Teoretyczne podstawy informatyki

Andrzej M. Borzyszkowski

Instytut Informatyki
Uniwersytet Gdański

sem. letni 2025/26

inf.ug.edu.pl/~amb/TPI

Podsumowanie tematyki

Automaty i języki regularne

- alfabety, słowa i języki, operacje na słowach i na językach
  - konkatenacja słów/ języków, nieskończona iteracja (gwiazdka Kleene'ego)
- automat, pojęcie akceptowania (rozpoznawania) słowa przez automat
- automaty deterministyczne, niedeterministyczne i niedeterministyczne z ε-przejściami
- wszystkie definiują tę samą klasę języków
- również wyrażenia regularne
- przykłady języków regularnych
  - $\{\varepsilon\}$, zbiór skończony, $\{w \in \{0,1\}^* \mid |w|_0, |w|_1$ są parzyste$\}$
  - $\{w \in \{0,1\}^* \mid$ ostatnie 100 cyfr pasuje do podanego wzorca$\}$

Operacje na językach regularnych

- suma, konkatenacja, iteracja
- dopełnienie, przekrój, różnica, przeplot
- operacja podstawienia (języka za literę) i homomorfizmu (słowo za literę)
- klasa języków regularnych jest zamknięta na powyższe operacje, również przeciwobraz przy homomorfiźmie
- wyrażenia regularne są definiowane przez sumę, konkatenację i iterację (oraz podanie skończonego zbioru), i definiują tę samą klasę języków

Języki regularne i nieregularne

- lemat o pompowaniu: jeśli słowo jest dostatecznie długie, to można dowolnie pompować pewien jego fragment
- wniosek: pewne języki nie są regularne
- przykłady
  - $\{wcw \mid w \in \{a, b\}^*\}$
  - $\{wcw^R \mid w \in \{a, b\}^*\}$
  - $\{a^m b^n \mid m < n\}$
  - $\{a^m b^n \mid m > n\}$
  - $\{a^n b^n \mid n > 0\}$
  - $\{w \mid |w|_a = |w|_b\}$
  - $\{a^p \mid p$ jest liczbą pierwszą $\}$
- wyrażenia regularne w językach programowania
  czasami dodatkowe możliwości definicyjne
  również możliwość rozpoznania i zamiany wzorca

Twierdzenie Myhilla-Nerode'a

- dla języka $L \subseteq A^*$ definiujemy R_L jako
  $x \; R_L \; y \iff (\forall \, z \in A^*. \, xz \in L \iff yz \in L)$
- dla automatu deterministycznego $\mathcal{A}$ nad A^* definiujemy $R_{\mathcal{A}}$ jako
  $x \; R_{\mathcal{A}} \; y \iff (\delta(q_0, x) = \delta(q_0, y))$
- tw.: następujące warunki są równoważne
  1. język $L \subseteq A^*$ jest regularny
  2. relacja R_L ma skończony indeks
  3. L jest sumą pewnej liczby klas abstrakcji pewnej relacji równoważności na A^*, prawostronnie niezmienniczej, skończonego indeksu
- zastosowanie: dany język $L \subseteq A^*$, automat zbudowany z klas abstrakcji relacji R_L jest minimalnym automatem deterministycznym akceptującym L

Gramatyki bezkontekstowe

- gramatyka bezkontekstowa
  - V: zbiór symboli nieterminalnych (kategorie syntaktyczne)
  - $S \in V$: symbol początkowy
  - T: alfabet (symbole terminalne, końcowe)
  - produkcje, zapisywane jako $v \to w_1 \mid \ldots \mid w_n$, $w_i \in (V \cup T)^*$
- przykłady gramatyk
  - $W \to L \mid WOW$, $O \to + \mid - \mid * \mid \div$, $L \to 0 \mid \ldots \mid 9$
  - $S \to ab \mid aSb$
  - $S \to \varepsilon \mid aB \mid bA$, $A \to aS \mid bAA$, $B \to bS \mid aBB$
  - $W \to \emptyset \mid \varepsilon \mid I \mid WW \mid W + W \mid W^* \mid (W)$, $I \to a$, $a \in A$

Język generowany przez gramatykę

- wywody słowa: dla produkcji $A \to v$ mamy kroki $uAw \Rightarrow uvw$
- język (bezkontekstowy) jest definiowany przez gramatykę
  $L(G) = \{w \in T^* \mid S \overset{*}{\Rightarrow} w\}$
  - $W \to L \mid WOW$, $O \to + \mid - \mid * \mid \div$, $L \to 0 \mid \ldots \mid 9$ – wyrażenia arytmetyczne (liczby jednocyfrowe)
  - $S \to ab \mid aSb$ – $\{a^n b^n \mid n > 0\}$
  - $S \to \varepsilon \mid aB \mid bA$, $A \to aS \mid bAA$, $B \to bS \mid aBB$ – $\{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$
  - $W \to \emptyset \mid \varepsilon \mid I \mid WW \mid W + W \mid W^* \mid (W)$, $I \to a$, $a \in A$ – wyrażenia regularne
- wywód słowa można zapisać w postaci drzewa, drzewo jednoznacznie określa wywód np. lewostronny
- to samo słowo może mieć wiele drzew wywodu – niejednoznaczność
- powyższe gramatyki są niejednoznaczne

