Spis treści

1 Literatura

- J.E. Hopencroft "Wprowadzenie do teorii automatów i obliczeń"
- M. Sipser "Wprowadzenie do teorii obliczeń"
- G.E. Revesz "Introduction to formal languages"
- H.R. Lewin, Papadimitriou "Elements of the Theory of Computation"

2 Język

$$\infty \text{ zdań} + n \text{ reguł} = \text{język}$$

2.1 Funkcje języka

1. Poznawcza
2. Społeczna
3. Ekspresywna

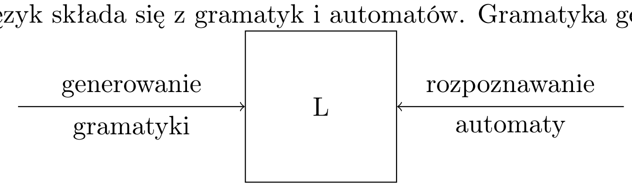
2.2 Nauka o języku

1. syntaktyka - budowa
2. semantyka - co znaczy?
3. pragmatyka - jak się używa?

Przykład:$2 + 3 \cdot 4$: różna semantyka $\rightarrow$ wieloznaczność syntaktyczna

2.3 Definicja

Język składa się z gramatyk i automatów. Gramatyka generuje język, automat rozpoznaje język.



3 Alfabet

Alfabet to zbiór atomowych dozwolonych symboli
Przykład: $\{a, b, c, d\}$

4 Słowo

Słowo to skończony ciąg symboli nad alfabetem.

- ε - słowo puste
- $\{K, L, O, P, S\} \neq$ "$KLOPS$", ponieważ słowa mają dodane znaczenie, w postaci tego do którego języka należą.

4.1 Konkatenacja

- Dla $P = a_1...a_n$ i $Q = b_1...b_n,$ to $PQ = a_1...a_nb_1...b_n$
- $P\epsilon = P$
- $\epsilon\epsilon = \epsilon$

4.2 Podsłowo

- $P = Q_1|Q|Q_2$
- $Q \subset P$

4.3 Długość słowa

- $|\epsilon| = 0$
- $|Pa| = |P| + 1$
- $|PQ| = |P| + |Q|$

4.4 Potęga słowa

- $P^0 = \epsilon$
- $P^{n+1} = P^nP$

4.5 Odbicie

- $\epsilon^-1 = \epsilon$
- $(Pa)^-1 = aP^-1$

5 Język

Zbiór dozwolonych słów nad alfabetem.

- V^* - zbiór wszystkich języków
- $V^+ = V^*\ \epsilon$
- $L \in V^*$
- $\{a, ab\} \neq \epsilon, a, ab$ ponieważ inaczej operacje na językach by nie działały

5.1 Konkatenacja języków

$$L_1 = \{a, aa\}, L_2 = \{b, aba\}, L_1L_2 = \{ab, aaba, aab, aaaba\}$$

L_2 \ L_1	a	aa
b	ab	aab
aba	aaba	*aaaba*

Tabela 1: Tabela konkatenacji języków L_1 i L_2

$|L_1L_2| \leq |L_1| \cdot |L_2|$ bo eps wszystko psuje

$$L_1 = \{a^n : n \geq 0\}, L_2 = \{b^n : n \geq 0\}, L_1L_2 = \{a^nb^m : n, m \geq 0\}$$

5.2 Potęga języka

$$L = \{a, ab\}, L^0 = \{\epsilon\}, L^1 = \{a, ab\}, L^2 = L \cdot L$$

Potęgowanie na językach jest dziwne

$$L = \{a^n : n \geq 0\}, L^2 = \{a^na^m : a, m \geq 0\} = \{a^n : a \geq 0\} = L$$

Potęgowanie języku **nie** zwiększyło mocy

$$L = \{a^n : n > 0\}, L^2 = \{a^na^m : a, m > 0\} = L \setminus \{a\} = \{a^n : n > 1\}$$

Potęgowanie języku **zmniejszyło** moc

5.3 Dzielenie słów

$$P \in L^n \rightarrow \text{ można podzielić } P \text{ na } n \text{ (niekoniecznie różnych) słów}$$

$$L = \{a, ab\}, ''aababaabab'' \in L^n, n =?$$

Jest to problem wykładniczy, który wymaga stworzenia drzewa różnych możliwości.

