Automaty i Języki Formalne

Soma Dutta

Wydział Matematyki i Informatyki, UWM w Olsztynie
soma.dutta@matman.uwm.edu.pl

Wykład - 5: Języki formalne i powiązania z automatami

Semestr zimowy 2023

Podsumowanie

- Omówiliśmy, w jaki sposób automat można przekształcić w automat z minimalną liczbą stanów.
- Następnie rozpoczęliśmy temat języka formalnego, definicji gramatyki formalnej i języka formalnego generowanego z zadanej gramatyki formalnej.
- Przedstawiono także mały przykład generowania języka w oparciu o gramatykę formalną.

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G ($L(G)$).

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G ($L(G)$).

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G $(L(G))$.

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

  $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^nCa^n \Rightarrow a^nba^n \quad n \geq 1$

  (poprzez zastosowanie $S \to aCa$, potem $C \to aCa$ (n-1)-razy i w końcu $C \to b$)

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G $(L(G))$.

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

  $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^n Ca^n \Rightarrow a^n ba^n \qquad n \geq 1$

  (poprzez zastosowanie $S \to aCa$, potem $C \to aCa$ (n-1)-razy i w końcu $C \to b$)

  Więc $\{a^n ba^n : n \geq 1\} \subseteq L(G)$.

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G $(L(G))$.

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

  $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^n Ca^n \Rightarrow a^n ba^n \quad n \geq 1$

  (poprzez zastosowanie $S \to aCa$, potem $C \to aCa$ (n-1)-razy i w końcu $C \to b$)

  Więc $\{a^n ba^n : n \geq 1\} \subseteq L(G)$.

- Dla odwrotnego włączenia musimy pokazać, że dowolny ciąg należący do $L(G)$ ma postać $a^n ba^n$ gdzie $n \geq 1$.

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G $(L(G))$.

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

  $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^nCa^n \Rightarrow a^nba^n \quad n \geq 1$

  (poprzez zastosowanie $S \to aCa$, potem $C \to aCa$ (n-1)-razy i w końcu $C \to b$)

  Więc $\{a^nba^n : n \geq 1\} \subseteq L(G)$.

- Dla odwrotnego włączenia musimy pokazać, że dowolny ciąg należący do $L(G)$ ma postać a^nba^n gdzie $n \geq 1$.

  Ponieważ jedyną produkcją S jest $S \to aCa$ i aby otrzymać ciąg terminali musimy na koniec zastosować $C \to b$, najprostszy sposób uzyskania ciągu terminali przy użyciu produkcji w G jest to: zastosuj powyższe dwie produkcje. Tzn., $S \Rightarrow aCa \Rightarrow aba$.

Przykład: Generowanie języka w oparciu o gramatykę

- Załóżmy, że mamy gramatykę formalną $G = (\{S, C\}, \{a, b\}, P, S)$ gdzie P składa się z produkcji $S \to aCa$, $C \to aCa|b$. Naszym zadaniem jest znalezienie języka wygenerowanego przez G $(L(G))$.

  Rozwązanie: $S \Rightarrow aCa \Rightarrow aba$. Więc, $aba \in L(G)$.

  $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^nCa^n \Rightarrow a^nba^n \quad n \geq 1$

  (poprzez zastosowanie $S \to aCa$, potem $C \to aCa$ (n-1)-razy i w końcu $C \to b$)

  Więc $\{a^nba^n : n \geq 1\} \subseteq L(G)$.

- Dla odwrotnego włączenia musimy pokazać, że dowolny ciąg należący do $L(G)$ ma postać a^nba^n gdzie $n \geq 1$.

  Ponieważ jedyną produkcją S jest $S \to aCa$ i aby otrzymać ciąg terminali musimy na koniec zastosować $C \to b$, najprostszy sposób uzyskania ciągu terminali przy użyciu produkcji w G jest to: zastosuj powyższe dwie produkcje. Tzn., $S \Rightarrow aCa \Rightarrow aba$.

  Albo, po zastosowaniu $S \to aCa$ jako pierwszej produkcji, możemy zastosować $C \to aCa$ skończenie wiele razy, zanim zastosujemy $C \to b$. Tzn., $S \Rightarrow aCa \Rightarrow \ldots \Rightarrow a^nCa^n \Rightarrow a^nba^n$. Więc, $L(G) \subseteq \{a^nba^n : n \geq 1\}$.

