Prof. J. Esparza
S. Sickert, J. Krämer
Vielen Dank an Jan Wagener für die erweiterten Aufgabenlösungen

Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2017 – Übungsblatt 5

Übungsblatt

Wir unterscheiden zwischen Übungs- und Abgabeblättern. Auf diesem *Übungsblatt* finden Sie eine Übersicht über die Kernaspekte, die Sie in Kalenderwoche 22 in den Tutorien diskutieren, üben und vertiefen. Die Aufgaben auf diesem Blatt dienen dem Üben und Verstehen des Vorlesungsstoffes, sowie dem *eigenständigen Erarbeiten* der Kernaspekte. Außerdem sollen Ihnen diese Aufgaben auch helfen, ein Gefühl dafür zu bekommen, was Sie inhaltlich in der Klausur erwartet. Klausuraufgaben können jedoch deutlich von den hier gestellten Aufgaben abweichen. Abschreiben und Auswendiglernen von Lösungen wird Ihnen daher keinen dauerhaften Erfolg in der Vorlesung bringen. Fragen zu den Übungsblättern können Sie montags bis donnerstags von 12 Uhr bis 14 Uhr in der *THEO-Sprechstunde* in Raum 03.11.034 stellen.

Kernaspekte

K5.1 korrektes Wiedergeben der folgenden Definitionen

- Myhill-Nerode-Relation
- Äquivalenzklasse
- inäquivalente Zustände in einem DFA

K5.2 zu einem gegebenen DFA den minimalen DFA angeben
K5.3 zu einem gegebenen Automaten den Quotientenautomaten angeben
K5.4 für reguläre Sprachen Sprachgleichheit mithilfe von Minimierung prüfen
K5.5 mithilfe der Myhill-Nerode-Relation beweisen bzw. widerlegen, dass eine gegebene Sprache regulär ist

AUFGABE 5.1.

Stufe B

Geben Sie jeweils eine formale Sprachen über dem Alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}\}$ mit folgenden Eigenschaften an:

(a) endlich (b) unendlich und regulär (c) nicht regulär und kontextfrei

Beschreiben Sie in eigenen Worten, wie Sie bei der Konstruktion der Sprachen vorgegangen sind, um sicher zu stellen, dass die Sprache die gewünschte Eigenschaft hat.

Erweiterte Lösungsskizze
Sei $\Sigma = \{a, b\}$.
Gib jeweils eine formale Sprache A über Σ mit gegebenen Eigenschaften an.

(a) Eine endliche Sprache

$$A = \emptyset$$

$\Longrightarrow$ Es ist lediglich darauf zu achten, dass die Sprache nur endlich viele Wörter enthält. "Keine Wörterïst wohl die einfachste Art, das sicherzustellen.

(b) Eine unendliche, reguläre Sprache

$$A = \Sigma^*$$

$\Longrightarrow$ Die Sprache soll unendlich und regulär sein, d.h. sie muss durch einen regulären Ausdruck darstellbar sein. Für Σ^* ist das offensichtlich der Fall: $(a|b)^*$
Außerdem enthält Σ^* auch unendlich viele Wörter, z.B. alle der Form a^n, $n \in \mathbb{N}_0$.

(c) Eine nicht reguläre, kontextfreie Sprache

$$A = \{a^n b^n \mid n \in \mathbb{N}_0\}$$

$\Longrightarrow$ Diese Sprache ist bekanntlich nicht regulär (leicht mit Pumping-Lemma für reguläre Sprachen zu zeigen).
Allerdings ist sie kontextfrei, da es möglich ist, eine kontextfreie (Typ-2) Grammatik zu konstruieren, die genau A erzeugt:

$G = (\{S\}, \Sigma, P, S)$ mit
$P : S \to aSb \mid \varepsilon$

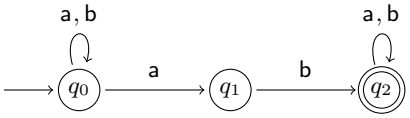
AUFGABE 5.2. Stufe C

In der Vorlesung haben Sie ein Verfahren zur Minimierung von DFAs gesehen. Wir erweitern diesen Algorithmus, damit er zusätzlich das unterscheidende Wort w für ein Paar inäquivalenter Zuständen berechnet.