6 Domknięcie Kleenego

$$L^* = \bigcup_{n \geq 0}^{\infty} L^n$$

$$L^+ = \bigcup_{n \geq 1}^{\infty} L^n$$

$$L_1 = \{a\}, L_1^* = \{a^n : n \geq 0\}, L_1^+ = \{a^n : n > 0\}$$

$$L_2 = \{\epsilon, a\}, L_2^* = \{a^n : n \geq 0\} = L_2^+$$

$$L = \{aa, ab, ba, bb\}, L^* = \{P \in \{a, b\}^* : 2||P|\} = \text{wszystkie słowa nad alfabetem a, b o parzystej długości}$$

- $L^+ \subset L^*$
- $\epsilon \in L \rightarrow L^+ = L^*$
- $(L^*)^* = L^*$
- $L_1 \subset L_2 \rightarrow L_1^* \subset L_2^*$

$$L = \{a^n : n > 1\}, L^1 \neq L^2, L^* = L$$

7 Automaty

- nieskończona taśma
- rejestry
- w każdym rejestrze symbol z alfabetu T
- głowica, która porusza się od lewej do prawej po rejestrach taśmy, aż do momentu, kiedy napotka pusty rejestr. Głowica zawsze jest w jednym ze stanów z zbioru stanów

7.1 Deterministyczne automaty skończone

Automat skończenie stanowy jest uporządkowaną piątką

$$\mathfrak{A} = \langle K, T, \delta, q_0, H \rangle$$

- K zbiór stanów
- T alfabet - symbole z tego alfabetu znajdują się w rejestrach
- $\delta : K \times T \to K$ funkcja przejścia automatu
- q_0 stan początkowy automatu
- H zbiór stanów akceptowalnych/końcowych

7.1.1 Funkcja przejść

Zbiory K i T są skończone, co oznacza, że funkcję δ można przedstawić w formie tabelki. Przykład:

$$K = \{q_0, q_1, q_2\}, T = \{a, b\}, H = \{q_2\}$$

$$\delta : K \times T \to K$$

$K \backslash T$	a	b
$\to q_0$	q_2	q_1
q_1	q_0	q_1
$\underline{q_2}$	q_1	q_2

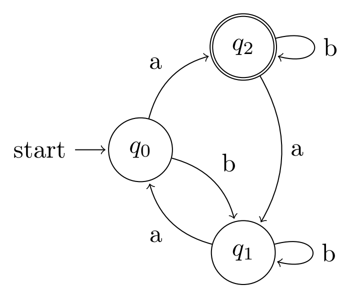


Diagram 1: Tabela konkatenacji języków L_1 i L_2 oraz graf przejść automatu

7.1.2 Rozszerzona funkcja przejść

$$\hat{\delta}: K \times T^* \to K$$

- $\hat{\delta}\,(q, \epsilon) = q$
- $\hat{\delta}\,(q, Pa) = \delta(\hat{\delta}\,(q, P), a)$

7.1.3 Przykład

Narysuj diagram przejścia deterministycznego automatu skończenie stanowego $\mathfrak{A}$ w którym $T = \{0, 1\}, P \in L(\mathfrak{A})$ wtedy i tylko wtedy gdy w P występuje na pierwszym od końca miejscu.

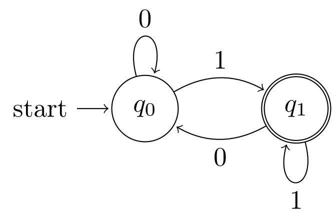


Diagram 2: Diagram przejścia automatu do wykrywania 1 na pierwszym miejscu od końca

Narysuj diagram przejścia deterministycznego automatu skończenie stanowego $\mathfrak{A}$ w którym $T = \{0, 1\}, P \in L(\mathfrak{A})$ wtedy i tylko wtedy gdy w P występuje na drugim od końca miejscu 1.