Operacje na gramatykach

- jednoznaczność: można skomplikować gramatykę by była jednoznaczna
  - wyrażenia arytmetyczne: pojęcie czynnika i składnika plus deklarowanie siły wiązania
  - wyrażenia regularne, boolowskie itd: podobne pomysły
  - ale istnieją języki bezkontekstowe nie posiadające gramatyki jednoznacznej
- usuwanie niepotrzebnych symboli nieterminalnych
- usuwanie ε (może tylko $S \to \varepsilon$)
- usuwanie produkcji jednostkowych $A \to B$
- postać normalna Chomksy'ego: produkcje $S \to \varepsilon$, $A \to a$, $a \in T$, $A \to BC$, gdzie $B, C \in V \setminus S$
- postać normalna Greibach: produkcje $S \to \varepsilon$, $A \to aA_1 \dots A_n$, gdzie $a \in T$, $A_1, \dots, A_n \in V \setminus S$

Lemat o pompowaniu dla języków bezkontekstowych

- jeśli L jest językiem bezkontekstowym, to istnieje stała n taka, że dla każde słowo $z \in L$ dłuższe niż n posiada rozkład $z = uvwxy$ taki, że
  - $|vwx| \leq n$
  - $|v| + |x| \geq 1$
  - $uv^\ell wx^\ell y \in L$ dla wszystkich $\ell \in \mathbb{N}$
- przykład zastosowania: nie są bezkontekstowe języki
  - $\{a^n b^n c^n \mid n \in \mathbb{N}\}$
  - $\{ww \mid w \in \{a, b\}^*\}$

Operacje na językach bezkontekstowych

- klasa języków bezkontekstowych jest zamknięta na sumę, konkatenację i iterację
- wniosek: każdy język regularny jest bezkontekstowy
- jeśli L i wszystkie języki L_a są bezkontekstowe, to podstawienie $subs(L)$ jest językiem bezkontekstowym
- podstawieniem jest m.in. suma, konkatenacja i iteracja
- obraz i przeciwobraz homomorficzny języka bezkontekstowego jest językiem bezkontekstowym
- klasa języków bezkontekstowych nie jest zamknięta na przekrój
  - $\{a^n b^n c^n \mid n \in \mathbb{N}\} = \{a^n b^n c^k \mid n, k \in \mathbb{N}\} \cap \{a^k b^n c^n \mid n, k \in \mathbb{N}\}$
- ani na dopełnienie: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

Automaty ze stosem

- automat ze stosem jest to automat (niedeterministyczny, z cichymi przejściami) zaopatrzony w dodatkową pamięć w postaci stosu, na który kładzione są pewne elementy (PDA – *push-down automata*)
- w każdym ruchu automat odczytuje symbol z taśmy wejściowej i pobiera symbol z wierzchu stosu, następnie zmienia stan i kładzie na stos słowo (ciąg symboli)
- może być również możliwy ruch bez czytania taśmy wejściowej, zależny tylko od wierzchołka stosu
- automat zaczyna działanie w stanie początkowym z pustym stosem
- dwie definicje akceptowania:
  1. stany akceptujące, jak dla zwykłych automatów
  2. akceptowane są słowa w momencie gdy stos jest pusty

Języki akceptowane przez automaty ze stosem

- konfiguracje i obliczenia definiujemy podobnie jak dla automatów niedeterministycznych: $\langle q, xw, X\beta\rangle \to \langle p, w, \alpha\beta\rangle$, $x \in A \cup \{\varepsilon\}$
  automat zmienił stan, została odczytana jedna litera (lub nie) ze słowa wejściowego, zdjęty jeden symbol ze stosu i zastąpiony słowem
- słowo jest zaakceptowane jeśli stan jest akceptujący albo jeśli stos jest pusty
- dla każdego automatu akceptującego przez stan istnieje równoważny automat akceptujący przez stos i na odwrót
- przykłady dla języków
  - $\{a^n b^n \mid n \in \mathbb{N}\}$
  - $\{wcw^R \mid w \in \{a, b\}^*\}$
  - $\{ww^R \mid w \in \{a, b\}^*\}$ – brak rozgranicznika, niedeterminizm

