Języki, Automaty i Obliczenia. Odcinek 4

Automaty ze stosem. Własności języków bezkontekstowych

Damian Niwiński

Listopad 2003

1 Automaty ze stosem

1.1 Podstawowe definicje

Automat ze stosem może być przedstawiony jako następujący układ:

$$A = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, F)$$

gdzie

- Σ jest *alfabetem symboli wejściowych*,
- Γ jest *alfabetem symboli stosowych*,
- Q jest zbiorem stanów,
- $q_I \in Q$ jest stanem początkowym,
- $Z_I \in \Gamma$ jest *symbolem startowym*,
- $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \times Q \times \Gamma^*$ jest relacją przejścia,
- $F \subseteq Q$ jest zbiorem stanów akceptujących.

Wszystkie występujące tu zbiory są skończone.

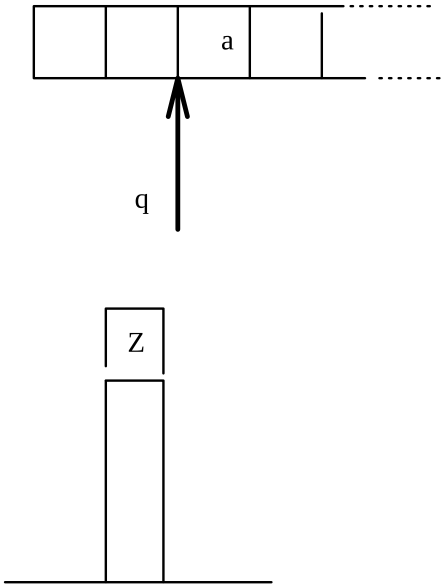
Przejście postaci (q, a, Z, q', γ) będziemy także zapisywać

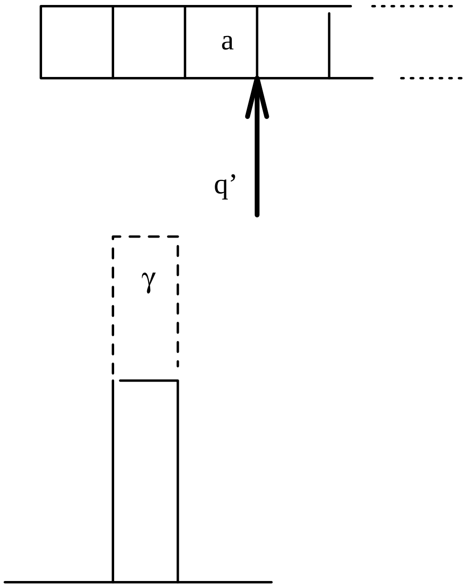
$$q, a, Z \to_A q', \gamma$$

Należy je rozumieć następująco.

Gdy $a \in \Sigma$:

Jeżeli automat znajduje się w stanie q, głowica automatu czyta na taśmie symbol a, a na wierzchołku stosu znajduje się symbol Z, to w następnej chwili czasu automat może zmienić stan na q', zastąpić symbol Z na szczycie stosu przez słowo γ i przesunąć głowicę o jedno miejsce w prawo.





Gdy $a = \epsilon$:

Jeżeli automat znajduje się w stanie q, a na wierzchołku stosu jest symbol Z, to w następnej chwili automat może zmienić stan na q' i zastąpić symbol Z na stosie przez słowo γ. Pozycja głowicy pozostaje przy tym nie zmieniona.

Zauważmy, że jeśli w rozważanym powyżej przejściu $\gamma = Z'Z$, to operację na stosie można określić jako

połóż Z' na stos,

a jeśli $\gamma = \epsilon$, to jako

zdejmij symbol z wierzchołka stosu.[1]

Konfigurację automatu w danej chwili czasu możemy przedstawić jako trójkę (q, w, γ), gdzie $q \in Q$, $w \in \Sigma^*$, $\gamma \in \Gamma^*$. Należy przez to rozumieć sytuację, kiedy automat znajduje się w stanie q, w jest słowem, jakie pozostało jeszcze do przeczytania, a γ jest zawartością stosu, przy czym przyjmujemy, że pierwsza litera słowa γ odpowiada symbolowi na wierzchołku stosu itd.

Za konfigurację *początkową* uważamy każdą konfigurację postaci

$$(q_I, w, Z_I)$$

[1]Można wykazać, że automaty ze stosem używające jedynie tych dwóch operacji mają nie mniejszą siłę obliczeniową niż zdefiniowane przez nas bardziej ogólne automaty.

natomiast konfiguracją *końcową* nazwiemy dowolną konfigurację postaci (q, ϵ, γ) (tzn. kiedy całe słowo zostało już przeczytane). Dalej, określamy relację na konfiguracjach

$$(q, w, \gamma) \vdash_A (q', w', \gamma')$$

odzwierciedlającą fakt, że konfiguracja (q', w', γ') może być osiągnięta z konfiguracji (q, w, γ) w jednym kroku przez wykonanie jakiegoś przejścia automatu A.

Mianowicie, relacja

$$(q, aw, Z\beta) \vdash_A (q', w, \alpha\beta)$$

zachodzi, o ile

$$q, a, Z \to_A q', \alpha$$

jest przejściem automatu A. Zauważmy, że trójka (q, w, ϵ) (pusty stos) jest dobrze określoną konfiguracją, z której jednak nie ma już przejścia do żadnej konfiguracji.

Jak zwykle, $\vdash_A^*$ oznacza zwrotno-przechodnie domknięcie relacji $\vdash_A$.

Niech teraz $w \in \Sigma^*$. Ciąg konfiguracji (q_0, w_0, γ_0), (q_1, w_1, γ_1), $\ldots$, (q_m, w_m, γ_m), gdzie (q_0, w_0, γ_0) jest konfiguracją początkową z $w_0 = w$ oraz, dla $i < m$, $(q_i, w_i, \gamma_i) \vdash_A (q_{i+1}, w_{i+1}, \gamma_{i+1})$, nazywamy *obliczeniem* automatu A na słowie w. Obliczenie nazwiemy *akceptującym* jeśli (q_m, w_m, γ_m) jest konfiguracją końcową (tj. $w_m = \epsilon$), a ponadto $q_m \in F$. Język rozpoznawany przez automat ze stosem A określamy jako zbiór tych słów, dla których istnieje obliczenie akceptujące; tak więc

$$L(A) = \{w \in \Sigma^* \, : \, (q_I, w, Z_I) \vdash_A^* (q_f, \epsilon, \gamma) \text{ dla pewnych } q_f \in F,\ \gamma \in \Gamma^*\}$$

1.2 Automaty akceptujące przez opróżnienie stosu

Dla automatu ze stosem A, określimy również język $Z(A)$ obejmujący te i tylko te słowa w, dla których istnieje obliczenie zakończone całkowitym opróżnieniem stosu,

$$Z(A) = \{w \in \Sigma^* \, : \, (q_I, w, Z_I) \vdash_A^* (q, \epsilon, \epsilon) \text{ dla pewnego } q \in Q\}$$

W ten sposób przyporządkowujemy automatowi język na ogół różny od $L(A)$, przy czym stany akceptujące nie odgrywają tutaj żadnej roli. Okazuje się jednak, że nowy sposób akceptacji jest w pewnym sensie równoważny poprzedniemu.