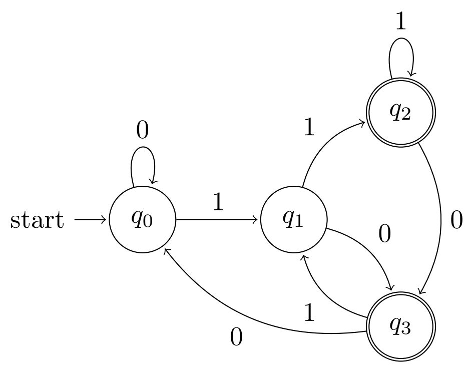


Diagram 3: Diagram przejścia automatu do wykrywania 1 na drugim miejscu od końca

Widać na diagramie 3 wprost zależność że w zależności od miejsca od końca na którym ma być jeden rośnie ilość stanów. Ilość stanów maszyny $|K|$ do wykrywania 1 na n-tym miejscu od końca można wyrazić w następujący sposób: $|K| = 2^n$

7.2 Niedeterministyczne automaty skończone

- zamiast jednego stanu początkowego jest zbiór stanów początkowych
- niedeterministyczna funkcja przejścia, która zwraca zbiór wyjściowych stanów

$$\mathfrak{A} = \langle K, T, \delta, Q_0, H \rangle$$

gdzie oznaczenia są identyczne jak dla deterministycznego automatu z dwoma różnicami:

- $\delta : K \times T \to a \in K$ funkcja przejścia automatu
- Q_0 zbiór stanów początkowych automatu

$$L(\mathfrak{A}) = \{P \in T^* : \overset{\wedge}{\delta} (Q_0, P) \cap H \neq \emptyset\}$$

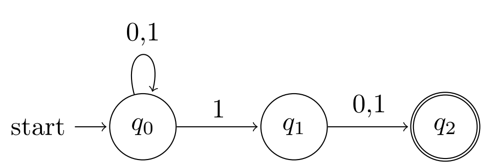


Diagram 4: Niedeterministyczna wersja automatu $\mathfrak{A}$ z rysunku 3

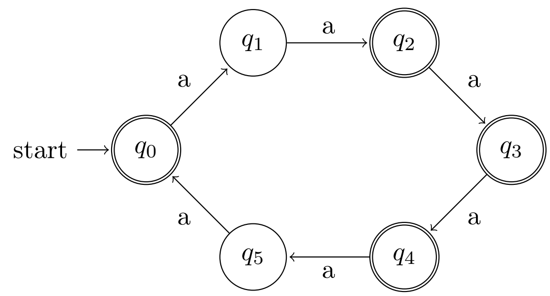


Diagram 5: Diagram przejścia automatu do wykrywania słów o długości podzielnej przez 2 lub 3

Jak widać zamiast 4 stanów potrzeba tylko 3, to dlatego, że dla wersji niedeterministycznej $|K| = n + 1$

7.2.1 Rozszerzona funkcja przejść

$$\overset{\wedge}{\delta}: P(K) \times T^* \to P(K)$$

- $\overset{\wedge}{\delta}(A, \epsilon) = A$
- $\overset{\wedge}{\delta}(A, Pa) = \bigcup_{q \in \overset{\wedge}{\delta}(A,P)} \delta(q, a)$

$$\overset{\wedge}{\delta}(\{p\}, a) = \delta(p, a)$$

7.2.2 Twierdzenie Scotta

- każdy **nie**deterministyczny automat skończony można zastąpić równoważnym deterministycznym automatem skończonym
$$\mathfrak{L}_{ndet} \subset \mathfrak{L}_{det}$$
- każdy deterministyczny automat skończony można zastąpić równoważnym **nie**deterministycznym automatem skończonym
$$\mathfrak{L}_{det} \subset \mathfrak{L}_{ndet}$$
- liczba stanów automatu deterministycznego jest wykładnicza w stosunku do liczby stanów automatu niedeterministycznego

$$\mathfrak{L}_{det} = \mathfrak{L}_{ndet}$$

Zatem co nam daje niedeterministyczność? Przede wszystkim prostotę, ale kosztem wykładniczej złożoności.

Narysuj diagram przejścia deterministycznego automatu skończenie stanowego $\mathfrak{A}$ w którym $T = \{a\}, P \in L(\mathfrak{A})$ wtedy i tylko wtedy gdy $(2||P|) \vee (3||P|)$.

Jak widzimy na diagramie 5 przyjmuje postać cyklu o okresie 6, ponieważ $NWW(3, 2) = 6$. Problem z diagramami deterministycznym się pojawia dla wyższych liczb, np.: 7 i 5, wtedy $NWW(7, 5) = 35$. Zatem narysujmy diagram niedeterministyczny 6.