Klasyfikacja języków według Noama Chomsky'ego

- Chomsky podzielił gramatyki na cztery typy pod względem produkcji.
- Typ-0: Za produkcję typu-0 uważa się każdą produkcję.
- Definicja: W produkcji w postaci $\phi A \psi \rightarrow \phi \alpha \psi$ gdzie $A \in V_N$, ϕ nazywa się lewym kontekstem, a ψ nazywa się prawym kontekstem zmiennej A.
- Typ-1: Produkcja w postaci $\phi A \psi \rightarrow \phi \alpha \psi$ gdzie $\alpha \neq \Lambda$ jest produkcją typu 1. (W produkcjach typu-1 kasowanie zmiennej A jest niedozwolone.)
- Typ-2: Produkcja typu-2 to produkcja posiadająca formę $A \rightarrow \alpha$ gdzie $A \in V_N$ i $\alpha \in (V_N \cup \Sigma)^*$. Oznacza to, że dla zmiennej A nie ma lewego ani prawego kontekstu.
- Typ-3: Produkcja w postaci $A \rightarrow a$ lub $A \rightarrow aB$, gdzie $A, B \in V_N$ i $a \in \Sigma$ nazywana jest produkcją typu-3.

Cztery typy gramatyki

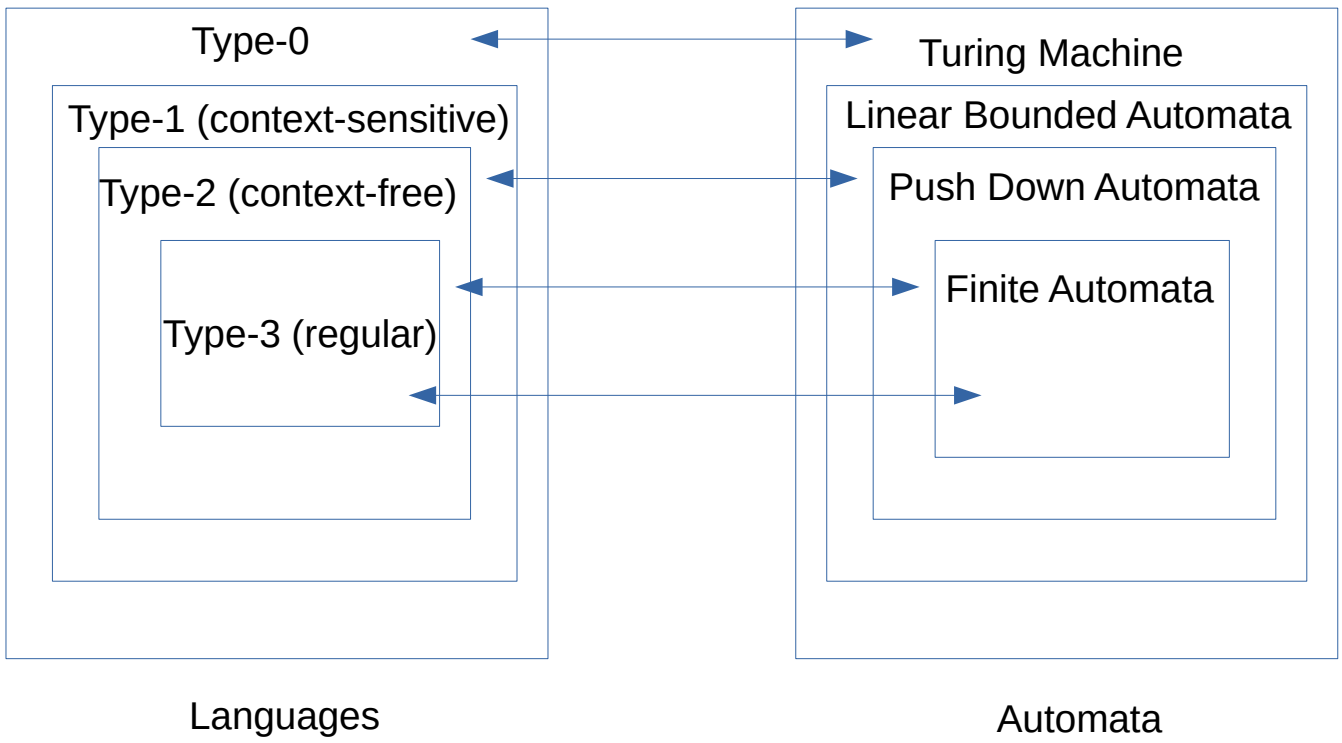
- Gramatyka typu-0: Każda gramatyka, czyli gramatyka bez żadnych ograniczeń, jest gramatykę typu-0.
- Gramatyka typu-1: Gramatyka nazywana jest typem-1 lub wrażliwą na kontekst (context-sensitive) lub kontekstową zależny (context-independent), jeżeli wszystkie jego produkcje są produkcjami typu-1. Produkcja $S \rightarrow \Lambda$ jest również dozwolone w gramatyce typu-1, ale w tym przypadku S nie pojawia się w prawiej stronie dowolnej produkcji.
- Gramatyka typu-2: Gramatykę nazywa się gramatykę typu-2, jeśli zawiera tylko produkcje typu-2. Nazywa się ją także gramatyką bezkontekstową (context-free) (bo może A być zastąpione przez α w dowolnym kontekście).
- Gramatyka typu-3: Gramatykę nazywa się gramatykę typu-3 lub gramatyką regularną (regular grammar), jeśli zawiera tylko produkcje typu-3. Produkcja $S \rightarrow \Lambda$ jest również dozwolone w gramatyce typu-3, ale w tym przypadku S nie pojawia się w prawiej stronie dowolnej produkcji.