Fakt 1

1. *Dla każdego automatu ze stosem A, istnieje automat ze stosem A', taki że $L(A) = Z(A')$.*

2. *Dla każdego automatu ze stosem B, istnieje automat ze stosem B', taki że $Z(B) = L(B')$.*

Dowód. Ad (1). Niech $A = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, F)$. Określamy automat

$$A' = (\Sigma, \Gamma \cup \{Z_I'\}, Q \cup \{q_I', q_e\}, q_I', Z_I', \delta', \emptyset)$$

gdzie $\delta' \supseteq \delta$ jest rozszerzeniem δ o następujące przejścia

$$\begin{array}{rcll} q_I', \epsilon, Z_I' & \to & q_I, Z_I Z_I' & \\ q_f, \epsilon, Z & \to & q_e, Z & \text{dla } q_f \in F,\ Z \in \Gamma \cup \{Z_I'\} \\ q_e, \epsilon, Z & \to & q_e, \epsilon & \text{dla } Z \in \Gamma \cup \{Z_I'\} \end{array}$$

A zatem, automat A' rozpoczyna działanie od położenia "starego" symbolu startowego Z_I na "nowy" symbol Z'_I. Następnie automat A' symuluje działanie automatu A, z tym, że kiedy A wchodzi w stan akceptujący, A' może niedeterministycznie przyjąć opcję (całkowitego) opróżniania stosu. Zanim to nastąpi, nowy symbol startowy pozostaje cały czas na dnie stosu, żeby uniknąć przypadkowej akceptcji przez A' (w sensie opróżnienia stosu) w sytuacji kiedy A opróżnia stos, ale nie wchodzi w stan akceptujący.

Szczegóły dowodu, że $L(A) = Z(A')$ pozostawiamy Czytelnikowi.

Ad (2). Niech $B = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, \emptyset)$. Określamy automat

$$B' = (\Sigma, \Gamma \cup \{Z'_I\}, Q \cup \{q'_I, q_f\}, q'_I, Z'_I, \delta', \{q_f\})$$

gdzie $\delta' \supseteq \delta$ jest rozszerzeniem δ o następujące przejścia

$$\begin{array}{rcll} q'_I, \epsilon, Z'_I & \to & q_I, Z_I Z'_I & \\ q, \epsilon, Z'_I & \to & q_f, \epsilon & \qquad \text{dla } q \in Q \end{array}$$

Intuicyjnie, automat B' symuluje działanie automatu B, z tym że na spodzie stosu pozostaje cały czas nowy symbol startowy Z'_I, który poza tym nie jest używany. Jeśli w pewnej chwili obliczenia symbol ten znajdzie się na wierzchołku stosu, oznacza to, że automat B, w odnośnej chwili symulowanego obliczenia opróżniłby swój stos. Dlatego automat B' wchodzi wówczas w stan akceptujący. Pamiętajmy jednak, że do tego, by obliczenie było akceptujące nie wystarczy samo pojawienie się stanu akceptującego, potrzeba jeszcze, by konfiguracja była końcowa (tzn. by całe słowo zostało przeczytane). $\Box$

2 Automaty ze stosem jako modele rozpoznawania języków bezkontekstowych

W tym rozdziale wykażemy ścisłą odpowiedniość pomiędzy automatami ze stosem a gramatykami bezkontekstowymi : automaty rozpoznają dokładnie te języki, które gramatyki generują. Zanim pokażemy właściwą konstrukcję, która automatowi przyporządkowuje równoważną gramatykę i na odwrót, wykażemy pewną nieco zaskakującą a bardzo przydatną własność automatów ze stosem.

2.1 Redukcja stanów w automacie ze stosem

Udowodnimy, że, w automacie ze stosem, symbole stosowe mogą w pewnym sensie całkowicie przejąć rolę stanów.

Twierdzenie 1 *Dla każdego automatu ze stosem A istnieje automat A'* z jednym stanem*, taki że $Z(A) = Z(A')$.*

Dowód. Niech $A = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, F)$. Określimy automat

$$A' = (\Sigma, \Gamma', \{e\}, e, Z'_I, \delta', \emptyset)$$

Nowym alfabetem symboli stosowych jest

$$\Gamma' = \{[q, Z, q'] \,:\, q, q' \in Q,\ Z \in \Gamma\} \cup \{Z'_I\}$$

Zbiór przejść δ' określony jest następująco:

- dla dowolnego $q \in Q$,
$$e, \epsilon, Z'_I \ \to_{A'}\ e, [q_I, Z_I, q]$$
jest przejściem automatu A';
- o ile $q, a, Z \to_A p, Z_1 Z_2 \dots Z_m$, $m \geq 1$, jest przejściem automatu A, a $p_1, p_2, \dots, p_{m-1}, q' \in Q$ są dowolne, to
$$e, a, [q, Z, q'] \ \to_{A'}\ e, [p, Z_1, p_1][p_1, Z_2, p_2] \dots [p_{m-2}, Z_{m-1}, p_{m-1}][p_{m-1}, Z_m, q']$$
jest przejściem automatu A';
- o ile $q, a, Z \to_A q', \epsilon$ jest przejściem automatu A, to
$$e, a, [q, Z, q'] \ \to_{A'}\ e, \epsilon$$
jest przejściem automatu A'.

Wykażemy, że dla dowolnego $w = uv$, następujące warunki są równoważne:

Warunek 1

$$q_I, w, Z_I \vdash_A^* q, v, Z_1 \dots Z_m$$

Warunek 2 Dla dowolnych $p_1, \dots, p_m \in Q$,

$$e, w, Z'_I \vdash_{A'}^* e, v, [q, Z_1, p_1][p_1, Z_2, p_2] \dots [p_{m-2}, Z_{m-1}, p_{m-1}][p_{m-1}, Z_m, p_m]$$

Zauważmy od razu, że dla $v = \epsilon$ i $m = 0$, z równoważności Warunków 1 i 2 otrzymamy, że

$$q_I, w, Z_I \vdash_A^* q, \epsilon, \epsilon \iff e, w, Z'_I \vdash_{A'}^* e, \epsilon, \epsilon$$

a zatem

$$w \in Z(A) \iff w \in Z(A')$$

co zakończy dowód.

Pozostaje więc dowieść (1)$\Leftrightarrow$ (2)

(1)$\Rightarrow$ (2) Przez indukcję ze względu na długość obliczenia automatu A prowadzącego od konfiguracji (q_I, w, Z_I) do $(q, v, Z_1 \dots Z_m)$, wykażemy że

$$e, w, Z'_I \vdash_{A'}^* e, v, [q, Z_1, p_1][p_1, Z_2, p_2] \dots [p_{m-2}, Z_{m-1}, p_{m-1}][p_{m-1}, Z_m, p_m]$$

przy dowolnych $p_1, \dots, p_m$.

Dla obliczenia o długości 0

$$q_I, w, Z_I \vdash_A^* q_I, w, Z_I$$

mamy, dla dowolnego $p \in Q$, obliczenie automatu A' o długości 1, postaci

$$e, w, Z'_I \vdash_{A'} e, w, [q_I, Z_I, p]$$

Z kolei rozważmy obliczenie automatu A o długości > 0. Mamy zatem

$$\begin{array}{lll} q_I, w, Z_I & \vdash_A^* & q', av, Y_1 Y_2 \ldots Y_m \\ & \vdash_A & q, v, Z_1 \ldots Z_k Y_2 \ldots Y_m \end{array}$$

dla pewnych $a \in \Sigma \cup \{\epsilon\}$ oraz $Y_1, \ldots, Y_m$, $Z_1, \ldots, Z_k$, gdzie

$$q', a, Y_1 \ \rightarrow_A \ q, Z_1 \ldots Z_k$$

jest przejściem automatu A.

