OPERACJE BINARNE

Na poprzednim wykładzie przyjęliśmy, że funkcja (pełna) jest taką relacją $f \subseteq A \times B$, że dla każdego $a \in A$ istnieje dokładnie jedna para $(a,b) \in f$, tj. $b = f(a)$. Zauważyliśmy również, że elementy zbioru A mogą być parami, trójkami, lub ogólnie n-tkami i wtedy mamy do czynienia z funkcją wielu zmiennych. Szczególnie często występującym przypadkiem jest odwzorowanie z A^2 w A.

Przykład 1 Niech $A = \mathcal{Z}$ oraz $a,b \in A$. Odwzorowania $(a,b) \rightarrow a+b$, $(a,b) \rightarrow a-b$, $(a,b) \rightarrow ab$ są przykładami funkcji $f : A^2 \rightarrow A$.

Funkcje takie jak powyższe określają sposób uzyskania z dwoch elementów pewnego zbioru elementu trzeciego i nazywane są *operacjami binarnymi*. W ogólności operacje nie muszą być definiowane na tym

samym zbiorze. Kombinacji mogą podlegać elementy dwóch różnych zbiorów, a jej wynikiem być element należacy do jeszcze innego zbioru.

Definicja 1 Niech A, B, C będą zbiorami. *Operacja binarna* na A i B jest odwzorowaniem $f : A \times B \to C$, tj. funkcją, która każdej parze $(a,b) \in A \times B$ przyporządkowuje pewną wartość $c \in C$. Operacja jest reprezentowana pewnym wybranym symbolem, jak np. $\circ$. Wówczas zamiast $f(a,b)$ piszemy $a \circ b$. Symbol $\circ$ jest nazywany *operatorem infiksowym*.

Na dzisiejszym wykładzie będą nas interesować operacje związane z pewnymi obiektami formalnymi takimi jak *litery, alfabety, słowa, języki*.

ALFABETY I SŁOWA

Definicja 2 *Alfabet* jest skończonym zbiorem. Elementy alfabetu są nazywane *literami*.

Najłatwiej wyobrazić sobie alfabet jako zbiór liter $a, b, c, \ldots$. Należy jednak pamiętać, że może to być dowolny zbiór, w tym taki gdzie litery są elementami złożonymi i posiadają określoną strukturę.

Z liter alfabetu A możemy tworzyć ciągi, zwane *słowami*, i określić nowy zbiór, który będzie zawierał wszystkie takie słowa.

Przykład 2 Zbiór słów, jakie można utworzyć z alfabetu $A = \{a\}$, wynosi $\{a,\ aa,\ aaa, \ldots\}$.

Przykład 3 Zbiór słów, jakie można utworzyć z alfabetu $A = \{a, b, c\}$, wynosi $\{a,\ b,\ c,\ aa,\ bb,\ cc,\ ab,\ ba,\ bc,\ cb,\ ac,\ ca,\ aaa,\ abc, \ldots\}$.

Inaczej mówiąc, słowa są n-tkami ze zbiorów A, A^2, A^3, ..., wobec czego ich zbiór możemy określić poprzez domknięcie zbioru A.

Definicja 3 Niech A będzie alfabetem. Elementy dodatniego domknięcia zbioru A

$$A^+ = A \cup A^2 \cup A^3 \cup \cdots = \bigcup_{n=1}^{\infty} A^n$$

są słowami określonymi nad alfabetem A. Długość słowa $w = a_1 a_2 \ldots a_n$ wynosi $|w| = n$, tj. jest równa liczbie elementów ciągu w.

Naturalną operacją określoną w zbiorze A^+ jest *konkatenacja*.

Definicja 4 Dla dowolnych słów $w_1 = a_1 a_2 \ldots a_p$ i $w_2 = b_1 b_2 \ldots b_q$ określonych na alfabecie A słowo $w_1 w_2 = a_1 a_2 \ldots a_p b_1 b_2 \ldots b_q$ jest nazywane *konkatenacją* słów w_1 i w_2.

Zbiór słów A^+ możemy rozszerzyć o słowo puste, oznaczane przez ε, którego długość wynosi 0.

Definicja 5 Zbiór $A^* = \{\varepsilon\} \cup A^+$ jest nazywany domknięciem alfabetu A ze względu na operację konkatenacji.