Przykłady

- $abAbcd \rightarrow abABbcd$: Tutaj po lewej stronie produkcji zmienna A ma lewy kontekst ab i prawy kontekst bcd.
- To jest produkcja typu-1, w której A zostaje zastąpione przez AB w obecności lewego kontekstu ab i prawego kontekstu bcd.
- Warto uważyć, że w gramatyce wrażliwej na kontekst G pozwalamy $S \rightarrow \Lambda$ na włączenie Λ w $L(G)$. Oprócz $S \rightarrow \Lambda$ wszystkie inne produkcje nie zmniejszają długość ciągu po lewej stronie.
- $S \rightarrow Aa$, $A \rightarrow aB$, $A \rightarrow a$, $B \rightarrow abc$, $A \rightarrow \Lambda$ gdzie $S, A, B \in V_N$ i $a, b, c \in \Sigma$ są to produkcje typu-2.
- Twierzenie: Każda produkcja typu-3 jest produkcją typu 2. (Widzimy, że $A \rightarrow aB$, $A \rightarrow a$ są to również produkcje typu-2)
- Niech L_0, L_{cs}, L_{cf} i L_r będą językami odpowiednio generowanymi przez typ-0, typ-1, typ-2 i typu 3 gramatyki.
- Twierdzenie: (i) $L_{cs} \subseteq L_0$, (ii) $L_{cf} \subseteq L_0$ i (iii) $L_r \subseteq L_{cf}$.
- $L_{cf} \subseteq L_{cs}$ w specjalnym przypadku. Zwykle w gramatyce typu-1 nie dopuszczamy $A \rightarrow \Lambda$ dla $A \neq S$, a gramatyka typu-2 pozwala na taką produkcję, dlatego $L_{cf} \subseteq L_{cs}$ wymaga pewnej konwersji.

Relacja między językami i automatami

- Pomiędzy tymi językami a różnymi rodzajami automatów istnieje bezpośrednie połączenie.
- Np. Każdy język regularny może zostać zaakceptowany przez automat skończony i odwrotnie, każdy język wygenerowany przez automat skończony jest językiem regularnym. Podobna jest relacja między innymi językami i odpowiadającymi im automatami.



Język rekurencyjny

- Procedure: Procedura rozwiązania problemu to skończona sekwencja instrukcji które można przeprowadzić mechanicznie przy dowolnym wejściu.
- Algorithm: Algorytm to procedura, która kończy się po skończonej liczbie kroków dla dowolnego wejścia.
- Recursive set: Zbiór X jest rekurencyjny, jeśli mamy algorytm do ustalenia czy dany element należy do X czy nie.
- Recursively enumerable set: Zbiór rekurencyjnie przeliczalny to zbiór X, dla którego mamy procedura sprawdzająca, czy dany element należy do X, czy nie.
- Każdy zbiór rekurencyjny jest rekurencyjnie przeliczalny.
- Język jest rekurencyjny, to oznacza, że jest jakiś algorytm sprawdzający, czy dany ciąg należy do tego języka, czy nie.

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

  Krok-2: Dla $|w| = n \geq 1$, algorytm opiera się na konstrukcji łańcucha podzbiorów $\{W_i\}$ z $(V_N \cup \Sigma)^*$. W_i jest po prostu zbiorem wszystkich form zdaniowych długość mniejsza lub równa n, które można wyprowadzić w co najwyżej i krokach. Konstrukcja odbywa się rekurencyjnie w następujący sposób.