Niech $q_1, \ldots, q_k, p_2, \ldots, p_m \in Q$, dowolne. Z założenia indukcyjnego, dla dowolnego p_1, mamy

$$(*) \ e, w, Z'_I \vdash_{A'}^* e, av, [q', Y_1, p_1][p_1, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m]$$

Rozważymy teraz dwa przypadki, w zależności od tego czy $k \neq 0$.

1. Gdy $k > 0$, mamy, z definicji A', przejście

$$e, a, [q', Y_1, q_k] \ \rightarrow_{A'} \ e, [q, Z_1, q_1][q_1, Z_2, q_2] \ldots [q_{k-1}, Z_k, q_k]$$

Następnie, kładąc w powyższej zależności $(*)$, $p_1 = q_k$, mamy

$$e, w, Z'_I \vdash_{A'}^* e, av, [q', Y_1, q_k][q_k, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m]$$

Stąd,

$$\begin{array}{lll} e, w, Z'_I & \vdash_{A'}^* & e, av, [q', Y_1, q_k][q_k, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \\ & \vdash_{A'} & e, v, [q, Z_1, q_1][q_1, Z_2, q_2] \ldots [q_{k-1}, Z_k, q_k][q_k, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \end{array}$$

2. Gdy $k = 0$, mamy, z definicji A',

$$e, a, [q', Y_1, q] \ \rightarrow_{A'} \ e, \epsilon$$

Kładąc w zależności $(*)$, $p_1 = q$, mamy

$$e, w, Z'_I \vdash_{A'}^* e, av, [q', Y_1, q][q, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m]$$

Stąd

$$\begin{array}{lll} e, w, Z'_I & \vdash_{A'}^* & e, av, [q', Y_1, q][q, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \\ & \vdash_{A'} & e, v, [q, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \end{array}$$

co kończy dowód implikacji (1)⇒ (2).

(2)⇒ (1) Wykażemy implikację silniejszą, a mianowicie, że z faktu iż

$$e, w, Z_I' \vdash_{A'}^* e, v, [q, Z_1, p_1][p_1, Z_2, p_2] \ldots [p_{m-2}, Z_{m-1}, p_{m-1}][p_{m-1}, Z_m, p_m]$$

dla *pewnych* $p_1, \ldots, p_m$, wynika

$$q_I, w, Z_I \vdash_A^* q, v, Z_1 \ldots Z_m$$

Podobnie jak poprzednio użyjemy indukcji ze względu na długość obliczenia, ale tym razem chodzić będzie o obliczenie automatu A' prowadzące od konfiguracji e, w, Z_I' do $e, v, [q, Z_1, p_1][p_1, Z_2, p_2] \ldots [p_{m-2}, Z_{m-1}, p_{m-1}][p_{m-1}, Z_m, p_m]$. Zgodnie z definicją automatu A', najkrótsze takie obliczenie ma oczywiście długość 1 i jest postaci

$$e, w, Z_I' \vdash_{A'} e, w, [q_I, Z_I, q]$$

Odpowiednie obliczenie automatu A jest obliczeniem trywialnym

$$q_I, w, Z_I \vdash_A^* q_I, w, Z_I$$

Z kolei rozważmy możliwe obliczenie postaci

$$\begin{aligned} e, w, Z_I' & \vdash_{A'}^* & e, av, [q', Y_1, p_1][p_1, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \\ & \vdash_{A'} & e, v, [q, Z_1, q_1][q_1, Z_2, q_2] \ldots [q_{k-1}, Z_k, p_1][p_1, Y_2, p_2] \ldots [p_{m-2}, Y_{m-1}, p_{m-1}][p_{m-1}, Y_m, p_m] \end{aligned}$$

(Pisząc w ten sposób dopuszczamy $k = 0$, czyli symbole Z_i są nieobecne i $p_1 = q$.) Ostatni krok tego obliczenia wykorzystuje przejście automatu A',

$$e, a, [q', Y_1, p_1] \to_{A'} e, [q, Z_1, q_1][q_1, Z_2, q_2] \ldots [q_{k-1}, Z_k, p_1]$$

gdzie

$$q', a, Y_1 \to_A q, Z_1 \ldots Z_k$$

jest przejściem automatu A. (Poniższy argument działa zarówno dla $k > 0$ jak i dla $k = 0$, gdy ciąg $Z_1 \ldots Z_k$ jest pusty.)

Z założenia indukcyjnego, mamy

$$q_I, w, Z_I \vdash_A^* q', av, Y_1 \ldots Y_m$$

Stąd

$$\begin{aligned} q_I, w, Z_I & \vdash_A^* & q', av, Y_1 \ldots Y_m \\ & \vdash_A & q, v, Z_1 \ldots Z_k Y_2 \ldots Y_m \end{aligned}$$

Ta uwaga kończy dowód twierdzenia. □

2.2 Równoważność gramatyk bezkontekstowych i automatów ze stosem

Twierdzenie 2 *Dla każdego automatu ze stosem A, można skonstruować gramatykę bezkontekstową G, taką że $L(A) = L(G)$.*

Dowód. Niech $L = L(A)$. Na podstawie Faktu 1 i Twierdzenia 2.1, możemy znaleźć automat z jednym stanem, A', taki że $L = Z(A')$, powiedzmy

$$A' = (\Sigma, \Gamma, \{e\}, e, Z_I, \delta, \emptyset)$$

Określamy gramatykę G w ten sposób, że

$$G = (\Sigma, \Gamma, Z_I, R)$$

gdzie zbiór reguł R jest zdefiniowany następująco:

$$Z \stackrel{G}{\to} aY_1 \dots Y_m \;\Leftrightarrow\; e, a, Z \to_{A'} e, Y_1 \dots Y_m$$

gdzie $Z, Y_1, \dots, Y_m \in \Gamma$, $a \in \Sigma \cup \{\epsilon\}$.

Pozostaje dowieść

$$Z(A') = L(G)$$

"$\subseteq$" Udowodnimy następujący fakt.

(*) Przypuśćmy, że $w = uv$ i $e, w, Z_I \vdash^*_{A'} e, v, \gamma$. Wówczas $Z_I \stackrel{G*}{\to} u\gamma$.

Dowód przeprowadzimy przez indukcję ze względu długość obliczenia częściowego prowadzącego od konfiguracji (e, w, Z_I) do (e, v, γ). Jeśli to jest obliczenie długości 0, to teza jest oczywista. Jeśli obliczenie jest dłuższe niż 0, to mamy

$$\begin{aligned} e, w, Z_I \;&\vdash^*_{A'}\; e, av, Y_1 \dots Y_m \\ &\vdash_A\; e, v, Z_1 \dots Z_k Y_2 \dots Y_m \end{aligned}$$

gdzie $a \in \Sigma \cup \{\epsilon\}$, $u = u'a$, dla pewnego u', $w = u'av$ i $e, a, Y_1 \to_{A'} e, Z_1 \dots Z_k$ jest przejściem automatu A'. Wówczas $Y_1 \stackrel{G}{\to} aZ_1 \dots Z_k$ jest regułą gramatyki G. Korzystając z założenia indukcyjnego mamy więc

$$Z_I \stackrel{G*}{\to} u'Y_1 \dots Y_m \stackrel{G}{\to} u'aZ_1 \dots Z_k Y_2 \dots Y_m$$

co kończy dowód (*). Biorąc pod uwagę przypadek gdy $v = \gamma = \epsilon$, otrzymujemy, że $e, w, Z_I \vdash^*_{A'} e, \epsilon, \epsilon$ implikuje $Z_I \stackrel{G*}{\to} w$.