Gramatyki bezkontekstowe a automaty ze stosem

- klasa języków bezkontekstowych to klasa języków akceptowanych przez automaty ze stosem – (CFL=PDA)
- przekrój języków bezkontekstowych nie musi być bezkontekstowy ale przekrój jest bezkontekstowy jeśli jeden z języków jest regularny
- rozpoznanie słowa z języka regularnego wymaga przygotowania automatu o ustalonej z góry wielkości
- ale rozpoznanie słowa z języka bezkontekstowego wymaga użycia stosu o nieograniczonej z góry wielkości
  co może być podstawą udanego ataku DOS (*denial of service* – odmowa usługi)

Deterministyczne automaty ze stosem

- DPDA: dopuszczalne są ciche przejścia, poza tym są deterministyczne
- dwie definicje akceptowania:
  1. stany akceptujące, jak dla zwykłych automatów
  2. akceptowane są słowa w momencie gdy stos jest pusty
- akceptowanie przez pusty stos wymaga dodatkowo własności przedrostka: $w \in L$, $u \sqsubset w$ implikuje $u \notin L$
- $\{wcw^R \mid w \in \{a, b\}^*\}$ jest DPDA
- $\{ww^R \mid w \in \{a, b\}^*\}$ nie jest DPDA
- przekrój języka DPDA i języka regularnego jest klasy DPDA
- klasa DPDA jest zamknięta na dopełnienie
- nie jest domknięta na przekrój, sumę, konkatenację czy iterację
- nie jest zamknięta na podstawienie czy obraz homomorficzny
- jest zamknięte na operację przeciwobrazu homomorficznego

Problemy dotyczące języków

- dane języki L, L_1, $L_2 \subseteq A^*$, słowo $w \in A^*$, możemy postawić pytania
  - czy język $L \neq \emptyset$, jest niepusty?
  - czy język $|L| < \infty$, jest skończony?
  - czy $L_1 = L_2$, języki są równe?
  - ogólniej, czy $L_1 \subseteq L_2$?
  - czy $w \in L$, słowo należy do języka?
- algorytmy rozstrzygające
  - pustość: automat i problem istnienie ścieżki
  - skończoność: pytanie o pętlę w automacie lub w grafie zależności symboli nieterminalnych
  - należenie słowa: wykonanie przejścia w automacie, algorytm CYK dla gramatyk
  - zawieranie języków: $L_1 \subseteq L_2$ jest równoważne $L_1 \cap \overline{L_2} = \emptyset$, daje to algorytm jeśli L_2 jest językiem regularnym

Problemy nierozstrzygalne, maszyny Turinga

- nie ma algorytmów rozstrzygających problem $L_1 \subseteq L_2$ w pełnej ogólności
- nawet problem równości, nawet problem $L_1 = A^*$
- nie ma algorytmu rozstrzygającego problem czy dopełnienie języka jest skończone, czy jest językiem bezkontekstowym

- pojęcie maszyny Turinga obejmuje wszystkie możliwe algorytmy
- automat z nieskończoną taśmą wejściową
- głowica automatu może poruszać się w obu kierunkach, nie tylko czytać po kolei
- automat może zapisywać litery na taśmie, zarówno nadpisywać istniejące jak i zapisywać miejsca dotychczas puste

Język maszyny Turinga

- obliczenie jest akceptujące jeśli jest skończone i ostatnia konfiguracja jest akceptująca: $L(\mathcal{A}) = \{w \in A^* \mid \mathcal{A} \text{ akceptuje } w\}$
  język rekurencyjnie przeliczalny
- $w \in L(\mathcal{A})$ oznacza, że maszyna Turinga zatrzyma swoje obliczenia i ujawni pozytywny fakt
- $w \notin L(\mathcal{A})$
  - może być zauważone po zatrzymaniu się maszyny Turinga w stanie nieakceptującym
  - ale może się okazać, że obliczenia maszyny Turinga ze słowem wejściowym w są nieskończone
- język L jest rekurencyjny/ rozstrzygalny/ obliczalny, jeśli
  - $L = L(\mathcal{A})$ dla pewnej maszyny Turinga
  - obliczenia tej maszyny są skończone dla każdego słowa $w \in A^*$
- tzn. podając słowo w na taśmie wejściowej czekamy na zatrzymanie się obliczeń, by rozstrzygnąć czy $w \in L(\mathcal{A})$ czy też nie

Języki rekurencyjne

- istnieją języki rekurencyjnie przeliczalne ale nie rekurencyjne
- tzn. istnieją maszyny Turinga nie gwarantujące własności stopu
- dopełnienie języka rekurencyjnego jest rekurencyjne
- dopełnienie języka rekurencyjnie przeliczalnego nie musi być rekurencyjnie przeliczalne
- jeśli język i jego dopełnienie są rekurencyjnie przeliczalne, to oba są rekurencyjne
  - uruchamiamy dwie maszyny Turinga, dla języka i dla dopełnienia, jedna z nich musi się zatrzymać rozstrzygając problem należenia słowa
    nie ma obawy nieskończonego obliczenia

