

-- DON'T PANIC! --

Abgeschlossenheitseigenschaften regulärer Sprachen

L, L' reguläre Sprachen \Rightarrow folgende Sprachen auch regulär:

$L \cup L', L^*, L \cdot L'$: nach Def.

$\bar{L} : \Sigma^* \setminus L$: Betr. DEA $A=(Q, \Sigma, \delta, s, F)$ mit $L(A)=L$. Sei

$\bar{A}=(Q, \Sigma, \delta, s, Q \setminus F)$. Dann gilt $L(\bar{A})=\bar{L}$.

$L \cap L' = \overline{\overline{L \cup L'}}$

$L \setminus L' = L \cap \bar{L}'$

L^R : (Spiegelung)

Entscheidungsprobleme

Typ	Wort-	Leerheits-	Äquivalenz	Schnitt
3	✓	✓	✓	✓
Det. KF	✓	✓	✓ [97]	✗
2	✓	✓	✗	✗
1	✓	✗	✗	✗
0	✗	✗	✗	✗

Abschlusseigenschaften

Typ	n	u	-	.	*
3	✓	✓	✓	✓	✓
Det. KF	✗	✗	✓	✗	✗
2	✗	✓	✗	✓	✓
1	✓	✓	✓	✓	✓
0	✓	✓	✗	✓	✓

Komplexität

Typ	Beschreibungsmittel
3	$O(n)$
Det. KF	$O(n)$
2	$O(n^3)$
1	$ \Sigma ^{O(n)}$, NP-hart
0	semientscheidbar

Gödelnummer

Kodiere $\delta(q, a)=(r, b, d)$ durch $0^r 10^{a+1} 10^b 10^{d+1} 10^d$, wobei $N=1, L=2, R=3$

Codierung: 111 code₁ 11 code₂ 11...11 code_z 111

Notationen

Grammatik $G=(V, \Sigma, P, S)$, Turingmaschine $T=(Q, \Sigma, \Gamma, \delta, s, F)$,

Endlicher Automat $A=(Q, \Sigma, \delta, s, F)$, Det. Kellerautomat

$K=(Q, \Sigma, \Gamma, \delta, s, \#, F)$, Nichtdet. Kellerautomat $K=(Q, \Sigma, \Gamma, \delta, s, \#)$

NLBTM

Linear beschränkte nichtdeterministische Turingmaschine

NTM $T=(Q, \Sigma, \Gamma, \delta, s, F)$ ist linear beschränkt, wenn

$\forall a=a_1 \dots a_n \in \Sigma^+ : a \rightarrow \alpha(q)\beta \rightarrow |\alpha\beta| \leq n$

Übergangsfunktionen

DEA: $\delta: Q \times \Sigma \rightarrow Q$

NEA: $\delta: Q \times (\Sigma \cup \epsilon) \rightarrow 2^Q$

DTM: $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, N\}$

NTM: $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R, N\}}$

Nichtd. Kellerautomat: $\delta: Q \times (\Sigma \cup \epsilon) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

Det. Kellerautomat: $\delta: Q \times (\Sigma \cup \epsilon) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$, mit

$\forall q \in Q, a \in \Sigma, A \in \Gamma : |\delta(z, a, A)| + |z, \epsilon, A| \leq 1$

Pumping Lemma

Reguläre Sprachen:

L regulär $\Rightarrow \exists n \in \mathbb{N} : \forall w \in L, |w| > n : \exists u, v, x \in \Sigma^* :$

$w = uvx \wedge |v| \geq 1 \wedge |uv| \leq n \wedge \forall i \in \mathbb{N}_0 : uv^i x \in L$

Kontextfreie Sprachen:

L kontextfrei $\Rightarrow \exists n \in \mathbb{N} : \forall z \in L : |z| > n$

$\exists u, v, w, x, y : z = uvwx \wedge |vx| \geq 1 \wedge |vwx| \leq n \wedge \forall i \in \mathbb{N}_0 : uv^i wx^i y \in L$

Pumping Lemma (regulär) Beispiel

Ann.: $L = \{a^n b a^m b a^{n+m} | n, m \geq 1\}$ sei regulär.

Mit dem Pumping Lemma folgt dann: $\exists n \in \mathbb{N}$, sodass $\forall w \in L$ mit

$|w| > n$ gilt: \exists Zerlegung $w = uvx$ mit $v \neq \epsilon$, $|uv| \leq n$ und

$uv^i x \in L$ für alle $i \in \mathbb{N}$.