"$\supseteq$" Udowodnimy następujący fakt.

(**) Przypuśćmy, że $w = uv \in \Sigma^*$ i $Z_I \stackrel{l-G*}{\to} u\gamma$, $\gamma \in \Gamma^*$. Wówczas $e, w, Z_I \vdash^*_{A'} e, v, \gamma$.

Dowód przeprowadzimy przez indukcję ze względu na długość lewostronnego wyprowadzenia słowa $u\gamma$ ze zmiennej Z_I. Kiedy wyprowadzenie ma długość 0, to $u = \epsilon$, $\gamma = Z_I$ i teza jest oczywista. Kiedy wyprowadzenie jest dłuższa niż 0, to mamy

$$\begin{array}{rcl} Z_I & \stackrel{l-G*}{\rightarrow} & u'Y_1 \dots Y_m \\ & \stackrel{l-G}{\rightarrow} & u'aZ_1 \dots Z_k Y_2 \dots Y_m \end{array}$$

gdzie, jak poprzednio, $a \in \Sigma \cup \{\epsilon\}$, $u = u'a$, $w = u'av$ i $Y_1 \stackrel{G}{\rightarrow} aZ_1 \dots Z_k$ jest regułą gramatyki G, a zatem, zgodnie z definicją G, $e, a, Y_1 \rightarrow_{A'} e, Z_1 \dots Z_k$ jest przejściem automatu A'. Korzystając z założenia indukcyjnego mamy

$$\begin{array}{rcl} e, w, Z_I & \vdash^*_{A'} & e, av, Y_1 \dots Y_m \\ & \vdash_{A'} & e, v, Z_1 \dots Z_k Y_2 \dots Y_m \end{array}$$

Przypuśćmy teraz, że $Z_I \stackrel{G*}{\rightarrow} w$. Na podstawie Twierdzenia 1 z Odcinka 3, istnieje wyprowadzenie *lewostronne* w z Z_I. Z (**) wynika, że wówczas $e, w, Z_I \vdash^*_{A'} e, \epsilon, \epsilon$, czyli $w \in Z(A')$.

Ta uwaga kończy dowód twierdzenia. □

Twierdzenie 3 *Dla każdej gramatyki bezkontekstowej G, można skonstruować automat ze stosem A, taki że $L(G) = L(A)$.*

Dowód. Niech G będzie gramatyką bezkontekstową. Jeśli $L(G) = \emptyset$, to zadanie jest trywialne, możemy więc założyć, że $L(G) \neq \emptyset$. Na podstawie Faktu 3 i Twierdzenia 2 z Odcinka 3, możemy skonstruować gramatykę w postaci normalnej Chomsky'ego, generującą język $L(G) - \{\epsilon\}$. Zbudujemy automat A', taki że $Z(A') = L(G')$. Na podstawie Faktu 1, można dalej znaleźć automat, powiedzmy A'', taki że $L(A'') = L(G')$. Modyfikacja tego automatu w taki sposób by akceptował również słowo puste, o ile należy ono do $L(G)$, jest łatwą konstrukcją, którą pozostawiamy Czytelnikowi.

Niech

$$G' = (\Sigma, V, X_I, R)$$

Określamy automat

$$A' = (\Sigma, V, \{e\}, e, X_I, \delta, \emptyset)$$

gdzie relacja przejść δ jest określona następująco:

- dla każdej reguły postaci $X \stackrel{G}{\rightarrow} YZ$,

$$e, \epsilon, X \rightarrow_{A'} e, YZ$$

- dla każdej reguły postaci $X \stackrel{G}{\rightarrow} \sigma$, $\sigma \in \Sigma$,

$$e, \sigma, X \rightarrow_{A'} e, \epsilon$$

Pozostaje dowieść

$$Z(A') = L(G')$$

Dowód tej równości opuszczamy, ponieważ jest analogiczny do dowodu podobnej równości w dowodzie Twierdzenia 2. Można powiedzieć więcej: jeżeli do określonego przed chwilą automatu zastosujemy konstrukcję z tamtego dowodu, to otrzymana gramatyka okaże się dokładnie (!) gramatyką G'. A zatem powyższa równość została już *de facto* udowodniona w dowodzie Twierdzenia 2. □

Wniosek 1 *Następujące warunki są równoważne, dla języka $L \subseteq \Sigma^*$.*

1. *L jest generowany przez gramatykę bezkontekstową,*
2. *L jest akceptowany przez niedeterministyczny automat ze stosem.*

Języki generowane przez gramatyki bezkontekstowe nazywamy *językami bezkontekstowymi* (ang. *context-free languages*). Gramatyki bezkontekstowe są metodą *syntezy* języków bezkontekstowych. Poznaliśmy już także narzędzie *analizy* tych języków: niedeterministyczny automat ze stosem. Udowodniliśmy, że obie metody: syntezy i analizy, dokładnie sobie odpowiadają, tzn. automaty ze stosem rozpoznają dokładnie te języki, które gramatyki syntezują.

Klasa języków bezkontekstowych stanowi rozszerzenie klasy języków regularnych i obejmuje wiele naturalnych przykładów języków nie będących regularnymi. Wiele elementów języków programowania, jak np. wyrażenia arytmetyczne, struktura bloków itp., da się opisać językami bezkontekstowymi. Stąd, ogromne znaczenie tych języków w konstrukcji kompilatorów.

Ćwiczenia

1. Skonstruować automaty ze stosem rozpoznające poznane wcześniej języki bezkontekstowe: zbiór palindromów, zbiór poprawnie uformowanych ciągów nawiasów, zbiór słów, które mają dwa razy więcej b niż a, zbiór ciągów, które nie są postaci ww.

2. Dla liczby naturalnej n, niech $\mathrm{bin}(n) \in \{0,1\}^*$ będzie binarnym przedstawieniem liczby n. Skonstruować automat ze stosem rozpoznający język

$$\{\mathrm{bin}(n)\$\mathrm{bin}(n+1)^R \,:\, n \in N\}$$

3. Dowieść, że dla każdego automatu ze stosem A, można skonstruować automat ze stosem *o dwóch stanach* A', taki że $L(A) = L(A')$.

4. Dowieść, że automatowi A' z poprzedniego zadania można postawić dalszy wymóg, że każde przejście jest postaci

$$q, a, Z \to_{A'} q', \alpha$$

gdzie $|\alpha| \leq 2$ (q, q' dowolne).

5. Dowieść, że dla każdego automatu ze stosem A, można skonstruować równoważny mu automat ze stosem A'', w którym każde przejście jest postaci

$$q, a, Z \rightarrow_{A''} q', YZ$$

lub

$$q, a, Z \rightarrow_{A''} q', \epsilon$$

(*) Czy dla tego automatu można nadal ograniczyć liczbę stanów?