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

  Krok-2: Dla $|w| = n \geq 1$, algorytm opiera się na konstrukcji łańcucha podzbiorów $\{W_i\}$ z $(V_N \cup \Sigma)^*$. W_i jest po prostu zbiorem wszystkich form zdaniowych długość mniejsza lub równa n, które można wyprowadzić w co najwyżej i krokach. Konstrukcja odbywa się rekurencyjnie w następujący sposób.

(i) $W_0 = \{S\}$.

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

  Krok-2: Dla $|w| = n \geq 1$, algorytm opiera się na konstrukcji łańcucha podzbiorów $\{W_i\}$ z $(V_N \cup \Sigma)^*$. W_i jest po prostu zbiorem wszystkich form zdaniowych długość mniejsza lub równa n, które można wyprowadzić w co najwyżej i krokach. Konstrukcja odbywa się rekurencyjnie w następujący sposób.

(i) $W_0 = \{S\}$.

(ii) $W_{i+1} = W_i \cup \{\beta \in (V_N \cup \Sigma)^* : \textit{istnieje } \alpha \in W_i,\ \alpha \Rightarrow \beta \textit{ i } |\beta| \leq n\}$.

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

  Krok-2: Dla $|w| = n \geq 1$, algorytm opiera się na konstrukcji łańcucha podzbiorów $\{W_i\}$ z $(V_N \cup \Sigma)^*$. W_i jest po prostu zbiorem wszystkich form zdaniowych długość mniejsza lub równa n, które można wyprowadzić w co najwyżej i krokach. Konstrukcja odbywa się rekurencyjnie w następujący sposób.

(i) $W_0 = \{S\}$.

(ii) $W_{i+1} = W_i \cup \{\beta \in (V_N \cup \Sigma)^* : \textit{istnieje } \alpha \in W_i,\ \alpha \Rightarrow \beta \textit{ i } |\beta| \leq n\}$.

Krok-3: Ta konstrukcja to gwarantuje, że $W_i \subseteq W_{i+1}$ dla dowolnego skończonego i i na pewnym etapie $i = k$, $W_k = W_{k+1}$

Język wrażliwy na kontekst jest rekurencyjny

Niech $G = (V_N, \Sigma, P, S)$ będzie gramatyką typu-1 i $w \in \Sigma^*$.

- Algorytm sprawdzający, czy dany ciąg należy do języka typu-1:

  Krok-1: Jeśli w $= \Lambda$, to $w \in L(G)$ wtedy i tylko wtedy, gdy $S \to \Lambda \in P$. Ponieważ w P jest tylko skończona liczba produkcji, musimy przetestować czy $S \to \Lambda$ jest w P czy nie.

  Krok-2: Dla $|w| = n \geq 1$, algorytm opiera się na konstrukcji łańcucha podzbiorów $\{W_i\}$ z $(V_N \cup \Sigma)^*$. W_i jest po prostu zbiorem wszystkich form zdaniowych długość mniejsza lub równa n, które można wyprowadzić w co najwyżej i krokach. Konstrukcja odbywa się rekurencyjnie w następujący sposób.

(i) $W_0 = \{S\}$.

(ii) $W_{i+1} = W_i \cup \{\beta \in (V_N \cup \Sigma)^* : \textit{istnieje } \alpha \in W_i,\ \alpha \Rightarrow \beta \textit{ i } |\beta| \leq n\}$.

Krok-3: Ta konstrukcja to gwarantuje, że $W_i \subseteq W_{i+1}$ dla dowolnego skończonego i i na pewnym etapie $i = k$, $W_k = W_{k+1}$

Krok-4: Dla najmniejszego takiego k spełniającego $W_k = W_{k+1}$ wystarczy sprawdzić, czy $w \in W_k$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \textit{ gdzie } \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

  $00112 \notin W_1$ - oznacza to, że $00112 \notin L(G)$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

  $00112 \notin W_1$ - oznacza to, że $00112 \notin L(G)$.

(ii) Ponieważ $|001122| = 6$, mamy W_0, W_1 tak sam jak powyżej, ale $W_2 = \{S, 012, 0\underline{S}A2, 0012A2\}$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

  $00112 \notin W_1$ - oznacza to, że $00112 \notin L(G)$.