Betrachte $w = a^n b a^n b a^{2n}$. Dann ist $|w| > n$. Sei $w = uvx$ eine Zerlegung

gemäß dem Pumping-Lemma. Wegen $|uv| < n$ und $v \neq \epsilon$ ist $v = a^k$ für ein

k mit $1 \leq k \leq n$. Dann ist aber $uv^0 x = a^{n-k} b a^n b a^{2n} \notin L$, da

$n - k + n \neq 2n$. Widerspruch!

Chomsky Hierarchie

Typ	Name	Definition	Beschreibungsmittel
0	Typ 0	beliebig	Turingmaschine
1	kontextsensitiv	$ l \leq r $ Sonderregel: $S \rightarrow \epsilon$ erlaubt, aber dann $S \notin r$	NLBTM
2	kontextfrei	Typ 1 und $l \in V$	(1Zustands) NKellerA
Det. KF	LR(k) Grammatik		DkellerA mit Endz.
3	regulär	Typ 2 und $r \in \Sigma \cup \Sigma V$	DEA, $\bar{\epsilon} NEA$, ϵNEA

Nerode Relation

Nerode Relation zur Sprache L :

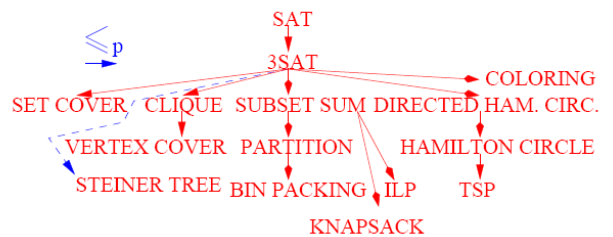
$R_L := \{(x, y) \in \Sigma^* \times \Sigma^* : \forall z \in \Sigma^* : xz \in L \Leftrightarrow yz \in L\}$

Äquivalenzklassenautomaten

Sprache regulär \Leftrightarrow index endlich

index = Anzahl Zustände im minimalen DEA = Anzahl der Äquivalenzklassen

Gesehene Reduktionen



Die Ackermannfunktion

Definition:

Function $a(x, y)$

if $x = 0$ then return $y+1$

if $y = 0$ then return $a(x-1, 1)$

return $a(x-1, a(x, y-1))$

Satz: a ist eine totale, TM-berechenbare Funktion

Monotonie der Ackermann Funktion

Lemma A: $y < a(x, y)$

Lemma B: $a(x, y) < a(x, y+1)$

Lemma C: $a(x, y+1) \leq a(x+1, y)$

Lemma D: $a(x, y) < a(x+1, y)$

Lemma BD: $a(x, y) \leq a(x', y')$, falls $x \leq x'$ und $y \leq y'$

-- DON'T PANIC! --

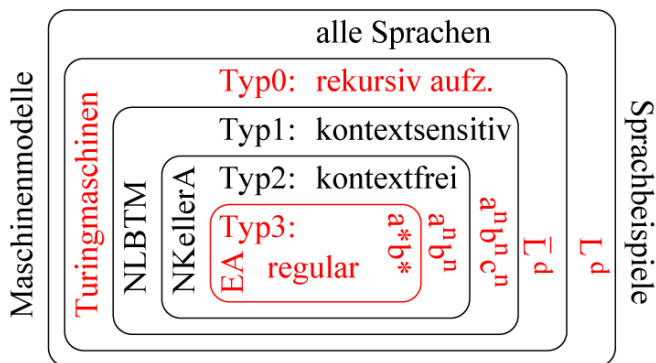
Grammatiken

Chomsky Normalform:

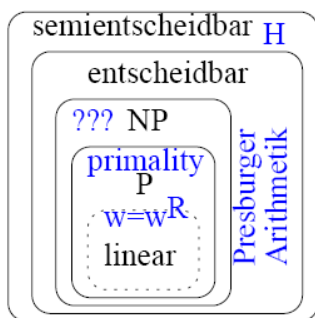
Vorbereitung: **Eliminierung von ϵ -Produktionen!**

- 1.) **Zyklische Einheitsproduktionen** entfernen
- 2.) **Nichtzykl. Einheitsprod.** Entfernen
- 3.) **gemischte rechte Seiten** entfernen
- 4.) **Elimination langer rechter Seiten**

Chomsky Hierarchie



Eine Komplexitätshierarchie



Komplexitätstheorie

Allgemeines:

- $A \leq_p B \wedge B \in P \Rightarrow A \in P$
- $A \leq_p B \wedge A \in NPV \Rightarrow B \text{ np-hart}$
- $A \leq_p B \wedge B \in NPV \Rightarrow A \in NP$
- $A \leq_p B \wedge A \in P \Rightarrow B \neq \emptyset \wedge B \neq \Sigma^*$

Beweistechniken $A \leq_p B$:

- Spezialfall:** A ist Spezialfall von B
- Gadger:** Jedes elementare Objekt in A wird auf mehrere Objekte in B abgebildet
- Randbedingungen:** Eigenschaften durch zusätzliche Konstrukte erzwingen

Polynomielle Reduzierbarkeit

- $A \leq_p B$
- $f_p: \Sigma^* \rightarrow \Sigma^*$
- $w \in A \Leftrightarrow f(w) \in B$

Approximationsalgorithmen

Aproximationsfaktor eines Minimierungsalgorithmus (Eingabe I, dazugeh. Lösung

$$x(I), x^*(I) \text{ optimale Lösung für Eingabe I: } \frac{f(x(I))}{f(x^*(I))} \leq \rho$$

Minimierung von Automaten

Zwei Automaten A, B äquivalent \Leftrightarrow Minimalautomat von A = Minimalautomat von B

Vorbereitung: **Eliminierung nicht erreichbarer Zustände!**

- 1.) oben nach unten: q_1, \dots, q_e
- links nach rechts: q_0, \dots, q_{e-1}
- 2.) Markierung aller (**Endzustand, nicht Endzustand**) - Paare
- 3.) Betrachtung der restlichen Zustandspaare: **markiertes Paar erreichbar** \Rightarrow markieren!
- 4.) Am Schluss: **nicht markierte** Zustandspaare zusammenfassen

DEA -> regulärer Ausdruck

$$R_{ij}^{m+1} = R_{ij}^m \cup R_{i, m+1}^m \cdot (R_{m+1, m+1}^m)^* \cdot R_{m+1, j}^m$$

While Programm

```

 $\mathbb{N}_0$  Name ( $\mathbb{N}_0 x_1, \dots, \mathbb{N}_0 x_k$ ) {
     $\mathbb{N}_0 x_0 = 0; \mathbb{N}_0 x_{k+1} = 0, \dots, \mathbb{N}_0 x_n = 0;$ 
    Body;
    return  $x_0$  }
    
```

Dabei sind Name ein geeigneter Bezeichner, Body die auszuführenden Anweisungen, $k \leq n$ zwei geeignete natürliche Zahlen

Primitive Anweisungen:

- $x_i := x_j$ (Zuweisung)
- x_i++ (Inkrement)
- x_i-- mit $0-- := 0$ (Dekrement)

Zusammengesetzte Anweisungen:

- $P; Q$ (Sequenz, wobei P und Q Anweisungen sind)
- while**($x_i \neq 0$) **do** P **end** (Schleife, wobei P eine Anweisung ist)

Beispiel:

```

main( $x_1, x_2$ );
{
  nat  $x_0$ ;
   $x_0 := x_1$ ;
  while  $x_2 \neq 0$  do {  $x_0 := x_0 + 1; x_2 := x_2 - 1$  }
  return  $x_0$  }
    
```

Loop Programm

```

 $\mathbb{N}$  main( $\mathbb{N} x_1, \dots, \mathbb{N} x_k$ ) {
   $\mathbb{N} x_0 = 0, \mathbb{N} x_{k+1} = 0; \mathbb{N} x_{k+2} = 0; \dots$ 
  Body; return  $x_0$ ; }
    
```

Zuweisung: $x_j := x_j + c, c \in -1, 0, 1, \dots, 0-1 := 0$

loop x_i : Schleife. Wiederhole x_i mal. Relevant ist der Inhalt von x_i vor der ersten Schleifenausführung.

Entscheidbarkeitsdinge

Rekursiv = entscheidbar
 rekursiv aufzählbar = semientscheidbar
 total rekursiv = berechenbar

primitiv rekursiv \Rightarrow total berechenbar

$L = L'$ entscheidbar \cap L'' semientscheidbar \Rightarrow L nicht entscheidbar

Fixpunktsatz

- 1.) Das Produkt zweier Wortmengen ist stetig
- 2.) Die Vereinigung zweier Mengen ist stetig
- 3.) Jede stetige Funktion ist monoton
- 4.) Jede stetige Funktion hat Fixpunkte