6. Mając dany automat ze stosem akceptujący język L, skonstruować automaty ze stosem akceptujące następujące języki:

   - $\text{Prefix}(L) = \{w \,:\, (\exists v) wv \in L\}$
   - $\text{Suffix}(L) = \{w \,:\, (\exists u) uw \in L\}$
   - $\text{Subword}(L) = \{w \,:\, (\exists u, v) uwv \in L\}$
   - (*) $L^R = \{w^R \,:\, w \in L\}$
   - (**) $\text{Cycle}(L) = \{vw \,:\, wv \in L\}$

7. Mając dany automat ze stosem akceptujący język L i automat skończony akceptujący język R, skonstruować automaty ze stosem akceptujące następujące języki:

   - LR^{-1}
   - $R^{-1}L$
   - $L \cap R$

8. Dla danych językow regularnych L i M, skonstruować automat ze stosem rozpoznający język $\bigcup_{i \in N}(L^i \cap M^i)$. Podać przykład, że zbiór ten nie musi być regularny.

9. (*) Dowieść, że dla każdego automatu ze stosem A istnieje stała C (zależna od automatu), taka, że dla każdego słowa $w \in Z(A)$, istnieje obliczenie akceptujące (przez pusty stos) o długości $\leq C|w|$. Wskazówka: oszacować wysokość stosu w obliczeniu akceptującym.

10. (*) Niech A będzie automatem ze stosem. Dowieść, że zbiór słów, które są możliwymi zawartościami stosu automatu A, jest językiem regularnym. Formalnie, mamy na myśli zbiór

$$\{\alpha \in \Gamma^* \,:\, (\exists w, v \in \Sigma^*)(\exists q \in Q)\, q_I, w, Z_I \vdash_A q, v, \alpha\}$$

Wywnioskować stąd, że zbiór słów, które są możliwymi zawartościami stosu automatu A w jakimś obliczeniu *akceptującym*, jest językiem bezkontekstowym.

11. Automat ze stosem $A = (\Sigma, \Gamma, Q, q_I, Z_I, \delta, F)$ nazywamy *deterministycznym*, jeśli w każdej sytuacji możliwy jest co najwyżej jeden ruch. Dokładniej:

   - jeśli, dla pewnej pary q, Z, zachodzi $q, \epsilon, Z \rightarrow_A p, \alpha$ przy pewnych p, α, to dla żadnego $\sigma \in \Sigma$, nie zachodzi $q, \sigma, Z \rightarrow_A p', \alpha'$, dla żadnych p', α';

- dla każdych q, σ, Z, istnieje co najwyżej jedna para p, α, taka że $q, \sigma, Z \to_A p, \alpha$.

Niech L będzie zbiorem palindromów nad alfabetem $\{a, b\}$. Dowieść, że jeśli $L = Z(A)$, to automat A jest niedeterministyczny. (Uwaga: analogiczny fakt można dowieść, gdy $L = L(A)$, ale jest to znacznie trudniejsze.)

12. (**) Niech

$$M = \{a^n b^n \,:\, n \in N\} \cup \{a^n b^{2n} \,:\, n \in N\}$$

Nietrudno jest podać niedeterministyczny automat ze stosem rozpoznający ten język. Wykazać, że *nie* istnieje *deterministyczny*[2] automat ze stosem A, taki, że $M = L(A)$.

3 Lematy o pompowaniu

W tym rozdziale poznamy dwa *lematy o pompowaniu* i ich zastosowanie do dowodzenia, że jakiś język nie jest bezkontekstowy.

Lemat 1 (o pompowaniu) *Niech $L \subseteq \Sigma^*$ będzie językiem bezkontekstowym. Wówczas istnieje stała M zależna tylko od L, taka, że dla każdego słowa α, jeśli $\alpha \in L$ i $|\alpha| \geq M$, to α można przedstawić $\alpha = \alpha_1\gamma_1\beta\gamma_2\alpha_2$ tak, że spełnione są następujące warunki:*

- $\gamma_1\gamma_2 \neq \epsilon$,
- $|\gamma_1\beta\gamma_2| \leq M$,
- *dla każdego $i \geq 0$, $\alpha_1\gamma_1^i\beta\gamma_2^i\alpha_2 \in L$.*

Dowód. Niech G będzie gramatyką bezkontekstową w postaci Chomsky'ego generującą język $L - \{\epsilon\}$, $G = (\Sigma, V, X_I, R)$. Niech $|V| = k$, kładziemy

$$M = 2^k$$

Niech D będzie drzewem wyprowadzenia słowa α o długości $|\alpha| \geq M$. Wówczas głębokość D jest $\geq k + 1$. Istotnie, nietrudno wykazać przez indukcję, że maksymalne słowo, jakie można wygenerować przy pomocy drzewa o głębokości i, ma długość 2^{i-1}. Niech $p_0, p_1, p_2, \ldots, p_\ell$ będzie pewną ścieżką o maksymalnej długości w drzewie D. Mamy oczywiście $\ell \geq k + 1$, przy czym $D(p_\ell) \in \Sigma$, natomiast, dla $m < \ell$, $D(m) \in V$. Muszą zatem istnieć dwa wierzchołki na tej ścieżce, w których pojawia się ta sama zmienna, powiedzmy $D(p_i) = D(p_j) = X$, $i < j$. Wybierzmy "pierwszą od końca" parę o tej własności, tzn. załóżmy, że $(\forall m, n > i) m \neq n \Rightarrow D(p_m) \neq D(p_n)$. Oczywiście, podciąg $p_i, p_{i+1}, \ldots, p_{\ell-1}$ ma wówczas nie więcej niż $k+1$ elementów. Rozważmy *poddrzewo* drzewa D wyznaczone przez wierzchołek p_i, które nazwiemy D', dokładniej (patrz odc. 3, str. 3)

$$\begin{aligned} \mathrm{dom}(D') &= \{u \,:\, p_i u \in \mathrm{dom}(D)\} \\ D'(u) &= D(p_i u)\,, \text{ dla } u \in \mathrm{dom}(D') \end{aligned}$$

[2] Więcej wiadomości o automatach deterministycznych będzie można znaleźć w przygotowywanym następnym odcinku.

Nietrudno sprawdzić, że D' jest również drzewem wyprowadzenia, lecz tym razem ze zmiennej $X = D(p_i)$. Zauważmy, że głębokość tego drzewa jest $\leq k+1$, co wynika z maksymalności ścieżki $p_1, p_2, \ldots, p_\ell$ i uczynionej wyżej uwagi o mocy podciągu $p_i, p_{i+1}, \ldots, p_{\ell-1}$. Niech γ będzie plonem drzewa D', $\gamma = y(D')$. Na mocy zauważonej wcześniej zależności pomiędzy głębokością drzewa wyprowadzenia a długością generowanego słowa,

$$|\gamma| \leq 2^k = M$$

Z kolei niech D'' będzie poddrzewem D wyznaczonym przez wierzchołek p_j, analogicznie do D'. Zauważmy, że jest to również drzewo wyprowadzenia ze zmiennej X. Niech $\beta = y(D'')$.

