

Disclaimer

Aufgaben aus dieser Vorlage stammen aus der Vorlesung *Algorithmen, Sprachen und Komplexität* und wurden zu Übungszwecken verändert oder anders formuliert! Für die Korrektheit der Lösungen wird keine Gewähr gegeben.

1. Definitionen der Automatentheorie. Vervollständige die folgenden Definitionen:

- (a) Eine Regel $(l \rightarrow r)$ einer Grammatik $G = (V, \Sigma, P, S)$ heißt rechtslinear, falls ...

Solution: immer das an der am weitesten rechts stehende Nicht-Terminal in ein Terminal umgewandelt wird. Dazu muss $l \in V$ und $r \in \Sigma^* V \cup \epsilon$.

- (b) Die Menge $Reg(\Sigma)$ der regulären Ausdrücke über dem Alphabet ist...

Solution: ist die kleinste Menge mit folgenden Eigenschaften:

- $\emptyset \in Reg(\Sigma), \lambda \in Reg(\Sigma), \Sigma \subseteq Reg(\Sigma)$
- Wenn $\alpha, \beta \in Reg(\Sigma)$, dann auch $(\alpha * \beta), (\alpha + \beta), (\alpha^*) \in Reg(\Sigma)$

- (c) Ein NFA ist ein Tupel $M = (\dots)$

Solution: ein nichtdeterministischer endlicher Automat M ist ein 5-Tupel $M = (Z, \Sigma, S, \delta, E)$ mit

- Z ist eine endliche Menge von Zuständen
- Σ ist das Eingabealphabet
- $S \subseteq Z$ die Menge der Startzustände (können mehrere sein)
- $\delta : Z \times \Sigma \rightarrow P(Z)$ ist die (Menge der) Überführungs/Übergangsfunktion
- $E \subseteq Z$ die Menge der Endzustände

- (d) Die von einem NFA $M = (Z, \Sigma, S, \delta, E)$ akzeptierte Sprache ist $L(M) = \dots$ (ohne Definition der Mehr-Schritt Übergangsfunktion δ)

Solution: $L(M) = \{w \in \Sigma^* \mid \hat{\delta}(S, w) \cap E \neq \emptyset\}$

(Das Wort wird akzeptiert wenn es mindestens einen Pfad vom Anfangs in den Endzustand gibt)

- (e) Die von einem PDA $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ akzeptierten Sprache ist $L(M) = \dots$

Solution: $L(M) = \{x \in \Sigma^* \mid \text{es gibt } z \in Z \text{ mit } (z_0, x, \#) \xrightarrow{*} (z, \epsilon, \epsilon)\}$

- (f) Sei L eine Sprache. Für $x, y \in \Sigma^*$ gilt $x R_L y$ genau dann, wenn ... (R_L ist die Myhill-Nerode-Äquivalenz zu L)

Solution: wenn $\forall z \in \Sigma^* : (xy \in L \leftrightarrow yz \in L)$ gilt

- (g) Sei $M = (Z, \Sigma, z_0, \delta, E)$ ein DFA. Die Zustände $z, z' \in Z$ heißen erkenntnisäquivalent, wenn

Solution: Zwei Zustände $z, z' \in Z$ heißen erkenntnisäquivalent ($z \equiv z'$) wenn für jedes Wort $w \in \Sigma^*$ gilt: $\hat{\sigma}(z, w) \in E \leftrightarrow \hat{\sigma}(z', w) \in E$.

2. Sätze und Lemmas aus der Automatentheorie. Vervollständige die folgenden Aussagen:

- (a) Sei $L \subseteq \Sigma^*$ eine Sprache. Dann sind äquivalent: 1) L ist regulär (d.h. wird von einem DFA akzeptiert), 2) ..., 3) ...

Solution:

1. L ist regulär (d.h. von einem DFA akzeptiert)
2. L wird von einem NFA akzeptiert
3. L ist rechtslinear (d.h. von einer Typ-3 Grammatik erzeugt)

- (b) Die Klasse der regulären Sprachen ist unter anderem abgeschlossen unter folgenden drei Operationen:

Solution:

- Vereinigung $L = L_0 \cup L_1$
- Verkettung $L = L_0 L_1$
- Abschluss $L = L_0^*$

- (c) Sei Σ ein Alphabet. Die Anzahl der Grammatiken über Σ ist ... und die Anzahl der Sprachen über Σ ist

Solution:

- (d) Unter anderem sind folgende (mind. drei) Probleme für kontextfreie Sprachen entscheidbar:

Solution:

- (e) Die Klasse der Kontextfreien Sprachen ist abgeschlossen unter den Operationen 1)... und 2)... . Sie ist aber nicht abgeschlossen unter 3)... und 4)... .

Solution:

- (f) Der Satz von Myhill-Nerode besagt,...

Solution: Sei L eine Sprache. L ist regulär $\leftrightarrow \text{index}(R_L) < \infty$ (d.h. nur wenn die Myhill-Nerode-Äquivalenz endliche Klassen hat).

- (g) Das Pumping-Lemma für kontextfreie Sprachen ...