Slowo puste jest elementem neutralnym konkatenacji, tj. dla każdego słowa $w \in A^+$ zachodzi $\varepsilon w = w\varepsilon = w$. Definiując $A^0 = \{\varepsilon\}$ otrzymujemy również (podobnie jak w przypadku relacji), że

$$A^* = \bigcup_{n=0}^{\infty} A^n$$

PRZYKŁAD ZASTOSOWANIA ALFABETU - STOSY

Wprowadzona abstrakcja ciągów liter, lub słow, dostracza prostego modelu dla zdefiniowania obiektu zwanego stosem (znanego z wykładu z informatyki). Stos jest listą obiektów, do której dostęp jest ograniczony do wierzcholka stosu, tj. początku listy. Stosy są najczęściej wykorzystywane w sytuacji występowania ciągu zdarzeń wymagających obsługi, z których każde kolejne jest bardziej pilne niż poprzednie.

Przykład 4 Gdy kompilator czyta wyrażenie takie jak $a \times (b + c) - d$, generowany jest kod pozwalający na wykonanie powyższych operacji arytmetycznych. Po napotkaniu znaku $\times$ kompilator jest gotowy do wygenerowania instrukcji mnożenia. Jednakże napotyka on wówczas znak nawiasu '(', co powoduje, że operacja mnożenia zostaje zawieszona, a aktualne dane - kod mnożenia i pierwszy argument (lub adres do tych danych), zapamiętane na stosie. Kompilator przechodzi do wytworzenia

kodu dodawania, po czym odzyskuje dane ze stosu, kontynuuje generację kodu mnożenia i kończy kodem odejmowania.

Stos może być w prosty sposób przedstawiony jako ciąg liter, na którym określone są pewne operacje.

Definicja 6 Niech A będzie alfabetem. Na słowach określonych na A definiuje się następujące operacje.

- operację *szczyt* $T : A^+ \to A$, która dla $w = a_1 a_2 \ldots a_n$ zwraca $T(w) = a_n$
- operację *reszta* $R : A^+ \to A^*$, która dla $w = a_1 a_2 \ldots a_n$ zwraca $R(w) = a_1 a_2 \ldots a_{n-1}$
- operację *na_stos* (lub *push*) $S : A \times A^* \to A^+$, która dla $a = a_n$ i $w = a_1 a_2 \ldots a_{n-1}$ zwraca $S(a, w) = a_1 a_2 \ldots a_n$

Alfabet użyty w powyższych operacjach nazywa się *stosem*.

Dodatkowo można zdefiniować operację *ze_stosu* (lub *pop*)
$P : A^+ \to A \times A^*$, która zwraca dwa obiekty $P(w) = (T(w), R(w))$.

Przykład 5 Jeżeli $w = abc$ i $d \in A$, to $T(w) = c$, $R(w) = ab$ i
$S(d, w) = abcd$.

Operacje te mogą być kombinowane ze sobą na dowolną głębokość, przy czym istnieją pewne zawsze prawdziwe identyczności.

Twierdzenie 1 Dla wyżej zdefiniowanego stosu zawsze zachodzi:

$$\forall x \in A, \forall w \in A^*,\ R(S(x, w)) = w$$

$$\forall w \in A^+,\ S(T(w), R(w)) = w$$

Dowód wynika bezpośrednio z definicji.

JĘZYKI FORMALNE

Definicja 7 Niech A bedzie alfabetem. Podzbiór A^* jest nazywany *językiem* lub *językiem formalnym* określonym na alfabecie A.

Przykład 6

$$
\begin{aligned}
A &= \{a, b.g\} \\
L_1 &= \{\varepsilon, a, abb\} \\
L_2 &= \{\text{wszystkie możliwe słowa o długości 3,} \\
&\qquad \text{które zaczynają się literą a}\} \\
L_3 &= \{\text{wszystkie możliwe słowa o skończonej długości,} \\
&\qquad \text{które zaczynają się literą a}\}
\end{aligned}
$$

Ponieważ słowa języka L można traktować jako litery alfabetu 'wyższego rzędu', te same operacje co dla alfabetu można definiować dla języków.

Definicja 8 Dla dwóch języków L_1 i L_2 określonych na alfabecie A ich *produkt* lub *konkatenacja* definiuje się jako

$$L_1L_2 = \{w : w = w_1w_2,\ w_1 \in L_1,\ w_2 \in L_2\}$$

Przykład 7 Jeżeli $A = \{a,b,c,\ldots,z,0,1,\ldots,9\}$ $L_1 = \{a,b,c,\ldots,z\}$ i $L_2 = A^*$ to L_1L_2 jest językiem zawierającym wszystkie słowa, które składają się z małej litery, po której następuje ciąg liter i cyfr.