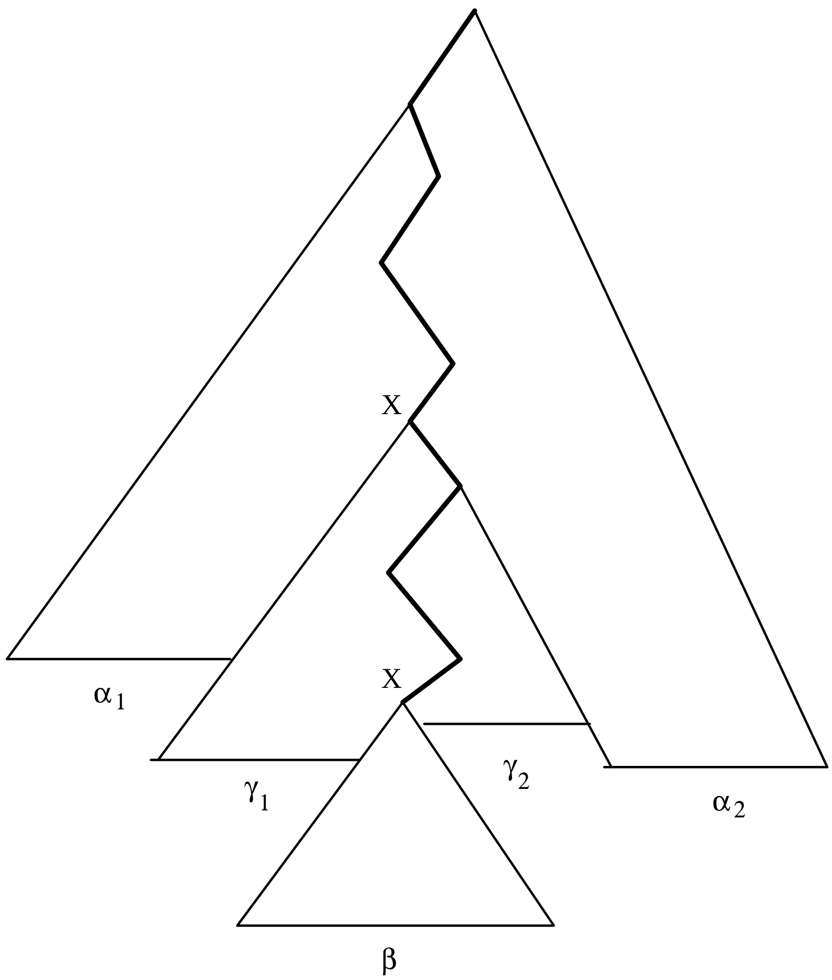
Mamy więc

$$\gamma = \gamma_1 \beta \gamma_2$$

dla pewnych γ_1, γ_2.

Twierdzimy, że

$$\gamma_1 \gamma_2 \neq \epsilon$$



Istotnie, zgodnie z definicją gramatyki Chomsky'ego, wierzchołek p_i ma dwa następniki w drzewie D, p_i1 i p_i2. Jednym z nich jest p_{i+1}, a drugi nie leży na wybranej przez nas ścieżce. Przypuśćmy, że $p_{i+1} = p_i2$. Niech D_1 i D_2 będą poddrzewami drzewa D wyznaczonymi przez wierzchołki p_i1 i p_i2 odpowiednio. Nietrudno zauważyć, że $y(D_1)$

jest prefiksem słowa γ_1. Ponieważ, z definicji postaci gramatyki Chomsky'ego, $y(D_1) \neq \epsilon$, otrzymujemy $\gamma_1 \neq \epsilon$. W przypadku, gdy wierzchołkiem leżącym na ścieżce jest p_i1, argument przebiega podobnie.

Dla zakończenia dowodu pozostaje jeszcze wykazać, że spełniony jest ostatni warunek lematu o pompowaniu. Niech $v \in \{1,2\}^*$ będzie tym słowem, że $p_j = p_iv$. (Intuicyjnie, v wyznacza ścieżkę z wierzchołka p_i do wierzchołka p_j.) Określimy ciąg drzew wyprowadzenia $E_0, E_1, \ldots$ i pomocniczo ciąg słów $u_2, u_3, \ldots \in \{1,2\}^*$, następująco:

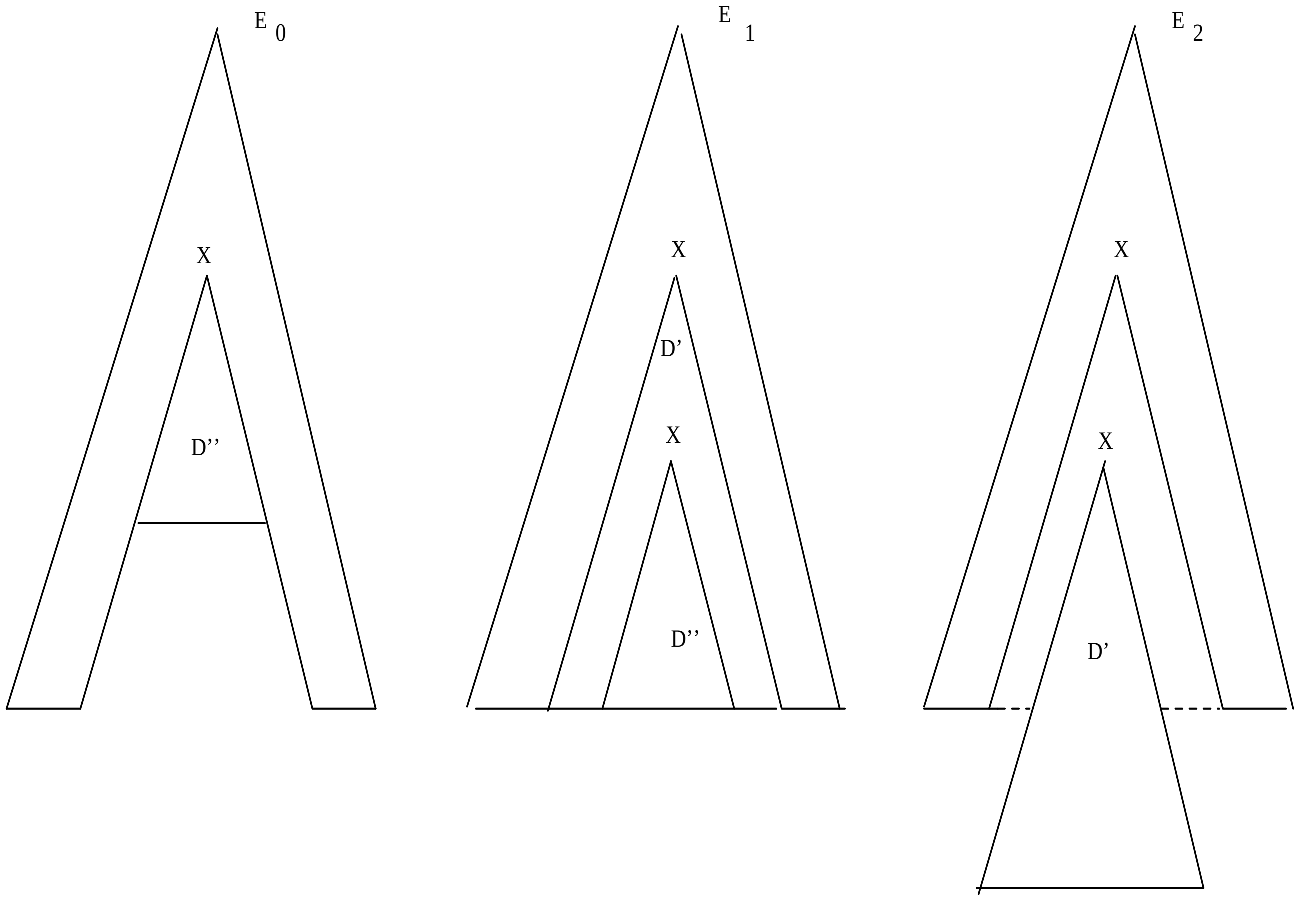
- E_0 jest rezultatem podstawienia w drzewie D drzewa D'' na wierzchołek p_i, w notacji symbolicznej (por. Odc. 3, str. 4), $E_0 = D[p_i \leftarrow D'']$.
- $E_1 = D$.
- E_2 jest rezultatem podstawienia w

  drzewie D drzewa D' na wierzchołek p_j, tj. $E_2 = D[p_j \leftarrow D']$, i $u_2 = p_jv$.
- E_3 jest rezultatem podstawienia w drzewie E_2 drzewa D' na wierzchołek u_2, tj. $E_3 = E_2[u_2 \leftarrow D']$, i $u_3 = u_2v$.
- Ogólniej, dla $n \geq 2$, E_{n+1} jest rezultatem podstawienia w drzewie E_n drzewa D' na wierzchołek u_n, tj. $E_{n+1} = E_n[u_n \leftarrow D']$, i $u_{n+1} = u_nv$.