Jako, że automaty niedetermnistyczne pozwalają na kilka stanów początkowych, to tworzymy diagram niespójny, który w zależności od tego czy $|P|$ jest podzielne przez 5 czy 7 przechodzi do odpowiedniego pod-automatu. Najłatwiej to można sobie wyobrazić jako dwa równoległe automaty z alternatywą na koniec.

7.2.3 Przekształcenie niedet $\to$ det

$$\mathfrak{A} = \langle K, T, \delta, Q_0, H \rangle$$

$$\mathfrak{A}' = \langle K', T', \delta', q_0', H' \rangle$$

$$L(\mathfrak{A}) = L(\mathfrak{A}')$$

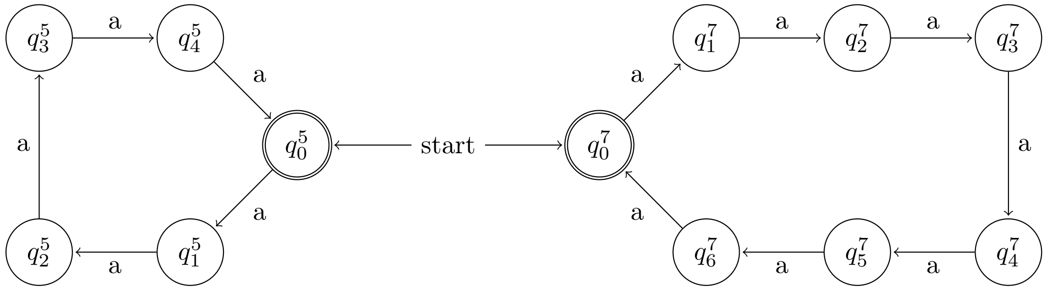


Diagram 6: Diagram przejścia automatu ndet do wykrywania słów o długości podzielnej przez 5 lub 7

- $T' = T$ bo nie ma sensu zmieniać taśmy
- $K' = P(K)$ **Wykładniczy wzrost liczby stanów**
  w przekształceniu będziemy używać systemu etykiet(konstrukcja potęgowa) aby zamieniać zbiory stanów na pojedyncze stany

$$\{q_0, q_1\} = q^{01}$$
$$P(K) = \{q^{\emptyset}, q^1, q^0, q^{10}, ...\}$$

- $q_0' = Q_0$ - stan odpowiadający zbiorowi stanów początkowych
- $H' = \{A \in K' : A \cap H \neq \emptyset\}$
- $\delta'(A, a) = \bigcup_{q \in A} \delta(q, a)$

Dla automatu 4 zbudujmy równoważny automat deterministyczny. Korzystając z powyższych zasad otrzymujemy:

- $K' = P(K) = \{\emptyset, \{q_0\}, ..., \{q_1, q_2\}, \{q_0, q_1, q_2\}\} = \{q^{\emptyset}, q^0, ..., q^{12}, q^{012}\}$
- $H' = \{q^2, q^{12}, q^{02}, q^{012}\}$
- $q_0' = \{q_0\} = q^0$

K' \ T	0	1
$q^{\emptyset}$	$q^{\emptyset}$	$q^{\emptyset}$
$\rightarrow q^0$	q^0	q^{01}
q^1	q^2	q^2
$\underline{q^2}$	$q^{\emptyset}$	$q^{\emptyset}$
q^{01}	q^{02}	q^{012}
$\underline{q^{12}}$	q^2	q^2
$\underline{q^{02}}$	q^0	q^{01}
$\underline{q^{012}}$	q^{02}	q^{012}

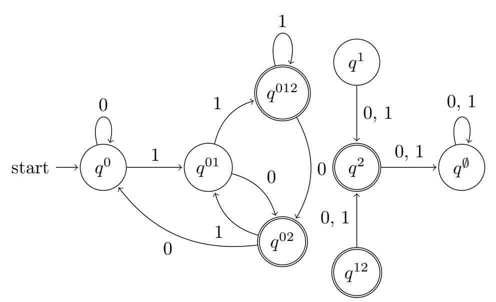


Diagram 7: Zdeterminizowany automat z rysunku 4 i jego tabela przejść

Jak widać na diagramie 7 liczba stanów wzrosła z 3 do 8, co jest zgodne z przewidywaniami. Jednocześnie widać, że diagram 3 zawiera się w diagramie 7, co niekoniecznie oznacza, że są sobie równoważne, lecz jako, że stany dodatkowe są nieosiągalne to te dwa automaty są równoważne. **Nie zawsze równoważność automatów będzie tak oczywista.**

