, odejmowanie, dzielenie, potęgowanie – notacja BNF – wersja krótsza przekształcona

W niniejszym podrozdziale została zaprezentowana krótsza, dodatkowo przekształcona definicja gramatyki w notacji BNF. W wersji tej przekształcono reguły dotyczące generowania działań algebraicznych. Zamiast tworzyć osobno reguły dotyczące poszczególnych działań algebraicznych, wydzielono je wszystkie do osobnej reguły, przy użyciu dodatkowego symbolu pomocniczego OP. W ten sposób, poza nowym wyrażeniem w nawiasach $(< E >)$, wystarcza tylko jedna reguła tworząca kolejne działania algebraiczne, bez względu na ich liczbę $< E >::=< E >< OP >< E >$.

$G =< V, \Sigma, P, \sigma >$
$V = \{a, b, ..., z, (,), +, *, -, /, \wedge\}$
$\Sigma = \{< E >, < OP >\}$
$P = \{$
$< E >::=< E >< OP >< E > |(< E >)|a|b|...|z$
$< OP >::= +| - | * |/|\wedge$
$\}$
$\sigma =< E >$

Efektem końcowym jest więc bardzo krótka, łatwa do analizy i modyfikacji gramatyka, generująca dokładnie taki sam język, jak gramatyka zaprezentowana w podrozdziale 3.2., 3.3., 4.2. oraz 4.3. Mając tak zdefiniowaną gramatykę, możemy przy jej użyciu wygenerować dowolne wyrażenie algebraiczne, zawierające argumenty, nawiasy oraz operatory dodawania, mnożenia, odejmowania, dzielenia, potęgowania. Przykładem może być wyrażenie $\frac{a * b + c + d}{(\frac{e}{f} - g^h)}$ które w postaci słowa generowanej gramatyki można zapisać jako: $a * b + c + d/(e/f - \mathrm{g}\hat{}h)$.

Poniżej przedstawiono wyprowadzenie/wygenerowanie wyrażenia/słowa z głowy gramatyki.

$\sigma = E \rightarrow$
$< E > < OP >< E >\rightarrow$
$< E > < OP >< E >< OP >< E >< OP >< E >\rightarrow$
$< E > < OP >< E >< OP >< E >< OP >< E >\rightarrow$
$< E > < OP >< E >< OP >< E >< OP >< E >< OP > < E > \rightarrow$
$< E > < OP >< E >< OP >< E >< OP >< E >< OP > (< E >) \rightarrow$
$< E > < OP >< E >< OP >< E >< OP >< E >< OP > (< E >< OP >< E >) \rightarrow$

$$<E><OP><E><OP><E><OP><E><OP>(<E><OP><E><OP><E><OP><E>) \rightarrow a*b+c+d/(e/f-g\hat{\ }h)$$

5. Wnioski końcowe

Gramatyka formalna umożliwia tworzenie wyrażeń, które będą mogły być poddane translacji, wartościowaniu, czy też dalszej analizie. Wygenerowane wyrażenia muszą być jednak prawidłowe oraz gramatyka musi dać możliwość wygenerowania *wszystkich* wyrażeń, które są prawidłowe. Trzeba przy tym pamiętać, że gramatyka sama w sobie nie jest ani translatorem tworzonych słów, ani nie służy do wartościowania wyrażeń arytmetycznych lub ich interpretacji. Jej celem jest generowanie słów poprawnych lub sprawdzanie poprawności gramatycznej podanych wyrażeń/słów.

Gramatyki zaprezentowane w rozdziale 3. zawierają w swojej definicji dodatkową informację na temat priorytetów związanych z poszczególnymi działaniami algebraicznymi. Są to jednak reguły odnoszące się do zależności pomiędzy poszczególnymi elementami alfabetu utworzonego słowa, nie ma natomiast informacji na temat ewentualnych zależności pomiędzy utworzonymi słowami.

Zaproponowane uproszczone wersje gramatyk (rozdział 4.) generują dokładnie taki sam język, jak te zazwyczaj przedstawiane jako gramatyki generujące proste wyrażenia algebraiczne (rozdział 3.). Takie uproszczenie pokazuje nie tylko to, że dany język może być generowany przez różne gramatyki oraz że istnieje swego rodzaju możliwość optymalizowania gramatyk, ale także to, że nastawienie wstępne z zakresu analizy tworzonego języka może mieć wpływ na reguły tworzonej gramatyki.

Podziękowania

Autor pragnie podziękować recenzentom za trud włożony w recenzje.

Literatura

1. L. Bloomfield, *Language*, The University of Chicago Press, Chicago London 1933.
2. N. Chomsky, *Three models for the description of language*, IRE Transactions on Information Theory 2(3) (1956), pp. 113-124.
3. N. Chomsky, *Syntactic Structures*, Mouton and Co., The Hague 1957.
4. A. Momot, *Gramatyki formalne - kilka sposobów rozwiązania prostego zadania*, MINUT 2019 (1), s. 10-16 ISSN 2719-3063.
5. A.A. Puntambekar, *Formal Languages And Automata Theory*, Technical Publications Pune, Pune 2008.