(ii) Ponieważ $|001122| = 6$, mamy W_0, W_1 tak sam jak powyżej, ale $W_2 = \{S, 012, 0\underline{S}A2, 0012A2\}$.

  $W_3 = \{S, 012, 0SA2, 001\underline{2A}2, 001A22\}$

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

  $00112 \notin W_1$ - oznacza to, że $00112 \notin L(G)$.

(ii) Ponieważ $|001122| = 6$, mamy W_0, W_1 tak sam jak powyżej, ale $W_2 = \{S, 012, 0\underline{S}A2, 0012A2\}$.

  $W_3 = \{S, 012, 0SA2, 001\underline{2A}2, 001A22\}$

  $W_4 = \{S, 012, 0SA2, 0012A2, 00\underline{1A}22, 001122\}$ i $W_5 = W_4$.

Przykład

- Rozważ następującą gramatykę:
  $G = (\{S, A\}, \{0, 1, 2\}, \{S \to 0SA2|012, 2A \to A2, 1A \to 11\}, S)$.
  Sprawdź czy (i) $00112 \in L(G)$ (ii) $001122 \in L(G)$.

  Rozwiązanie: (i) Tutaj $w = 00112$ i $|w| = 5$.

  $W_0 = \{S\}$

  $W_1 = W_0 \cup \{\beta : \alpha \Rightarrow \beta \ gdzie \ \alpha \in W_0, |\beta| \leq 5\} = \{S, 012, 0SA2\}$.

  $W_2 = \{S, 012, 0\underline{S}A2\}$. Więc $W_1 = W_2$.
  (Tutaj mamy tylko jedną możliwość wyprowadzenia nowego ciągu bo z $0SA2 \in W_1$ możemy mieć $0SA2 \Rightarrow 0012A2$, ale $|0012A2| > 5$).

  $00112 \notin W_1$ - oznacza to, że $00112 \notin L(G)$.

(ii) Ponieważ $|001122| = 6$, mamy W_0, W_1 tak sam jak powyżej, ale $W_2 = \{S, 012, 0\underline{S}A2, 0012A2\}$.

  $W_3 = \{S, 012, 0SA2, 001\underline{2A}2, 001A22\}$

  $W_4 = \{S, 012, 0SA2, 0012A2, 00\underline{1A}22, 001122\}$ i $W_5 = W_4$.

  $001122 \in W_4$ i tak więc $001122 \in L(G)$.

Operacje na językach

- Jako zbiory, języki podlegają operacjom i powstają w ten sposób nowe języki.
- Concatenation: Niech A i B będą dowolnymi dwoma zbiorami ciągów. Konkatenacja AB elementów A i B wynosi zdefiniowany przez $AB = \{uv : u \in A, v \in B\}$ (uv to konkatenacja ciągi u i v)
- Definiujemy $A^1 = A$ i $A^{n+1} = A^n A$.
  $(A^n = \{u_1 u_2 \dots u_n : u_1, \dots, u_n \in A\})$.
- Transpose: Zbiór transpozycji A jest zdefiniowany przez $A^T = \{u^T : u \in A\}$ gdzie u^T jest ciągiem otrzymanym z u poprzez odwrócenie kolejności jego symboli.
- Twierdzenie: Jeśli L_1 i L_2 to dwa języki typu-i (i = 0, 1, 2, 3), to konkatenacja L_1 i L_2, oznaczony jako $L_1 L_2$, jest także językiem typu-i. Oznacza to, że możemy znaleźć gramatykę G typu-i taką, że $L(G) = L_1 L_2$.
- Twierdzenie: Jeśli L to język typu-i (i = 0, 1, 2, 3), to transpozycja L, oznaczony jako L^T, jest także językiem typu-i. Oznacza to, że możemy znaleźć gramatykę G typu-i taką, że $L(G) = L^T$.

Contd.

- Twierdzenie: Jeśli L_1 i L_2 to dwa języki typu-i (i = 0, 1, 2, 3), to suma L_1 i L_2, oznaczony jako $L_1 \cup L_2$, jest także językiem typu-i. Oznacza to, że możemy znaleźć gramatykę G typu-i taką, że $L(G) = L_1 \cup L_2$.
- Twierdzenie: Jeśli L_1 i L_2 to dwa języki typu-i (i = 0, 1, 3), to przecięcie L_1 i L_2, oznaczony jako $L_1 \cap L_2$, jest także językiem typu-i Oznacza to, że możemy znaleźć gramatykę G typu-i taką, że $L(G) = L_1 \cap L_2$.
- Warto uważyć, że przecięcie dwóch języków bezkontekstowych (context-free) może nie być bezkontekstowe.