Niech α_1, α_2 będą tymi słowami, że $\gamma = \alpha_1\beta\alpha_2$. Opierając się wprost na powyższych definicjach, nietrudno wykazać, że

$$\begin{array}{rcc} y(E_0) & = & \alpha_1\beta\alpha_2 \\ y(E_1) & = & \alpha_1\gamma_1\beta\gamma_2\alpha_2 \\ y(E_2) & = & \alpha_1\gamma_1\gamma_1\beta\gamma_2\gamma_2\alpha_2 \\ y(E_3) & = & \alpha_1\gamma_1\gamma_1\gamma_1\beta\gamma_2\gamma_2\gamma_2\alpha_2 \\ \ldots & \ldots & \ldots \\ y(E_n) & = & \alpha_1\gamma_1^n\beta\gamma_2^n\alpha_2 \end{array}$$

□



Przykład 1 Zbiór $\{a^n b^n a^n : n \in N\}$ nie jest bezkontekstowy. Istotnie, niech M będzie stałą z lematu i niech $n > M$. Przypuśćmy, że $a^n b^n a^n = \alpha_1 \gamma_1 \beta \gamma_2 \alpha_2$ jest dekompozycją z lematu. Nietrudno zauważyć, że $\gamma_1, \gamma_2 \in \{a\}^* \cup \{b\}^*$. Analizując poszczególne przypadki, łatwo dochodzimy do sprzeczności. Na przykład, gdy $\gamma_1 \in \{a\}^*$ i $\gamma_2 \in \{b\}^*$, to również słowo $a^{n+|\gamma_1|} b^{n+|\gamma_2|} a^n$ powinno należeć do języka, wbrew definicji.

Jeśli $w \in \Sigma^*$ i $|w| = m$, to elementy zbioru $\{1, \ldots, m\}$ będziemy nazywali *pozycjami* w słowie w. W kolejnym lemacie będziemy rozważali słowa z wyróżnionymi pozycjami, tj., formalnie, pary (w, P), gdzie $P \subseteq \{1, \ldots, |w|\}$.

Lemat 2 (lemat Ogdena) *Niech $L \subseteq \Sigma^*$ będzie językiem bezkontekstowym. Wówczas istnieje stała M zależna tylko od L, taka, że dla każdego słowa $\alpha \in L$ z dowolnie wyróżnionymi co najmniej M pozycjami, α można przedstawić $\alpha = \alpha_1 \gamma_1 \beta \gamma_2 \alpha_2$ tak, że spełnione są następujące warunki:*

- *γ_1 i γ_2 zawierają wspólnie co najmniej jedną wyróżnioną pozycję,*
- *słowo $\gamma_1 \beta \gamma_2$ zawiera nie więcej niż M wyróżnionych pozycji,*
- *dla każdego $i \geq 0$, $\alpha_1 \gamma_1^i \beta \gamma_2^i \alpha_2 \in L$.*

Dowód (szkic). Argument jest podobny jak w dowodzie lematu o pompowaniu. Niech $G = (\Sigma, V, X_I, R)$ będzie gramatyką bezkontekstową w postaci Chomsky'ego generującą język $L - \{\epsilon\}$ i niech $|V| = k$. Kładziemy tym razem

$$M = 2^{k+1}$$

Przypuśćmy, że D jest drzewem wyprowadzenia słowa α, w którym wyróżnione jest $\geq M$ pozycji (oczywiście wówczas także $|\alpha| \geq M$). Strategia wyboru ścieżki $p_0, p_1, \dots, p_\ell$ jest następująca. Oczywiście, $p_0 = \epsilon$. Z kolei przypuśćmy, że już wybraliśmy do naszej ścieżki wierzchołki $p_0, p_1, \dots, p_m$, $0 \leq m < \ell$. Wierzchołek p_m ma w drzewie D dwa następniki: $p_m 1$ i $p_m 2$, które wyznaczają dwa poddrzewa drzewa D, powiedzmy, D_1 i D_2 odpowiednio. Wybieramy ten następnik, w którego poddrzewie znajduje się *więcej* wyróżnionych pozycji, a w przypadku remisu wybieramy, powiedzmy, $p_m 1$. Nietrudno zauważyć, że tak określona ścieżka $p_0, p_1, \dots, p_\ell$ prowadzi do liścia odpowiadającego jakiejś wyróżnionej pozycji w generowanym słowie α.

Każdy wierzchołek p_m na ścieżce o tej własności, że *oba* poddrzewa wyznaczone przez następniki p_m zawierają jakieś wyróżnione pozycje, nazwiemy *punktem rozgałęzienia*. Zauważmy, że jeżeli p_m jest punktem rozgałęzienia, to poddrzewo wyznaczone przez p_{m+1} zawiera istotnie *mniej* ale *co najmniej połowę* pozycji, jakie znajdowały się w drzewie wyznaczonym przez p_m. Pamiętając o tym, że w drzewie wyznaczonym przez p_0 (tj. drzewie D) jest $\geq 2^{k+1}$ wyróżnionych pozycji, nietrudno wykazać, że na ścieżce $p_0, p_1, \dots, p_\ell$ musi leżeć co najmniej $k+1$ punktów rozgałęzienia. Wśród tych punktów możemy więc znaleźć dwa takie, powiedzmy p_i, p_j, że $i < j$ i $D(p_i) = D(p_j)$. Dalej dowód przebiega bardzo podobnie jak w przypadku lematu o pompowaniu. Mianowicie, wybieramy "pierwszą od końca" parę p_i, p_j o powyższej własności i słowa α_1, γ_1, β, γ_2, α_2 określamy analogicznie, jak w tamtym dowodzie.