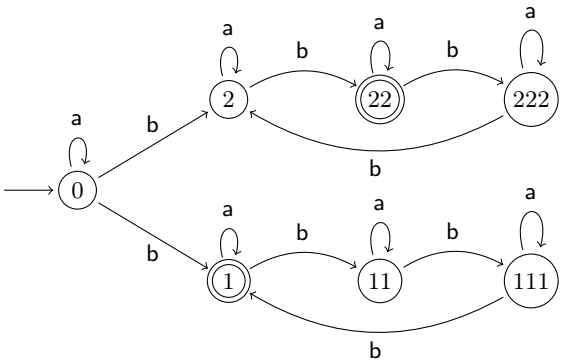
(a) Erweitern Sie den Minimierungsalgorithmus so, dass er für jedes Paar von inäquivalenten Zuständen $q, p \in \mathsf{Q}$ eines DFAs $\mathsf{D} = (\mathsf{Q}, \Sigma, \delta, \mathsf{q}_0, \mathsf{F})$ ein kürzestes Wort $w_{\{q,p\}}$ mit $\delta(q, w_{\{q,p\}}) \in \mathsf{F} \Leftrightarrow \delta(p, w_{\{q,p\}}) \notin \mathsf{F}$ berechnet.

(b) Determinisieren und dann minimieren Sie die folgenden NFAs. Verwenden Sie hierfür den erweiterten Minimierungsalgorithmus aus (a). Bestimmen Sie hierbei auch die Abhängigkeiten $D[\{p, q\}]$ zwischen den Paaren der Zustände gemäß dem Verfahren der Vorlesung.

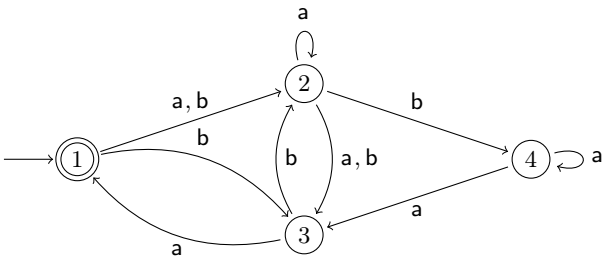
(i) NFA N_1:



(ii) NFA N_2:



(iii) NFA N_3:



Hinweise:

- Konstruieren Sie bei der Determinisierung nur die vom Startzustand erreichbaren Zustände.
- Wir verwenden das Alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}\}$.

Erweiterte Lösungsskizze

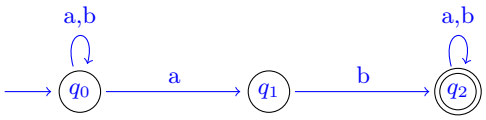
(a) Erweitere den Minimierungsalgorithmus aus der Vorlesung, sodass er für ein Paar von nicht äquivalenten Zuständen $p, q \in Q$ ein kürzestes Wort w berechnet, welches die beiden Zustände unterscheidet, d.h.:

$\delta(p, w) \in F$ und $\delta(q, w) \notin F$ **oder**
$\delta(p, w) \notin F$ und $\delta(q, w) \in F$

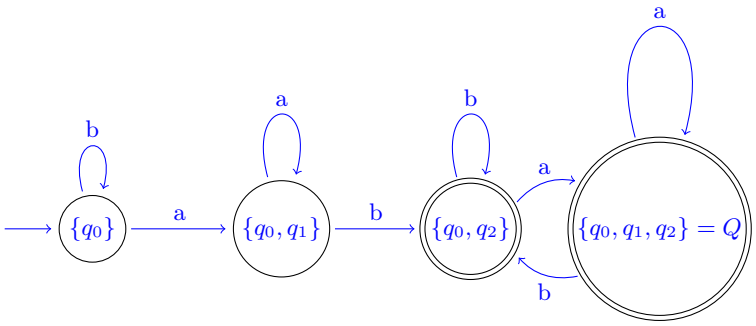
Idee: Trage beim Ausfüllen der Tabelle inäquivalenter Zustände statt eines Kreuzes an der jeweiligen Stelle ein unterscheidendes Wort w ein.
Falls einer der beiden Zustände ein Endzustand ist und der andere nicht, so ist das kürzeste, die beiden unterscheidende Wort $w = \varepsilon$.
Sonst können zwei Zustände p und q nur dann unterschieden werden, wenn man von ihnen aus mit einem Zeichen $a \in \Sigma$ in ein bereits durch ein Wort v unterscheidbares Zustandspaar p' und q' gelangt.
Dann ist aber $w = av$ ein Wort, welches p und q unterscheidet.

(b) Determinisiere den jeweils gegebenen NFA und minimiere ihn dann mit dem erweiterten Minimierungsverfahren aus Teilaufgabe (a).