Definicja 9 Dla języka L określonego na A, potęgi L są zdefiniowane jako: $L^0 = \{\varepsilon\}$, $L^1 = L$, $L^n = LL^{n-1}$, $n \geq 2$.

Przykład 8 Jeżeli $A = \{0,1,\ldots,9\}$ i $L = \{00,11,22,\ldots,99\}$ to

$$L^2 = \{0000,0011,0022,\ldots,9988,9999\}$$

Określenie potęgi pozwala nam także zdefiniować domknięcia.

Definicja 10 Niech L będzie językiem określonym na alfabecie A. *Dodatnie domknięcie* i *domknięcie L* jest określone jako

$$L^+ = \bigcup_{n=1}^{\infty} L^n \qquad L^* = \bigcup_{n=0}^{\infty} L^n$$

Zauważmy, że $L^+ = LL^*$ oraz $A^+ = AA^*$.

Przykład 9 Jeżeli $A = \{0,1\}$ oraz $L = \{0,11\}$ to L^+ jest językiem zawierającym wszystkie możliwe słowa określone na A, które składają się z ciągu jednego lub więcej zer lub ciągu zer z dowolną liczbą 11-tek wstawionych w ten ciąg lub ciągu zawierającego parzystą liczbę jedynek większą od 0. Niektóre takie elementy to: 0, 00, 000, 0110, 11011000, 11, 1111, itd.

GRAMATYKI FORMALNE

Najważniejszym sposobem opisywania języków formalnych są gramatyki formalne. Opis w postaci gramatyki składa się z:

- Opisania, symbole jakiego alfabetu są używane przez język. Są to tzw. *symbole terminalne*.
- Wyboru dowolnego skończonego zbioru symboli pomocniczych, tzw. *symboli nieterminalnych*
- Wyboru z symboli nieterminalnych jednego *symbolu startowego*
- Wyboru pewnego skończonego zbioru *reguł przepisywania*, zwanych też *produkcjami*. Każda z reguł składa się z jednego słowa będącego lewą stroną reguły, oraz drugiego będącego prawą stroną. Reguły zapisuje się w postaci $\alpha \rightarrow \beta$.

Do tak opisanego języka należy każde słowo, dla którego potrafimy zbudować taki ciąg, że:

- pierwszym elementem ciągu jest słowo złożone z symbolu startowego
- każde następne słowo powstało przez:
  - wybranie dowolnego fragmentu poprzedniego słowa, które jest równe lewej stronie dowolnej reguły
  - i zamienienie go na prawą stronę tej samej reguły
- ostatnim elementem ciągu jest dane słowo.

Przykładowa gramatyka:

- alfabet składa się z 0 i 1
- symbole pomocnicze to A, B i C
- symbolem startowym jest A
- reguły przepisywania to:

$$A \to BC$$
$$A \to CB$$
$$B \to 0B0$$
$$C \to 1C1$$
$$B \to 0$$
$$C \to 1$$

Przykładowe wyprowadzenie słowa 00011111 w tej gramatyce:

$$
\begin{array}{ll}
A & \\
BC & (A \to BC) \\
0B0C & (B \to 0B0) \\
0B01C1 & (C \to 1C1) \\
0B011C11 & (C \to 1C1) \\
00011C11 & (B \to 0) \\
00011111 & (C \to 1)
\end{array}
$$

KLASY GRAMATYK

Czy gramatyki formalne są wystarczające do opisu wszystkich języków, które chcemy opisać?

Każda gramatyka formalna jest opisem skończonym. Gramatyk formalnych jest więc przeliczalnie wiele (pomijając możliwość różnego oznaczenia zbiorów symboli terminalnych oraz nieterminalnych), a zatem nie wszystkie języki formalne da się opisać za pomocą gramatyk formalnych.

Mając jednoznaczny opis języka, chcielibyśmy mieć też możliwie efektywną procedurę, która sprawdzałaby czy dane słowo należy do danego języka. Niestety, ogólne gramatyki formalne nie dostarczają nam takiej metody.

Niektóre opisywane przez nie języki będą tylko semirozstrzygalne – jeśli dane słowo należy do języka, w końcu znajdziemy jego wyprowadzenie, jednak nie istnieje ogólna metoda stwierdzania, czy wyprowadzenie nie istnieje, czy też po prostu jeszcze go nie znaleźliśmy – niektóre zaś z tych rozstrzygalnych będą miały zbyt dużą złożoność obliczeniową – metoda będzie istniała, ale będzie o wiele za wolna.