Aby wykazać, że słowo $\alpha_1 \beta \alpha_2$ zawiera nie więcej niż M wyróżnionych pozycji, wystarczy zauważyć, że p_i jest co najwyżej "$k + 1$-szym od końca" punktem rozgałęzienia na ścieżce, oraz, że, ogólnie, poddrzewo wyznaczone przez "m-ty od końca" punkt rozgałęzienia na naszej ścieżce, zawiera co najwyżej 2^m pozycji. Fakt, że jedno ze słów γ_1, γ_2 zawiera co najmniej jedną wyróżnioną pozycję, wynika z tego, że p_i jest punktem rozgałęzienia. Ostatni warunek lematu jest otrzymany analogicznie, jak w przypadku dowodu lematu o pompowaniu. □

Przykład 2 Język

$$L = \{a^i b^j c^k \; : \; i \neq j, \; j \neq k, \; i \neq k\}$$

nie jest bezkontekstowy. Istotnie, niech n będzie stałą z lematu Ogdena. Rozważmy słowo $\alpha = a^n b^{n+n!} c^{n+2 \cdot n!}$, w którym są wyróżnione wszystkie pozycje a i tylko te. Przypuśćmy, że $\alpha = \alpha_1 \gamma_1 \beta \gamma_2 \alpha_2$ jest dekompozycją z lematu. Wówczas przynajmniej jedno ze słów γ_1, γ_2 musi zawierać przynajmniej jedno a. Nietrudno widzieć, że musi zachodzić $\gamma_1 \in \{a\}\{a\}^*$ oraz $\gamma_2 \in \{a\}^* \cup \{b\}^* \cup \{c\}^*$. W każdym przypadku możemy łatwo dojść do sprzeczności. Przypuśćmy na przyład, że $\gamma_2 \in \{b\}^*$. Niech $p = |\gamma_1|$, $q = |\gamma_2|$. Mamy $1 \leq p \neq n$, niech więc $p_1 = n!/p$. Wówczas, z trzeciego punktu lematu Ogdena, również słowo

$$a^{n+p \cdot (2p_1)} b^{n+p \cdot (2 \;\, p_1)} c^{n+2 \cdot n!}$$

powinno należeć do L, ale $n + p \cdot (2p_1) = n + 2 \cdot n!$, co przeczy definicji L. □

4 Własności domknięcia

W tym punkcie rozważymy *własności domknięcia*, tzn. na jakie operacje jest, a na jakie nie jest zamknięta klasa języków bezkontekstowych. Dowody poniższych faktów pozostawiamy jako ćwiczenia.

Fakt 2 *Rodzina języków bezkontekstowych jest zamknięta na operacje skończonej sumy, konkatenacji i iteracji ("gwiazdki").*

Uwaga. Rodzina języków bezkontekstowych nie jest natomiast zamknięta na operacje przecięcia i uzupełnienia. Istotnie, niech

$$M = \{a^n b^n a^n \ : \ n \in N\}$$

będzie językiem, o którym wykazaliśmy, że nie jest bezkontekstowy (Przykład 1). Nietrudno pokazać, że

$$\begin{aligned} M &= \{a^n b^n \ : \ n \in N\} a^* \cap a^* \{b^n a^n \ : \ n \in N\} \\ M &= \{a, b\}^* - \{a^i b^j a^k \ : \ i \neq j \text{ lub } j \neq k\} \end{aligned}$$

gdzie języki występujące po prawych stronach równości *są* bezkontekstowe.

Fakt 3 *Rodzina języków bezkontekstowych jest zamknięta na przecięcia z językami regularnymi, tzn. jeśli język L jest bezkontekstowy a język R jest* regularny*, to język $L \cap R$ jest bezkontekstowy.*

Fakt 4 *Rodzina języków bezkontekstowych jest zamknięta na ilorazy przez języki regularne, tzn. jeśli język L jest bezkontekstowy a język R jest* regularny*, to języki LR^{-1} i $R^{-1}L$ są bezkontekstowe.*

Fakt 5 *Rodzina języków bezkontekstowych jest zamknięta na* podstawienia*, tzn. jeśli język L jest bezkontekstowy i $f : \Sigma \to P(\Gamma^*)$ jest funkcją taką, że, dla każdego $\sigma \in \Sigma$, $f(\sigma)$ jest językiem bezkontekstowym, to $\hat{f}(L)$ jest językiem bezkontekstowym.*

Fakt 6 *Rodzina języków bezkontekstowych jest zamknięta na przeciwobrazy homomorficzne, tzn. jeśli $f : \Sigma \to \Gamma^*$ jest homomorfizmem i $M \subseteq \Gamma^*$ jest bezkontekstowy, to język*

$$\hat{f}^{-1}(M) = \{w \in \Sigma^* \ : \ \hat{f}(w) \subseteq M\}$$

jest bezkontekstowy.

Ćwiczenia

1. Dowieść, że każdy język bezkontekstowy nad jednoliterowym alfabetem jest regularny.

   Wskazówka. Skorzystać z lematu o pompowaniu i relacji przystawania *modulo*.

2. Czy przy pomocy automatu skończonego można rozpoznać, ze w słowie nad alfabetem $\{a, b, c\}$ jedna z liter występuje co najmniej tyle samo razy co dwie pozostałe? Czy można rozpoznać tę własność przy pomocy automatu ze stosem?

3. Dowieść, że język $\{ww : w \in \Sigma^*\}$ nie jest bezkontekstowy, o ile $|\Sigma| \geq 2$.

4. Niech L będzie językiem regularnym. Które z następujących języków są regularne lub bezkontekstowe?

   - $\{ww : w \in L\}$
   - $\{ww^R : w \in L\}$
   - $\{wv^R : w, v \in L\}$

5. Dowieść, że język $\{a^i b^j c^k : i \neq j,\ j \neq k\}$ nie jest bezkontekstowy.

6. Dowieść, że język $\{w\$v : v$ jest podsłowem $w\}$ nie jest bezkontekstowy, natomiast język $\{w\$v^R : v$ jest podsłowem $w\}$, jest bezkontekstowy.

7. Czy następujacy zbiór jest językiem bezkontekstowym:

   Zbior słów $p\$w$, gdzie p jest wyrażeniem regularnym nad alfabetem $\{a, b\}$, a w jest słowem należącym do języka wyznaczonego przez p.

   Analogiczne pytanie dla $p\$w^R$.

8. Niech A – alfabet, $A' = \{a' : a \in A\}$. Określamy homomorfizmy h, h_1, h_2 z $A \cup A'$ w A następująco: $h(a) = h(a') = a$, $h_1(a) = a$, $h_1(a') = \epsilon$, $h_2(a) = \epsilon$, $h_2(a') = a$.

   Definiujemy *przeplot* języków L_1 i L_2:

   $$\mathrm{Shuffle}(L_1, L_2) = \{x : (\exists y \in h^{-1}(x)) h_1(y) \in L_1 \wedge h_2(y) \in L_2\}$$

   Dowieść, że

   (a) przeplot dwóch jezyków regularnych jest regularny,

   (b) przeplot języka bezkontekstowego z regularnym jest bezkontekstowy,

   (c) przeplot dwóch jezykow bezkontekstowych nie musi być bezkontekstowy (wskazówka: $\{a^n b^m c^n d^m : n, m \in N\}$ nie jest bezkontekstowy)

9. Język bezkontekstowy nazywamy *liniowym*, jeśli można go wygenerować gramatyką, w której po prawej stronie dowolnej reguły występuje co najwyżej jedna zmienna. Dowieść, że następujące języki bezkontekstowe nie są liniowe:

   (a) $\{a^i b^i c^j d^j : i, j \in \mathbf{N}\}$,

   (b) $L = \{x \in (a+b)^*, \#_a(x) = \#_b(x)\}$

   Wskazówka. Sformułować lemat o pompowaniu dla języków liniowych oparty na dekompozycji $\alpha = \alpha_1\gamma_1\beta\gamma_2\alpha_2$, w której ograniczone jest $\alpha_1\gamma_1\gamma_2\alpha_2$.