

Lösungsvorschläge zu den Staatsexamina:
Theoretische Informatik und Algorithmik

Frühjahr 2023

Inhaltsverzeichnis

1	Thema	2
1.1	Teilaufgabe 1: Algorithmik	2
1.2	Teilaufgabe 2: Theoretische Informatik	2
1.2.1	Aufgabe 1: Reguläre Sprachen	2
1.2.2	Aufgabe 2: Regularität und Kontextfreiheit	4
1.2.3	Aufgabe 3: Entscheidbarkeit	6
1.2.4	Aufgabe 4: Komplexität	7
2	Thema	8
2.1	Teilaufgabe 1: Algorithmik	8
2.2	Teilaufgabe 2: Theoretische Informatik	8
2.2.1	Aufgabe 1: Reguläre Sprachen	8
2.2.2	Aufgabe 2: Kontextfreie Sprachen	10
2.2.3	Aufgabe 3: Entscheidbarkeit	12
2.2.4	Aufgabe 4: NP-Vollständigkeit	12
2.2.5	Aufgabe 5: Aussagen	12

1 Thema

1.1 Teilaufgabe 1: Algorithmik

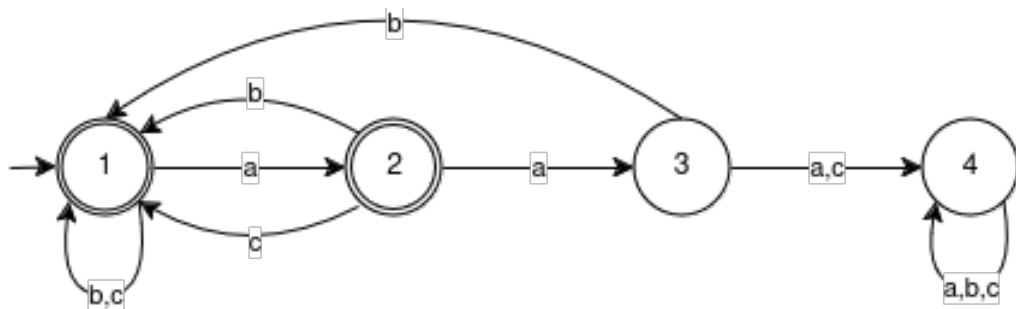
1.2 Teilaufgabe 2: Theoretische Informatik

1.2.1 Aufgabe 1: Reguläre Sprachen

(a) $DEA = (\{1, 2, 3\}, \Sigma, \delta, 1, \{1, 2\})$, wobei δ gegeben ist durch:

Zustand	a	b	c
1	2	1	1
2	3	1	1
3	-	1	1

Grafische Darstellung:



(b) Die Anzahl der Äquivalenzklassen entspricht der Anzahl an Zuständen, die der DEA aufweist, der diese Sprache erkennt. Unser DEA aus a) verfügt über drei Zustände, also hat L_1 vier Äquivalenzklassen. Wörter aus diesen Äquivalenzklassen müssen sich aus diesen Zuständen heraus bilden lassen:

Drei Wörter je Äquivalenzklasse:

1. bc, ccc, bbb
2. a, aca, aba
3. aa, aabaa, abaaa
4. aaa, aac, aaabc

(c) Übergangstabelle

Zustand	a-Übergang	b-Übergang	ϵ -Übergang
1	2	1	4
2	3	\emptyset	\emptyset
3	1	2	1
4	\emptyset	3	\emptyset