Dlatego istnieje wiele innych metod opisu języków, które dostarczają bardziej efektywnych metod testowania przynależności danego słowa. Metody te są jednak z natury rzeczy mniej ogólne od gramatyk formalnych, i generalnie czym lepszą mają złożoność, tym mniej języków potrafią opisać. Wśród gramatyk formalnych wydziela się pewne klasy języków, które można opisać za pomocą różnego typu *automatów*.

W szczególności rozważymy poniżej tzw. *automaty skończone*, pozwalające opisać *gramatyki regularne*.

DETERMINISTYCZNY AUTOMAT SKOŃCZONY

Deterministyczny automat skończony (ang. Deterministic Finite-state Automaton, DFA) to abstrakcyjna maszyna o skończonej liczbie stanów, która zaczynając w stanie początkowym czyta kolejne symbole pewnego słowa, po przeczytaniu każdego zmieniając swój stan na stan będący wartością funkcji jednego przeczytanego symbolu oraz stanu aktualnego. Jeśli po przeczytaniu całego słowa maszyna znajduje się w którymś ze stanów oznaczonych jako akceptujące (końcowe), słowo należy do języka regularnego, do rozpoznawania którego jest zbudowana.

Deterministyczny automat skończony, podobnie jak inne automaty skończone może być reprezentowany za pomocą tabeli przejść pomiędzy stanami lub diagramu stanów.

Przykład 10 Zbudujmy na przykład maszynę rozpoznającą takie słowa nad alfabetem binarnym, które są podzielne przez 5. Żeby zbudować tę

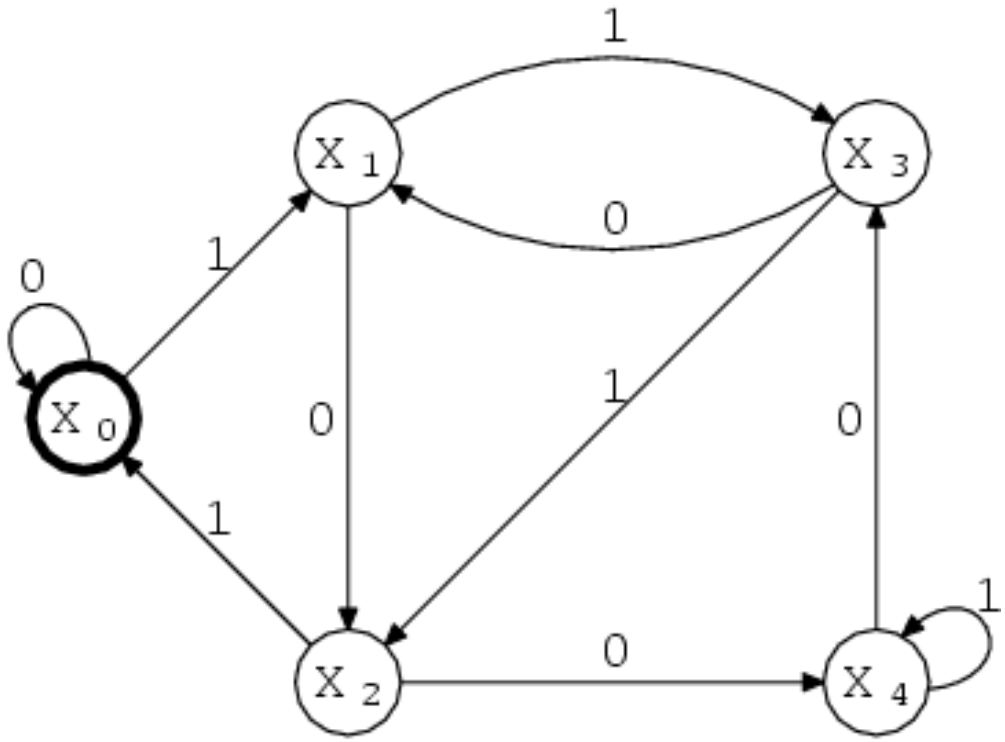
maszynę skorzystajmy z faktu, że:

$$w \cdot 0 = 2w, \qquad w \cdot 1 = 2w + 1$$

(wartość liczby to ostatnia cyfra plus dwa razy wartość liczby zbudowanej z pozostałych cyfr)

Czyli: $c_n \cdot c_{n-1} \cdots c_1 \cdot c_0 = c_0 + 2(c_1 + 2(\cdots(c_{n-1} + 2(c_n))\cdots))$

Ale jako że obchodzi nas nie wynik, a jedynie jego podzielność przez 5, możemy wykonywać obliczenia w arytmetyce modulo 5. Czyli zaczynamy od stanu x_0, i po przeczytaniu każdej cyfry c_i przechodzimy ze stanu x_j do stanu $x_{2j+c_i \bmod 5}$. Jeśli po przeczytaniu całego słowa jesteśmy w stanie x_0, oznacza to, że reszta z dzielenia słowa przez 5 wynosi 0, a więc słowo jest podzielne przez 5.



$x_0 \to^0 x_0, \quad x_0 \to^1 x_1, \quad x_1 \to^0 x_2, \quad x_1 \to^1 x_3, \quad x_2 \to^0 x_4,$
$x_2 \to^1 x_0, \quad x_3 \to^0 x_1, \quad x_3 \to^1 x_2, \quad x_4 \to^0 x_3, \quad x_4 \to^1 x_4$

stan startowy: x_0, stany akceptujące: tylko x_0

DFA - DEFINICJA FORMALNA

Definicja 11 Deterministyczny automat skończony jest piątką

$$(A, X, x_0, X_M, d)$$

gdzie:

- A jest alfabetem
- X jest zbiorem stanów
- x_0 jest wyróżnionym stanem początkowym należącym do Q
- $X_A \subseteq X$ jest zbiorem stanów akceptujących (końcowych)
- $d : X \times A \rightarrow X$ jest funkcją przejścia, przypisującą parze (x, a) nowy stan x', w którym znajdzie się automat po przeczytaniu symbolu a w stanie x.

Funkcja d może być częściowo określona (tj. mogą istnieć takie pary (x, a), dla których nie jest określony nowy stan x').

W powyższym przykładzie mamy:

$$A = \{0,1\}$$

$$X = \{x0, x1, x2, x3, x4\}$$

$$x_0 = x_0$$

$$X_A = \{x_0\}$$

d jest określona przez:

$$d(x_0,0) = x_0, \quad d(x_0,1) = x_1, \quad d(x_1,0) = x_2,$$

$$d(x_1,1) = x_3, \quad d(x_2,0) = x_4, \quad d(x_2,1) = x_0,$$

$$d(x_3,0) = x_1, \quad d(x_3,1) = x_2, \quad d(x_4,0) = x_3,$$

$$d(x_4,1) = x_4$$

JĘZYKI i GRAMATYKI REGULARNE

Język regularny to język formalny taki, że istnieje automat skończony potrafiący zdecydować czy dane słowo należy do języka.

Każdy język regularny można zapisać w postaci *gramatyki regularnej*, tj. takiej gramatyki, że po lewej stronie każdej reguły jest jeden symbol nieterminalny, a po prawej dowolna liczba symboli terminalnych, po których występuje co najwyżej jeden symbol nieterminalny.

Regułami gramatyki regularnej są więc na przykład:

$$A \to B$$
$$A \to xB$$
$$A \to x$$
$$A \to xyz$$
$$A \to xyzB$$
$$A \to \varepsilon$$

nie są zaś nimi na przykład:

$$A \rightarrow BC$$
$$AB \rightarrow CD$$

Zależności między językami regularnymi a gramatykami regularnymi są następujące:

- Każdy język regularny można zapisać za pomocą gramatyki regularnej
- Każda gramatyka regularna generuje pewien język regularny
- Język, który nie jest regularny, nie posiada gramatyki regularnej
- Gramatyka, która nie jest regularna, może generować język regularny, ale nie musi. Jeśli takowy generuje, ma on też inną gramatykę, która jest regularna.

Regularnymi są np. języki:

- zbiór wszystkich słów alfabetu 0,1
- zbiór wszystkich słów alfabetu 0,1 o długości n
- zbiór wszystkich słów alfabetu 0,1 o parzystej długości
- zbiór wszystkich słów alfabetu 0,1 zaczynających się od zera
- zbiór wszystkich słów alfabetu 0,1 nie zaczynających się od zera
- zbiór wszystkich słów alfabetu 0,1 w których na przemian występują zera i jedynki
- zbiór wszystkich słów alfabetu 0,1,2 w których na przemian występują zera, jedynki i dwójki