7.3 Automaty z przejściem

Co jeśli moglibyśmy zmienić stan ale nie ruszyć głowicy? Wtedy mamy do czynienia z automatem z przejściem.

$$\epsilon \in T, \epsilon = \text{nie ruszaj głowicy automatu}$$

$$\delta : K \times (T \cup \{\epsilon\}) \to P(K)$$

Każdy automat z przejściem jest niedeterministyczny

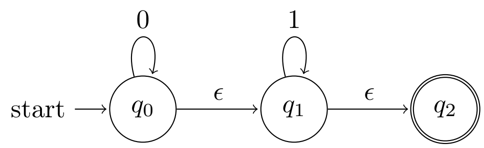


Diagram 8: Automat z przejściem

Automat przedstawiony na rysunku 8 akceptuje języki o następującej postaci $L(\mathfrak{A}) = \{0^n 1^m 2^k : n, m, k \geq 0\}$.
Nie istnieją różne epsilony: $00\epsilon 1 \epsilon 222 = 00122$

7.3.1 Domknięcie stanu

$$E(q) = \text{zbiór stanów osiągalnych z } q \text{ przez dowolną liczbę epsilonów}$$

1. $q \in E(q)$
2. $r \in E(q) \wedge p \in \delta(r, \epsilon) \to p \in E(q)$

7.3.2 Domknięcie zbioru stanów

$$E(A) = \bigcup_{q \in A} E(q)$$

7.3.3 Rozszerzona funkcja przejść

$$\hat{\delta} : P(K) \times T^* \to P(K)$$

- $\hat{\delta}(A, \epsilon) = E(A)$
- $\hat{\delta}(A, Pa) = \bigcup_{q \in \hat{\delta}(A,P)} E(\delta(q, a))$

7.3.4 Przekształcenie $\epsilon \to$ ndet

$$\mathfrak{A}' = \langle K', T', \delta', Q_0', H' \rangle$$

$$T' = T, K' = K, H' = H, Q_0' = E(Q_0), \delta'(A, a) = E(\delta(q, a))$$

$$\mathfrak{L}_{\mathfrak{ndet}} = \mathfrak{L}_{\epsilon} = \mathfrak{L}_{\mathfrak{det}}$$

δ_ϵ	a	b	ϵ
$\to q_0$	$\emptyset$	$\{q_1\}$	$\{q_2\}$
q_1	$\{q_1, q_2\}$	$\{q_2\}$	$\emptyset$
$\underline{q_2}$	$\{q_0\}$	$\emptyset$	$\emptyset$

$\Longrightarrow$

δ_{ndet}	a	b
$\to q_0$	$E(\emptyset) = \emptyset$	$\{q_1\}$
q_1	$\{q_1, q_2\}$	$\{q_2\}$
$\to \underline{q_2}$	$E(\{q_0\}) = \{q_0, q_2\}$	$\emptyset$

Diagram 9: Przekształcenie automatu z przejściem na niedeterministyczny

8 Wyrażenia regularne

$$Reg(V) = \text{zbiór wyrażeń regularnych nad alfabetem } V$$

- $o \in Reg(V)$
- $e \in Reg(V)$
- $a \in V \to a \in Reg(V)$
- $u, v \in Reg(V) \to (u + v), (u \cdot v), (u^*) \in Reg(V)$

8.1 Operacje

W kolejności od najwyższego priorytetu do najniższego

1. $P \in Reg(V) \to L(P) \neq \emptyset$
2. $L(u^*) = (L(u))^*$
3. $L(uv) = L(u) \cdot L(v)$
4. $L(u + v) = L(u) \cup L(v)$
5. $L(u) = \{u\}$

8.2 Przykłady

$$L(ba^*) = \{ba^n : n \geq 0\}$$

$$L(ba^*) = L(b)L(a^*) = \{b\} \cdot (L(a))^* = \{b\} \cdot \{a\}^* = \{ba^n : n \geq 0\}$$

$L((a + b)^* ab(a + b)^*)$ - wszystkie słowa nad alfabetem $\{a, b\}$ zaczynające się od a i kończące się b

8.3 Tw. Kleenego

$$\forall_{v \in Reg(V)} L(v) \subset \mathfrak{L}_{det}$$

Każdy język generowany przez wyrażenie regularne jest językiem akceptowanym przez automat skończony. Dowód opiera się na konstrukcji automatu ϵ odpowiadającego operatorowi wyrażenia regularnego.