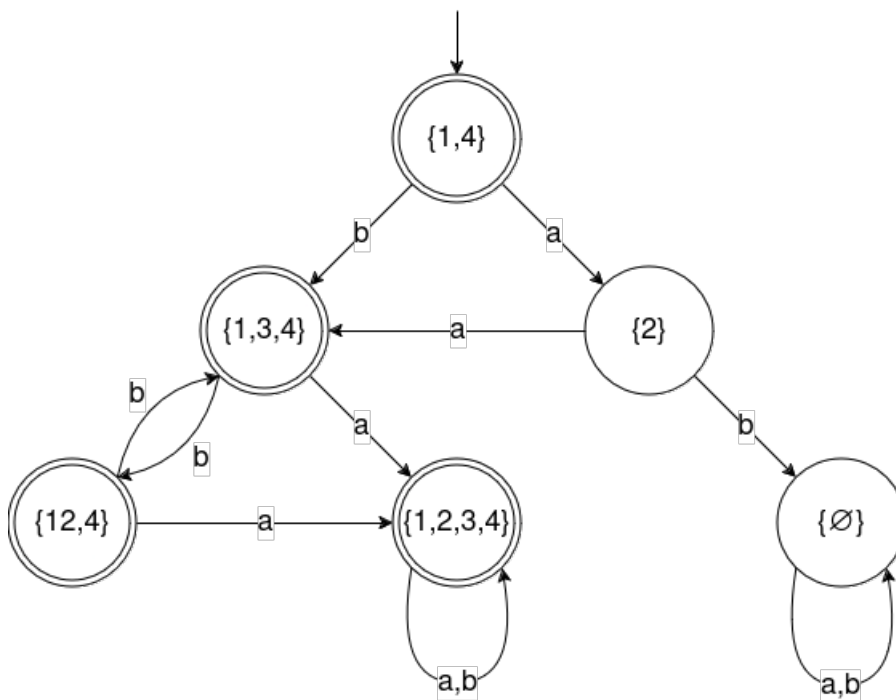
 ϵ -Funktionsabschlussstabelle

Zustand	Funktionsabschluss
1	$\{1,4\}$
2	$\{2\}$
3	$\{1,3,4\}$
4	$\{4\}$

Potenzmengenkonstruktionstabelle

Zustand	a-Übergang	b-Übergang	Endzustand?
$\{1,4\}$	$\{2\}$	$\{1,3,4\}$	Ja
$\{2\}$	$\{1,3,4\}$	$\{\emptyset\}$	
$\{1,3,4\}$	$\{1,2,4\}$	$\{1,2,3,4\}$	Ja
$\{1,2,4\}$	$\{1,2,3,4\}$	$\{1,3,4\}$	Ja
$\{1,2,3,4\}$	$\{1,2,3,4\}$	$\{1,2,3,4\}$	Ja

Fertiger DEA



1.2.2 Aufgabe 2: Regularität und Kontextfreiheit

(a) $L_1 = \{a^m b^n c^n \mid n, m \geq 1; n, m \in \mathbb{N}\} \cup \{b^m c^n \mid n \geq 0; n, m \in \mathbb{N}\}$

T3-Pumplemma

Sei L_1 regulär. Dann gilt:

$\exists p \geq 1 : \forall z \in L, |z| \geq p :$

$\exists z = uvw$ mit

1. $|v| \geq 1$

2. $|uv| \leq p$

$\forall i \in \mathbb{N} : uv^i w \in L_1$

Betrachte $z = a^1 b^{p-1} c^p$ mit $|z| \geq 2p$.

Aus den Voraussetzungen 1 und 2 folgt: u besteht aus einem a , v besteht ausschließlich aus b 's und $|v|_b \geq 1$. Setze $i = 2$. Dann gilt: $z = uv^2 w = a^1 b^{p-1+2} c^p$.

Aus $|v| \geq 1$ folgt dann aber $z \notin L_1$.

$\Rightarrow L_1$ ist nicht regulär.


Die erste Vereinigungsmenge von L_1 ist also nicht regulär. Von der zweiten könnten wir zeigen, dass sie regulär ist, indem wir einen DEA für sie angeben. Da aber die Vereinigung einer regulären Sprache mit einer kontextfreien Sprache niemals regulär sein kann, ist auch L_1 nicht regulär.

(b) L_2 ist regulär. Als Beweis nennen wir einen regulären Ausdruck, der L_2 erzeugt:

$abc(abcabc)^*$

Man beachte, dass die mehreren gleichnamigen Zeichenwiederholungsexponenten eine **fiese Blendgranate** darstellen, die die Aufgabenstellung wirft, um einen zu einem Nichtreguläritätsbeweis mittels Pumplemma oder Myhill-Nerode zu verlocken.

(c) Als **Beweis** geben wir eine kontextfreie Grammatik an, die L_3 erzeugt:

S \rightarrow ASE | T 
T \rightarrow BTD | c
AA \rightarrow aa
AB \rightarrow ab
BB \rightarrow bb
E \rightarrow e
D \rightarrow d

Erklärung:

Die Geradheit der Summe überprüfen wir mit den mittleren drei Ableitungsregeln mit Doppel-Nichtterminalen auf der linken Seite. Nur mit ihnen lässt sich ein Wort vollständig in Terminale ableiten. Damit wird die Geradheit also erzwungen.

Es genügt jedoch, die Geradheit der Summe aus n und m auf *einer* von beiden Seiten zu überprüfen, da beide Seiten **parallel** sind, da wir links gleich viele A's erzeugen wie E's rechts sowie gleich viele B's links wie D's rechts.

(d) 

1.2.3 Aufgabe 3: Entscheidbarkeit

- (a) Bei $L_1 - L_2$ handelt es sich um die Differenz beider Sprachen, also der Sprache, die alle Wörter aus L_1 enthält, abzüglich derer, die in L_2 enthalten sind.

Wir können uns eine deterministische Turingmaschine (DTM) bauen, die folgendermaßen funktioniert:

- Prüfe, ob das Wort in L_1 liegt. Das Wortproblem für reguläre Sprachen ist entscheidbar und kann von einer DTM in polynomieller Zeit beantwortet werden.
- Falls die erste Prüfung zutreffend war, prüfe weiter, ob das Wort in L_2 liegt. Auch das Wortproblem für kontextfreie Sprachen ist entscheidbar und kann von einer DTM in polynomieller Zeit beantwortet werden.
- Ergab Prüfung 1 wahr und Prüfung 2 falsch, ist das Wort in L . Auch das ist problemlos in polynomieller Zeit feststellbar.

$L_1 - L_2$ kann somit durch eine DTM berechnet werden, also liegt $L_1 - L_2$ in P.

- (b) $L \cap \bar{L}$ entspricht der leeren Sprache, denn zwischen einer Sprache und ihrem Komplement gibt es lediglich eine leere Schnittmenge. Die leere Menge ist eine reguläre Sprache, und für eine reguläre Sprache ist das Wortproblem entscheidbar.

Also ist $L \cap \bar{L}$ entscheidbar.

Dagegen ist $L \cup \bar{L}$ die Sprache *aller* Wörter, denn eine Sprache, die mit ihrem Komplement vereinigt wird, muss alle denkbaren Wörter enthalten. $L \cup \bar{L}$ bildet damit das Komplement zur leeren Sprache $L \cap \bar{L}$, von der wir bereits gezeigt haben, dass es entscheidbar ist.

Also ist auch $L \cup \bar{L}$ entscheidbar.

- (c) Für semi-entscheidbare Sprachen gilt, dass nur entscheidbar ist, ob ein Wort in der Sprache liegt, nicht aber, ob es *nicht* in der Sprache liegt.

Wenn L semi-entscheidbar ist, und ebenso \bar{L} , dann ist auch für **die Nein-Instanzen** von L entscheidbar, ob sie nicht in der Sprache L liegen, denn dies sind genau diejenigen Wörter, für die in \bar{L} entschieden werden kann, dass sie in \bar{L} liegen.

Also ist L entscheidbar.

- (d) Dadurch, dass L in NP liegt, gibt es eine NTM M , die L entscheidet, und damit auch eine **zu** DTM M' , die die gleiche Sprache erkennt wie M , und damit L entscheidet. Damit ist L entscheidbar.

1.2.4 Aufgabe 4: Komplexität



2 Thema

2.1 Teilaufgabe 1: Algorithmik

2.2 Teilaufgabe 2: Theoretische Informatik

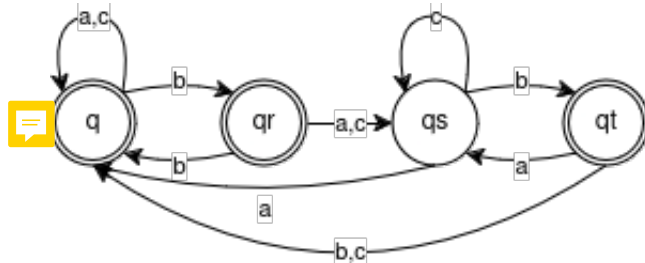
2.2.1 Aufgabe 1: Reguläre Sprachen

(a) $(a|b|c)^*b(a|c)(c|ba)^*$

(b) Schritt 1: Umwandlung des NEA zum DEA mittels Potenzmengenkonstruktion:

Zustand	Eingabe a	Eingabe b	Eingabe c	Anmerkung
q	q	qr	q	Ergänze Zustand qr.
qr	qs	q	qs	Ergänze Zustand qs.
qs	q	qt	qs	Endzustand. Ergänze Zustand qt.
qt	qs	q	q	

Der entstandenen Tabelle können wir aus der ersten Spalte die für den DEA benötigten Zustände entnehmen, und haben darin außerdem angemerkt, welche der Zustände im DEA, der L erkennen soll, Endzustände sein müssen. Unser neuer DEA soll aber das Komplement von L erkennen. Um das zu erreichen, müssen wir Endzustände und Nicht-Endzustände vertauschen:



(c)

Zustand	q	r	s	t	x	y	z
Zeuge	1	11	ε	-	01	0	00

Der Zustand t ist nicht erreichbar, entfällt also im minimierten Automaten.

Tabellenfüllverfahren

z und q sind Endzustände. Zustandspaare, die einen (nicht zwei) von beiden Zuständen enthalten, dürfen wir bereits markieren:

q						
r	X0					
s	X0	X1				
x	X0	X2	X3			
y	X0		X4	X5		
z		X0	X0	X0	X0	
	q	r	s	x	y	z

Nebenrechnung

Zustandspaar	0	1	Erläuterung
(s,r)	(y,q)	(q,x)	Eingabe 0 führt mit (y,q) zu X0. Ergänze X1.
(x,r)	(r,q)	(z,x)	Eingabe 0 führt mit (r,q) zu X0. Ergänze X2.
(x,s)	(r,y)	(z,q)	Eingabe 1 führt mit (z,q) zu X0. Ergänze X3.
(y,s)	(z,y)	(x,q)	Eingabe 1 führt mit (z,q) zu X0. Ergänze X4.
(y,x)	(z,r)	(z,x)	Eingabe 1 führt mit (z,x) zu X0. Ergänze X5.
(z,q)	(x,x)	(y,r)	z und q sind äquivalent.
(y,r)	(z,q)	(x,x)	y und r sind äquivalent.

Die Zustände z und q sowie y und r sind äquivalent, da beide Eingaben jeweils in äquivalente Zustände führen. Damit bleiben für den minimierten Automaten folgende Äquivalenzklassen: $[z,q], [y,r], [s], [x]$

Der minimierte Automat A' lautet also:

$$A' = (\{[z, q], [y, r], [s], [x]\}, \{1, 2\}, \delta, s, \{q, z\})$$

2.2.2 Aufgabe 2: Kontextfreie Sprachen

(a) **Konstruktionsidee**

Im Kern beruht diese Grammatik auf der Idee, dass jedes Wort in L aus zwei Teilwörtern besteht, nämlich aus

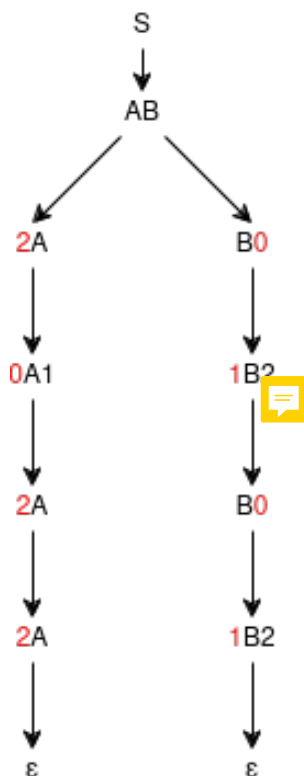
- einem linken Teilwort w , dass an seinem rechten Rand (also dem Mittelteil des Gesamtwortes) so viele 1en aufweist wie es selbst 0en enthält, wobei links und rechts aller 0en beliebig viele 2en zulässig sind, und analog dazu
- einem rechten Teilwort v , dass an seinem linken Rand (also dem Mittelteil des Gesamtwortes) so viele 1en aufweist wie es selbst 2en enthält, wobei links und rechts der 2en beliebig viele 0en zulässig sind.

Beide Teilwörter können beliebig weit aufgepumpt werden. Sie müssen nur beide sicherstellen, dass ihr Anteil an 1en übereinstimmt mit der Anzahl ihres mitzuzählenden Terminalzeichens (im linken Teilwort 0en und im rechten Teilwort 2en). Damit ist nämlich sichergestellt, dass im Gesamtwort gilt, dass $n = |w|_0 + |v|_2$, denn n entspricht der Summe der 1en in v und w .

In diesem Sinne konstruieren wir unsere **Grammatik** wie folgt:

$S \rightarrow AB \mid \epsilon$
 $A \rightarrow 0A1 \mid 2A \mid \epsilon$
 $B \rightarrow 1B2 \mid B0 \mid \epsilon$

(b)



(c) $L = \{w1^n v \mid n \in \mathbb{N}, w, v \in \{0, 2\}^*, n = |w|_0 \cdot |v|_2\}$

Nichtreguläritätsbeweis mit Myhill-Nerode

Sei L eine formale Sprache.

Die binäre Relation $\sim_L \subseteq \Sigma^*$ ist gegeben durch:

$x \sim_L y \Leftrightarrow \forall w \in \Sigma^* : xw \in L \Leftrightarrow yw \in L$.

Ist L regulär, dann ist der Index von \sim_{L_2} endlich.

Betrachte die Wörter 01^n und 01^k mit $k < n$.

Hängt man daran das Wort 2^n , gilt:

$01^n 2^n \in L \forall n \in \mathbb{N}$

$01^k 2^n \notin L \forall n \in \mathbb{N}$, denn $n \neq |w|_0 \cdot |v|_2$, da $k < n$.

Durch paarweise gleichmäßiges Erhöhen des Zeichenwiederholungsexponenten erhält man somit unendlich viele Wörter, die sich alle paarweise nicht in derselben Äquivalenzklasse befinden.

$\Rightarrow |\sim_L| = \infty$

$\Rightarrow L$ ist nicht regulär.

Nichtreguläritätsbeweis mittels Pumplemma für reguläre Sprachen

Sei L regulär. Dann gilt:

$\exists p \geq 1 :$

$\forall z \in L, |z| \geq p :$

$\exists z = uvw$ mit:

1. $|v| \geq 1$

2. $|uv| \leq p$

$\forall i \in \mathbb{N} : uv^i w \in L$

Betrachte $z = 0^p 1^p 2$

Aus 1 und 2 folgt: v besteht ausschließlich aus 0en und $|v|_0 \geq 1$.

Setze $i = 0 : z = uv^0 w = uw = 0^{p-x} 1^p 2$ mit $0 < x \leq p$.

Aus $|v| \geq 1$ folgt dann aber $z \notin L$.

$\Rightarrow L$ ist nicht regulär.

2.2.3 Aufgabe 3: Entscheidbarkeit

- (a) Hier soll nicht die Entscheidbarkeit widerlegt, sondern die Semientscheidbarkeit bewiesen werden. Daher sollten wir hier nicht *von einem Problem reduzieren*, sondern stattdessen *ein Entscheidungsverfahren zeigen*.

Als Beweis geben wir also einen Algorithmus an, der die Ja-Instanzen von L in endlicher Zeit entscheidet:

Gegeben M und w , konstruiere M' wie folgt:

```

1 for (n = 0; n <= 42; n++) {
2     Wähle nichtdeterministisch ein Wort  $w$  aus  $\Sigma^*$  mit
         $|w| = n$ .
3     Verhalte dich wie  $M$  auf  $w$ .
4     Falls  $M$  auf  $w$  hält, halte auch.
5     Sonst: ...
6 }
```


- (b) **Satz von Rice**

Sei E eine semantische, nicht-triviale Eigenschaft von DTMs. Dann ist das Problem „Gegeben eine DTM M , hat $L(M)$ Eigenschaft E ?“ unentscheidbar.

E bezieht sich hier auf die Eigenschaft der Sprache, dass sie Wörter der Länge $n \leq 42$ enthält. Sie ist semantisch, da sie die Sprache betrifft. Und sie ist nicht-trivial, da es DTMs gibt, die diese Eigenschaft haben, und welche, die sie nicht haben.

2.2.4 Aufgabe 4: NP-Vollständigkeit

2.2.5 Aufgabe 5: Aussagen

- (a) 
- (b) Richtig.
Reguläre Sprachen sind im Komplement abgeschlossen. Wenn L regulär ist, dann ist auch ihr Komplement L^* regulär. Insofern gilt auch die Rückrichtung. L ist also regulär.
- (c)
- (d) Falsch.
 L kann, muss aber nicht zwingend regulär sein. Das Wortproblem ist auch entscheidbar für kontextfreie und kontextsensitive Sprachen. Also könnte L genau so gut kontextfrei oder kontextsensitiv sein.