Konstruowanie gramatyki dla wymaganego języka

- Skonstruuj gramatykę G akceptującą zbiór L wszystkich ciągów ponad $\{a, b\}$ mający więcej a niż b.

Konstruowanie gramatyki dla wymaganego języka

- Skonstruuj gramatykę G akceptującą zbiór L wszystkich ciągów ponad $\{a, b\}$ mający więcej a niż b.
- Do generowania ciągów zawierających a, ale bez b, możemy mieć produkcje $S \to a$, $S \to aS$, $S \to Sa$. (Jeśli $xa \in L$, to $ax \in L$. Musimy więc mieć oboje $S \to aS$, $S \to Sa$.)

Konstruowanie gramatyki dla wymaganego języka

- Skonstruuj gramatykę G akceptującą zbiór L wszystkich ciągów ponad $\{a, b\}$ mający więcej a niż b.
- Do generowania ciągów zawierających a, ale bez b, możemy mieć produkcje $S \to a$, $S \to aS$, $S \to Sa$. (Jeśli $xa \in L$, to $ax \in L$. Musimy więc mieć oboje $S \to aS$, $S \to Sa$.)
- Jeśli $x, y \in L$, to $xy \in L$ oraz $yx \in L$. Zatem xy lub yx ma co najmniej dwa więcej a niż b. Można tego dokonać poprzez produkcje $S \to bSS$, $S \to SbS$, $S \to SSb$. (b można dodać na początku, lub na końcu SS, lub pomiędzy S i S.
(i) Na przykład, jeśli najpierw zostanie zastosowana jedna z tych produkcji, to aby uzyskać ciąg terminali, należy użyć powyższej produkcji $S \to a$. W ten sposób wyprowadzane są ciągi takie jak baa, aba, aab.)

Konstruowanie gramatyki dla wymaganego języka

- Skonstruuj gramatykę G akceptującą zbiór L wszystkich ciągów ponad $\{a, b\}$ mający więcej a niż b.
- Do generowania ciągów zawierających a, ale bez b, możemy mieć produkcje $S \to a$, $S \to aS$, $S \to Sa$. (Jeśli $xa \in L$, to $ax \in L$. Musimy więc mieć oboje $S \to aS$, $S \to Sa$.)
- Jeśli $x, y \in L$, to $xy \in L$ oraz $yx \in L$. Zatem xy lub yx ma co najmniej dwa więcej a niż b. Można tego dokonać poprzez produkcje $S \to bSS$, $S \to SbS$, $S \to SSb$. (b można dodać na początku, lub na końcu SS, lub pomiędzy S i S.
(i) Na przykład, jeśli najpierw zostanie zastosowana jedna z tych produkcji, to aby uzyskać ciąg terminali, należy użyć powyższej produkcji $S \to a$. W ten sposób wyprowadzane są ciągi takie jak baa, aba, aab.)
- Więc $G = (\{S\}, \{a, b\}, \{S \to a|aS|Sa|bSS|SbS|SSb\}, S)$.

Konstruowanie gramatyki dla wymaganego języka

- Skonstruuj gramatykę G akceptującą zbiór L wszystkich ciągów ponad $\{a, b\}$ mający więcej a niż b.
- Do generowania ciągów zawierających a, ale bez b, możemy mieć produkcje $S \to a$, $S \to aS$, $S \to Sa$. (Jeśli $xa \in L$, to $ax \in L$. Musimy więc mieć oboje $S \to aS$, $S \to Sa$.)
- Jeśli $x, y \in L$, to $xy \in L$ oraz $yx \in L$. Zatem xy lub yx ma co najmniej dwa więcej a niż b. Można tego dokonać poprzez produkcje $S \to bSS$, $S \to SbS$, $S \to SSb$. (b można dodać na początku, lub na końcu SS, lub pomiędzy S i S.
(i) Na przykład, jeśli najpierw zostanie zastosowana jedna z tych produkcji, to aby uzyskać ciąg terminali, należy użyć powyższej produkcji $S \to a$. W ten sposób wyprowadzane są ciągi takie jak baa, aba, aab.)
- Więc $G = (\{S\}, \{a, b\}, \{S \to a|aS|Sa|bSS|SbS|SSb\}, S)$.
- Sprawdź, G akceptuje wszystkie ciągi nad $\{a, b\}$, mając więcej a niż b.