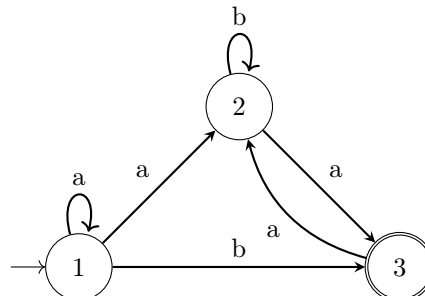
Solution: Man versucht auszunutzen, daß eine kontextfreie Sprache von einer Grammatik mit endlich vielen Nichtterminalen erzeugt werden muss. Das bedeutet auch: wenn ein Ableitungsbaum ausreichend tief ist, so gibt es einen Ast, der ein Nichtterminal mehrfach enthält. Die durch diese zwei Vorkommen bestimmten Teilbäume werden „gepumpt“.

Wenn L eine kontextfreie Sprache ist, dann gibt es $n \geq 1$ derart, dass für alle z in L mit $|z| \geq n$ gilt: es gibt Wörter u, v, w, x, y in Σ^* mit

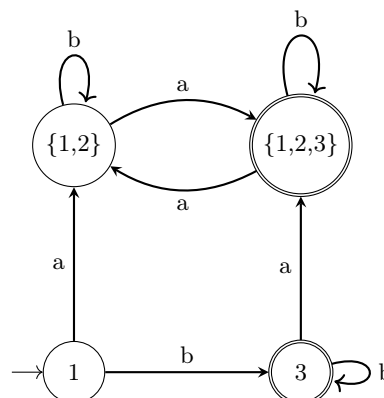
1. $z = uvwxy$,
2. $|vwx| \leq n$,
3. $|vx| \geq 1$ und
4. $uv^iwx^iy \in L$ für alle $i \geq 0$

3. Konstruktionen der Automatentheorie

- (a) Betrachte den folgenden NFA X . Berechne einen DFA Y mit $L(X) = L(Y)$.

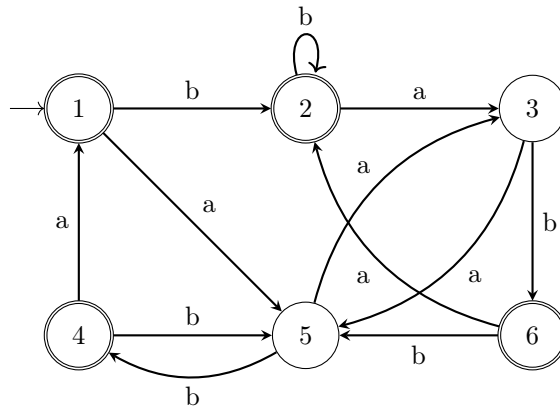


Solution:



- (b) Betrachte den folgenden NFA X . Berechne einen DFA Y mit $L(X) = L(Y)$.

(d) Betrachte den folgenden DFA X. Berechne den minimalen DFA Y mit $L(X) = L(Y)$.



Solution:

4. Algorithmen für reguläre Sprachen. Sei $\Sigma = \{a, b, c\}$. Gebe einen Algorithmus an, der bei Eingabe eines NFA X entscheidet, ob alle Wörter $\omega \in L(X)$ ungerade Längen besitzen und abc als Infix enthalten.

Solution:

5. Kontextfreie Sprachen: Sei $\Sigma = \{a, b, c\}$. Betrachte die Sprache $K = \{a^k b^l c^m \mid k \leq l \text{ oder } k \leq m\}$.

(a) Zeige, dass K eine kontextfreie Sprache ist.

Solution:

(b) Zeige, dass $L = \Sigma^* \setminus K$ (Komplement von L) nicht kontextfrei ist.

Solution:

(c) Begründe warum K deterministisch kontextfrei ist oder warum nicht.

Solution:

6. Kontextfreie Grammatiken: Sei $\Sigma = \{a, b, c, \}$

(a) Sei G die kontextfreie Grammatik mit Startsymbol S und der Regelmenge $S \rightarrow AB$, $A \rightarrow aBS \mid a$ und $B \rightarrow bBa \mid b$. Überführe G in eine äquivalente Grammatik in Chomsky Normalform.

Solution:

(b) Sei G' die kontextfreie Grammatik mit Startsymbol S und der Regelmenge $S \rightarrow AB$, $A \rightarrow CD \mid CF$, $F \rightarrow AD$, $B \rightarrow c \mid EB$, $C \rightarrow a$, $D \rightarrow b$, $E \rightarrow c$. Entscheide mit dem CYK-Algorithmus, ob die Wörter $w_1 = aaabbbcc$ oder $w_2 = aaabbccc$ von G' erzeugt werden.

Solution:

(c) Gebe für die Wörter aus b), die von G' erzeugt werden, den Ableitungsbaum an.

Solution:

7. Definitionen der Berechnbarkeitstheorie. Vervollständige die Definitionen

(a) Ein While Programm ist von der Form...

Solution:

(b) Eine Turingmaschine ist ein 7-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$, wobei...