Maszyny Turinga

- maszyna Turinga może być wyposażona w więcej niż jedną taśmę, inne taśmy mogą być dostępne tylko do czytania, tylko do przesuwania się wprzód – nie zmienia to klasy języków
- automat ze stosem można zaimplementować jako maszyna Turinga
- maszynę Turinga można zaimplementować jako automat z dwoma stosami
- automat maszyny Turinga może być deterministyczny lub niedeterministyczny – nie zmienia to klasy języków

Obliczalność – Teza Church'a-Turinga

- funkcja $f : A^* \to \Sigma^*$ jest *obliczalna* jeśli istnieje (deterministyczna) maszyna Turinga $\mathcal{A}$ z taśmę wejściową i wyjściową, taka, że
  - jeśli na wejściu znajduje się $w \in A^*$, to maszyna zatrzyma swoje działanie
  - i na wyjściu znajdzie się słowo $f(w)$
- teza: wszystkie pojęcia obliczalności są równoważne

Złożoność obliczeniowa czasowa

- deterministyczna maszyna Turinga działa w czasie $f(n)$, $f : \mathbb{N} \to \mathbb{N}$, jeśli dla dowolnego słowa w długości n, zaczynając działanie w konfiguracji początkowej, kończy po najwyżej $f(n)$ krokach
  - w szczególności ma własność stopu
- język należy do klasy $TIME(f(n))$ jeśli jest akceptowany przez maszynę Turinga działającą w czasie $f(n)$
- dodajemy literkę N do oznaczenia klasy, jeśli maszyna akceptująca jest niedeterministyczna
  - w tym kontekście nie żądamy rozstrzygnięcia tzn. gwarancji stopu maszyny Turinga

Czas działania	deterministyczna	niedeterministyczna
logarytm $O(\log(n))$	L	NL
wielomian $O(p(n))$	P	NP
wykładniczy $2^{p(n)}$	EXPTIME	NEXPTIME

Problemy *NP*-zupełne

- dla każdej maszyny niedeterministycznej da się zbudować maszynę deterministyczną akceptującą ten sam język kosztem złożoności
- problem SAT: wejście: formuła boolowska w postaci koniunkcyjnej spełnialność = istnienie wartościowania zdań atomowych dającego *True*
- maszyna niedeterministyczna zaczyna od wyboru wartościowania, sprawdzenie wartości formuły jest liniowe, problem *NP*
- dowolny problem z klasy *NP* można zredukować wielomianowo do problemu SAT (bez dowodu) – problem SAT jest *NP*-zupełny
- inne problemy *NP*-zupełne
  - problem 3SAT
  - problem kliki w grafie
  - problem pokrycia wierzchołkowego
  - k-kolorowalność grafu
  - istnienie ścieżki Hamiltona

Uniwersalna maszyna Turinga

- uniwersalna maszyna Turinga M_u dla wejścia $\langle M, w\rangle$ gdzie $\langle M\rangle$ jest kodem maszyny M, a $w \in \Sigma^*$, symuluje obliczenie M na słowie w
- maszyna M_u akceptuje parę $\langle M, w\rangle$ wtedy i tylko wtedy, gdy maszyna M akceptuje w
- uniwersalna maszyna Turinga to w zasadzie komputer, mamy zapisany program M, dane w i wykonujemy obliczenia, które może przyniosą wynik a może okażą się nieskończone
- język uniwersalny $L_u = \{\langle M, w\rangle \mid M \text{ akceptuje } w\}$ jest rekurencyjnie przeliczalny
- ale nie jest rekurencyjny, gdyby był istniałaby maszyna rozpoznająca język $\{\langle M_i, w_i\rangle \mid M_i \text{ akceptuje } w_i\}$ (numerowane maszyny i słowa) czyli również język $L_{\mathcal{D}} = \{w_i \mid w_i \notin L(M_i)\}$
- ale ten język nie może być rozpoznany przez żadna maszynę M_k

Rozstrzygalność

- problem jest rozstrzygalny gdy istnieje (deterministyczna) maszyna Turinga zatrzymująca się dla każdej instancji i akceptująca te instancje, dla których odpowiedź jest pozytywna
- wszystkie algorytmy dowodzą rozstrzygalności problemów
- języki nierekurencyjne są przykładami problemów nierozstrzygalnych
- główny przykład: problem stopu maszyny Turinga
- i wspomniane wcześniej problemy dotyczące języków bezkontekstowych