▶ I NFA N_1:



Determinisieren mittels Potenzmengenkonstruktion liefert D_1:



Ausfüllen der Minimierungstabelle liefert folgende Ergebnisse:

	$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$	Q
$\{q_0\}$	=	-	-	-
$\{q_0, q_1\}$		=	-	-
$\{q_0, q_2\}$	ε	ε	=	-
Q	ε	ε		=

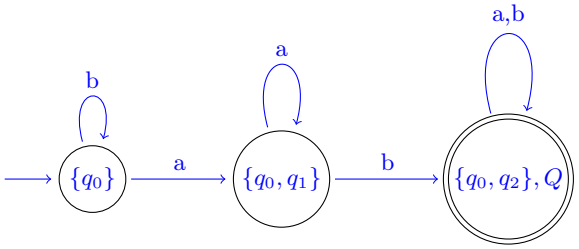
Beim ersten Durchlauf der Tabelle sind Paare von End- und Nicht-End-Zuständen zu markieren. Diese unterscheiden sich stets durch das Wort ε.

	$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$	Q
$\{q_0\}$	=	-	-	-
$\{q_0, q_1\}$	b	=	-	-
$\{q_0, q_2\}$	ε	ε	=	-
Q	ε	ε		=

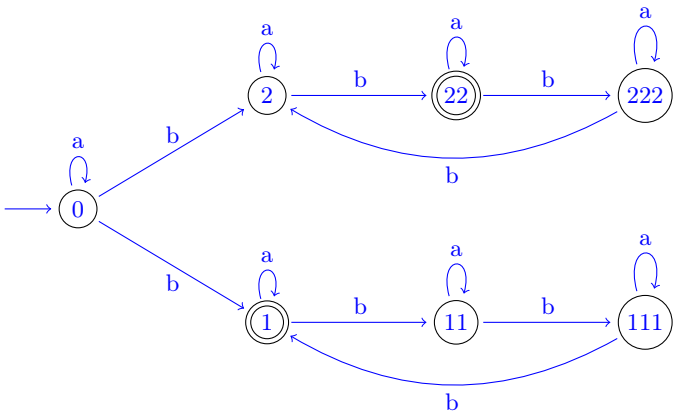
Beim zweiten Durchlauf können q_0,q_2 und q_0,q_1,q_2 nicht unterschieden werden. q_0 und q_0,q_1 führen allerdings mit dem Zeichen b in das Zustandspaar q_0 und q_0,q_2. Diese beiden können bereits durch ε unterschieden werden, d.h. q_0 und q_0,q_1 können durch $b\varepsilon{=}b$ unterschieden werden.

Beim nächsten Durchlaufen der Tabelle können keine weiteren Felder ausgefüllt werden, d.h. alle noch nicht als unterschiedlich markierten Zustandspaare können als äquivalent gekennzeichnet werden. Damit können die beiden Zustände $\{q_0, q_2\}$ und $\{q_0, q_1, q_2\}$ zu einem Zustand zusammengefasst werden.

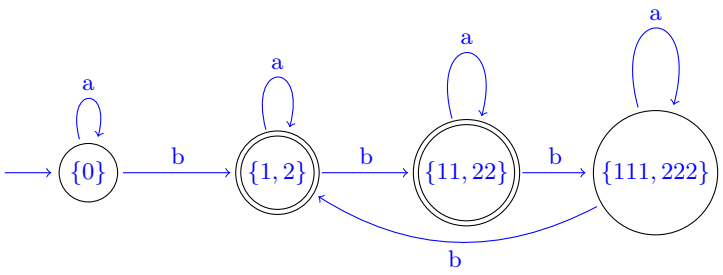
Das ergibt dann den Minimierten DFA D_1^m:



▶ II NFA N_2:



Determinisieren mittels Potenzmengenkonstruktion liefert D_2:



Ausfüllen der Minimierungstabelle liefert folgende Ergebnisse:

	$\{0\}$	$\{1,2\}$	$\{11,22\}$	$\{111,222\}$
$\{0\}$	=	-	-	-
$\{1,2\}$	ε	=	-	-
$\{11,22\}$	ε		=	-
$\{111,222\}$		ε	ε	=

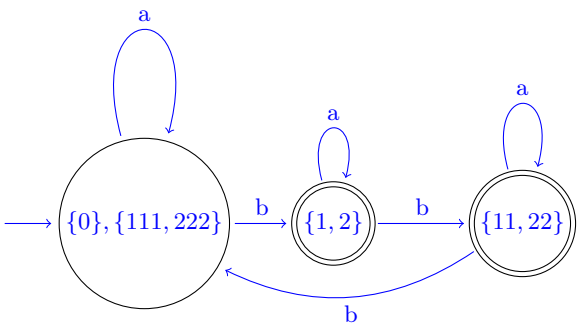
Beim ersten Durchlauf der Tabelle sind Paare von End- und Nicht-End-Zuständen zu markieren. Diese unterscheiden sich stets durch das Wort ε.

	$\{0\}$	$\{1,2\}$	$\{11,22\}$	$\{111,222\}$
$\{0\}$	=	-	-	-
$\{1,2\}$	ε	=	-	-
$\{11,22\}$	ε	b	=	-
$\{111,222\}$		ε	ε	=

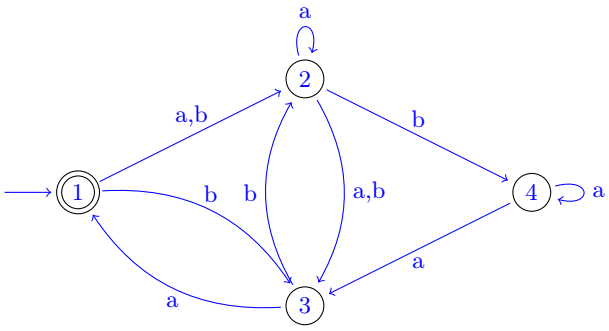
Beim zweiten Durchlauf können 0 und 111,222 nicht unterschieden werden. 1,2 und 11,22 führen allerdings mit dem Zeichen b in das Zustandspaar 11,22 und 111,222. Diese beiden können bereits durch ε unterschieden werden, d.h. 1,2 und 11,22 können durch $b\varepsilon{=}b$ unterschieden werden.

Beim nächsten Durchlaufen der Tabelle können keine weiteren Felder ausgefüllt werden, d.h. alle noch nicht als unterschiedlich markierten Zustandspaare können als äquivalent gekennzeichnet werden. Damit können die beiden Zustände $\{0\}$ und $\{111, 222\}$ zu einem Zustand zusammengefasst werden.

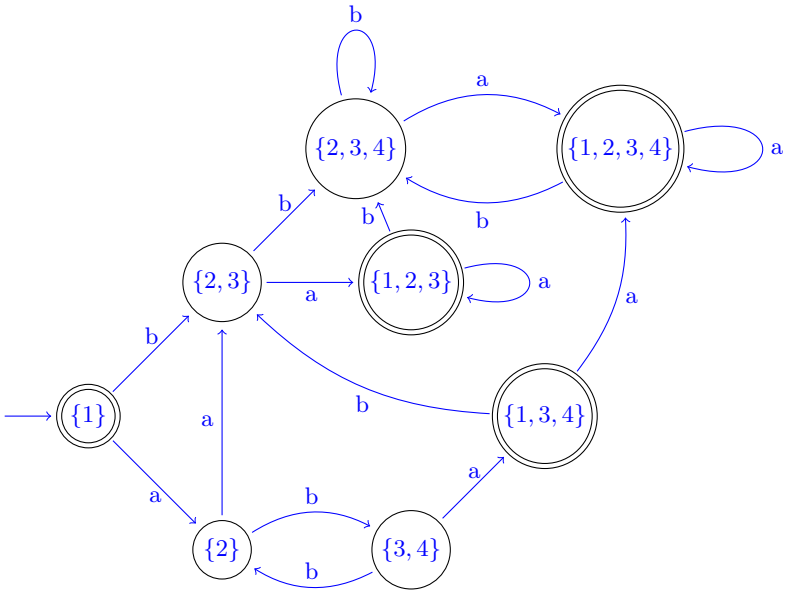
Das ergibt dann den Minimierten DFA D_2^m:



▶ III NFA N_3:



Determinisieren mittels Potenzmengenkonstruktion liefert D_3:



Ausfüllen der Minimierungstabelle liefert folgende Ergebnisse:

	$\{1\}$	$\{2,3\}$	$\{2\}$	$\{2,3,4\}$	$\{1,2,3\}$	$\{3,4\}$	$\{1,2,3,4\}$	$\{1,3,4\}$
$\{1\}$	$=$	-	-	-	-	-	-	-
$\{2,3\}$	ε	$=$	-	-	-	-	-	-
$\{2\}$	ε		$=$	-	-	-	-	-
$\{2,3,4\}$	ε			$=$	-	-	-	-
$\{1,2,3\}$		ε	ε	ε	$=$	-	-	-
$\{3,4\}$	ε				ε	$=$	-	-
$\{1,2,3,4\}$		ε	ε	ε		ε	$=$	-
$\{1,3,4\}$		ε	ε	ε		ε		$=$

Beim ersten Durchlaufen der Tabelle werden wie gehabt Paare von einem End- und einem Nicht-

End-Zustand als inäquivalent markiert.
Das jeweilige unterscheidende Wort ist ε.

Beim zweiten Durchlaufen der Tabelle erhält man

	$\{1\}$	$\{2,3\}$	$\{2\}$	$\{2,3,4\}$	$\{1,2,3\}$	$\{3,4\}$	$\{1,2,3,4\}$	$\{1,3,4\}$
$\{1\}$	=	-	-	-	-	-	-	-
$\{2,3\}$	ε	=	-	-	-	-	-	-
$\{2\}$	ε	a	=	-	-	-	-	-
$\{2,3,4\}$	ε		a	=	-	-	-	-
$\{1,2,3\}$	a	ε	ε	ε	=	-	-	-
$\{3,4\}$	ε		a		ε	=	-	-
$\{1,2,3,4\}$	a	ε	ε	ε		ε	=	-
$\{1,3,4\}$	a	ε	ε	ε		ε		=

- $\{2\}$ und $\{2,3\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2,3\}$, $\{1,2,3\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{2\}$ und $\{2,3\}$ durch $a\varepsilon = a$ unterscheidbar.
- $\{2\}$ und $\{2,3,4\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2,3\}$, $\{1,2,3,4\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{2\}$ und $\{2,3,4\}$ durch $a\varepsilon = a$ unterscheidbar.
- $\{1\}$ und $\{1,2,3\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2\}$, $\{1,2,3\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{1\}$ und $\{1,2,3\}$ durch $a\varepsilon = a$ unterscheidbar.
- $\{2\}$ und $\{3,4\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2,3\}$, $\{1,3,4\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{2\}$ und $\{3,4\}$ durch $a\varepsilon = a$ unterscheidbar.
- $\{1\}$ und $\{1,2,3,4\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2\}$, $\{1,2,3,4\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{1\}$ und $\{1,2,3,4\}$ durch $a\varepsilon = a$ unterscheidbar.
- $\{1\}$ und $\{1,3,4\}$ sind unterscheidbar, da man von dort aus mit einem a in das Zustandspaar $\{2\}$, $\{1,2,3,4\}$ kommt.
  Dieses ist bereits durch ε unterscheidbar. Daher sind $\{1\}$ und $\{1,3,4\}$ durch $a\varepsilon = a$ unterscheidbar.

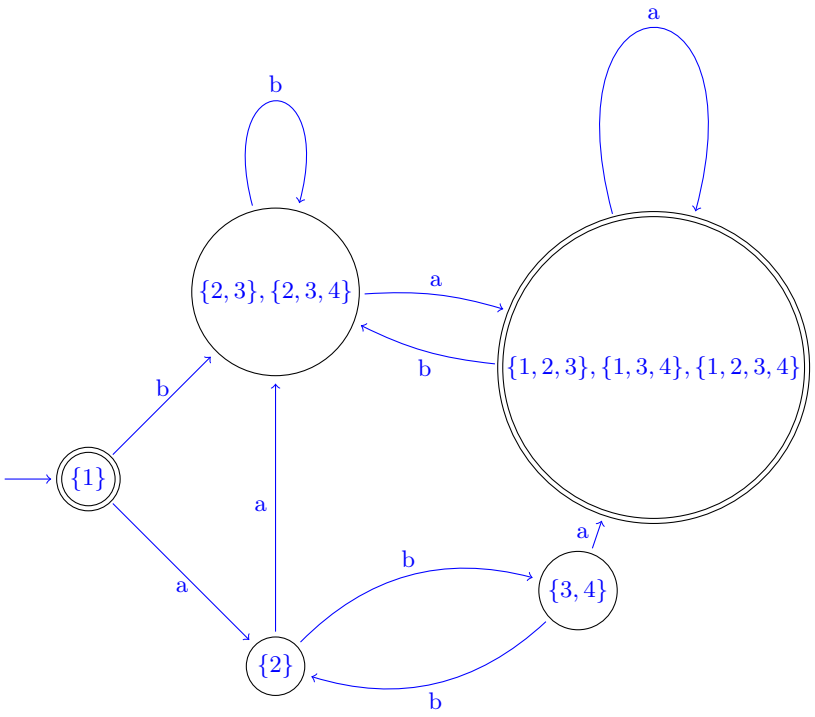
Der dritte Durchlauf der Tabelle ergibt dann

	$\{1\}$	$\{2,3\}$	$\{2\}$	$\{2,3,4\}$	$\{1,2,3\}$	$\{3,4\}$	$\{1,2,3,4\}$	$\{1,3,4\}$
$\{1\}$	=	-	-	-	-	-	-	-
$\{2,3\}$	ε	=	-	-	-	-	-	-
$\{2\}$	ε	a	=	-	-	-	-	-
$\{2,3,4\}$	ε		a	=	-	-	-	-
$\{1,2,3\}$	a	ε	ε	ε	=	-	-	-
$\{3,4\}$	ε	ba	a	ba	ε	=	-	-
$\{1,2,3,4\}$	a	ε	ε	ε		ε	=	-
$\{1,3,4\}$	a	ε	ε	ε		ε		=

- $\{2,3\}$ und $\{3,4\}$ sind unterscheidbar, da man von dort aus mit einem b in das Zustandspaar $\{2\}$, $\{2,3,4\}$ kommt.
  Dieses ist bereits durch a unterscheidbar. Daher sind $\{2,3\}$ und $\{3,4\}$ durch ba unterscheidbar.
- $\{3,4\}$ und $\{2,3,4\}$ sind unterscheidbar, da man von dort aus mit einem b in das Zustandspaar $\{2\}$, $\{2,3,4\}$ kommt.
  Dieses ist bereits durch a unterscheidbar. Daher sind $\{3,4\}$ und $\{2,3,4\}$ durch ba unterscheidbar.

Beim nächsten Durchlaufen der Tabelle können keine weiteren Felder ausgefüllt werden, d.h. alle noch nicht als unterschiedlich markierten Zustandspaare können als äquivalent gekennzeichnet werden.
Damit können die beiden Zustände $\{2,3\}$ und $\{2,3,4\}$ sowie die drei Zustände $\{1,2,3\}$, $\{1,3,4\}$ und $\{1,2,3,4\}$ jeweils zu einem Zustand zusammengefasst werden.

Das ergibt dann den Minimierten DFA D_3^m:



AUFGABE 5.3. Stufe C

Sei $\mathsf{L} = \mathsf{L}(\mathsf{a}^*\mathsf{b}^*\mathsf{c}^*)$ über dem Alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$.

(a) Entscheiden Sie, welche der folgenden Äquivalenzen wahr sind und begründen Sie Ihre Antwort:

- $\varepsilon \overset{?}{\equiv}_{\mathsf{L}} \mathsf{a}$
- $\mathsf{b} \overset{?}{\equiv}_{\mathsf{L}} \mathsf{c}$
- $\mathsf{abc} \overset{?}{\equiv}_{\mathsf{L}} \mathsf{cba}$

(b) Sei $v = \mathsf{aababc}$. Geben Sie ein Wort $u \neq v$ an, so dass $u \equiv_{\mathsf{L}} v$.

(c) Geben Sie die Mengen $[\mathsf{ab}]_{\mathsf{L}}$, $[\mathsf{bc}]_{\mathsf{L}}$ und $[\mathsf{ca}]_{\mathsf{L}}$ an.

(d) Geben Sie nun L', so dass $\mathsf{c} \equiv_{\mathsf{L}'} \mathsf{ba}$, $\mathsf{c} \not\equiv_{\mathsf{L}'} \mathsf{ab}$ und $\mathsf{aba} \equiv_{\mathsf{L}'} \mathsf{bab}$. Weiterhin soll $\varepsilon, \mathsf{aba} \in \mathsf{L}'$ gelten.

Erweiterte Lösungsskizze

Seien $\Sigma = \{a, b, c\}$
und $L = L(a^*b^*c^*)$

Die Frage, ob $u \overset{?}{\equiv}_L v$ ist:
Gilt **für alle** Wörter $w \in \Sigma^*$: $uw \in L \iff vw \in L$?

Analog dazu ist Frage, ob $u \overset{?}{\not\equiv}_L v$:
Gibt es ein Wort $w \in \Sigma^*$, sodass zwar
$uw \in L$ aber $vw \notin L$ bzw. sodass
$uw \notin L$ aber $vw \in L$?

Die Äquivalenzklasse eines Wortes v ist dann:
$[v]_L = \{u \in \Sigma^* \mid u \equiv_L v\}$

$[v]_L$ ist die Menge aller Wörter, die äquivalent zu v sind:

Erste Frage:
Welche Wörter w dürfen an v drangehängt werden, sodass vw noch in der Sprache L liegt?

$[v]_L$ ist **nicht** die Menge dieser Wörter w, sondern die Menge der Wörter u an die noch **genau** die gleichen Wörter w angehängt werden dürfen, nicht mehr und nicht weniger.

(a) Sind die folgenden Äquivalenzen korrekt? Warum (nicht)?

▶ $\varepsilon \overset{?}{\equiv}_L a$

Wahr, da sowohl nach einem ε als auch nach einem a noch genau Wörter der Form $a^*b^*c^*$ folgen dürfen.
D.h. formal ausgedrückt:
$\forall w \in \Sigma^*.\ \varepsilon w \in L \iff aw \in L \quad \left(\iff w \in L(a^*b^*c^*) = L \right)$

▶ $b \overset{?}{\equiv}_L c$

Falsch!
Gegenbeispiel ist $w = b$.
Es gilt zwar $bw = bb \in L$, aber $cw = cb \notin L$.

▶ $abc \overset{?}{\equiv}_L cba$

Falsch!
Gegenbeispiel ist $w = \varepsilon$.
Es gilt zwar $abcw = abc \in L$, aber $cbaw = cba \notin L$.

(b) $v = aababc$
Gib ein Wort u an, sodass $u \neq v$,
aber $u \equiv_L v$.

Welche Worte w darf man noch an v anhängen,
sodass $uw \in L$?

$\Longrightarrow$ Keine!
In v steht bereits ein b vor einem a,
was für kein Wort in L erlaubt ist:
$aababc$

Für welche Worte u gilt das noch,
dass man nicht mehr in der Sprache liegen kann,
egal welches Suffix w man noch an u anhängt?

$\Longrightarrow$ Alle, die ein ba, ca oder cb enthalten.
Diese Zeichenkombinationen dürfen in keinem Wort in L auftauchen,
d.h. wenn ein Wort u bereits mindestens eine der drei enthält,
kann man auch durch anhängen eines beliebigen Suffixes
nicht mehr in der Sprache landen.

Beispiel für ein solches Wort wäre

$u = cba$.

(c) Gib alle Elemente der gegebenen Äquivalenzklassen an (z.B. als regulären Ausdruck).

▶ $[ab]_L$

Welche Wörter $w \in \Sigma^*$ dürfen an ab drangehängt werden,
sodass $ab\,w$ noch in L liegt?

$\Longrightarrow$ a's dürfen keine mehr angehängt werden, da ja bereits ein b vorkam,
d.h. nur noch Wörter der Form b^*c^* dürfen angehängt werden.

$L(b^*c^*)$ ist aber **nicht** die gesuchte Menge.
Die gesuchte Menge ist die Menge aller Wörter, an die auch noch **genau**
Wörter aus $L(b^*c^*)$ angehängt werden dürfen:

$\Longrightarrow$ Wenn bisher in einem Wort nur a's vorkamen, so darf man zwar
Wörter der Form b^*c^* anhängen, aber auch beispielsweise
das Wort abc. D.h. Wörter der Form a^* sind nicht in der
gesuchten Menge.
Kam in einem Wort hingegen schon ein c vor, so darf beispielsweise
bc nicht mehr angehängt werden.

Wir suchen also gerade nach Wörtern, in denen beliebig viele a's
und dann **mindestens ein** b vorkommt, aber kein c:

$$[ab]_L = L(a^*bb^*)$$

▶ $[bc]_L$

Welche Wörter $w \in \Sigma^*$ dürfen an bc drangehängt werden, sodass $bc\,w$ noch in L liegt?

$\Longrightarrow$ a's und b's dürfen keine mehr angehängt werden, da ja bereits ein c vorkam, d.h. nur noch Wörter der Form c^* dürfen angehängt werden.

$L(c^*)$ ist aber **nicht** die gesuchte Menge.
Die gesuchte Menge ist die Menge aller Wörter, an die auch noch **genau** Wörter aus $L(c^*)$ angehängt werden dürfen:

$\Longrightarrow$ Wenn bisher in einem Wort nur a's und dann b's vorkamen, dann dürfen zwar auch noch beliebig viele c's angehängt werden, aber eben noch mehr; z.B. nämlich auch bbc.
D.h. Wörter der Form a^*b^* sind nicht äquivalent zu bc.

Wir suchen also gerade nach den Wörtern, die (nach beliebig vielen a's und dann b's) mindestens ein c enthalten:

$$[bc]_L = L(a^*b^*cc^*)$$

▶ $[ca]_L$

Welche Wörter $w \in \Sigma^*$ dürfen an ca drangehängt werden, sodass $ca\,w$ noch in L liegt?

$\Longrightarrow$ Keine!
Egal, was man an ca noch anhängt, das resultierende Wort wird immer die Zeichenkette ca enthalten.
Das ist für Wörter in L nicht erlaubt.

$\emptyset$ ist aber **nicht** die gesuchte Menge.
Die gesuchte Menge ist die Menge aller Wörter, an die auch nichts mehr angehängt werden kann, um zu erreichen, dass das resultierende Wort noch in der Sprache L liegt.

$\Longrightarrow$ Ein Wort liegt genau dann nicht in L, wenn es mindestens eine der drei Zeichenketten ba, ca oder cb enthält.
Gdw. ein Wort u eine dieser drei bereits enthält, kann auch ein beliebiger angehängter Suffix w nicht mehr erreichen, dass das resultierende Wort uw noch in L liegt. Schließlich enthält logischerweise uw die entsprechende Zeichenkette auch.