8.3.1 $w = u + v$

$$L(u + v) = L(u) \cup L(v)$$
$$L(u) = L(\mathfrak{A}_u), L(v) = L(\mathfrak{A}_v)$$
$$L(\mathfrak{A}) = L(\mathfrak{A}_u) \cup L(\mathfrak{A}_v)$$

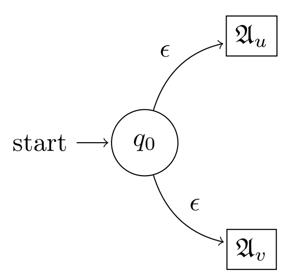


Diagram 10: Konstrukcja automatu dla wyrażenia regularnego $u + v$

8.3.2 $w = uv$

$$L(uv) = L(u) \cdot L(v)$$

$$L(u) = L(\mathfrak{A}_u), L(v) = L(\mathfrak{A}_v)$$

$$L(\mathfrak{A}) = L(\mathfrak{A}_u) \cdot L(\mathfrak{A}_v)$$

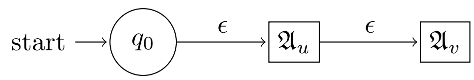


Diagram 11: Konstrukcja automatu dla wyrażenia regularnego uv

8.3.3 $w = u^*$

$$L(u^*) = (L(u))^*$$

$$L(u) = L(\mathfrak{A}_u)$$

$$L(\mathfrak{A}) = (L(\mathfrak{A}_u))^*$$

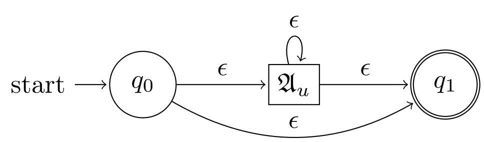


Diagram 12: Konstrukcja automatu dla wyrażenia regularnego u^*

9 Klasy języków

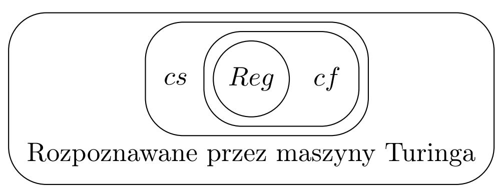


Diagram 13: Nadzbiory języków regularnych

9.1 Języki regularne

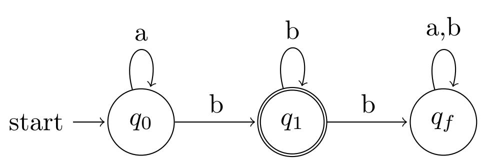


Diagram 14: Konstrukcja automatu dla automatu a^*b^*

Język regularny to język akceptowany przez wyrażenie regularne, czyli język akceptowany przez automat skończony.

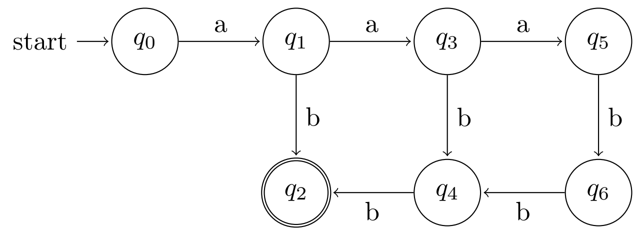


Diagram 15: Konstrukcja automatu akceptującego język $L = \{a^n b^n : 1 \leq n \leq 3\}$ **bez śmietnika**

Czy można zapisać automat deterministyczny (lub nie) skończenie stanowy, akceptujący język $L = \{a^n b^n : n \geq 1\}$ **?**

Nie da się, ponieważ wymagałoby to nieskończonej ilości stanów. Zatem język L nie jest językiem regularnym.

9.1.1 Lemat o pompowaniu dla języków regularnych

Służy do dowodzenia, że język **nie** jest regularny.

Jeżeli $L = L(\mathfrak{A})$

to: istnieje $k \geq 0$, taki, że każde słowo $P \in L$ o długości $|P| \geq k$ można zapisać jako $P = XYZ$, spełniające warunki:

Lemat o pompowaniu jest warunkiem koniecznym ale nie wystarczającym. To oznacza, że wszystkie języki regularne spełniają warunek lematu o pompowaniu, ale nie wszystkie języki spełniający warunek lematu o pompowaniu są regularne. Warunkiem dostatecznym jest zdefiniowanie automatu skończenie stanowego akceptującego język.