Solution:

(c) Die von einer Turingmaschine M akzeptierte Sprache ist $L(M) = \dots$

Solution:

- (d) Seien $A \supseteq \Sigma^*$ und $B \supseteq \Gamma^*$. Eine Reduktion von A auf B ist ...

Solution:

- (e) Eine Sprache L heißt rekursiv aufzählbar, falls ...

Solution:

- (f) Sei $f : N \rightarrow N$ eine monotone Funktion. Die Klasse $TIME(f)$ besteht aus allen Sprachen L , für die es eine Turingmaschine M gibt mit ...

Solution:

8. Sätze der Berechnbarkeitstheorie: Vervollständige die folgenden Aussagen

- (a) Zu jeder Mehrband-Turingmaschine M gibt es ...

Solution:

- (b) Sei $f : N^k \rightarrow N$ eine Funktion für ein $k \in N$. Die folgenden Aussagen sind äquivalent: 1) f ist Turing-berechenbar, 2) ..., 3) ..., 4) ...

Solution:

- (c) Sei $L \subseteq \Sigma^*$ eine Sprache. Sind L und $\Sigma^* \setminus L$ semi-entscheidbar, dann...

Solution:

- (d) Der Satz von Rice lautet...

Solution: dass es unmöglich ist, eine beliebige nicht-triviale Eigenschaft der erzeugten Funktion einer Turing-Maschine (oder eines Algorithmus in einem anderen Berechenbarkeitsmodell) algorithmisch zu entscheiden.

Es sei \mathcal{P} die Menge aller partiellen Turing-berechenbaren Funktionen und $\mathcal{S} \subsetneq \mathcal{P}$ eine nicht-leere, echte Teilmenge davon. Außerdem sei eine effektive Nummerierung vorausgesetzt, die einer natürlichen Zahl $n \in N$ die dadurch codierte Turing-Maschine M_n zuordnet. Dann ist die Menge $\mathcal{C}(\mathcal{S}) = \{n \mid \text{die von } M_n \text{ berechnete Funktion liegt in } \mathcal{S}\}$ nicht entscheidbar. „Sei U eine nicht-triviale Eigenschaft der partiellen berechenbaren Funktionen, dann ist die Sprache $L_U = \{ \langle M \rangle \mid M \text{ berechnet } f \in U \}$ nicht entscheidbar.“

9. Berechnungsmodelle

- (a) Gebe ein Loop-Programm an, das die Funktion $n \rightarrow n^2 - n$ berechnet

Solution:

- (b) Gebe ein Loop Programm an, das die Funktion $f : N \rightarrow N$ mit $f(n_1, n_2) = 2n_1n_2$ berechnet. Verwende nur elementare Anweisungen und keine Abkürzungen.

Solution:

- (c) Gebe ein GoTo Programm an, das die Funktion $g : N \rightarrow N$ mit $g(n_1, n_2) = |n_1 - n_2|$ berechnet. Verwende nur elementare Anweisungen und keine Abkürzungen.

Solution:

- (d) Gebe eine deterministische Turingmaschine M für das Eingabealphabet $\{0, 1\}$ an, das folgende Funktion berechnet: Für Eingabe $a_1a_2...a_{n-1}a_n$ berechnet M die Ausgabe $a_na_1...a_{n-1}$ (letzte Symbol der Eingabe an erste Stelle).

Solution:

10. Reduktionen

- (a) Seien $A, L \subseteq \Sigma^*$ nichtleere Sprachen und A entscheidbar. Gebe eine Reduktion von $L \cup A$ auf L an.

Solution:

- (b) Gebe eine Bedingung für A an, sodass $L \cup A \leq_p L$ für alle nichtleeren Sprachen $L \subseteq \Sigma^*$ gilt. Begründe.

Solution:

11. Komplexitätsklassen. Ergänze zu den Paaren von Komplexitätsklassen das Relationssymbol zur Teilmengenbeziehung.

(a) EXPSPACE ? EXPTIME

Solution: $\text{EXPSPACE} \geq \text{EXPTIME}$

(b) NP ? P

Solution: $\text{NP} \geq \text{P}$

(c) NP ? NPSPACE

Solution: $\text{NP} \leq \text{NPSPACE}$

(d) NPSPACE ? PSPACE

Solution: $\text{NPSPACE} = \text{PSPACE}$

12. Unentscheidbare Probleme: Gebe (mind vier) unterscheidbare Probleme an (als Menge oder als Eingabe-Frage-Paar).

Solution:

13. NP-vollständiges Problem: Gebe (mind. zwei) NP-vollständige Probleme an (als Menge oder Eingabe-Frage-Paar).

Solution:

Hamilton Kreis

- Eingabe: $\text{Graph}(V, E)$
- Frage: Kann der Graph so durchlaufen werden, dass jeder Knoten genau ein mal besucht/abgelaufen wird?

14. Polynomialzeitreduktion: Betrachte das Problem 4C, also die Menge der ungerichteten Graphen die sich mit vier Farben färben lassen.

(a) Gebe eine Polynomialzeitreduktion von 3C auf 4C an.

Solution:

(b) Zeige, dass wenn $4C \in P$, dann gilt $P = NP$.

Solution: