Frank Bergmann
Bockenheimer Landstasse 135 / 221

60325 Frankfurt / Main

bergmann@cs.uni-frankfurt.de

Theoretische Informatik

Frank Bergmann

13. Juli 2000

Inhaltsverzeichnis

1 Übungsblatt 1 **4**
1.1 Aufgabe 1 4
1.1.1 Aufgabenstellung 4
1.1.2 Lösung 4
1.2 Aufgabe 2 5
1.2.1 Aufgabenstellung 5
1.2.2 Lösung 5
1.3 Aufgabe 3 6
1.3.1 Aufgabenstellung 6
1.3.2 Lösung 6
1.4 Aufgabe 4 7
1.4.1 Aufgabenstellung 7
1.4.2 Lösung 7

2 Übungsblatt 2 **9**
2.1 Aufgabe 5 9
2.1.1 Aufgabenstellung 9
2.1.2 Lösung 9
2.2 Aufgabe 6 10
2.2.1 Aufgabenstellung 10
2.2.2 Lösung 10
2.3 Aufgabe 7 11
2.3.1 Aufgabenstellung 11
2.3.2 Lösung 11
2.4 Aufgabe 8 13
2.4.1 Aufgabenstellung 13
2.4.2 Lösung 13

3 Übungsblatt 3 **15**
3.1 Aufgabe 9 15
3.1.1 Aufgabenstellung 15
3.1.2 Lösung 15
3.2 Aufgabe 10 16
3.2.1 Aufgabenstellung 16
3.2.2 Lösung 16
3.3 Aufgabe 11 18
3.3.1 Aufgabenstellung 18

3.3.2 Lösung 18

4 Übungsblatt 4 **19**
4.1 Aufgabe 12 19
4.1.1 Aufgabenstellung 19
4.1.2 Lösung 20
4.2 Aufgabe 13 20
4.2.1 Aufgabenstellung 20
4.2.2 Lösung 21
4.3 Aufgabe 14 22
4.3.1 Aufgabenstellung 22
4.3.2 Lösung 22
4.4 Aufgabe 15 24
4.4.1 Aufgabenstellung 24
4.4.2 Lösung 24

5 Übungsblatt 5 **25**
5.1 Aufgabe 16 25
5.1.1 Aufgabenstellung 25
5.1.2 Lösung 25
5.2 Aufgabe 17 27
5.2.1 Aufgabenstellung 27
5.2.2 Lösung 27
5.3 Aufgabe 18 28
5.3.1 Aufgabenstellung 28
5.3.2 Lösung 29
5.4 Aufgabe 19 31
5.4.1 Aufgabenstellung 31
5.4.2 Lösung 31

6 Übungsblatt 6 **32**
6.1 Aufgabe 20 32
6.1.1 Aufgabenstellung 32
6.1.2 Lösung 32
6.2 Aufgabe 21 34
6.2.1 Aufgabenstellung 34
6.2.2 Lösung 34
6.3 Aufgabe 22 35
6.3.1 Aufgabenstellung 35
6.3.2 Lösung 35
6.4 Aufgabe 23 37
6.4.1 Aufgabenstellung 37
6.4.2 Lösung 37

7 Übungsblatt 7 **39**
7.1 Aufgabe 24 39
7.1.1 Aufgabenstellung 39
7.1.2 Lösung 40
7.2 Aufgabe 25 41
7.2.1 Aufgabenstellung 41

7.2.2 Lösung . . . 41
7.3 Aufgabe 26 . . . 43
7.3.1 Aufgabenstellung . . . 43
7.3.2 Lösung . . . 43

8 Übungsblatt 8 **44**
8.1 Aufgabe 27 . . . 44
8.1.1 Aufgabenstellung . . . 44
8.1.2 Lösung . . . 45
8.2 Aufgabe 28 . . . 45
8.2.1 Aufgabenstellung . . . 45
8.2.2 Lösung . . . 46
8.3 Aufgabe 29 . . . 47
8.3.1 Aufgabenstellung . . . 47
8.3.2 Nachtrag zur Vorlesung: Abschlusseigenschaften von Kontextfreien Sprachen . . . 47
8.3.3 Lösung . . . 48
8.4 Aufgabe 30 . . . 49
8.4.1 Aufgabenstellung . . . 49
8.4.2 Lösung . . . 49

1 Übungsblatt 1

1.1 Aufgabe 1

1.1.1 Aufgabenstellung

Seien $\{M_i, \circ_i\}$ algebraische Strukturen $(i = 1, 2)$und $h : M_1 \to M_2$ ein Homomorphismus. Beweisen Sie:

1. Ist $\{M_2, \circ_2\}$ assoziativ und h surjektiv, dann ist auch $\{M_1, \circ_1\}$ assoziativ.

2. Ist $\{M_1, \circ_1\}$ assoziativ und h injektiv, dann ist auch $\{M_2, \circ_2\}$ assoziativ.

1.1.2 Lösung

Zuerst sollte ich vielleicht nochmals festhalten, was *Injektivität* bzw. *Surjektivität* eigentlich ist. Eine Abbildung ist dann injektiv wenn gilt: zwei Elemente $a, b \in S_1$ und $h(a) = h(b)$ dann ist $a = b$. Eine Abbildung ist dann surjektiv, wenn gilt: Für alle Elemente $b \in S_2$existiert ein $a \in S_1$ der Art, dass $h(a) = b$

1. Gegeben sind die Strukturen $\{M_1, \circ_1\}$ und $\{M_2, \circ_2\}$, wobei die Struktur $\{M_2, \circ_2\}$ assotiativ ist. Desweiteren ist ein bijektiver Homomorphismus $h : M_1 \to M_2$gegeben.
   Zu Zeigen ist nun, dass $\{M_1, \circ_1\}$ assoziativ ist.
   Der Beweis erfolgt nun wie folgt:
   Seien $a_1, a_2, a_3 \in M_1$ dann gilt:

$$\begin{aligned} h(a_1 \circ_1 (a_2 \circ_1 a_3)) &= h(a_1) \circ_2 h(a_2 \circ_1 a_3) \\ &= h(a_1) \circ_2 (h(a_2) \circ_2 h(a_3)) \\ &= (h(a_1) \circ_2 h(a_2)) \circ_2 h(a_3) \\ &= (a_1 \circ_1 a_2) \circ_2 h(a_3) \\ &= h((a_1 \circ_1 a_2) \circ_1 a_3) \end{aligned}$$

Aus der Injektivität folgt also die Assoziativität. Und so gilt:

$$a_1 \circ_1 (a_2 \circ_1 a_3) = (a_1 \circ_1 a_2) \circ_1 a_3$$

2. Beim zweiten geforderten Beweis dieser Aufgabe sind wieder die beiden Strukturen $\{M_1, \circ_1\}$ und $\{M_2, \circ_2\}$gegeben. Diesmal ist $\{M_1, \circ_1\}$assoziativ und es soll gezeigt werden, dass der surjektive Homomorphismus $h: M_1 \to M_2$ausreicht, um zu zeigen, dass $\{M_2, \circ_2\}$ assoziativ ist.
Seien $b_1, b_2, b_3 \in M_2$, dann gibt es Elemente $a_1, a_2, a_3 \in M_1$der Art, dass $h(a_1) = b_1$, $h(a_2) = b_2$ und $h(a_3) = b_3$sind. Dann gilt:

$$\begin{aligned} b_1 \circ_2 (b_2 \circ_2 b_3) &= h(a_1) \circ_2 (h(a_2) \circ_2 h(a_3)) \\ &= h(a_1) \circ_2 (a_2 \circ_1 a_3) \\ &= h(a_1 \circ_1 (a_2 \circ_1 a_3)) \\ &= h(a_1 \circ_1 a_2) \circ_2 h(a_3) \\ &= (h(a_1) \circ_2 h(a_2)) \circ_2 h(a_3) \\ &= (b_1 \circ_2 b_2) \circ_2 b_3 \end{aligned}$$

1.2 Aufgabe 2

1.2.1 Aufgabenstellung

Seien Σ und $\triangle$ zwei Aplphabete. Beweisen Sie:

1. Zu jeder Abbildung $h: \Sigma \to \triangle^*$ gibt es genau einen Homomorphismus $\tilde{h}: \Sigma^* \to \triangle^*$mit $h(a) = \tilde{h}(a)$ für alle $a \in \Sigma$.

2. Zu jedem Homomorphismus $\tilde{h}: \Sigma^* \to \triangle^*$gibt es genau eine Abbildung $h: \Sigma \to \triangle^*$mit $h(a) = \tilde{h}(a)$ für alle $a \in \Sigma$.

1.2.2 Lösung

Zuerst noch zwei Hinweise zu den Symbolen. Σ und $\triangle$ stellen endliche Mengen von Wörtern dar. Währendessen ist Σ^* ein freies Monoid über Σ.

1. Wir habene eine Abbilgung $h: \Sigma \to \triangle^*$gegeben. Nun ist zu zeigen, dass es genau einen Homomorphismus $\tilde{h}: \Sigma^* \to \triangle^*$gibt, so dass $h(a) = \tilde{h}(a)$ für alle Ellemente $a \in \Sigma$ gilt. Es ist notwendig zum einen die *Existenz* und zum anderen die *Eindeutigkeit* des Homomorphismusses zu zeigen. Zuerst zeige ich die *Existenz*
Man definiert $\tilde{h}: \Sigma^* \to \triangle^*$durch $\tilde{h}(w) = h(a_1)\,h(a_2)\ldots h(a_n)$ für $w = a_1a_2\ldots a_n$ mit $a_i \in \Sigma$, $1 \leq i \leq n$. Zu zeigen ist nun, dass $\tilde{h}$ auch eine homomorphe Abbildung von $\Sigma^* \to \triangle^*$ist.
Für alle $u, v \in \Sigma^*$(also $u = a_1 \ldots a_n$ und $v = b_1 \ldots b_n$ für $a_i, b_i \in \Sigma$, $1 \leq i \leq n$) gilt:
$$h(uv) = h(u)\,h(v)$$
damit folgt unittelbar:
$$\begin{aligned} \tilde{h}(uv) &= (h(a_1)\,h(a_2)\ldots h(a_n))\,(h(b_1)\,h(b_2)\ldots h(b_n)) \\ &= \tilde{h}(u)\,\tilde{h}(v) \end{aligned}$$

2. Es bleibt also noch die *Eindeutigkeit* zu zeigen. Seien $\overline{h}$ und $\tilde{h}$ zwei Homomorphismen von $\Sigma^* \rightarrow \triangle^*$mit:

$$\overline{h}(a) = \tilde{h}(a) = h(a), \forall a \in \Sigma$$

Dann gilt für ein beliebiges Wort $w = a_1 a_2 \dots a_n \in \Sigma^*$:

$$\begin{aligned} \tilde{h}(w) &= \tilde{h}(a_1) \dots \tilde{h}(a_n) \\ &= h(a_1) \dots h(a_n) \\ &= \overline{h}(a_1) \dots \overline{h}(a_n) \\ &= \overline{h}(w) \\ \Rightarrow \tilde{h}\,\text{und}\,\overline{h}\,\text{sind gleich} \end{aligned}$$

1.3 Aufgabe 3

1.3.1 Aufgabenstellung

Sei A eine Menge von $R \subseteq A \times A$ eine transitive Relation. Beweisen oder wiederlegen Sie folgende Aussagen:

1. Die Relation R_1 enthalte $(a,b) \in R_1$definitionsgemäSS genau dann wenn $(a,b) \in R \wedge (b,a) \in R$. Dann ist R_1 eine Äquivalenzrelation.
2. Die Relation R_2 enthalte $(a,b) \in R_2$definitionsgemäSS genau dann wenn $((a,b) \in R \wedge (b,a) \in R) \vee (a = b)$. Dann ist R_2 eine Äquivalenzrelation.
3. Die Relation R_3 enthalte $(a,b) \in R_3$definitionsgemäSS genau dann wenn $(a,b) \in R \vee (b,a) \in R$. Dann ist R_3 eine Äquivalenzrelation.
4. Die Relation R_4 enthalte $(a,b) \in R_4$definitionsgemäSS genau dann wenn $((a,b) \in R \vee (b,a) \in R) \vee (a = b)$. Dann ist R_4 eine Äquivalenzrelation.

1.3.2 Lösung

Um zu zeigen, dass eine Relation eine Äquivalentrelation ist, ist zu zeigen, dass sie reflexiv, transitiv und symmetrisch ist.

1. Die Aussage ist falsch. Deswegen reicht es ein Gegenbeispiel zu finden. Ein Gegenbeispiel zu der Aussage wäre:

$$\begin{aligned} &A = \{1,2,3\} \\ &\Rightarrow R = \{(1,2),(2,3),(1,3)\} \\ &\Rightarrow R_1 = \{\emptyset\} \end{aligned}$$

2. Die Aussage ist wahr. Also bleibt zu zeigen, dass die Relation R_2 reflexiv, transitiv und symmetrisch ist.
Die *Reflexivität* folgt aus der Definition von R_2.
Die *Transitivität* ist wie folgt zu beweisen: Seien $(a,b) \in R_2$und $(b,c) \in R_2$ so ist auch $(a,c) \in R_2$. Aus der Definition der Relation folgt:

$$\begin{aligned} (a,b) \in R_2 &\iff ((a,b) \in R \wedge (b,a) \in R) \vee (a = b) \\ (b,c) \in R_2 &\iff ((b,c) \in R \wedge (c,b) \in R) \vee (b = c) \end{aligned}$$

Es treten also die folgenden Fälle auf:

$$\begin{array}{ll} a = b) & (b,c) \in R_2 \Rightarrow (a,c) \in R_2 \\ b = c) & (a,b) \in R_2 \Rightarrow (a,c) \in R_2 \\ a \neq b,\, b \neq c) & (a,b) \in R_2 \wedge (b,c) \in R_2 = (a,b) \in R \wedge (b,a) \in R \wedge (b,c) \in R \wedge (c,b) \in R \\ & \Rightarrow (a,c) \in R \wedge (c,a) \in R \\ & \Rightarrow (a,c) \in R_2 \end{array}$$

Nun bleibt also nur noch die *Symmetrie* zu zeigen. Die folgt aber schon ganz leicht aus der Definition.

$$\begin{array}{rcl} (a,b) \in R_2 & \Longleftrightarrow & ((a,b) \in R \wedge (b,a) \in R) \vee (a = b) \\ & \Longleftrightarrow & ((b,a) \in R \wedge (a,b) \in R) \vee (b = a) \\ & \Longleftrightarrow & (b,a) \in R_2 \end{array}$$

Und damit ist gezeigt, dass R_2 eine Äquivalenzrelation.

3. R_3 ist keine Äquivalenzrelation. Ein Gegenbeispiel dazu ist:

$$\begin{array}{l} A = \{1,2\} \\ \Rightarrow R = \{(1,2)\} \\ \Rightarrow R_3 = \{(1,2),(2,1)\} \end{array}$$

4. Bleibt noch die Relation R_4, doch auch diese Relation ist keine Äquivalenzrelation. ihr fehlt die Transitivität. Ein Gegenbeispiel ist:

$$\begin{array}{l} A = \{1,2,3\} \\ \Rightarrow R = \{(1,2),(3,2)\} \\ \Rightarrow R_3 = \{(1,2),(2,1),(2,3),(3,2),(1,1),(2,2),(3,3)\} \end{array}$$

1.4 Aufgabe 4

1.4.1 Aufgabenstellung

Sei $\{M_1, \circ_1\}$ein Monoid, $\{M_2, \circ_2\}$ eine algebraische Struktur und $h\colon M_1 \to M_2$ ein Homomorphismus. Beweisen Sie folgende Aussagen:

1. $\{h(M_1), \circ_2\}$ ist ein Monoid.

2. Die Relation $R \subseteq M_1 \times M_1$ enthalte $(a,b) \in R$ definitionsgemäSS genau dann, wenn $h(a) = h(b)$ ist eine Kongruenzrelation, d.h. R ist Äquivalenzrelation und falls $(x,y) \in R$, dann gilt für alle $z, z' \in M_1$ auch $(z \circ_1 x \circ_1 z', z \circ_1 y \circ_1 z') \in R$.

3. Sei $[a]_R = \{b \in M_1 |\ (a,b) \in R\}$, $M_3 = \{[a]_R \mid a \in M_1\}$ die Menge alle Äquivalenzklassen bezüglich R und $\circ_3$ die Operation $[a] \circ_3 [b] = [a \circ_1 b]$. Dann ist $\circ_3$ wohldefiniert und $\{M_3, \circ_3\}$ isomorph zu $\{h(M_1), \circ_2\}$.

1.4.2 Lösung

Gegeben ist also zum einen ein Monoid $\{M_1, \circ_1\}$, eine algebraische Struktur $\{M_2, \circ_2\}$ undder Homomophismus $h\colon M_1 \to M_2$.

1. Um zu zeigen, dass $\{h(M_1), \circ_2\}$ ein Monoid ist. Sind 3 Sachen zu beweisen. Zum ersten muss gezeigt werden, dass $\{h(M_1), \circ_2\}$ eine algebraische Struktur ist, dass die Assoziativität gilt und schlieSSlich, dass ein neutrales Element existiert. Zuerst zeige ich, dass $\{h(M_1), \circ_2\}$eine algebraische Struktur ist.
   Wir haben gegeben, dass für zwei Elemente $b_1, b_2 \in h(M_1)$ gilt:
   $$\exists a_1, a_2 \in M_1 \text{ mit } h(a_1) = b_1;\ h(a_2) = b_2$$
   Und damit folgt:
   $$\begin{aligned} b_1 \circ_2 b_2 &= h(a_1) \circ_2 h(a_2) \\ &= h(a_1 \circ_1 a_2) \end{aligned}$$
   Da nun aber $a_1 \circ_1 a_2 \in M_1$ist, gilt $h(a_1 \circ_1 a_2) \in h(M_1)$. Die *Assoziativität* folgt aus der *Surjektivität* des Homomorphismusses h und der *Assoziativität* von M_1. (Siehe auch Aufgabe 1).
   Bleibt noch die Existenz des neutralen Elements zu zeigen. Wir wissen, dass es ein neutrales Element $\epsilon_1 \in M_1$ gibt, so dass
   $$a_1 \circ_1 \epsilon_1 = \epsilon_1 \circ_1 a_1 = a_1,\ \forall a_1 \in M_1$$
   $h(e_1)$ ist nun das neutrale Element in $\{h(M_1), \circ_2\}$. Weiterhin sei $b \in h(M_1)$. Dann gibt es ein $a \in M_1$ mit $h(a) = b$.Und es gilt:
   $$\begin{aligned} a \circ_1 \epsilon_1 &= \epsilon_1 \circ_1 a = a \\ h(a) \circ_2 h(\epsilon_1) &= h(\epsilon_1) \circ_2 h(a) = h(a) \end{aligned}$$

2. Es gilt $R \subseteq M_1 \times M_1$ und wenn $(a, b) \in R$ sind, dann gilt auch $h(a) = h(b)$. Zu zeigen ist nun, dass R eine Kongruenzrelation ist. Dazu müssen wir zeigen, dass R eine Äquivalenzrelation ist, danach folgern wir, dass es eine Kongruenzrelation ist.
   Betrachten wir zuerst, ob R eine Äquivalenzrelation ist. Die *Reflexivität* ist geschenkt, und folgt durch zweimaliges Einsetzen eines Elementes $a \in M_1$. Die *Symmetrie* ergibt sich durch:
   $$\begin{aligned} (a, b) \in R &\iff h(a) = h(b) \\ &\iff h(b) = h(a) \\ &\iff (b, a) \in R \end{aligned}$$
   Bleibt noch die *Transitivität*. Sie folgt durch:
   $$\begin{aligned} (a, b) \in R,\ (b, c) \in R &\iff h(a) = h(b) \wedge h(b) = h(c) \\ &\iff h(a) = h(c) \\ &\iff (a, c) \in R \end{aligned}$$
   Und schlieSSlich bleibt noch zu überprüfen, ob es sich nicht um eine *Kongruenzrelation* handelt. Sei nun $x \sim y$ und $z, z' \in M_1$ beliebig. Zu zeigen ist nun, dass $zxz' \sim zyz$ dazu:
   $$\begin{aligned} h(z \circ_1 x \circ_1 z') &= h(z) \circ_2 h(x) \circ_2 h(z') \\ &= h(z) \circ_2 h(y) \circ_2 h(z') \\ &= h(z \circ_1 y \circ_1 z') \end{aligned}$$
   $\Rightarrow z \circ_1 x \circ_1 z' \sim z \circ_1 y \circ_1 z'$

3. Gegeben haben wir hier zum einen die $[a]_R = \{b \in M_1 |\ (a, b) \in R\}$ Menge der Äquivalenzklassen der Relation R, die algebraische Struktur $M_3 = \{[a]_R \mid a \in M_1\}$ und schlieSSlich die Operation $\circ_3$ mit $[a] \circ_3 [b] = [a \circ_1 b]$. Um nun zu zeigen, dass die algebraische Struktur $\{M_3, \circ_3\}$ isomorph zu der Sruktur $\{h(M_1), \circ_2\}$ ist, zeigen wir zuerst, dass $\circ_3$ wohldefiniert ist. Wohldefiniert bedeutet, dass die Definition unabhängig von der Wahl der Repräsentanten ist. Es seien also $a_1, a_2, b_1, b_2 \in M_1$ mit $a_1 \sim_R a_2$ und $b_1 \sim_R b_2$dann gilt:

$$\begin{aligned}[a_1] \circ_3 [b_1] &= [a_1 \circ_1 b_1] \\ &= [a_1 \circ_1 b_2] \\ &= [a_2 \circ_1 b_2] \\ &= [a_2] \circ_3 [b_2]\end{aligned}$$

Nun bleibt noch die *Isomorphie* zu zeigen. Sprich zum einen *Injektivität* und zum anderen *Surjektivität*. Der Beweis erfolgt wie bei der ersten Aufgabe.

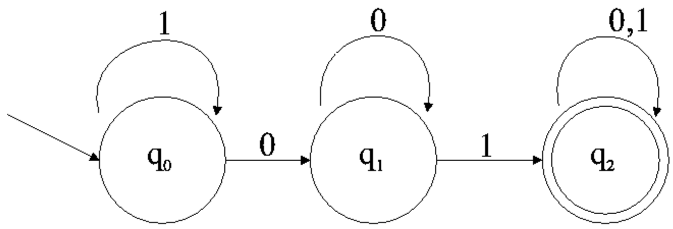
2 Übungsblatt 2

2.1 Aufgabe 5

2.1.1 Aufgabenstellung

Konstruieren Sie einen DFA für die Sprache $L = \{x01y \mid x, y \in \{0,1\}^*\}$. Beweisen Sie exakt (durch vollständige Induktion), dass der Automat diese Sprache akzeptiert.

2.1.2 Lösung



Wir zeigen durch Indiktion, dass der abgebilgete Automat die gegebene Sprache akzeptiert:

$$\begin{array}{lcl}\delta(q_0, w) = q_0 & \Longleftrightarrow & w = 1^i \text{ für } i \geq 0 \\ \delta(q_0, w) = q_0 & \Longleftrightarrow & w = 1^i 0^j \text{ für } i \geq 0, j \geq 1 \\ \delta(q_0, w) = q_0 & \Longleftrightarrow & w = x01y \text{ für alle } x, y \in \Sigma^*\end{array}$$

Verankerung: Für die Wortlängen 0 und 1 ist die Behauptung trivial.
Schritt: Sei dei Behauptung für alle $w \in \Sigma^*$ mit $|w| = n$ bewiesen. Dann gilt

für $x = w\sigma$ mit $|w| = n$ und $\sigma \in \Sigma$:

$$\begin{array}{lcl}
\delta(q_0, w\sigma) = q_0 & \Longleftrightarrow & \delta(q_0, w) = q_0 \wedge \sigma = 1 \\
& \Longleftrightarrow & w = 1^i \text{ für } i \geq 0 \wedge \sigma = 1 \\
& \Longleftrightarrow & w\sigma = 1^i \text{ für } i \geq 0 \\
\delta(q_0, w\sigma) = q_1 & \Longleftrightarrow & (\delta(q_0, w) = q_0 \wedge \sigma = 0) \vee (\delta(q_0, w) = q_1 \wedge \sigma = 0) \\
& \Longleftrightarrow & \left(w = 1^i \text{ für } i \geq 0 \wedge \sigma = 0\right) \vee \left(w = 1^i 0^j \text{ für } i \geq 0, j \geq 1 \wedge \sigma = 0\right) \\
& \Longleftrightarrow & \left(w\sigma = 1^i 0 \text{ für } i \geq 0\right) \vee \left(w\sigma = 1^i 0^j \text{ für } i \geq 0, j \geq 1\right) \\
& \Longleftrightarrow & w\sigma = 1^i 0^j \text{ für } i \geq 0, j \geq 1 \\
\delta(q_0, w\sigma) = q_2 & \Longleftrightarrow & (\delta(q_0, w) = q_1 \wedge \sigma = 1) \vee (\delta(q_0, w) = q_2 \wedge \sigma \in \{0,1\}) \\
& \Longleftrightarrow & \left(w = 1^i 0^j \text{ für } i \geq 0, j \geq 1 \wedge \sigma = 1\right) \vee \left(w = u01v \text{ für } u, v \in \{0,1\}^* \wedge \sigma \in \{0,1\}\right) \\
& \Longleftrightarrow & w\sigma = u01v \text{ für } u, v \in \{0,1\}^*
\end{array}$$

Dann gilt für alle Wörter w aus Σ^*:

$$\begin{array}{lcl}
w \in T(M) & \Longleftrightarrow & \delta(q_0, w) \in F \\
& \Longleftrightarrow & \delta(q_0, w) = q_2 \\
& \Longleftrightarrow & w = u01v \text{ für } u, v \in \{0,1\}^*
\end{array}$$

Und eben das war zu zeigen.

2.2 Aufgabe 6

2.2.1 Aufgabenstellung

Sei m eine natürliche Zahl. $L \subseteq \{0,1\}^*$ sei definiert als die Menge aller $w = a_n \ldots a_1 a_0$ für welche die durch dieses Wort dargestellte Zahl $\sum_{0 \leq i \leq n} a_i 2^i$ durch m teilbar ist.

1. Konstruieren Sie einen DFA für L_m mit höchstens m Zuständen.
2. Konstruieren Sie für Zweierpotenzen $m = 2^k$ einen DFA für L_m mit $k+1$ Züständen.

Begründen Sie jeweils die Korrektheit der Automaten.

2.2.2 Lösung

1. Die Idee ist es, in den Zuständen die Restklassen modulo m zu speichern, welche der Zahl des gelesenen Wortes entspricht. Es gilt:

$$\begin{array}{ll}
\text{Falls} & \phi(x) \bmod m = r, \text{ dann ist} \\
& \phi(x0) \bmod m = 2r \bmod m \text{ und} \\
& \phi(x) \bmod m = 2r + 1 \bmod m \\
\text{Dabei ist } \phi(x1) & \text{die mit dem Wert assoziierter Zahl}
\end{array}$$

Man definiert nun den DFA $M = (Q, \Sigma, \delta, q_0, F)$ mit:

$$\begin{array}{lcl}
Q & = & \{q_i \mid 0 \leq i \leq n-1\} \\
\delta(q_i, 0) & = & q_{(2i \bmod m)} \text{ für alle } q_i \in Q \\
\delta(q_i, 1) & = & q_{(2i+1 \bmod m)} \text{ für alle } q_i \in Q \\
F & = & \{q_0\}
\end{array}$$

Durch Induktion kann man nun zeigen, dass für alle Wörter aus Σ^*gilt: $\delta(q_0, w) = q_{\phi(w) \bmod r}$.

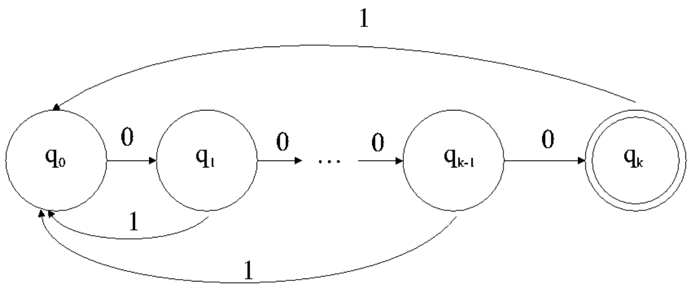
$$\begin{aligned}\forall w \in \Sigma^* : w \in T(M) &\iff \delta(q_0, w) \in F \\ &\iff \delta(q_0, w) = q_0 \\ &\iff \phi(w) \bmod r = 0\end{aligned}$$

Bei der Klausur sind Induktionsbeweise nicht unbedingt notwendig. Jedoch wird zumindest Wert darauf gelegt, dass man die Behauptung aufstellen kann.

2. Man kann zeigen, dass für alle Wörter w aus Σ^* $\phi(w)$ gerade, dann durch $m = 2^k$ teilbar, wenn die letzten k Stellen von w 0 sind. Das Wort kann also nur 2 Formen annehmen:

$$w = x0^k, k \geq 1 \text{ der } w = 0^i, i \geq 0$$

Ein DFA für die Sprache $L_m = \{x0^k | \, x \in \Sigma^*\} \vee \{0^l \, | \, l \geq 0\}$ kann aussehen, wie in Bild 2.2.2 gezeigt. Der Beweis soll hier nicht geführt werden.



2.3 Aufgabe 7

2.3.1 Aufgabenstellung

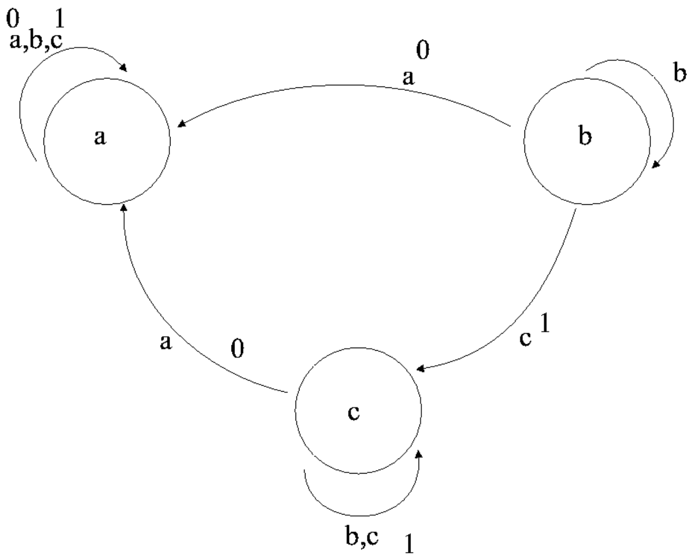
Sei $\langle M, \circ \rangle$ ein endglicher Moniod $M = \{a, b, c\}$ und der Verknüpfung "$\circ$" definiert durch:

$\circ$	a	b	c
a	a	a	a
b	a	b	c
c	a	c	c

Sei h: $\{0,1\}^* \rightarrow M$ ein Homomorphismus mit $h(0) = a$ und $h(1) = c$. Für ein Element $x \in M$ sei $h^{-1}(x) = \{w \in \{0,1\}^* \, | \, h(w) = x\}$ Berechnen Sie $h^{-1}(a)$ und $h^{-1}(c)$.

2.3.2 Lösung

Der Graph im Bild zeigt gut, wie man sich die Verknüpfung vorstellen kann.



Damit komit man schnell auf die Idee, dass $h^{-1}(a) = \left\{u0v \,|\, u, v \in \{0,1\}^*\right\}$ und $h^{-1}(c) = \left\{1^i \,|\, i \geq 1\right\}$ gilt. Wir zeigen nun zuerst, dass für alle $w \in \Sigma^*$ gilt:

$$h(w) = c \iff w = 1^k \text{ für } k \geq 1$$

Zuerst die Richtung: "$\Leftarrow$" mittels Induktion über die Länge der Wörter:
Verankerung: Für Wörter der Länge 1 idt die Behauptung trivial.
Schritt $n \to n+1$:

$$\begin{aligned} h\left(1^{k+1}\right) &= h\left(1^k\right) \circ h(1) \\ &= c \circ c \\ &= c \end{aligned}$$

Und nun wieder durch Induktion über die Wortlänge $|w|$ die Richtung "$\Rightarrow$"
Verankerung: Für die Wortlänge $|w| = 1$ gilt $h(0) = a$ und $h(1) = c$
Schritt $|w| = n+1$*:* Für $w = x$ sei die behauptung schon gezeigt. Nun ist also $w = x\sigma, \sigma \in \Sigma$. Auch hierbei gilt wieder:

$$h(x\sigma) = h(x) \circ h(\sigma)$$

Fall: $\sigma = 0$:

$$\begin{aligned} h(x) \circ h(0) &= h(x) \circ a \\ &= a \\ \text{also ist } h(x\sigma) &\neq c \end{aligned}$$

Fall $\sigma = 1$:

$$\begin{array}{lcl} h(x) \circ h(1) & = & h(x) \circ c \\ h(x) = a & \Rightarrow & h(x) \circ c = a \circ c = a \\ h(x) = b & \Rightarrow & h(x) \circ c = b \circ c = c \\ h(x) = c & \Rightarrow & h(x) \circ c = c \circ c = c \end{array}$$

Dabei ist zu beachten, dass $h(x) = b$ nicht vorkommt für $x \neq \epsilon$.
Für $h(x) = c$ ist nach der Verankerung: $x = 1^n$, also $x\sigma = 1^{n+1}$. Für alle

$w \in \Sigma^*$ gilt: $h(w) = a \vee h(w) = c$ also ist:

$$h^{-1}(a) = \Sigma^* - h^{-1}(c) = \{u0v \mid u, v \in \Sigma^*\}$$

und das war zu zeigen.

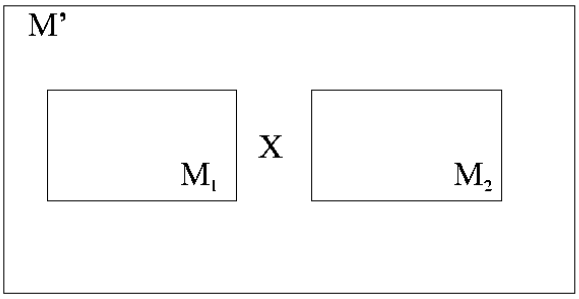
2.4 Aufgabe 8

2.4.1 Aufgabenstellung

Gegeben seien zwei DFA $M_i = \left(Q_i, \Sigma_i, \delta_i, q_0^i, F_i\right)$ für $(i = 1, 2)$

1. Konstruieren Sie *direkt* (ohne Verwendung der Potenzmengenkonstruktion) einen DFA M' mit $T(M) = T(M_1) \cup T(M_2)$. Beweisen Sie, dass der von Ihnen definierte Automat tatsächlich diese Sprache akzeptiert.
2. Konstruieren Sie einen NFA M'' mit $T(M'') = T(M_1) \cup T(M_2)$, der nicht mehr als $|Q_1| + |Q_2| + 1$ Zustände und beweisen Sie dessen Korrektheit.

2.4.2 Lösung



1. Die Lösungsidee ist, wie im Bild schon gezeigt, die Konstruktion des DFA's mittels einer Methode, inspiriert, durch das Kreuzprodukt. Der DFa ist danach: $M' = \left(Q, \Sigma, \delta, \left[q_0^1, q_0^2\right], F\right)$ mit:

$$\begin{array}{lcl} Q & = & \{[p,q] \mid p \in Q_1, q \in Q_2\} \\ \delta([p,q], \sigma) & = & [\delta_1(p,\sigma), \delta_2(q,\sigma)] \text{ für alle } [p,q] \in Q, \sigma \in \Sigma \\ F & = & \{[p,q] \mid p \in F_1 \vee q \in F_2\} \end{array}$$

zu Zeigen ist nun, dass: $T(M') = T(M_1) \cup T(M_2)$. Dies zeigt man am besten durch Induktion über die Länge der Wörter. Zu zeigen ist nun:

$$\forall [p,q] \in Q, w \in \Sigma^* \colon \delta([p,q], w) = [\delta_1(p,w), \delta_2(q,w)]$$

Verankerung: $|w| = 0 \Rightarrow w = \epsilon$

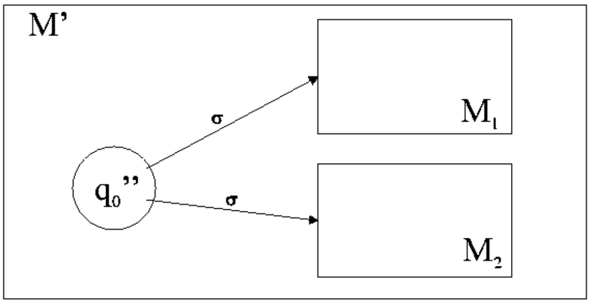
$$\delta([p,q], \epsilon) = [p,q] = [\delta_1(p,\epsilon), \delta_2(q,\epsilon)]$$

Schritt: Sei die Behauptung für $|w| = n$ bewiesen und nun $|w| = n + 1$. Also ist $w = x\sigma$, $\sigma \in \Sigma$. Es ist nun:

$$\begin{aligned}
\delta\left([p,q],w\right) &= \delta\left([p,q],x\sigma\right) \\
&= \delta\left(\delta\left([p,q],x\right),\sigma\right) \\
&= \delta\left(\left[\delta_1(p,x),\delta_2(q,x)\right],\sigma\right) \\
&= \left[\delta_1\left(\delta_1(p,x),\sigma\right),\delta_2\left(\delta_2(q,x),\sigma\right)\right] \\
&= \left[\delta_1(p,x\sigma),\delta_2(q,x\sigma)\right]
\end{aligned}$$

Nun bleibt noch die Frage, wie daraus $T(M') = T(M_1) \cup T(M_2)$ folgen soll. Der Beweis sieht wie folgt aus:

$$\begin{aligned}
w \in T(M') &\iff \delta\left(\left[q_0^1,q_0^2\right],w\right) \in F \\
&\iff \left[\delta_1\left(q_0^1,w\right),\delta_2\left(q_0^2,w\right)\right] \in F \\
&\iff \delta_1\left(q_0^1,w\right) \in F_1 \vee \delta_2\left(q_0^2,w\right) \in F_2 \\
&\iff w \in T(M_1) \vee w \in T(M_2) \\
&\iff w \in T(M_1) \cup T(M_2)
\end{aligned}$$



2.

Die Konstruktion der Teilaufgabe 1) hat $|Q_1| \cdot |Q_2|$ Zustände. Dies sind jedoch weit mehr, als gefordert sind. Das Bid zeigt die neue Idee. Die formale Definition für M'' ist dann:

$$M'' = \left(Q'',\Sigma,\delta'',q_0^{''},F''\right)$$

mit:

$$\begin{aligned}
Q'' &= \left\{q_0^{''}\right\} \cup Q_1 \cup Q_2 \wedge Q_1 \cap Q_2 = \emptyset \\
\delta''\left(q_0^{''},\sigma\right) &= \left\{\delta_1\left(q_0^1,\sigma\right),\delta_2\left(q_0^2,\sigma\right)\right\}.\ \forall \sigma \in \Sigma \\
\delta''(q,\sigma) &= \left\{\delta_1\left(q_0^1,\sigma\right)\right\}.\ \forall q \in Q_1, \sigma \in \Sigma \\
\delta''(q,\sigma) &= \left\{\delta_2\left(q_0^2,\sigma\right)\right\}.\ \forall q \in Q_2, \sigma \in \Sigma \\
F'' &= \left\{q \in Q'' \mid q \in F_1 \cup F_2\right\} \cup \left\{q_0^{''} \mid q_0^1 \in F_1 \vee q_0^2 \in F_2\right\}
\end{aligned}$$

Durch Induktion über die Wortlänge $|w|$, ergibt sich:

$$\forall w \in \Sigma^* \text{ ausser } w = \epsilon$$
$$\delta''\left(q_0^{''},w\right) = \left\{\delta_1\left(q_0^1,w\right),\delta_2\left(q_0^2,w\right)\right\}$$

$$
\begin{array}{lll}
\forall w \in \Sigma^{'} & & \\
w \in T\left(M''\right) & \Longleftrightarrow & \delta''\left(q_0^{''}, w\right) \cap F \neq \emptyset \\
& \Longleftrightarrow & \left\{\delta_1\left(q_0^1, w\right), \delta_2\left(q_0^2, w\right)\right\} \cap \left(F_1 \cap F_2\right) = \emptyset \\
& \Longleftrightarrow & \left\{\delta_1\left(q_0^1, w\right)\right\} \cap F_1 \neq \emptyset \cup \left\{\delta_2\left(q_0^2, w\right)\right\} \cap F_2 \neq \emptyset \\
& \Longleftrightarrow & \delta_1\left(q_0^1, w\right) \in F_1 \vee \delta_2\left(q_0^2, w\right) \in F_2 \\
& \Longleftrightarrow & w \in T\left(M_1\right) \vee w \in T\left(M_2\right) \\
& \Longleftrightarrow & w \in T\left(M_1\right) \cup T\left(M_2\right)
\end{array}
$$

AuSSerdem gilt:

$$
\begin{array}{l}
\epsilon \in T\left(M''\right) \Longleftrightarrow \epsilon T\left(M_1\right) \cup T\left(M_2\right) \\
T\left(M''\right) = T\left(M_1\right) \cup T\left(M_2\right)
\end{array}
$$

Damit ist bewiesen, dass der NFA die Sprache Akzeptiert.

3 Übungsblatt 3

3.1 Aufgabe 9

3.1.1 Aufgabenstellung

Ein nichtdeterministischer endlicher Automat mit mehrerenAnfangszuständen(MNFA) sei definiert durch $M = (Q, \Sigma, \delta, Q_0, F)$ wobei $Q_0 \subseteq Q$ eine nicht-leere endliche Menge von Start-zuständen bezeichnet, alle anderen Komponenten entsprechen denen eines NFA. Die durch den Automaten akzeptierte Sprache sei $T(M) = \{w \in \Sigma^* \mid \delta(Q_0, w) \cap F \neq \emptyset\}$, wobei δ wie üblich auf Zustandsmengen und Eingabewörter erweitert sei. Beweisen Sie: Jede von einem MNFA akzeptierte Sprache ist regulär.

3.1.2 Lösung

Zunächst interssiert erst einmal die Frage, wie ein δ', welches auf Zustandsmengen erweitert ist eigentlich aussieht, dazu helfen die folgenden Überlegungen:

$$
\begin{array}{l}
\delta : Q \times \Sigma \rightarrow 2^Q \\
\delta' : 2^Q \times \Sigma^* \rightarrow 2^Q \\
\delta'(A, \epsilon) = A, \; \forall A \in 2^Q \\
\delta'(A, w\sigma) = \bigcup_{p \in \delta(A,w)} \delta(p, \sigma) \, \forall A \in 2^Q, w \in \Sigma^*, \sigma \in \Sigma
\end{array}
$$

Wieder gilt definitionsgemäSS, dass $\delta = \delta'$ ist. Um nun zu zeigen, dass die von dem MNFA akzeptierte Sprache regulär ist, konstruiert man einen DFA M' mit $T(M') = T(M)$. Dann ist ein NFA zu konstruieren. Mit dem Wissen, dass der Index der Nerode Relation endlich ist, kann man dann zeigen, dass $T(M')$ regulär ist.
Man beginnt also damit, dass man einen DFA mittels Potenzmengenkonstruktion erzeugt wird. Für den DFA M' gilt:

$$
\begin{array}{lll}
M' & \stackrel{def}{=} & (Q', \Sigma, \delta'[Q_0], F') \\
Q' & \stackrel{def}{=} & \left\{[A] \mid A \in 2^Q\right\} \\
F' & \stackrel{def}{=} & \{[A] \in Q' \mid A \cap F \neq \emptyset\} \\
\delta'([A], \sigma) & \stackrel{def}{=} & [\delta(A, \sigma)] \; \forall [A] \in Q', \sigma \in \Sigma
\end{array}
$$

Von diesem DFA kann man zeigen, dass $T(M') = T(M)$ gilt. Es ist auch zu zeigen, dass sich die Definition auf die Erweiterung der δ-Funktion fortsetzt. Durch Induktion zeigt man also für alle $w \in \Sigma^*$:

$$\begin{array}{lcl} \delta(Q_0, w) & \stackrel{def}{=} & A \iff \delta'([Q_0], w) = [A] \\ w \in T(M) & \iff & \delta(Q_0, w) = A \wedge A \cap F \neq \emptyset \\ & \iff & \delta'([Q_0], w) = [A] \wedge [A] \in F' \\ & \iff & w \in T(M') \end{array}$$

Also gilt $T(M') = T(M)$. Nun konstruieren wir einen äquivalenten NFA dazu:

$$\begin{array}{lcl} M'' & \stackrel{def}{=} & (Q'', \Sigma, \delta'', q_0', F'') \\ Q'' & \stackrel{def}{=} & Q' \cup \{q_0'\}, q_0' \notin Q' \\ F'' & \stackrel{def}{=} & F' \cup \{q_0' \mid \epsilon \in T(M)\} \\ \delta''(q_0', \sigma) & \stackrel{def}{=} & \delta(Q_0, \sigma), \forall \sigma \in \Sigma \\ \delta''(q, \sigma) & \stackrel{def}{=} & \delta(q, \sigma), \forall q \in Q', \sigma \in \Sigma \end{array}$$

Damit ist gezeigt, dass auch die Sprache des MNFAs regulär ist.

3.2 Aufgabe 10

3.2.1 Aufgabenstellung

Für ein Eingabewort $w = \sigma_1\sigma_2 \ldots \sigma_n \in \Sigma^*, \sigma_i \in \Sigma, \forall 1 \leq i \leq n$, bezeichne $w^R = \sigma_n \ldots \sigma_2\sigma_1$ das Reversal-Wort zu w. Für eine Sprache $L \subseteq \Sigma^*$ sei die Reversal-Sprache definiert als:

$$L^R = \left\{w^R \in \Sigma^* \mid w \in L\right\}$$

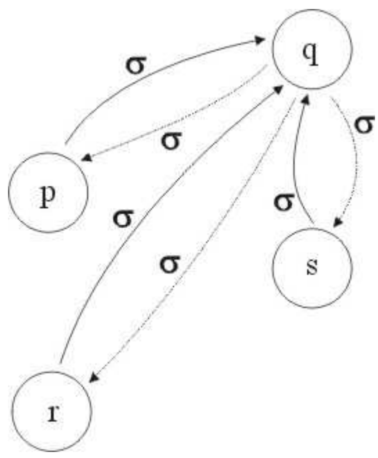
Sei weiter $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA und der zu M gehörende Reversal-MNFA M^R definiert durch $M^R = \left(Q, \Sigma, \delta^R, F, \{q_0\}\right)$ mit:

$$\delta^R(p, \sigma) = \{q \in Q \mid p = \delta(q, \sigma)\} \text{ fr alle } p \in Q, \sigma \in \Sigma.$$

Beweisen Sie: $T\left(M^R\right) = T(M)^R$.

3.2.2 Lösung

Hier erst einmal ein Bild, welches etwas zum Verständnis der Aufgabe beitragen sollte:



Das Beispiel zeigt, dass zu einem Zustand q mehrere Zustände existieren, welche mit dem Zeichen σ erreichbar sind. In anderen Worten:

$$\delta^R(q,\sigma) = \{p,r,s\}$$

Dies zeigt uns, dass wir auf jeden Fall in einem NFA landen werden. Zu zeigen ist nun, dass beide Automaten das selbe tun, also dass $T\left(M^R\right) = T(M)^R$ gilt.

Dies kann man durch Induktion nach der Wortlänge $|w|$ zeigen. Für alle $w \in \Sigma^*, p, q \in Q$ gilt:

$$\delta(q,w) = q \iff \delta^R\left(q, w^R\right) \ni p$$

Verankerung$|w| = n = 0$ *:* Es gilt:

$$\delta(p,\epsilon) = q \iff \delta^R(q,\epsilon) = p \text{ da } p = q \text{ ist.}$$

Schritt $n \to n+1$*:* Seien $p, q \in Q$ und $w\sigma \in \Sigma^{n+1}, \sigma \in \Sigma$ dann gilt:

$$\begin{aligned}
&\delta(p,w\sigma) = q \iff \exists p' \in Q\colon \delta(p,q) = p' \wedge \delta(p,\sigma) = q \\
&\iff \exists p' \in Q\colon p \in \delta^R\left(p', w^R\right) \wedge p' \in \delta^R(q,\sigma) \\
&\iff p \in \delta^R\left(q, \sigma w^R\right) \\
&\iff p \in \delta^R\left(q, (w\sigma)^R\right)
\end{aligned}$$

damit wäre gezeigt ,dass die Übergangsfunktion auch für Wörter definiert ist. Um nun zu zeigen, dass $T\left(M^R\right) = T(M)^R$ ist:

$$\begin{aligned}
w \in T(M) &\iff \delta(q_0, w) \in F \\
&\iff \delta(q_0, w) = f \wedge f \in F \\
&\iff q_0 \in \delta^R\left(f, w^R\right) \wedge f \in F \\
&\iff q_0 \in \delta^R\left(F, w^R\right) \\
&\iff \delta^R\left(F, w^R\right) \cap \{q_0\} \neq 0 \\
&\iff w^R \in T\left(M^R\right)
\end{aligned}$$

Und somit ist $T\left(M^R\right) = T(M)^R$.

3.3 Aufgabe 11

3.3.1 Aufgabenstellung

Beweisen Sie den *Satz von Brzozowski*: Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA für L ohne nicht-erreichbare Zustände und M^R der zu M gehörende Reversal-MNFA. Dann ist der aus M^R durch Potenzmengenkonstruktion erzeugte DFA $M' = (Q', \Sigma, \delta', [F], F')$ mit:

$$\begin{array}{lll} Q' & \overset{def}{=} & \left\{[A] \mid A \in 2^Q \wedge \exists w \in \Sigma^*\colon \delta^R(F, w) = A\right\} \\ F' & \overset{def}{=} & \{[A] \in Q' \mid q_0 \in A\} \\ \delta'([A], \sigma) & \overset{def}{=} & \left[\delta^R(A, \sigma)\right] \text{ fuer alle } [A] \in Q', \sigma \in \Sigma \end{array}$$

ein minimaler DFA für L^R.
Hinweis: Zeigen Sie zunächst, dass der DFA L^R akzeptiert und beweisen Sie dann, dass der Autmoat minimal ist. Wenn Sie geschickt sind, kann der erste Teil des Beweises bereits durch Aufgabe 9 erfolgen.

3.3.2 Lösung

Zuerste einmal eine Anmerkung: Geschickt bedeutet , bei der 9 die Potenzmengenkonstruktion agewandt zu haben, damit folgt die Behauptung. Somit ist $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA ohne nichterreichbare Zustände. Ein Zustand $q$ist dann nicht ereichbar, falls es kein Wort $w \in \Sigma^*$ gibt mit $\delta(q_0, w) = q$. Sei nun weiterhin M^R ein MNFA mit $T\left(M^R\right) = T(M)^R$ und M' durch Potenzmengenkonstruktion aus M^R gewonnen. Nach den Aufgaben 9 und 10 folgt:

$$T(M') = T\left(M^R\right) = T(M)^R$$

Zu zeigen ist nun, dass M' ein minimaler DFA ist:
Minimalität bedeutet:

- jeder Zustand ist erreichbar,
- M' ist reduziert, d.h. für alle $[A], [B] \in Q'$mit $[A] \neq [B]$ existiert ein $w^R \in \Sigma^*$ mit:

  $$\left(\delta'\left([A], w^R\right) \in F' \wedge \delta'\left([B], w^R\right) \notin F\right) \vee \left(\delta'\left([A], w^R\right) \notin F' \wedge \delta'\left([B], w^R\right) \in F\right)$$

  Diese beiden Zustände können nicht zusammenfallen, da sonst eine falsche Sprache akzeptiert werden würde.

Die Erreichbarkeit der Zustände folgt direkt aus der Definition. Also bleibt nur die Reduzibilität zu zeigen. Wir betrachten dazu zwei verschiedene Zustände. Verschieden bedeutet:

$$\exists q \in Q\colon (q \in A \wedge q \notin B) \vee (q \notin A \wedge q \in B)$$

Sei nun M ein DFA ohne nicht-erreichbare Zustände, also:

$$\exists w \in \Sigma^*\colon \delta(q_0, w) = q$$

Bei Aufgabe 10 wurde gezeigt, dass:

$$\begin{array}{l}\delta\left(q_0,w\right)=q \iff \delta^R\left(q,w\right)=q_0\\ \forall p\in Q-\{q\}:\delta\left(q_0,w\right)=p \iff q_0\notin\delta^R\left(p,w\right)\end{array}$$

Es sei $[A],[B]\in Q'$ mit $[A]\neq[B]$. Dann gibt es ein $q\in Q$ mit:

1. $q\in A\wedge q\notin B$ oder
2. $q\notin A\wedge q\in B$

Wir betrachten ObdA nur den Fall:

$$q\in A\wedge q\notin B$$

Da in M jeder Zustand erreichbar ist, gibt es ein Wort $w\in\Sigma^*$ mit:

$$\delta\left(q_0,w\right)=q$$

Damit folgt nach Aufgabe 10:

$$q_0\in\delta^R\left(q,w^R\right)$$

und wegen

$$q_0\in\delta^R\left(q,w^R\right) \iff \delta\left(q_0,w\right)=q$$

folgt

$$q_0\in\delta^R\left(p,w^R\right),\ \forall p\in Q \text{ mit } p\neq q$$

denn es ist $\delta\left(q_0,w\right)=p$. Damit ist

$$\delta'\left(A,w^R\right)\ni q_0\wedge\delta^R\left(B,w^R\right)\ni q_0$$

Also gilt, dass:

$$\delta'\left([A],w^R\right)\in F'\wedge\delta'\left([B],w^R\right)\notin F'$$

und damit ist M' reduzibel.

4 Übungsblatt 4

4.1 Aufgabe 12

4.1.1 Aufgabenstellung

Sei $R\subseteq\Sigma^*$ eine reguläre Sprache. Zeigen Sie, dass die Sprache:

$$FH\left(R\right)\overset{def}{=}\{x\mid\exists y\in\Sigma^*:(|x|=|y|\wedge xy\in R)\}$$

ebenfalls regulär ist.

4.1.2 Lösung

Angenommen wir haben einen DFA für R. Dieser muss existieren, da R regulär ist. Die Idee ist nun, mit einem Wort x in q_0 zu starten und gleichzeitig mit einem Wort y in f zu starten. Dies zeigt auch das Bild recht gut.

Da nun $R \subseteq \Sigma^*$ regulär ist, gibt es einen DFA $M = (Q, \Sigma, \delta, q_0, F)$ mit $T(M) = R$. Dazu konstruiert man nun den NFA $M' = (Q', \Sigma, \delta', q_0', F')$ mit:

$$\begin{aligned}
&Q' = \{[p,q] \mid p, q \in Q\} \cup \{q_0' \mid q_0' \in Q \times Q\} \\
&\delta'([p,q], \sigma) = \{[\delta(p,\sigma), q'] \mid \exists f \in \Sigma \colon \delta(q', f) = q\} \\
&\text{Für alle } [p,q] \in Q', \sigma \in \Sigma \text{ gilt :} \\
&\delta'(q_0', \sigma) = \textstyle\bigcup_{f \in F} \delta'([q_0, f], \sigma) \ \text{ für alle } \sigma \in \Sigma \\
&F' = \{[p,q] \mid q \in Q\} \cup \{q_0' \mid q_0 \in F\}
\end{aligned}$$

Durch Induktion über die Wortlänge $|w|$ zeigt man, dass für alle $x \in \Sigma^*$ und $p, q \in Q$ gilt:

$$\delta'(q_0', x) = [p,q] \iff \delta(q_0, x) = p \wedge \exists y \in \Sigma^*, f \in F \colon (|y| = |x| \wedge \delta(q, y) = f)$$

Der Beweis ist nun schon so oft durchgeführt worden, dass an dieser Stelle darauf verzichtet werden soll.
Damit gilt für alle Wörter $x \in \Sigma^*$:

$$\begin{aligned}
x \in T(M') &\iff \delta'(q_0', x) \cap F \neq \emptyset \\
&\iff \exists [p,q] \in F' \colon [p,q] \in \delta'(q_0', x) \\
&\iff \exists [q,q] \in Q' \colon [q,q] \in \delta'(q_0', x) \\
&\iff \exists [q,q] \in Q' \colon \delta(q_0, x) = q \wedge \exists y \in \Sigma^*, f \in F \colon (|x| = |y| \wedge \delta(q, y) = f) \\
&\iff \exists y \in \Sigma^*, q \in Q \colon \delta(q_0, x) = q \wedge \delta(q, y) = y \wedge |y| = |x| \\
&\iff \exists y \in \Sigma^* \colon |x| = |y| \wedge xy \in T(M) \\
&\iff x \in FH(T(M))
\end{aligned}$$

Wegen $\epsilon \in T(M') \iff \epsilon \in T(M)$ gilt:

$$T(M') = FH(T(M))$$

4.2 Aufgabe 13

4.2.1 Aufgabenstellung

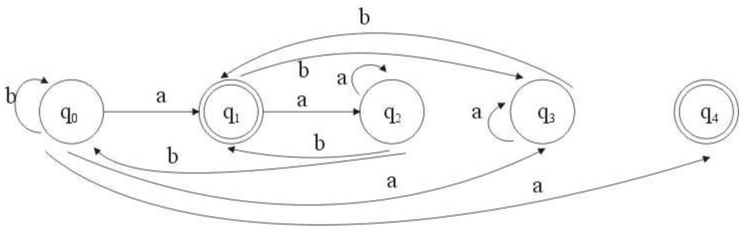
Gegeben sei der NFA $M = (\{q_0, \ldots, q_4\}, \{a, b\}, \delta, q_0, \{q_1, q_4\})$ mit der Übergangsfunktion:

$$\begin{array}{ll}
\delta(q_0, a) \stackrel{def}{=} \{q_1, q_3, q_4\} & \delta(q_0, b) \stackrel{def}{=} \{q_0\} \\
\delta(q_1, a) \stackrel{def}{=} \{q_2\} & \delta(q_1, b) \stackrel{def}{=} \{q_3\} \\
\delta(q_2, a) \stackrel{def}{=} \{q_2\} & \delta(q_2, b) \stackrel{def}{=} \{q_0, q_1\} \\
\delta(q_3, a) \stackrel{def}{=} \{q_3\} & \delta(q_3, b) \stackrel{def}{=} \{q_1\} \\
\delta(q_4, a) \stackrel{def}{=} \emptyset & \delta(q_4, b) \stackrel{def}{=} \emptyset
\end{array}$$

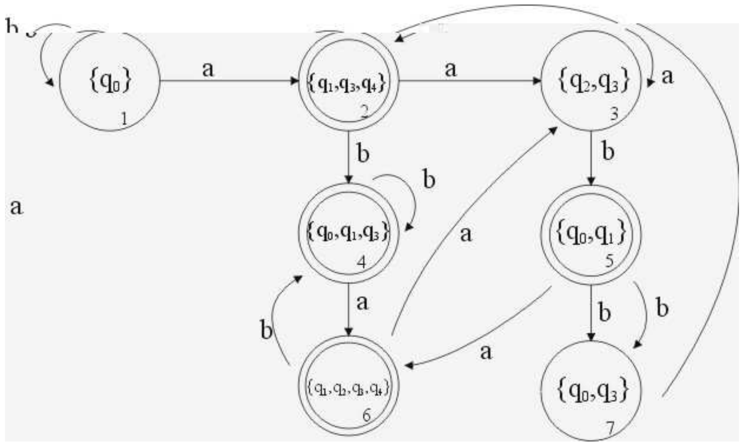
Konstruieren Sie mittels der Potenzmengenkonstruktion einen äquivalenten DFA und wenden Sie auf diesen den aus der Vorlesung bekannten Minimierungsalgorithmus an. Dokumentieren Sie alle Zwischenschritte.

4.2.2 Lösung

die Übergangsfunktion führt zu folgendem Automaten:



damit kann man den folgenden DFA konstruieren:



Dieser ist nun zu Minimieren. Der Algorithmus besagt, dass zunächst alle nicht-erreichbaren Zustände entfernt werden sollen. In unserem Fall sind jedoch keine solchen Zustände vorhanden. Der nächste Schritt besteht darin, die Klasse $\overset{0}{\sim}$ zu definieren. Dies ist:

$$\overset{0}{\sim}: [2,4,5,6], [1,3,7]$$

Damit ist die Basis geschaffen, um über mehrere Iterationen alle benötigten Klassen zu bestimmen. Das Verfahren Terminiert, sobald sich zwei Klassen entsprechen. Dazu erstelle ich die folgende Tabelle:

$\overset{1}{\sim}$	a	b
1	$[2,4,5,6]_{\overset{0}{\sim}}$	$[1,3,7]_{\overset{0}{\sim}}$
2	$[1,3,7]_{\overset{0}{\sim}}$	$[2,4,5,6]_{\overset{0}{\sim}}$
3	$[1,3,7]_{\overset{0}{\sim}}$	$[2,4,5,6]_{\overset{0}{\sim}}$
4	$[2,4,5,6]_{\overset{0}{\sim}}$	$[2,4,5,6]_{\overset{0}{\sim}}$
5	$[2,4,5,6]_{\overset{0}{\sim}}$	$[1,3,7]_{\overset{0}{\sim}}$
6	$[1,3,7]_{\overset{0}{\sim}}$	$[2,4,5,6]_{\overset{0}{\sim}}$
7	$[2,4,5,6]_{\overset{0}{\sim}}$	$[2,4,5,6]_{\overset{0}{\sim}}$

Die Klassen von $\overset{1}{\sim}$ sind also:

$$[2,6], [1], [3], [4], [5], [7]$$

Und nun das Ganze nun noch für die Klasse $\overset{2}{\sim}$:

$\overset{2}{\sim}$	a	b
1	$[2,6]_{\overset{1}{\sim}}$	$[1]_{\overset{1}{\sim}}$
2	$[3]_{\overset{1}{\sim}}$	$[4]_{\overset{1}{\sim}}$
3	$[3]_{\overset{1}{\sim}}$	$[5]_{\overset{1}{\sim}}$
4	$[2,6]_{\overset{1}{\sim}}$	$[4]_{\overset{1}{\sim}}$
5	$[2,6]_{\overset{1}{\sim}}$	$[7]_{\overset{1}{\sim}}$
6	$[3]_{\overset{1}{\sim}}$	$[4]_{\overset{1}{\sim}}$
7	$[2,6]_{\overset{1}{\sim}}$	$[5]_{\overset{1}{\sim}}$

Die Klassen von $\overset{2}{\sim}$ sind also:

$$[1], [2], [3], [4], [5], [6], [7]$$

Damit sind wir auch schon am Ende angelangt, da keine Zustände mehr zusammengefaSSt sind, werden alle weiteren Relationen kein neues Ergebnis bringen können.

4.3 Aufgabe 14

4.3.1 Aufgabenstellung

Sei $L_n \overset{def}{=} \left\{x1y \,|\, x, y \in \{0,1\}^*, |x| \leq n-1, |y| = n\right\}$. Beweisen Sie:

1. Es gibt einen NFA für L_n mit $2n+1$ Zuständen.
2. Jeder NFA für L_n hat mindestens $2n+1$ Zustände.
3. Jeder DFA für L_n hat mindestens 2^{n+1} Zustände.

4.3.2 Lösung

1. Durch Induktion für alle Wörter $w \in \Sigma^*$ beweist man:

$$\begin{array}{lcl} q_i \in \delta(q_0, w) & \Longleftrightarrow & |w| = i \wedge 0 \leq |w| \leq n-1 \\ p_i \in \delta(q_0, w) & \Longleftrightarrow & w = w'1w'', |w''| = i, 0 \leq |w''| \leq n, 0 \leq |w'| \leq n-1 \end{array}$$

   Damit gilt also für Wörter aus $T(M)$:

$$\begin{array}{rcl} w \in T(M) & \Longleftrightarrow & \delta(q_0, w) \cap F \neq \emptyset \\ & \Longleftrightarrow & p_n \in \delta(q_0, w) \\ & \Longleftrightarrow & w = w'1w'', |w''| = n, 0 \leq |w'| \leq n-1 \\ & \Longleftrightarrow & w \in L_n \end{array}$$

2. Sei nun $M = (Q, \Sigma, \delta, q_0, F)$ ein beliebiger NFA für die Sprache L_n. Zu zeigen ist nun:

$$|Q| \geq 2 \cdot n + 1$$

Dazu nehmen wir an, $|Q|$ wäre kleiner als $2 \cdot n + 1$. Offenstichtlich ist dann L_n endlich und die maximale Länge eines Wortes $w \in L_n$ beträgt $|w| = 2 \cdot n$. Wie sieht nun ein solches Wort w aus?
Behauptung:

$$\begin{array}{l} \forall 0 \leq i \leq l \leq 2 \cdot n \colon w = w'_i = w''_i, |w'_i| = l, |w''_i| = 2 \cdot n - i \\ \exists q_i \in Q \colon \delta\left(q_0, w'_i\right) \ni q_i \wedge \delta\left(q'_0, w''_i\right) \cap F \neq \emptyset \end{array}$$

Beweis: Dann gilt aber:

$$\text{Wenn gilt: } \nexists q_i \text{ dann ist } \delta\left(q_0, w\right) \cap F = \emptyset$$

und dies stellt einen Widerspruch dar. Wegen $|Q| < 2 \cdot n + 1$ gibt es ein i und ein j $(i < j)$ mit:

$$q_i = q_j - p$$

Daraus folgt:

$$\begin{array}{l} p \in \delta\left(q_0, w'_j\right) \wedge \left|w'_j\right| = j \\ p \in \delta\left(q_0, w'_i\right) \wedge \left|w'_i\right| = i \\ \delta\left(p, w''_j\right) \cap F \neq \emptyset, \left|w''_j\right| = 2 \cdot n - j \\ \delta\left(p, w''_i\right) \cap F \neq \emptyset, \left|w''_i\right| = 2 \cdot n - i \end{array}$$

Da aber $i < j$ ist folgt:

$$\delta\left(q_0, w'_j w''_i\right) > \delta\left(p, w''_i\right) \Rightarrow \delta\left(q_0, w'_j w''_i\right) \cap F \neq \emptyset$$

Damit folgt aber $w'_j w''_i \in T(M)$. Damit ist $\left|w'_j w''_i\right| = 2 \cdot n - i + j > 2 \cdot n + 1$ was einen Widerspruch darstellt.

3. Zu zeigen ist hier, dass jeder DFA für $L_n \geq 2^{n+1}$ Zustände hat. Seien nun $w_1, w_2 \in L_n$ mit $w_1 \neq w_2$. Dann ist:

$$\begin{array}{l} w_1 = a_r, a_{r-1}, \ldots, a_2, a_1 \text{ und} \\ w_2 = b_s, b_{s-1}, \ldots, b_2, b_1 \end{array}$$

mit $1 \leq r, s \leq n$ und $a_i, b_i \in \Sigma$.
Es sind nun zwei Fälle zu beachten. Zum einen:

- $r \neq s$ oBdA sei nun $r < s$ mit $u = 0^{n-r-1}10^n$ gilt:

$$\begin{array}{rcl} w_1 u & \in & L_n \text{ aber es gilt} \\ w_2 u & \notin & L_n \text{ wegen } |w_2 u| > 2 \cdot n \end{array}$$

$\Rightarrow \delta\left(q_0, w\right) \neq \delta\left(q_0, w_2\right)$

- Der Zweite Fall ist: $r = s$ Der Beweis hierzu geht analog wie der im Skript. Es se $u = 0^{n-i+1}$ dann folgt:

$$\delta\left(q_0, w_i u\right) \neq \delta\left(q_0, w_2 u\right)$$

Es gibt also insgesamt $2^{n+1} - 2$ solcher Worte. Hinzu kommen noch die akzeptierenden Zustände und die Startzustände.

4.4 Aufgabe 15

4.4.1 Aufgabenstellung

Beweisen Sie ohne Benutzung des Satzes von Nerode: Die folgenden drei Aussagen sind äquivalent:

1. L ist regulär.
2. L ist die Vereinigung von einigen Äquivalenzklassen einer Kongruenzrelation mit endlichem Index.
3. Der Index der Relation:

$$x \sim y \iff (\forall w, w' \in \Sigma^* \colon wxw' \in L \iff wyw' \in L)$$

ist endlich.

4.4.2 Lösung

Der Beweis funktioniert analog zu dem im Skript. Hier ist jedoch die $\rho -$ *Relation* definiert durch:

$$x\rho y \iff \delta(q,x) = \delta(q,y), \forall q \in Q$$

$1 \to 2$: Sei nun $R \subseteq \Sigma^*$ regulär. Damit gibt es einen DFA M mit $T(M) = R$. Für jedes $x \in \Sigma^*$ definiert man:

$$\begin{aligned} \delta_x &: Q \to Q \\ \delta_x(q) &= \delta(q,x) \text{ für alle } q \in Q \end{aligned}$$

Man definiert die Relation $\sim$ wie folgt:

$$\begin{aligned} x \sim y &\iff \delta_x = \delta_y \\ &\iff \forall q \in Q \colon \delta(q,x) = \delta(q,y) \end{aligned}$$

Diese Relation ist reflexiv und symmetrisch. Die Transitivität folgt aus:

$\delta_x = \delta_y$ und $\delta_y = \delta_z$ d.h. es gilt auch $\delta_x = \delta_z$ und damit $x \sim z$.

Also ist $\sim$ transitiv und damit eine Äquivalenzrelation.
Für jedes $x, y \in \Sigma^*$ und $q \in Q$ gilt:

$$\begin{aligned} \delta_{xy}(q) = \delta(q,xy) &= \delta(\delta(q,x), y) \\ &= \delta_y(\delta_x(q)) \\ &= \delta_x \circ \delta_y(q) \end{aligned}$$

Also ist $\delta_{xy} = \delta_x \circ \delta_y$. Sei $x \sim y$ und $z \in \Sigma^*$ beliebig gewählt. Es gilt:

$$\delta_{xy} = \delta_x \circ \delta_z = \delta_y \circ \delta_z = \delta_{yz}$$

damit ist $xz \sim yz$ und $\sim$ ist rechtsinvariant. Ebenso gilt:

$$\delta_{zx} = \delta_z \circ \delta_x = \delta_z \circ \delta_y = \delta_{zy}$$

Damit haben wir die Linksinvarianz. Somit ist $\sim$ eine Kongruenzrelation.
Es gibt $|Q|^{|Q|}$ verschiedene Funktionen von $Q \to Q$. Damit ist der Index von $\sim$ endlich.
Und es ist weiter:

$$T(M) = \{\langle x\rangle_\sim \mid \delta_x(q_0) \in F\}$$

Das sollte reichen um zu zeigen wie der Beweis funktioniert. Ansonsten funktioniert der Beweis genau, wie der Beweis im Skript.

5 Übungsblatt 5

5.1 Aufgabe 16

5.1.1 Aufgabenstellung

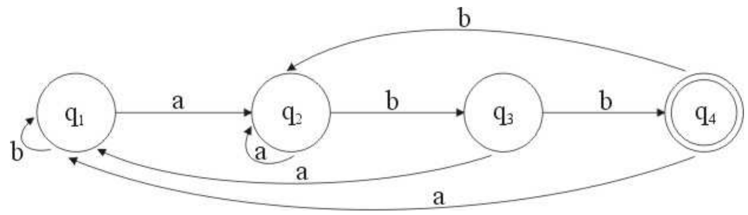
Konstruieren Sie mit Hilfe des aus der Vorlesung bekannten Verfahrens aus dem folgendem DFA $M = (\{q_0, \ldots, q_3\}, \{a, b\}, \delta, q_0, \{q_3\})$ mit:

$$\begin{array}{ll}
\delta(q_0, a) \overset{def}{=} q_1 & \delta(q_0, b) \overset{def}{=} q_0 \\
\delta(q_1, a) \overset{def}{=} q_1 & \delta(q_1, b) \overset{def}{=} q_2 \\
\delta(q_2, a) \overset{def}{=} q_0 & \delta(q_2, b) \overset{def}{=} q_3 \\
\delta(q_3, a) \overset{def}{=} q_0 & \delta(q_3, b) \overset{def}{=} q_1
\end{array}$$

einen äquivalenten regulären Ausdruck. Dokumentieren Sie alle Zwischenschritte.

5.1.2 Lösung

Hier erst einmal ein Bild des oben definierten Automaten:



Um diesen Automaten umzuwandeln sind folgende zwei Regeln wiederholt anzuwenden:

$$\begin{array}{lcl}
R_{ij}^{k} & = & R_{ik}^{k-1}\left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1} \cup R_{ij}^{k-1} \\
R_{ij}^{0} & = & \begin{cases} \{a \mid \delta(q_i, a) = q_j\} & \text{falls } i \neq j \\ \{a \mid \delta(q_i, a) = q_j\} \cup \{\epsilon\} & \text{falls } i = j \end{cases}
\end{array}$$

Diese rekursiven Regeln sind nun für k durchzuführen: Zunächst für $k = 0, k = 1$:

	$k=0$	$k=1$
r_{11}^k	$\epsilon+b$	$(\epsilon+b)\,(\epsilon+b)\,(\epsilon+b)\cup(\epsilon+b)=b^*$
r_{12}^k	a	$(\epsilon+b)\,(\epsilon+b)\,a\cup a=b^*a$
r_{13}^k	$\emptyset$	$(\epsilon+b)\,(\epsilon+b)\,\emptyset\cup\emptyset=\emptyset$
r_{14}^k	$\emptyset$	$(\epsilon+b)\,(\epsilon+b)\,\emptyset\cup\emptyset=\emptyset$
r_{21}^k	$\emptyset$	$\emptyset\,(\epsilon+b)\,(\epsilon+b)\cup\emptyset=\emptyset$
r_{22}^k	$\epsilon+a$	$\emptyset\,(\epsilon+b)\,a\cup(\epsilon+a)=(\epsilon+a)$
r_{23}^k	b	$\emptyset\,(\epsilon+b)\,\emptyset\cup b=b$
r_{24}^k	$\emptyset$	$\emptyset\,(\epsilon+b)\,\emptyset\cup\emptyset=\emptyset$
r_{31}^k	a	$a\,(\epsilon+b)\,(\epsilon+b)\cup a=ab^*$
r_{32}^k	$\emptyset$	$a\,(\epsilon+b)\,a\cup\emptyset=ab^*a$
r_{33}^k	ϵ	$a\,(\epsilon+b)\,\emptyset\cup\epsilon=\epsilon$
r_{34}^k	b	$a\,(\epsilon+b)\,\emptyset\cup b=b$
r_{41}^k	a	$a\,(\epsilon+b)\,(\epsilon+b)\cup a=ab^*$
r_{42}^k	b	$a\,(\epsilon+b)\,a\cup b=ab^*a+b$
r_{43}^k	$\emptyset$	$a\,(\epsilon+b)\,\emptyset\cup\emptyset=\emptyset$
r_{44}^k	ϵ	$a\,(\epsilon+b)\,\emptyset\cup\epsilon=\epsilon$

Und nun das selbe noch für $k=2$ und $k=3$:

	$k=2$	$k=3$
r_{11}^k	$(b^*a)\ (\epsilon+a)\ \emptyset\cup b^*=b^*$	
r_{12}^k	$(b^*a)(\epsilon+a)\ (\epsilon+a)\cup b^*a=b^*a^+$	
r_{13}^k	$(b^*a)(\epsilon+a)\ b\cup\emptyset=b^*a^+b$	
r_{14}^k	$(b^*a)(\epsilon+a)\ \emptyset\cup\emptyset=\emptyset$	
r_{21}^k	$(\epsilon+a)\ (\epsilon+a)\ \emptyset\cup\emptyset=\emptyset$	
r_{22}^k	$(\epsilon+a)\ (\epsilon+a)\ (\epsilon+a)\cup(\epsilon+a)=a^*$	
r_{23}^k	$(\epsilon+a)\ (\epsilon+a)\ b\cup b=a^*b$	
r_{24}^k	$(\epsilon+a)\ (\epsilon+a)\ \emptyset\cup\emptyset=\emptyset$	
r_{31}^k	$(ab^*a)(\epsilon+a)\ \emptyset\cup ab^*=ab^*$	
r_{32}^k	$(ab^*a)(\epsilon+a)\ (\epsilon+a)\cup ab^*a=ab^*a^+$	
r_{33}^k	$(ab^*a)(\epsilon+a)\ b\cup\epsilon=ab^*a^+b$	
r_{34}^k	$(ab^*a)(\epsilon+a)\ \emptyset\cup b=b$	
r_{41}^k	$(ab^*a+b)(\epsilon+a)\ \emptyset\cup ab^*=ab^*$	
r_{42}^k	$(ab^*a+b)(\epsilon+a)\ (\epsilon+a)\cup ab^*a+b=ab^*a^++ba^+$	
r_{43}^k	$(ab^*a+b)(\epsilon+a)\ b\cup\emptyset=ab^*a^+b+ba^*b$	
r_{44}^k	$(ab^*a+b)(\epsilon+a)\ \emptyset\cup\epsilon=\epsilon$	

Zu berechnen ist nun der akzeptierende Ausdruck:

$$r_{14}^3=r_{13}^2\left(r_{33}^2\right)^*r_{34}^2+r_{14}^2$$

Damit erspare ich mir mal die Spalte $k=3$, denn es gilt:

$$r_{14}^3=b^*a^+b\left(ab^*a^+b\right)^*b+\emptyset$$

Und damit hätten wir einen regulären Ausdruck gefunden, welcher den Automaten beschreibt.

5.2 Aufgabe 17

5.2.1 Aufgabenstellung

Sei $R \subseteq \Sigma^*$ eine reguläre Sprache. Zeigen Sie, dass die Sprache

$$MT\left(R\right) \stackrel{def}{=} \{y \mid \exists x, z \in \Sigma^* \colon (|x| = |y| = |z| \wedge xyz \in R)\}$$

ebenfalls regulär ist.

5.2.2 Lösung

Wenn die Sorache R regulär ist, dann existiert ein DFA M mit $M = (Q, \Sigma, \delta, q_0, F)$ für welchen gilt: $T\left(M\right) = R$. Die Idee, um zum *mittleren Drittel (MT)* zu gelangen ist, ein Zeichen zu lesen und dann die anderen beiden Werte von dort aus zu ermitteln. Wir definieren nun einen MNFA $M' = (Q', \Sigma, \delta', Q_0, F')$ mit:

$$\begin{aligned}
Q' &= Q \times Q \times Q \times Q \times Q \\
Q_0 &= \{[q_0, p, p, q, q] \mid p, q \in Q\} \\
F' &= \{[p', p', q', q', f] \mid p', q' \in Q, f \in F\} \\
\delta'\left([q_1, q_2, q_3, q_4, q_5], a\right) &= \{[\delta\left(q_1, \sigma\right), q_2, \delta\left(q_3, a\right), q_4, \delta\left(q_5, \tau\right)] \mid \sigma, \tau \in \Sigma\} \ \forall a \in \Sigma
\end{aligned}$$

δ' sei wie üblich auf Worte $w \in \Sigma^*$ erweitert. (Die Zustände q_2 und q_4 dienen zum Speichern der Startpositionen.)

Behauptung:

$$\begin{aligned}
[p_1, p_2, p_3, p_4, p_5] \in \delta'\left([q_1, q_2, q_3, q_4, q_5], w\right) \iff & (\exists x \in \Sigma^* \colon |x| = |w| \wedge \delta\left(q_1, x\right) = p_1) \wedge \\
& (q_2 = p_2) \wedge \\
& (\delta\left(q_3, w\right) = p_3) \wedge \\
& (q_4 = p_4) \wedge \\
& (\exists z \in \Sigma^* \colon |z| = |w| \wedge \delta\left(q_5, z\right) = p_5)
\end{aligned}$$

Der Beweis der Behauptung erfolgt durch Induktion über alle Worte w.

Für Wörter der Länge 0 und 1 ist die Behauptung klar, nach der Definition. Es gibt nur die Worte:

$$\begin{aligned}
|w| = 0 &\Rightarrow w = \epsilon \\
|w| = 1 &\Rightarrow w \in \Sigma
\end{aligned}$$

Gelte nun die Behauptung für alle Wort der Wortlänge $|w| \leq n$. Nun folgt der Schritt auf $|w| = n + 1$. Dazu betrachten wir ein Wort $w = w_1, \ldots, w_n, w_{n+1}$. Für diese gilt:

$$\begin{aligned}
[p_1, p_2, p_3, p_4, p_5] \in \delta'\left([q_1, q_2, q_3, q_4, q_5], w\right) \iff & (\exists x \in \Sigma^* \colon |x| = |w| \wedge \delta\left(q_1, x\right) = p_1) \wedge \\
& (q_2 = p_2) \wedge \\
& (\delta\left(q_3, w\right) = p_3) \wedge \\
& (q_4 = p_4) \wedge \\
& (\exists z \in \Sigma^* \colon |z| = |w| \wedge \delta\left(q_5, z\right) = p_5)
\end{aligned}$$

damit ist:

$$\delta'\left(\left[q_1,\ldots,q_5\right],w_1\ldots w_{n+1}\right)=\bigcup_{\left[s_1,\ldots,s_5\right]\in\delta'\left(\left[q_1,\ldots,q_5\right],w_1\ldots w_n\right)}\delta'\left(\left[s_1,\ldots,s_5\right],w_{n+1}\right)$$

$$=\bigcup_{\substack{[s_1,\ldots,s_5] \text{ mit } (\exists x\in\Sigma^*:\delta(q_1,x)=s_1)\wedge\\(s_2=q_2)\wedge\\(\delta(q_3,w_1\ldots w_n)=s_3)\wedge\\(s_4=q_4)\wedge\\(\exists z\in\Sigma^*:\delta(q_5,z)=s_5)\wedge\\(|x|=|z|=n)}}\left\{\left[\delta\left(s_1,\sigma\right),s_2,\delta\left(s_3,w_{n+1}\right),s_4,\delta\left(s_5,\tau\right)\right]\mid\sigma,\tau\in\Sigma\right\}$$

Durch einsetzen erhält man dann:

$$=\left\{\left[\delta\left(\delta\left(q_1,x\right),\sigma\right),q_2,\delta\left(\delta\left(q_3,w_1\ldots w_n\right),w_{n+1}\right),s_4,\delta\left(\delta\left(q_5,z\right),\tau\right)\right]\mid\sigma,\tau\in\Sigma\wedge\exists x,z\in\Sigma^*:|x|=|y|=n\right\}$$

$$=\left\{\left[\delta\left(q_1,x'\right),q_2,\delta\left(q_3,w\right),q_4,\delta\left(q_5,z'\right)\right]\mid\exists x',z'\in\Sigma^*:|x|=|y|=n\right\}$$

$$\begin{aligned}\Rightarrow\left[p_1,\ldots,p_5\right]\in\delta'\left(\left[q_1,\ldots,q_5\right],w_1\ldots w_{n+1}\right)\quad\Longleftrightarrow\quad&\left(\exists x\in\Sigma^*:|x|=n+1\wedge\delta\left(q_1,x\right)=p_1\right)\wedge\\&\left(q_2=p_2\right)\wedge\\&\left(\delta\left(q_3,w_1\ldots w_{n+1}\right)=p_3\right)\wedge\\&\left(q_4=p_4\right)\wedge\\&\left(\exists z\in\Sigma^*:|z|=n+1\wedge\delta\left(q_5,\epsilon\right)=p_5\right)\end{aligned}$$

Somit ist die Behauptung bewiesen. Bleibt nur noch zu zeigen, dass für Wörter $w\in T\left(M'\right)$ auch gilt $w\in MT\left(R\right)$.

$$w\in T\left(M'\right)\iff\delta'\left(q_0,y\right)\cap F'\neq\emptyset\Rightarrow\exists p,y,p',y'\in Q,f\in F$$

mit:

$$\begin{aligned}\left[p,q,w,q,f\right]\in\delta'\left(\left[q_0,p',p',q',q'\right],w\right)\quad\Longleftrightarrow\quad&\left(\exists x\in\Sigma^*:|x|=|w|\wedge\delta\left(q_0,x\right)=p\right)\wedge\\&\left(p'=p\right)\wedge\\&\left(\delta\left(q',w\right)=q\right)\wedge\\&\left(q'=q\right)\wedge\\&\left(\exists x,y\in\Sigma^*,p,q\in Q,f\in F:\left(\delta\left(q_0,xyz\right)=\delta\left(\delta\left(q_0,x\right),yz\right)\right)\right)\end{aligned}$$

weiterhin ist:

$$\begin{aligned}\delta\left(p,yz\right)\quad&=\quad\delta\left(\delta\left(p,y\right),z\right)=\delta\left(q,z\right)=f\wedge\left(|x|=|y|=|z|\right)\\&\Longleftrightarrow\quad\exists x,y\in\Sigma^*:|x|=|y|=|z|\wedge xyz\in R\iff w\in MT\left(R\right)\end{aligned}$$

Damit ist gezeigt, dass $MT\left(R\right)$ regulär ist.

5.3 Aufgabe 18

5.3.1 Aufgabenstellung

Seien r,s und t reguläre Ausdrücke. Beweisen oder widerlegen Sie die folgenden Aussagen:

1. $r\,(s+t) = rs + rt$
2. $s\,(rs+s)^*\,r = rr^*s\,(rr^*s)^*$
3. $(r+s)^* = r^* + s^*$
4. $(r+s)\,t = rt + st$
5. $(rs+r)^*\,r = r\,(sr+r)^*$
6. $(rs)\,t = r\,(st)$

5.3.2 Lösung

Zunächst sollen folgende Definitionen gegeben werden, die das Lösen der Aufgaben um einiges erleichtern:

$$\begin{aligned} r^* &= \bigcup_{i\geq 0}^{\infty} r^i \\ w \in r+s &\iff w \in r \vee w \in s \\ w \in r \cdot s &\iff \exists x,y\colon (x \in r \wedge y \in s \wedge w = xy) \end{aligned}$$

Damit ist es nun möglich die Ausdrücke zu überprüfen.

1. $r\,(s+t) = rs + rt$
   Betrachten wir nun ein Wort $w \in r \cdot (s+t)$.
   Mit der Definition von $\cdot$ folgt:
   $$\exists x,y\colon (y \in (s+t) \wedge x \in r \wedge w = xy)$$
   Aus der Definition von $+$ kann man folgern:
   $$\exists x,y\colon ((y \in s \vee y \in t) \wedge (x \in r \wedge w = xy))$$
   Mittels Definition der Distributivität folgt dann:
   $$\exists x,y\,(((y \in s) \wedge (x \in r \wedge w = xy)) \vee ((y \in t) \wedge (x \in r \wedge w = xy)))$$
   Nutzt man nun nochmal die Definition von $\cdot$ aus, so erhält man:
   $$\exists x,y\,((w = xy \in rs) \vee (w = xy \in rt))$$
   Mit der Definition von $+$ schlieSSlich, kommt man zu:
   $$w \in rs + rt$$
   Und das war zu zeigen.

2. Die Aussage $s\,(rs+s)^*\,r = rr^*s\,(rr^*s)^*$ ist falsch. Um dies zu zeigen genügt ein Gegenbeispiel:
   Dazu wählt man $r = 0, s = 1$. Dann gilt:
   $$10 \in 1\,(01+1)^*\,0 = s\,(rs+s)^*\,r$$
   und
   $$10 \notin 00^*1\,(00^*1)^* = rr^*s\,(rr^*s)^*$$
   damit ist die Aussage falsch.

3. Auch diese Aussage ($(r+s)^* = r^* + s^*$) ist falsch. Für $r = 0$, $s = 1$ gilt:
$$1010 \in (0+1)^* = (r+s)^*$$
und
$$1010 \notin (0^* + 1^*) = r^* + s^*$$
somit ist auch diese Aussage widerlegt.

4. Die Aussage $(r+s)\,t = rt{+}st$ scheint zu gelten. Für ein Wort $w \in (r+s)\,t$ gilt nach der Definition von $\cdot$:
$$\exists x, y \colon (x \in (r+s) \wedge y \in t \wedge w = xy)$$
Mit der Definition von $+$ erhält man daraus:
$$\exists x, y \colon ((x \in r \vee x \in s) \wedge (y \in t \wedge w = xy))$$
Nun erhäkt man nach dem Distributivgesetz:
$$\exists x, y \colon (((x \in r) \wedge (y \in t \wedge x = xy)) \vee ((x \in s) \wedge (y \in t \wedge x = xy)))$$
Mittels der Definition von $\cdot$ überführt man dies in:
$$\exists x, y \colon ((w = xy \in rt) \vee (w = xy \in st))$$
und damit ergibt die Definition von $+$:
$$w \in rt + st$$
Damit ist die Aussage bewiesen.

5. Auch die Aussage $(rs+r)^*\,r = r\,(sr+r)^*$ ist war. Wir betrachten ein Wort $w \in (rs+r)^*\,r$. Nach der Definition von $*$ gilt:
$$\bigcup_{i \geq 0}^{\infty} (rs+r)^i\,r$$
Dies entspricht der Vereinigung von i-Faktoren $(r\,(s+\epsilon)\,r)$. Durch Vergleichen mit Aufgabenteil 6 und 4 gelangt man so schnell zu:
$$\bigcup_{i \geq 0}^{\infty} r\,(sr+r)^i$$
und somit gilt:
$$w \in r\,(sr+r)^* \,.$$

6. SchlieSSlich bleibt die Aussage $(rs)\,t = r\,(st)$ zu überprüfen. Diese ist wahr, wie der kurze folgende Beweis zeigen wird. Wir betrachten dazu ein Wort $w \in (rs)\,t$ Mit der Definition von $\cdot$ gelangt man schnell zu:
$$\exists x, y, z \colon ((xy \in rs) \wedge x \in t \wedge w = xyz)$$

Nochmaliges Anwenden führt dann zu:

$$\exists x, y, z\colon ((x \in r \wedge y \in s) \wedge x \in t \wedge w = xyz)$$

Durch die Assoziativität folgt:

$$\exists x, y, z\colon (x \in r \wedge (y \in s \wedge z \in t) \wedge w = xyz)$$

Nun verwendet man wieder die Definition von $\cdot$ und gelangt zunächst zu:

$$\exists x, y, z\colon (x \in r \wedge (yz \in st) \wedge w = xyz)$$

und schlieSSlich zu:

$$w \in r\,(st)$$

womit die Aussage bewiesen wäre.

5.4 Aufgabe 19

5.4.1 Aufgabenstellung

Beweisen oder widerlegen Sie:

1. Sie $L_1 \subseteq \Sigma^*$ eine reguläre Sprache und $L_2 \subseteq \Sigma^*$ eine beliebige Sprache. Dann ist die Sprache

$$L_1/L_2 \overset{def}{=} \{x \mid \exists y \in L_2\colon xy \in L_1\}$$

   regulär.

2. Reguläre Sprachen sind abgeschlossen unter abzählbar unendlicher Vereinigung.

5.4.2 Lösung

1. Die Aussage ist wahr. Es sei $L_1 \subseteq \Sigma^* \in REG$ und $L_2 \subseteq \Sigma^*$ beliebig. Dann ist

$$L_1/L_2 = \{x \mid \exists y \in L_2\colon xy \in L_1\} \in REG.$$

   Hier nun der einfache Beweis: Da L_1 regulär ist, existiert ein DFA $M\,(Q, \Sigma, \delta, q_0, F)$ mit $T\,(M) = L_1$. Man definiert nun einen Automaten $M'\,(Q, \Sigma, \delta, q_0, F')$ mit

$$F' = \{q \in Q \mid \exists y \in L_2\colon \delta\,(q, y) \in F\}\,.$$

   Dann gilt:

$$\begin{aligned} x \in T\,(M') &\iff \delta\,(q_0, x) \in F' \iff \delta\,(q_0, x) = q \wedge q \in F' \\ &\iff \delta\,(q_0, x) = q \wedge (\exists y \in L_2\colon \delta\,(q, y) \in F) \\ &\iff \exists y \in L_2\colon \delta\,(q_0, xy) = \delta\,(q, y) \in F \\ &\iff \exists y \in L_2\colon xy \in L_1 \\ &\iff x \in L_1/L_2 \end{aligned}$$

   Also ist $T\,(M') = L_1/L_2$ und damit ist L_1/L_2 regulär.

2. Reguläre Sprachen sind nicht abgeschlossen gegenüber abzählbarer unendlicher Vereinigung. Zum Beweis soll ein Gegenbeispiel genügen. Sei $L_n = \{a^n b^n\}$ definiert für $n \in N$. Jede Sprache ist endlich und somit regulär. Durch Vereinigung erhält man:

$$L = \bigcup_{n=0}^{\infty} L_n = \bigcup_{n=0}^{\infty} \{a^n b^n\} = \{a^n b^n \mid n \geq 0\}$$

diese ist jedoch nicht regulär.

6 Übungsblatt 6

6.1 Aufgabe 20

6.1.1 Aufgabenstellung

Welche der folgenden Sprachen sind regulär? Beweisen Sie Ihre Aussagen:

1. $L_1 \stackrel{def}{=} \{wcw \mid w \in \{0,1\}^n\}$ für ein festes $n \in N$
2. $L_2 \stackrel{def}{=} \left\{wcw \mid w \in \{0,1\}^*\right\}$
3. $L_3 \stackrel{def}{=} \left\{w \in \{0\}^* \mid |w| \neq 2^n \text{ für alle } n \in N\right\}$
4. $L_4 \stackrel{def}{=} \left\{ww^R \mid w \in \{0,1\}^*\right\}$

6.1.2 Lösung

1. Angenommen die Sprache $L_1 \stackrel{def}{=} \{wcw \mid w \in \{0,1\}^n\}$ wäre für ein festes $n \in N$ regulär. Man betrachtet dann DFAs M_i, die genau einen Wert aus L_1 erkennen. So gilt zum Beispiel für $n = 2$:

$$\begin{array}{lll} M_1 & \text{akzeptiert} & \{00c00\} \\ M_2 & \text{akzeptiert} & \{01c01\} \end{array}$$

Zu einem gegebenem $n \in N$ gibt es dann 2^n solcher DFAs M_i. Dies sind jedoch endlich viele und somit ist L_1 dann die Vereinigung regulärer Mengen und ist somit regulär.

2. Angenommen $L_2 \stackrel{def}{=} \left\{wcw \mid w \in \{0,1\}^*\right\}$ wäre regular. Dann gibt es nach dem Pumping-Lemma eine Konstante $n \in N$, desweiteren gibt es für das Wort $\{0^n c 0^n\} \in L_2$ eine Zerlegung:

$$0^n c 0^n = xyz \text{ mit } 0 < |y| < n \text{ und } xy^i z \in L_2 \forall i \geq 0$$

Daraus ergeben sich die folgenden Fälle:
1. Fall: y enthält ein c daraus folgt: $xy^2z \in L_2$ würde 2 c's enthalten. In dem Fall wäre das Wort aber kein Element der Sprache mehr.
2. Fall: y enthält kein c und z enthält ein c. Durch das Pumping-Lemma erhält man so links des c's mehr Nullen, als rechts davon. Diese Worte müssten in L_2 sein, sind es aber nicht. Also auch hier ein Widerspruch.

3. Fall: Dieser Fall ist analog zu *Fall 2:* y enthält kein c jedoch x enthält eines. Somit erhält man diesmal rechts von c mehr Nullen, als links davon. Wieder stellt dies einen Widerspruch dar.
Zusammenfassend lässt sich also sagen, dass L_2 nicht regulär ist.

3. Angenommen die Sprach $L_3 \overset{def}{=} \left\{w \in \{0\}^* \mid |w| \neq 2^n \text{ für alle } n \in N\right\}$ wäre regulär. Dann wäre auch das Komplement der Sprache

$$\overline{L}_3 \overset{def}{=} \left\{w \in \{0\}^* \mid |w| = 2^n \text{ für alle } n \in N\right\}$$

regulär. Sei nun n die Konstante des Pumping-Lemmas und $2^m > n$ gewählt. Dann besitzt das Wort $w = 0^{2^n}$ eine Zerlegung $w = xyz$ mit $0 < |y| < n$ und $xy^iz \in \overline{L}_3$ für alle $i \geq 0$.
Insbesondere gilt auch $xy^2z \in \overline{L_3}$ dass heist, $\left|xy^2z\right|$ ist eine 2er Potenz. Damit gilt:

$$2^n < 2^m + 1 \leq \left|xy^2z\right| \leq 2^m + n < 2^n + 2^m = 2^{m+1}$$

Daraus folgt aber, dass $\left|xy^2z\right|$ keine 2er Potenz gewesen sein kann. Dies stellt einen Widerspruch zur Annahme dar und somit kann L_3 nicht regulär gewesen sein.

4. Angenommen die Sprache $L_4 \overset{def}{=} \left\{ww^R \mid w \in \{0,1\}^*\right\}$ wäre regulär. Mit dem wissen, dass aufgrund der Abschlusseigenschaften dann auch die Sprache:

$$L_4' = L_4 \cap \{0^r1^s0^n \mid n, r, s \geq 0\} = \left\{0^n1^{2r}0^n \mid n, r \geq 0\right\}$$

regulär ist. Nun kann man einen Homomorphismus h_1 aufstellen mit:

$$\begin{aligned} h_1(a) &= 0 \\ h_1(b) &= 0 \\ h_1(1) &= 1 \end{aligned}$$

Dann ist die Sprache:

$$L_4'' = h_1^{-1}(L_4') \cap \left\{a^r1^{2r}b^r \mid r, n \geq 0\right\}$$

ebenfalls regulär. Betrachtet man nun den Homomorphismus h_2 mit:

$$\begin{aligned} h_2(a) &= a \\ h_2(1) &= \epsilon \\ h_2(b) &= b \end{aligned}$$

Die Sprache definiert durch:

$$L_4''' = h_2(L_4'') = \{a^nb^n \mid n \geq 0\}$$

müsste dann ebenfalls regulär sein, jedoch wissen wir, dass L_4''' nicht regulär ist. Demzufolge kann auch L_4 nicht regulär sein.

6.2 Aufgabe 21

6.2.1 Aufgabenstellung

Beweisen Sie, dass die Sprache

$$L \stackrel{def}{=} \left\{0^i 1^j \mid i,j \geq 1 \wedge ggT\left(i,j\right) = 1\right\}$$

nicht regulär ist.
Hinweis: Sie können folgenden Satz benutzen.

Satz

> Zu jedem $i,j \in Z$ mit $ggT\left(i,j\right) = 1$ existieren $u,v \in Z$ mit $ui+vj = 1$.

6.2.2 Lösung

Zuerst eine Überlegungen zu dem Hinweis: Zu jedem $i,j \in Z$ mit $ggT\left(i,j\right) = 1$ existieren $u,v \in Z$ mit $ui + vj = 1$.

Folgerung: zu $i,j \in N \mid i,j > 1$ mit $ggT\left(i,j\right) = 1$ existieren $u,v \in Z$ mit $ui + vj = 1$.

Beweis: Nach dem Satz existieren $u,v \in Z$ mit $ui + vj = 1$. Da $i,j > 0$ sind, muss entwerde $u < 0$ und $v > 0$ gelten, oder aber $u > 0$ und $v < 0$ gelten. Im ersten Fall ist die Folgerung bereits bewiesen. Für den zweiten Fall gilt:

> Sei nun also $u \geq 0$ und $v \leq 0$. Dann gilt:
>
> $$\begin{array}{rlll} ui + vj = 1 & \Longleftrightarrow & & ui - uij + uij + vj = 1 \\ & \Rightarrow & \left(u - uj\right)i & + \left(v + uj\right)j = 1 \end{array}$$
>
> man definiert nun $u' = \left(u - uj\right)$ und $v' = \left(v - uj\right)$ dann gilt:
>
> $$u'i + v'j = 1$$
>
> Da weiterhin gelten soll, dass $u < 0$ und $v > 0$ (bzw. $u' < 0$ und $v' > 0$) sind noch folgende Fälle zu betrachten:
>
> $u = 0 \Rightarrow vj = 1 \Rightarrow j = 1$ Widerspruch
> $v = 0 \Rightarrow ui = 1 \Rightarrow i = -1$ Widerspruch
> $u' = 0 \Rightarrow u - uj = 0 \Rightarrow j = 1$ Widerspruch
> $v' = u \Rightarrow u + uj = 1 \Rightarrow r\left(j - 1\right) = 1 \Rightarrow j + \frac{1}{r}n$ Widerspruch
> $\Rightarrow j = 0$, da aber $r < 0$ auch hier Widerspruch

Nun aber zurück zur Aufgabe. Zuerst wird angenommen, dass die Sprache $L \stackrel{def}{=} \left\{0^i 1^j \mid i,j \geq 1 \wedge ggT\left(i,j\right) = 1\right\}$ regulär sei. Dann gibt es nach dem Pumping-Lemma ein $n \in N$, so dass für $w \in L$ mit $|w| \geq n$ gilt:

$$w = xyz,\ 0 < |y| < n,\ xy^i z \in L \text{ für alle } i \geq 0$$

Sei nun $w = 0^n 1^{n'}$, wobei n' die kleinste Primzahl gröSSer als n ist. Somit ist der $ggT\left(n, n'\right) = 1$. Damit ist $w \in L$. Da weiterhin gilt, dass die Wortlänge

$|w| \geq n$ ist, existiert eine Zerlegung $w = xyz$. Wenn y Nullen und Einsen enthalten würde, so würde es beim "Pumpen" einen Widerspruch geben, da dann das Wort nicht nur aus einem Block Nullen und einem Block Einsen bestehen würde. Also enthält y nur Nullen, oder nur Einsen. OBdA seien dies hier nur Nullen.

Dann ist $w = xyz_1z_2$ mit $z_1 \in \{0\}^*, z_2 \in \{1\}^*$. Weiterhin gilt: $|yz_1| = n$ und $|z_2| = n'$. Daraus folgt aber: $n = |y| + |xz_1| = s + r$ (mit $0 < s < n, r > 0$). Da aber n' Primzahl ist folgt, dass der $ggT(s, n') = 1$ ist. Nach unserer Folgerung gibt es nun $u, v \in Z$ mit $u < 0, v > 0$ und es gilt:

$$\begin{array}{rcl} us + vn' & = & 1 \\ u < 0 \Rightarrow u & = & -|u| \\ us + rn' = 1 & \Longleftrightarrow & uvs + vrn' = r \iff |u|rs + r \in n' = r \\ \Longleftrightarrow \quad v|u|s + r & = & vrn' \text{ mit } r, |u|, s, v > 0 \end{array}$$

Nach dem Pumping-Lemma gilt, $xy^{r|u|}z \in L$, dass heist der $ggT\left(\left|xy^{r|u|}z_1\right|, |z_2|\right) = 1$. Jedoch gelten:

$$\begin{array}{l} \left|xy^{r|u|}z_1\right| = |y|\,r\,|u| + |xz_1| = sr\,|u| + r \text{ und} \\ ggT\left(\left|xy^{r|u|}z_1\right|, |z_2|\right) = ggT(sr\,|u| + r, n') = ggT(vrn', n') = n' > 1 \end{array}$$

Da dies einen Widerspruch zur Anname, dass der $ggT(\cdot, \cdot) = 1$ ist, darstelt ist L nicht regulär.

6.3 Aufgabe 22

6.3.1 Aufgabenstellung

Sei L die Menge aller korrekt geformten arithmetischen Ausdrücke über dem Alphabet $\{\text{op}, (,), \cdot, +\}$.

1. Geben Sie eine kontextfreie Grammatik für L an.
2. Beweisen Sie, dass Ihre Grammatik L erzeugt.
3. Beweisen Sie: L ist nicht regulär

6.3.2 Lösung

1. Zu generieren ist also die Grammatik $G = (V, \Sigma, P, S)$ dabei sind:

$$\begin{array}{l} \Sigma = \{op, (,), \cdot, +\} \\ S = S \\ V = \{S\} \end{array}$$

Für P gilt:

$$P = \{ \quad \begin{array}{l} S \to op \\ S \to S + S \\ S \to S \cdot S \\ S \to (S) \quad \} \end{array}$$

Die so definierte Grammatik ist kontextfrei, da gilt:

$$\forall p \in P : p \in (V - \Sigma) \times V^*$$

2. Nun ist zu beweisen, dass diese Grammatik auch L erzeugt. Sei nun $w \in \Sigma^+$. Zu zeigen ist nun, dass es eine Ableitung $S \overset{*}{\Rightarrow} w$ gibt, für die gilt, dass w ein korrekt geformter arithmetischer Ausdruck ist. Dies soll mittels Induktion über die Wortlänge $|w|$ bewiesen werden.
   Verankerung $|w| = 1$*:* "$\Rightarrow$"

   $$S \overset{*}{\Rightarrow} w \text{ mit } |w| = 1 \Rightarrow S \Rightarrow w \Rightarrow op \Rightarrow w \text{ ist korrekt geformt}$$

   "$\Leftarrow$"

   $w = op$ ist der einzige korrekt geformte Ausdruck der Länge $|w| = 1$, $S \to op \in P$

   somit folgt: $S \overset{*}{\Rightarrow} w = op$
   Annahme: Die Grammatik erzeuge für Wortlängen $1, \ldots, n$ korregt geformte arithmetische Ausdrücke.
   Schritt $|w| = n + 1$*:* "$\Rightarrow$" Es gilt:

   $S \overset{*}{\Rightarrow} w$ $|w| > 1 \Rightarrow w \neq op$, also gibt es drei Möglichkeiten, wie das Wort aussehen kann :
   1) $S \Rightarrow S + S \overset{+}{\Rightarrow} w_1 + w_2 = w$, mit $S \overset{*}{\Rightarrow} w_1 \wedge S \overset{*}{\Rightarrow} w_2$
   2) $S \Rightarrow S \cdot S \overset{+}{\Rightarrow} w_1 \cdot w_2 = w$, mit $S \overset{*}{\Rightarrow} w_1 \wedge S \overset{*}{\Rightarrow} w_2$
   3) $S \Rightarrow (S) \overset{+}{\Rightarrow} (w_1) = w$, mit $S \overset{*}{\Rightarrow} w_1$

   In jedem Fall ist $|w_1| \leq |w|, |w_2| \leq n$, also gilt nach Induktionsvoraussetzung, dass w_1 und w_2 korrekt geformte Ausdrücke sind. Und damit sind auch die Ausdrücke $w_1 + w_2, w_1 \cdot w_2$ und (w_1) wohlgeformte arithmetische Ausdrücke.
   "$\Leftarrow$" Sei nun w ein korrekt geformter Ausdruck der Länge $|w| = n + 1$. Das bedeutet, dass w nicht nur aus *op* besteht, sondern entweder $+, \cdot$ oder Klammern enthält. damit sind für:

   $$\begin{aligned} 1)&\quad w = w_1 + w_2 \\ 2)&\quad w = w_1 \cdot w_2 \\ 3)&\quad w = (w_1) \end{aligned}$$

   w_1 und w_2 korrekt geformte Ausdrücke der Länge $\leq n$.
   Nach Induktionsvoraussetzung gilt also $S \overset{*}{\Rightarrow} w_1$ und $S \overset{*}{\Rightarrow} w_2$. Damit ist:

   $$\begin{aligned} &1)\, S \Rightarrow S + S \overset{+}{\Rightarrow} w_1 + S \overset{+}{\Rightarrow} w_1 + w_2 = w \\ &2)\, S \Rightarrow S \cdot S \overset{+}{\Rightarrow} w_2 \cdot S \overset{+}{\Rightarrow} w_1 \cdot w_2 = w \\ &3)\, S \Rightarrow (S) \overset{+}{\Rightarrow} (w_1) = w \end{aligned}$$

   Also gilt in jedem Fall $S \overset{*}{\Rightarrow} w$ für Wörter der Wortlänge $|w| = n + 1$. Und dies war zu zeigen.

3. Angenommen L wäre regulär. Dann kann man einen Homomorphismus definieren mit:

   $$\begin{aligned} &h\,(op) = \epsilon \\ &h\,() = \epsilon \\ &h\,(\cdot) = \epsilon \\ &h\,('(') = a \\ &h\,(')') = b \end{aligned}$$

dann wäre auch die Sprache

$$L' = h(L) = \{a^r b^s \mid r, s \geq 0\} = \{a^n b^n \mid n \geq 0\}$$

regulär. Jedoch wissen wir von dieser, dass sie nicht regulär ist. Und somit kann auch L nicht regulär sein.

6.4 Aufgabe 23

6.4.1 Aufgabenstellung

Wandeln Sie folgende kontextfreie Grammatik $G = (\{S, A, B, C, D, E\}, \{a, b\}, P, S)$ mit

$$P \overset{def}{=} \{ \quad \begin{array}{rl} S \rightarrow AE|BE|C & A \rightarrow aAa|B \\ B \rightarrow bB|bb & C \rightarrow aCaa|D \\ D \rightarrow baD|abD|aa & E \rightarrow EE|\epsilon \quad \} \end{array}$$

mittels der aus der Vorlesung bekannten Algorithmen in eine reduzierte kontextfreie Grammatuik um. Dokumentieren Sie alle Zwischenergebnisse.

6.4.2 Lösung

Um G zu reduzieren sind drei Schritte zu tun.

1. Zuerst sind alle ϵ-Produktionen zu entfernen. Die Produktion $S \rightarrow \epsilon$ ist nur dann erlaubt, wenn $\epsilon \in L(G)$ ist.

2. Dann sind alle zyklischen Produktionen zu entfernen.

3. Zum Schluss sind dann noch alle überflüssigen Symbole zu entfernen. Das heiSSt also:

$$\begin{array}{lcl} a) & \forall A \in V - \Sigma \exists w \in \Sigma^*: & A \overset{*}{\Rightarrow} w \\ b) & \forall A \exists \alpha: & S \overset{*}{\Rightarrow} \alpha A \beta \end{array}$$

Dann also los. Doch zunächst schreibe ich die gegebenen Produktionen nochmals ausführlich auf, dann sieht man leichter, welche Produktionen zu entfernen sind :)

$$\begin{array}{rrr} S \rightarrow AE & S \rightarrow BE & S \rightarrow C \\ B \rightarrow bB & B \rightarrow bb & \\ D \rightarrow baD & D \rightarrow abD & D \rightarrow aa \\ A \rightarrow aAa & A \rightarrow B & \\ C \rightarrow aCaa & C \rightarrow D & \\ E \rightarrow EE & E \rightarrow \epsilon & \end{array}$$

Damit lässt sich doch schon mehr anfangen.

1. Zu der gegebenen Grammatik ergibt sich:

$$
\begin{aligned}
V_1 &= \{E\} \\
V_2 &= \{E\} \cup \{E\} = \{E\} = V_n \\
P_1 &= \{P\} \cup \{S \to A, S \to B, E \to E\} \\
P_2 &= P_1 - \{E \to \epsilon\} \\
P' &= P_2 \cup \{S' \to S\}
\end{aligned}
$$

Nun somit fällt die Produktion $E \to \epsilon$ herraus. Und man bekommt die Grammatik G' definiert durch:

$$G' = (\{S, S', A, B, C, D, E\}, \{a, b\}, P', S')$$

mit den Produktionen:

$$
\begin{aligned}
P' = \{ \quad & S \to AE, S \to BE, S \to C, S \to A, S \to B \\
& A \to aAa, A \to B \\
& B \to bB, B \to bb \\
& C \to aCaa, C \to D \\
& D \to baD, D \to abD, D \to aa \\
& E \to EE, E \to E \\
& S' \to S \qquad \}
\end{aligned}
$$

Das neue Startsymbol ist demnach S'.

2. Als nächstes sind alle Zyklischen Produktionen zu entfernen. Der Algorithmus liefert hier:

$$
\begin{aligned}
& V_{S'}^{(1)} = \{S\}, V_{S'}^{(2)} = \{S\} \cup \{A, B, C\} = \{S, A, B, C\} \\
& V_{S'}^{(3)} = \{S, A, B, C\} \cup \{A, B, C, D\} = \{S, A, B, C, D\} \\
& V_{S'}^{(4)} = \{S, A, B, C, D\} \cup \{A, B, C, D\} = \{S, A, B, C, D\} = V_{S'}^{(N)} \\
& V_{S}^{(1)} = \{A, B, C, D\}, V_{S}^{(2)} = \{A, B, C, D\}, V_{S}^{(3)} = \{A, B, C, D\} = V_{S}^{(N)} \\
& V_{A}^{(1)} = \{B\} = V_{A}^{(N)} \\
& V_{B}^{(N)} = \emptyset \\
& V_{C}^{(1)} = \{D\}, V_{C}^{(2)} = \{D\} = V_{C}^{(N)} \\
& V_{D}^{(1)} = \emptyset \\
& V_{E}^{(1)} = \{E\}, V_{E}^{(2)} = \{E\} = V_{E}^{(N)}
\end{aligned}
$$

Damit kommt man nun auf die Grammatik G'', welche definiert ist durch:

$$G'' = (\{S', S, A, B, C, D, E\}, \{a, b\}, P'', S')$$

mit:

$$
\begin{aligned}
P'' = \{ \quad & P' - \{S \to A, S \to B, S \to C, A \to D, C \to D, E \to E, S' \to S\} \cup \\
& \{ S' \to AE, S' \to BE, S' \to aAa, S' \to bB, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD \\
& S' \to abD, S' \to aa, S \to aAa, S \to bB, S \to bB, S \to bb, S \to aCaa, S \to baD, S \to abD, S - \\
& A \to bB \\
& C \to baD, C \to abD, C \to aa \\
& E \to EE\}
\end{aligned}
$$

man gelangt so schlieSSlich zu den Produktionen:

$$
\begin{aligned}
P'' = \{ \quad & S' \to AE, S' \to BE, S' \to aAa, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD, S' \to abD, S' \to aa \\
& S \to AE, S \to BE, S \to aAa, S \to bB, S \to bb, S \to aCaa, S \to baD, S \to abD, S \to aa \\
& A \to aAa, A \to bB, A \to bb \\
& B \to bB, B \to bb \\
& C \to baD, C \to abD, C \to aCaa, C \to aa \\
& D \to baD, D \to abD, D \to aa \\
& E \to EE
\end{aligned}
$$

3. Zum Schluss sind alle Symbole zu entfernen, welche nicht benötigt werden. Dies funktioniert analog, wie im Skript auf Seite 55. Der Algorithmus greift in zwei Schritten:
   1. Schritt:

$$
\begin{aligned}
& V_1 = \{S', S, A, B, C, D\} \\
& V_2 = \{S', S, A, B, C, D\} = V_N \\
& \overline{G} = \left(\overline{V}, \{a, b\}, \overline{P}, S'\right) \text{ mit} \\
& \overline{V} = \{S', S, A, B, C, D\} \cup \{a, b\} \\
& \overline{P} = P'' \cap \left(\overline{V} \times \overline{V}^*\right) \\
& \overline{P} = P'' - \{S' \to AE, S' \to BE, S \to AE, S \to BE, E \to EE\}
\end{aligned}
$$

   2. Schritt:

$$
\begin{aligned}
& H_0 = \{S'\}, H_1 = \{S'\} \cup \{A, B, C, D\} = \{S', A, B, C, D\} \\
& H_2 = \{S', A, B, C, D\} = H_N \\
& G''' = (V''', \{a, b\}, P''', S') \text{ mit} \\
& V''' = \{S', A, B, C, D\} \cup \{a, b\} \\
& P''' = \overline{P} \cap \left(V''' \times (V''')^*\right)
\end{aligned}
$$

   so gelangt man zu den endgültigen Produktionen:

$$
\begin{aligned}
P''' = \{ \quad & S' \to aAa, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD, S' \to abD, S \to aa \\
& A \to aAa, A \to bB, A \to bb \\
& B \to bB, B \to bb \\
& C \to aCaa, C \to baD, C \to abD, C \to aa \\
& D \to baD, D \to abD, D \to aa \qquad \}
\end{aligned}
$$

   dies ist nun die endgültige reduzierte Grammatik.

7 Übungsblatt 7

7.1 Aufgabe 24

7.1.1 Aufgabenstellung

Konstruieren Sie eine kontextfreie Grammatik G für die Sprache

$$L \overset{def}{=} \left\{wcx \in \{a, b, c\}^* \mid w, x \in \{a, b\}^* \wedge w \neq x\right\}$$

und begründen Sie, dass ihre Grammatik G die Sprache L generiert.

7.1.2 Lösung

Um unterschiedliche Worte zu erhalten gilt es zwei Fälle zu beachten:

1. einmal ist es möglich, dass die beiden Worte eine unterschiedliche Länge haben. Sprich die Wortlänge von w ist ungleich der von x ($|w| \neq |x|$). Oder aber

2. die Wortlänge ist gleich, jedoch unterscheiden sich beide Worte an mindestens einer Stelle. Es gilt also:

$$|w| = |x| \wedge \exists i \colon 1 \leq i \leq n \wedge x_i \neq w_i$$

Nun gilt es also eine Grammatik zu konstruieren, welche jeweils einen der beiden Fälle beachtet. Dazu betrachten wir Worte $v \in \{a, b, c\}^*$ mit

$$G \overset{*}{\Rightarrow} v \wedge (\text{ entweder Fall 1 oder Fall 2 ist erfüllt})$$

zunächst betrachten wir die Produktionen für den ersten Fall, es sei dazu OE $|w| > |x|$:

$$\begin{array}{lcl} S & \rightarrow & TS_L \\ S_L & \rightarrow & TS_LT \,|\, TS_L \,|\, c \\ T & \rightarrow & a \,|\, b \end{array}$$

Dann gilt: $S \Rightarrow TS_L \overset{*}{\Rightarrow} Tw'cx \Rightarrow wcx$ mit $|w| > |x|$. Ebenso wählt man die Produktionen für den Fall $|w| < |x|$:

$$\begin{array}{lcl} S & \rightarrow & S_RT \\ S_R & \rightarrow & TS_RT \,|\, S_RT \,|\, c \\ T & \rightarrow & a \,|\, b \end{array}$$

und es gilt $S \Rightarrow S_RT \overset{*}{\Rightarrow} wcx'T \Rightarrow wcx$ mit $|w| < |x|$. Und nun zum zweiten Fall. Hier gilt nun zunächst zu spezifizieren, welcher Unterschied gefunden wird. Wir betrachten dazu zuerst $w_i = a \wedge x_i = b$:

$$\begin{array}{lcl} S & \rightarrow & S_abT' \\ S_a & \rightarrow & TS_aT \,|\, aT'c \\ T' & \rightarrow & T'T' \,|\, T \,|\, \epsilon \\ T & \rightarrow & a \,|\, b \end{array}$$

Mit diesen Produktionen gilt offensichtlich:

$$S_a \overset{n}{\rightarrow} T^nS_aT^n \rightarrow T^naT'cT^n$$

Da die Wortlänge der beiden Wörter gleich ist, stehen vor einem a genausoviele Terminalsymbole wie nach einem c. Also gilt:

$$\begin{array}{l} S \rightarrow S_abT' \overset{i-1}{\rightarrow} T^{i-1}S_aT^{i-1}bT' \Rightarrow T^{i-1}aT'cT^{i-1}bT' \\ \quad \overset{*}{\Rightarrow} w_1 \dots w_{i-1}aw_{i+1} \dots w_ncx_1 \dots x_{i-1}bx_{i+1} \dots x_n \end{array}$$

und somit unterscheiden sich die Worte x und w genau an derstelle i mit $w_i = a \wedge x_i = b$. Ebenso verfährt man mit dem Fall$w_i = b \wedge x_i = a$:

$$\begin{aligned} S &\rightarrow S_b a T' \\ S_b &\rightarrow T S_b T \mid b T' c \\ T' &\rightarrow T'T' \mid T \mid \epsilon \\ T &\rightarrow a \mid b \end{aligned}$$

Mit diesen Produktionen gilt offensichtlich

$$\begin{aligned} S \rightarrow S_b a T' \stackrel{i-1}{\rightarrow} T^{i-1} S_b T^{i-1} a T' \Rightarrow T^{i-1} a T' c T^{i-1} b T' \\ \stackrel{*}{\Rightarrow} w_1 \dots w_{i-1} b w_{i+1} \dots w_n c x_1 \dots x_{i-1} a x_{i+1} \dots x_n \end{aligned}$$

und somit unterscheiden sich die Worte x und w genau an derstelle i mit $w_i = b \wedge x_i = a$. SchlieSSlich fasst man nun noch all diese Produktionen zusammen zu:

$$\begin{aligned} P = \{ \quad S &\rightarrow T S_L \mid S_R T \mid S_a b T' \mid S_b a T' \\ S_L &\rightarrow T S_L T \mid T S_L \mid c \\ S_R &\rightarrow T S_R T \mid S_R T \mid c \\ S_a &\rightarrow T S_a T \mid a T' c \\ S_b &\rightarrow T S_b T \mid b T' c \\ T' &\rightarrow T'T' \mid T \mid \epsilon \\ T &\rightarrow a \mid b \qquad \} \end{aligned}$$

Diese erfüllen nun die Aufgabe.

7.2 Aufgabe 25

7.2.1 Aufgabenstellung

Konstruieren Sie einen Kellerautomaten $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ der die Sprache

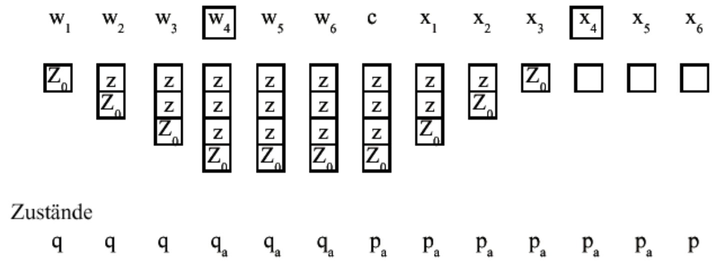
$$L_n \stackrel{def}{=} \left\{ wcx \in \{a, b, c\}^* \mid w, x \in \{a, b\}^n \wedge w \neq x \right\}$$

für ein festes $n \in N$ erzeugt. Weiter soll für die Anzahl der Zustände gelten: $|Q| = O(n)$. Begründen Sie, dass Ihr Kellerautomat M die Sprache L_n generiert.

7.2.2 Lösung

Jedes Wort $v \in L_n$ der Sprache besteht aus zwei gleichlangen Teilwörtern w und x. Diese sind durch ein c getrennt. Betrachtet man die Überlegungen der letzten Aufgabe, so stellt man schnell fest, dass sie w und x an mindestens einer Stelle i unterscheiden müssen. Dies soll auch das Bild[1] illustrieren:

[1] Das Bild stammt von http://www.cs.uni-frankfurt.de/˜demmer



Wie spezifiziert man nun den Automaten? Nun die Idee ist die folgende:

- die Länge der Teilworte ($|w| = |x| = n$) zählt man mittels der Zustände
- der Automat merkt sich mittels des Kellers, an welcher Stelle sich w und x das erste mal unterscheiden. Und zwar wie folgt:
  - Stack aufbauen
  - an der i-ten Stelle wird geraten $w_i = a \wedge x_i = b$ oder eben $w_i = b \wedge x_i = a$
  - nun geht man über alle weiteren Symbole bis zum c und lässt den Stack unberührt,
  - dann lässt man den Stack abbauen, bis das Startsymbol erreicht ist
  - der Automat kontrolliert nun über die Zustände, ob sich w_i und x_i wirklich unterscheiden,
  - nun bleibt nur noch zu überprüfen, ob auch beide Teilwörter die selbe Länge haben. Dies kann der Automat jedoch, da er die Länge von w mitzählen konnte, bis er c erreicht hatte.
  - der Automat akzeptiert, wenn die Länge der Teilworte gleich ist und sie sich an der Stelle i unterscheiden.

Formal sieht das ganze so aus: Man definiert den Automaten:

$$M = (Q, \{a, b, c\}, \{z, z_0\}, \delta, q_0, z_0, \emptyset)$$

mit:

$$\begin{aligned} Q = \{ \quad & q_0, \ldots, q_{n-1}, \\ & q_1^a, \ldots, q_n^a, \\ & q_1^b, \ldots, q_n^b, \\ & p_0, \ldots, p_n, \\ & p_0^a, \ldots, p_{n-1}^a, \\ & p_0^b, \ldots, p_{n-1}^b \quad \} \end{aligned}$$

damit ist die Anzahl der Zustände offensichtlich ein Vielfaches von n und es gilt $|Q| = O(n)$.

Nun fehlt nur noch die Definition der Übergangsfunktion δ. In der Übung hieSS es, man solle dies in der Bibliothek nachschlagen, um Euch jedoch den

Weg zu ersparen (wer weiSS wie lang die Bib noch auf hat) soll diese nun hier folgen:

$$
\begin{array}{ll}
\delta(q_0, a, z_0) = \{(q_1, zz_0), (q_1^a, z_0)\} & \delta(q_0, b, z_0) = \{(q_1, zz_0), (q_1^b, z_0)\} \\
\delta(q_1, a, z) = \{(q_2, zz), (q_2^a, z)\} & \delta(q_1, b, z) = \{(q_2, zz), (q_2^b, z)\} \\
\delta(q_1^a, \sigma, z_0) = \{(q_2^a, z_0)\} & \delta(q_1^b, \sigma, z_0) = \{(q_2^b, z_0)\} \quad \sigma \in \{a, b\} \\
\delta(q_2, a, z) = \{(q_3, zz), (q_3^a, z)\} & \delta(q_2, b, z) = \{(q_3, zz), (q_3^b, z)\} \\
\delta(q_2^a, \sigma, \overline{z}) = \{(q_3^a, \overline{z})\} & \delta(q_2^b, \sigma, \overline{z}) = \{(q_3^b, \overline{z})\} \quad \overline{z} \in \{z, z_0\} \\
\vdots & \\
\delta(q_{n-1}, a, z) = \{(q_n^a, z)\} & \delta(q_{n-1}, b, z) = \{(q_n^b, z)\} \\
\delta(q_{n-1}^a, \sigma, \overline{z}) = \{(q_n^a, \overline{z})\} & \delta(q_{n-1}^b, \sigma, \overline{z}) = \{(q_n^b, \overline{z})\} \\
\delta(q_n^a, c, \overline{z}) = \{(p_0^a, \overline{z})\} & \delta(q_n^b, c, \overline{z}) = \{(p_0^b, \overline{z})\} \\
\delta(p_0^a, \sigma, z) = \{(p_1^a, \epsilon)\} & \delta(p_0^b, \sigma, z) = \{(p_1^b, \epsilon)\} \\
\delta(p_0^a, b, z_0) = \{(p_1, z_0)\} & \delta(p_0^b, a, z_0) = \{(p_1, z_0)\} \\
\delta(p_1^a, \sigma, z) = \{(p_2^a, \epsilon)\} & \delta(p_1^b, \sigma, z) = \{(p_2^b, \epsilon)\} \\
\delta(p_1^a, b, z_0) = \{(p_2, z_0)\} & \delta(p_1^b, a, z_0) = \{(p_2, z_0)\} \\
\delta(p_1, \sigma, z_0) = \{(p_2, z_0)\} & \\
\vdots & \\
\delta(p_{n-1}^a, b, z_0) = \{(p_n, z_0)\} & \delta(p_{n-1}^b, a, z_0) = \{(p_n, z_0)\} \\
\delta(p_{n-1}, \sigma, z_0) = \{(p_n, z_0)\} & \\
\delta(p_n, \epsilon, z_0) = \{(p_n, \epsilon)\} &
\end{array}
$$

7.3 Aufgabe 26

7.3.1 Aufgabenstellung

Gegeben sei folgende kontextfreie Grammatik $G = (\{S\}, \{if, then, else, expr, stmt\}, P, S)$ mit

$$
P \overset{def}{=} \{ \quad \begin{array}{rl} S \rightarrow & \text{if expr then } S \text{ else } S \\ S \rightarrow & \text{if expr then } S \\ S \rightarrow & \text{stmt} \end{array} \quad \}
$$

1. Zeigen Sie, dass G mehrdeutig ist.
2. Geben Sie eine eindeutige Grammatik für $L(G)$ an.
3. Geben Sie für die in Aufgabe 22 beschriebene Sprache L eine mehrdeutige und eine eindeutige Grammatik an.

7.3.2 Lösung

1. Um zu zeigen, dass die Grammatik mehrdeutig ist, genügt es ein Wort zu finden, zu dem es zwei Ableitungen gibt. Dieses Wort ist:

   if expr then if expr then stmt else stmt

   Für dieses Wort gibt es zwei Ableitungen:

(a) die erste ist:

$$\begin{aligned} S &\to \text{ if expr then } S \text{ else } S \\ &\to \text{if expr then if expr then } S \text{ else } S \\ &\to \text{if expr then if expr then stmt else } S \\ &\to \text{if expr then if expr then stmt else stmt} \end{aligned}$$

(b) eine zweite Ableitung für dieses Wort ist:

$$\begin{aligned} S &\to \text{ if expr then } S \\ &\to \text{if expr then if expr then } S \text{ else } S \\ &\to \text{if expr then if expr then stmt else } S \\ &\to \text{if expr then if expr then stmt else stmt} \end{aligned}$$

es existieren also zwei Linksableitungen und somit ist die Grammatik mehrdeutig.

2. Die Grammatik $G' = (\{S', S_n, S_m\}, \{if, then, else, expr, stmt\}, P', S')$ mit

$$\begin{aligned} P' = \{ \qquad\qquad & S' \to S_n \mid S_m \\ & S_n \to \text{ if expr then } S' \mid \text{if expr then } S_m \text{ else } S_n \\ & S_m \to \text{if expr then } S_m \text{ else } s_n \mid stmt \qquad \} \end{aligned}$$

ist eindeutig.

3. Eine Grammatik für die Aufgabe 22 ist: $G = (\{\epsilon\}, \{(,), \cdot, +, op\}, P, E)$ mit

$$P = \{E \to E + E \mid E \cdot E \mid (\,E\,) \mid op\}$$

Diese Grammatik ist mehrdeutig, denn zum Beispiel das Wort $op \cdot op + op$ besitzt 2 verschiedene Linksableitungen:

(a) $E \Rightarrow E \cdot E \Rightarrow op \cdot E \Rightarrow op \cdot E + E \Rightarrow op \cdot op + E \Rightarrow op \cdot op + op$

(b) $E \Rightarrow E + E \Rightarrow op + E \Rightarrow op + E \cdot E \Rightarrow op + op \cdot E \Rightarrow op + op \cdot op$

Eine eindeutige Grammatik für diese Aufgabe ist die Grammatik $G' = (\{E\}, \{(,), \cdot, +, op\}, P', E)$ mit:

$$P' = \{E \to op + E \mid op \cdot E \mid (E) + E \mid (E) \cdot E \mid (E) \mid op\}.$$

8 Übungsblatt 8

8.1 Aufgabe 27

8.1.1 Aufgabenstellung

Geben Sie einen (wenn möglich deterministischen) PDA für die Sprache:

$$L_k = \{a^n b_i^n \mid n \in N,\, 1 \le i \le k\}$$

an. Begründen Sie, warum der PDA die Sprache akzeptiert.

8.1.2 Lösung

Die Grundidee ist, zuerst alle $a's$ auf den Keller zu speichern, um dann das erste b_i einzulesen und ein Zeichen vom Stack abzubauen. Dann werden weiterhin nur noch für b_i Zeichen abgebaut. Landet man schlieSSlich bei einem leeren Keller, so akzeptiert man.

Ein deterministischer PDA, der dies erfüllt sieht zum Beispiel wie der folgende aus:

$$M = (\{q_0, \ldots, q_k, \tilde{q}\}, \{a, b_1, \ldots, b_k\}, \{Z, Z_0\}, \delta, Z_0, \{\tilde{q}\})$$

die Übergangsfunktion δ ist durch folgende Definitionen festgelegt. Zuerst schiebt man für jedes a ein Z auf den Keller.

$$\begin{aligned} \delta(q_0, a, Z_0) &= (q_0, Z, Z_0) \\ \delta(q_0, a, Z) &= (q_0, Z, Z_0) \end{aligned}$$

Abhängig vom nun gelesenem b_i geht man nun in einen der Zustände $q_1 \vee \ldots \vee q_k$ und konsummiert ein Zeichen des Kellers:

$$\begin{aligned} \delta(q_0, b_1, Z) &= (q_1, \epsilon) \\ \delta(q_0, b_2, Z) &= (q_2, \epsilon) \\ &\vdots \\ \delta(q_0, b_k, Z) &= (q_k, \epsilon) \end{aligned}$$

Nun sind weitere $n-1$ Zeichen vom Stack zu nehmen. Dabei bleibt man jedoch immer im gleichen Zustand:

$$\begin{aligned} \delta(q_1, b_1, Z) &= (q_1, \epsilon) \\ &\vdots \\ \delta(q_k, b_1, Z) &= (q_k, \epsilon) \end{aligned}$$

Und schlieSSlich gilt es genau dann zu akzeptieren, wenn man kein Zeichen mehr vorfindet und man nur noch das Kellerstartsymbol Z_0 auf dem Stack findet:

$$\begin{aligned} \delta(q_1, \epsilon, Z_0) &= (\tilde{q}, \epsilon) \\ &\vdots \\ \delta(q_k, \epsilon, Z_0) &= (\tilde{q}, \epsilon) \end{aligned}$$

Dieser PDA ist deterministisch, da es für jedes Symbol nur eine δ-Definition existiert. Somit gilt:

1) $\forall q \in \{q_0, \ldots, q_k, \tilde{q}\} \wedge \forall a \in (\Sigma \cup \{\epsilon\}) \wedge \forall A \in \{Z, Z_0\} : |\delta(q, a, A)| \leq 1$
2) $\delta(q, \epsilon, A) \neq \emptyset \Rightarrow \delta(q, a, A) \neq \emptyset \forall a \in \Sigma \forall A \in \{Z, Z_0\}$

8.2 Aufgabe 28

8.2.1 Aufgabenstellung

Wandeln Sie folgende Grammatik:

$$G = (\{A, S\}, \{a, b\}, P, S)$$

mit

$$P = \{S \to AA \,|\, a; S \to SS \,|\, b\}$$

in Greibach Normalform um. Dokumentieren Sie alle Zwischenschritte.

8.2.2 Lösung

Die Lösung erfolgt in zwei Schritten zunächst ist die Grammatik G in Chomsky-Normalform zu bringen, sprich sie ist zu reduzieren. Im zweiten Schritt dann wird sie schlieSSlich in Greibach-Normalform (GNF)gebracht.

Schritt 1:

Durch einführen des neuen Startsymbol S' wird die Grammatik G in Chomsky-Normalform (CNF) gebracht. Damit ergibt sich die Grammatik $G = (\{A, S, S'\}, \{a, b\}, P', S')$ mit den Produktionen:

$$\begin{aligned} P' = \{ \quad & S' \to AA \,|\, a \\ & S \to AA \,|\, a \\ & A \to SS \,|\, b \quad \} \end{aligned}$$

Das war's denn auch schon für diesen Schritt.

Schritt 2:

Nun gilt es die Grammatik von CNF auf GNF zu bringen. Dazu benenne ich zunächst die Nichtterminale wie folgt um:

$$\begin{aligned} S' &= A_1 \\ S &= A_2 \\ A &= A_3 \end{aligned}$$

Somit gilt:

$$\begin{aligned} P' = \{ \quad & A_1 \to A_3A_3 \,|\, a \\ & A_2 \to A_3A_3 \,|\, a \\ & A_3 \to A_2A_2 \,|\, b \quad \} \end{aligned}$$

Die Umwandlung wird nochmals in drei Schritte unterteilt.

1. Für $i = 1, 2$ gilt bereits:

   $$A_i \to A_j\gamma \text{ mit } j > i$$

   Also bleibt nur noch A_3 zu behandeln. Man ersetzt dazu $A_3 \to A_3A_3$ durch $A_3 \to A_3A_3A_2 \,|\, aA_2$. So gelangt man zu den Produktionen:

   $$\begin{aligned} P' = \{ \qquad & A_1 \to A_3A_3 \,|\, a \\ & A_2 \to A_3A_3 \,|\, a \\ & A_3 \to A_3A_3A_2 \,|\, aA_2 \,|\, b \quad \} \end{aligned}$$

Nun gilt es $A_3 \to A_3A_3A_2$ durch $A_3 \to aA_2B_3 \mid bB_3$ zu ersetzen. Dann wird $B_3 \to A_3A_2 \mid A_3A_2B_3$ eingefügt. Somit gelangt man zu:

$$\begin{aligned} P' = \{ \quad A_1 &\to A_3A_3 \mid a \\ A_2 &\to A_3A_3 \mid a \\ A_3 &\to aA_2B_3 \mid aA_2 \mid b \\ B_3 &\to A_3A_2 \mid A_3A_2B_3 \quad \} \end{aligned}$$

2. Nun ersetzt man $A_2 \to A_3A_3$ durch $A_2 \to aA_2B_3A_3 \mid bB_3A_3 \mid aA_2A_3 \mid bA_3$ weiterhin ersetzt man $A_1 \to A_3A_3$ durch $A_1 \to aA_2B_3A_3 \mid bB_3A_3 \mid aA_2A_3 \mid bA_3$. Nun haben alle Produktionen haben nun als erstes Zeichen ein Terminalsymbol.

3. SchlieSSlich ist $B_3 \to A_3A_2$ durch $B_3 \to aA_2B_3A_2 \mid bB_3A_2 \mid aA_2A_2 \mid bA_2$ und $B_3 \to A_3A_2B_3$ durch $B_3 \to aA_2B_3A_2B_3 \mid bB_3A_2B_3 \mid aA_2A_2B_3 \mid bA_2B_3$. Benennt man nun die Nichtterminale wieder zurück, so erhält man die Grammatik $\tilde{G}$ in GNF.

$$\tilde{G} = \left(\{S', S, A, B_3\}, \{a, b\}, \overline{P}, S'\right)$$

mit den Produktionen:

$$\begin{aligned} \overline{P} = \{ \quad S' &\to aSB_3A \mid bB_3A \mid aSA \mid bA \mid a \\ S &\to aSB_3A \mid bB_3A \mid aSA \mid bA \mid a \\ A &\to aSB_3 \mid bB_3 \mid aS \mid b \\ B_3 &\to aSB_3|bB_3|aSS|bS|aSB_3SB_3|bB_3SB_3|aSSB_3|bSB_3 \quad \} \end{aligned}$$

8.3 Aufgabe 29

8.3.1 Aufgabenstellung

Beweisen Sie, dass die folgenden Sprachen nicht kontextfrei sind:

1. $L_1 \stackrel{def}{=} \left\{a^{n^2} \mid n \in N\right\}$
2. $L_2 \stackrel{def}{=} \{a^{n_1}ba^{n_2}ca^{m_1}ba^{m_2} \mid n_1 + n_2 = m_1 + m_2 \land n_1 \neq m_1\}$
3. $L_3 \stackrel{def}{=} \{wcx \mid w, x \in \{a,b\}^* \land w \neq x \land |w| = |x|\}$

8.3.2 Nachtrag zur Vorlesung: Abschlusseigenschaften von Kontextfreien Sprachen

Hier nur eine kurze Aufzählung der Abschlusseigenschaften von kontextfreien Sprachen. Diese sind aus unbekanntem Grund nicht in der Vorlesung behandelt worden.
Kontextfreie Sprachen sind abgeschlossen gegenüber:

- Vereinigung,
- Konkatenation,

- Kleene Star,
- Schnitt mit regulären Mengen,
- Homomorphismen,
- inverse Homomorphismen

Nicht abgeschlossen sind kontextfreie Sprachen gegenüber:

- Schnitt
- Komplement

Damit ist es doch schon um einiges einfacher die Aufgaben zu lösen.

8.3.3 Lösung

1. Wir werden einen Widerspruchsbeweis anwenden, um zu zeigen, dass die Sprache L_1 nicht kontextfrei ist. Angenommen die Sprache L_1 wäre kontextfrei. Nach dem Pumpinglemma existiert eine Konstante $k \in N$, so dass es für ein Wort $z = n^{k^2}$ eine Zerlegung $z = uv^iwx^iy$ existiert, mit $|vx| \geq 1$, $|vwx| \leq k$ und $uv^iwx^iy \in L_1$ für $i \geq 0$. Insbesondere gilt:

   $$uv^2wx^2y \in L_1$$

   das bedeutet, dass $\left|uv^2wx^2y\right|$ ist Quadratzahl. Es gilt jedoch:

   $$k^2 < k^2 + |vx| = \left|uv^2wx^2y\right| = k^2 + |wx| \leq k^2 + k < k^2 + 2 \cdot k + 1 = (k+1)^2$$

   also ist $\left|uv^2wx^2y\right|$ keine Quadratzahl. Damit ist L_1 nicht kontextfrei.

2. Wieder folgt ein Widerspruchsbeweis, mit dem wir zeigen werden, dass L_2 nicht kontextfrei ist. Angenommen L_2 wäre kontextfrei. Sei $k \in N$ die Konstante des Pumpinglemmas. Wir betrachten das Wort $z = a^{k!+k}ba^kca^kba^{k!+k} \in L_2$. Dann existiert eine Zerlegung $z = uvwxy$ mit $|vx| \geq 1$, $|vwx| \leq k$ und $uv^iwx^iy \in L_2$ für alle $i \geq 0$. Für das Pumpen gelten die folgenden Argumente:

   (a) v und x enthalten kein b und kein c, da sonst ein Widerspruch beim Pumpen entstehen würde,

   (b) v ist links von c und x ist rechts vom c positioniert, da sonst ein Widerspruch entstehen würde, da sonst die Bedingung $n_1 + n_2 = m_1 + m_2$ verletzt würde.

   (c) Also ist vwx ein Teil von a^kca^k. Dabei ist c ein Teil von w.

   (d) SchlieSSlich ist $|v| = |x| \geq 1$, da sonst auch die Bedingung $n_1 + n_2 = m_1 + m_2$ verletzt würde.

   Damit gilt also,

   $$uv^iwx^iy = a^{k!+k}ba^{k-|v|}\left(a^{|v|}\right)^i c \left(a^{|v|}\right)^i a^{k-|v|}ba^{k!+k} \in L_2$$

für alle $i \in N$. Setzt man nun $i = \frac{k!}{|v|} + 1$, so erhält man:

$$uv^iwx^iy = a^{k!+k}ba^{k-|v|}\left(a^{|v|}\right)^{\frac{k!}{|v|}+1}c\left(a^{|v|}\right)^{\frac{k!}{|v|}+1}ba^{k!+k}$$
$$= a^{k!+k}ba^{k!+k}ca^{k!+k}ba^{k!+k}$$

Das verletzt jedoch die Bedingung, dass $n_1 \neq m_1$ ist. Damit ist L_2 nicht kontextfrei.

3. Auch diese Sprache ist nicht kontextfrei. Der Beweis erfolgt wieder über Widerspruch. Angenommen L_3 ist kontextfrei, dann ist auch die Sprache

$$L' = L_3 \cap \{a^nba^mca^rba^s \mid n, m, r, s \geq 0\}$$

kontextfrei. Denn kontextfreie Sprachen sind abgeschlossen unter dem Schnitt mit regulären Mengen. Somit gilt:

$$\begin{aligned} L' &= \{a^{n_1}ba^{n_2}ca^{m_1}ba^{m_2} \mid n_1 + n_2 = m_1 + m_2 \wedge (n_1 \neq m_1 \vee n_2 \neq m_2)\} \\ &= \{a^{n_1}ba^{n_2}ca^{m_1}ba^{m_2} \mid n_1 + n_2 = m_1 + m_2 \wedge n_1 \neq m_1\} = L_2 \end{aligned}$$

denn aus $n_2 \neq m_2 \Rightarrow n_1 \neq m_1$. Zum Beweis nehmen wir mal an, dass $n_1 = m_1$ wäre. Dann gilt für $n_2 \neq m_2 \Rightarrow n_1 + n_2 = m_1 + m_2 \Rightarrow n_1 = m_1$, dies ist jedoch ein Widerspruch zur Annahme. Also gilt obige Gleichung. Und von L_2 wissen wir schon, dass es nicht kontextfrei ist, also kann auch L_3 nicht kontextfrei sein.

8.4 Aufgabe 30

8.4.1 Aufgabenstellung

Sei $L \subseteq \Sigma^*$ eine beliebige Sprache. In Aufgabe 12 wurde der Operator FH eingeführt:

$$FH\,(L) \stackrel{def}{=} \{x \mid \exists y \in \Sigma^*: (|x| = |y| \wedge xy \in L)\}$$

Zeigen Sie, dass die Klasse der kontextfreien Sprachen nicht unter FH abgeschlossen ist.

Hinweis

Zeigen Sie durch Widerspruch, dass die Sprache $FH\,(L)$ mit

$$L \stackrel{def}{=} \left\{a^nb^nc^m\#d^{3m}\# \mid n, m \in N\right\}$$

nicht kontextfrei ist.

8.4.2 Lösung

Nun, dann hält man sich wohl besser an den Hinweis.

Behauptung: Kontextfreie Sprachen sind nicht abgeschlossen gegenüber FH.

Beweis: Dazu betrachtet man die Spache $L = \left\{a^nb^nc^m\#d^{3\cdot m}\# \mid n, m \in N\right\}$ dann gilt:

$$\begin{aligned} FH\,(L) \cap \{a^nb^mc^r\# \mid n.m.r \geq 0\} &= \{a^nb^nc^m\# \mid n, m \geq 0 \wedge (2 \cdot n + m) = 3 \cdot m\} \\ &= \{a^nb^nc^n\# \mid n \geq 0\} \end{aligned}$$

Diese Sprache müsste kontextfrei sein. Es gilt dann also:

$$\begin{aligned} |a^n b^n c^m \#| &= \left|d^{3\cdot m} + 1\right| \\ \Leftrightarrow 2 \cdot n + n + 1 &= 3 \cdot m + 1 \\ \Leftrightarrow n &= m \end{aligned}$$

Betrachtet man nun den Homomorphismus, mir folgenden Überführungen:

$$\begin{aligned} h(a) &= a \\ h(b) &= b \\ h(c) &= c \\ h(\#) &= \epsilon \end{aligned}$$

Dann wäre die Sprache $h(\{a^n b^n c^n \# \mid n \geq 0\}) = \{a^n b^n c^n \mid n \geq 0\}$ kontextfrei. Dies ist jedoch nicht der Fall. Somit sind die kontextfreien Sprachen nicht abgeschlossen gegenüber FH.