- $Y \neq \epsilon$
- $|XY| \leq k$
- $\forall_{i \geq 0} XY^i Z \in L$

Parafrazując: jeśli język jest regularny, to nie ma w nim słowa, którego nie mógłbyś podzielić na trzy części, takie, że środkowa część jest powtarzalna.

Przykład zgodny

Dlaczego język $L = \{a^n b^m : n, m \geq 1\}$ jest regularny?

- $k > 2, P \in L$
- $|P| \geq k \to n + m \geq k$
- $P = a^{n-x} a^x b^m$ czyli $P = XYZ$ gdzie $X = a^{n-x}, Y = a^x, Z = b^m$
- $n - x + x < k \leftrightarrow |XY| \leq k$
- dla $i \geq 0$ $XY^i Z = a^{n-x} a^{xi} b^m = a^{n+x(i-1)} b^m$, co jak widzimy jest w języku L

Przykład niezgodny

Dlaczego język $L = \{a^n b^n : n \geq 1\}$ nie jest regularny?

- $k > 2, P \in L$
- $|P| \geq k \to 2n \geq k$
- $|XY| \leq k \to XY = a^n \vee XY = b^n$
- dla $i = 2$ $XY^2 Z = a^n b^n b^n \notin L$

Zatem muszą istnieć języki nieregularne

9.1.2 Przechodniość regularności

Z tw. Kleenego:

$$L_1, L_2 \in \mathfrak{L}_{reg} \to L_1 \cup L_2, L_1 \cdot L_2, L_1^*, L_2^* \in \mathfrak{L}_{reg}$$

$$L_1 \cup L_2 = L(v+w) = L(v) \cup L(w) = L_1 \cup L_2$$

$$L_1 \cap L_2 = \overline{\overline{L_1 \cap L_2}} = \overline{\overline{L_1} \cup \overline{L_2}}$$

10 Gramatyki

Gramatyki służą do generowania języków. Działanie gramatyk wyraża się przy pomocy reguły przepisującej.

$$(P, Q) - \text{słowa}$$

$$\text{Jeżeli } P \to Q \text{ oraz } U = P_1 P P_2 \text{ to } U \Rightarrow P_1 Q P_2$$

Jeżeli reguła przepisująca jest wykonywana wielokrotnie to używamy oznaczenia $\overset{*}{\Rightarrow}$.

10.1 Typu 3 (regularne)

$$G = \langle V_N, V_T, S, F \rangle$$

gdzie:

- V_N - zbiór nieterminali (symboli zastępowanych)
- V_T - zbiór terminali (symboli niezastępowanych)
- S - symbol początkowy
- F - zbiór reguł przepisujących

$$V_N \cap V_T = \emptyset$$

10.1.1 Wyprowadzanie słowa w jednym kroku

Dla gramatyki G mówimy, że W jest wyprowadzane w jednym kroku z U (oznaczane $U \Rightarrow W$) jeżeli istnieje reguła $P \to Q \in F$ taka, że $U = P_1 P Q_2$ oraz $W = P_1 Q Q_2$.

10.1.2 Wyprowadzanie słowa w wielu krokach

Dla gramatyki G mówimy, że W jest wyprowadzane w wielu krokach z U (oznaczane $U \overset{*}{\Rightarrow} W$) wtedy gdy $U = W$, lub gdy istnieją słowa $R_1 \ldots R_n : n > 1$ gdzie $R_1 = U$ a $R_n = W$.

10.1.3 Język generowany przez gramatykę

$$L(G) = \{W \in V_T^* : S \overset{*}{\Rightarrow} W\}$$

10.1.4 Przykład prosty

$$L(G) = \{a^n : n \geq 0\} = L(a^*)$$

$$G = \langle \{S\}, \{a\}, S, \{S \to aS, S \to \epsilon\} \rangle$$

10.1.5 Przykład złożony

$$L(G) = \{a^n b^n : n \geq 1\} \notin \mathfrak{L}_{reg}$$

$$G = \langle \{S\}, \{a, b\}, S, \{S \rightarrow aSb, S \rightarrow ab\} \rangle$$