Wir suchen also gerade nach Wörtern, die eine der drei Zeichenketten ba, ca oder cb enthalten:

$$[ca]_L = L\Big(\Sigma^*\,(ba \mid ca \mid cb)\,\Sigma^*\Big) = \overline{L}$$

$\rightsquigarrow$ In diesem Fall! Dass $[w]_L = \overline{L}$ für $w \notin \Sigma^*$ gilt bei Weitem nicht immer!

(d) Gib eine Sprache L' an, für die folgende Bedigungen gilt:

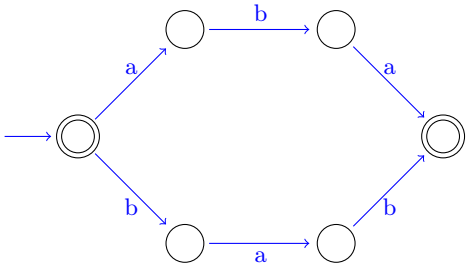
$c \equiv_{L'} ba$
$c \not\equiv_{L'} ab$
$aba \equiv_{L'} bab$

$\varepsilon \in L'$
$aba \in L'$

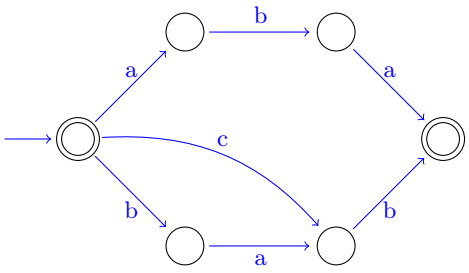
Idee: Wie in der VL gesehen, korrespondieren die Äquivalenzklassen einer Sprache mit den Zuständen des minimalen DFAs für diese Sprache.
Also könnte man versuchen, einen DFA für L' zu konstruieren und dabei äquivalente Wörter in denselben Zustand zu führen. Danach kann man die Sprache direkt vom DFA ablesen.

Erst einmal sollen ε und aba in L' liegen.
Man braucht also Läufe für diese Wörter:

Dann soll noch $aba \equiv_{L'} bab$ sein, d.h. die beiden Wörter aba und bab können in denselben Zustand führen:



Außerdem muss noch $c \equiv_{L'} ba$ sein, d.h. das Wort c sollte in den gleichen Zustand führen, wie das Wort ba:



Als letztes muss noch $c \not\equiv_{L'} ab$ sein.
Das ist aber in der Sprache des bisherigen DFA bereits der Fall, da das Wort cb akzeptiert wird, während das Wort abb nicht akzeptiert wird.
Also $cb \in L'$ und $abb \notin L'$.

Folglich kann L' einfach vom Automaten abgelesen werden.
Da dieser nicht viele Wörter akzeptiert, können sie auch einfach aufgelistet werden:

$$L' = \{\varepsilon, cb, bab, aba\}$$

AUFGABE 5.4. Stufe D

Entscheiden Sie, ob folgende Sprachen über dem Alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ regulär sind. Bestimmen Sie hierzu die Äquivalenzklassen der dazugehörigen Myhill-Nerode-Relation. Falls die Sprache regulär ist, zählen Sie alle Äquivalenzklassen auf und zeichnen Sie den kanonischen Minimalautomat $M_{\mathsf{L}} = (\Sigma^* / \equiv_{\mathsf{L}}, \Sigma, \delta_{\mathsf{L}}, [\varepsilon]_{\equiv_{\mathsf{L}}}, F_{\mathsf{L}})$. Falls

die Sprache nicht regulär ist, reicht es eine unendliche Menge von Äquivalenzklassen zu bestimmen.

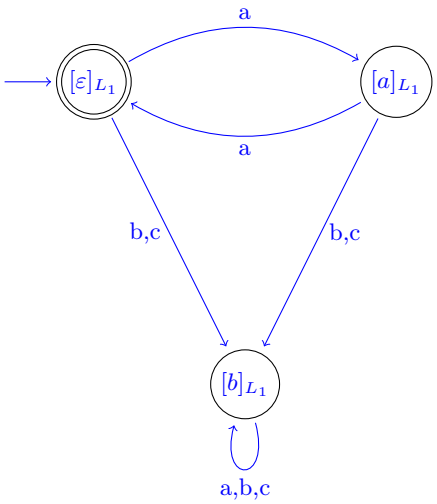
(a) $\mathsf{L}_1 = \{\mathsf{a}^{2i} \mid i \in \mathbb{N}_0\}$
(b) $\mathsf{L}_2 = \{\mathsf{a}^i\mathsf{b}^i\mathsf{c}^i \mid i \in \mathbb{N}_0\}$
(c) $\mathsf{L}_3 = \{w \in \Sigma^* \mid |w|_\mathsf{a} = 2 \cdot |w|_\mathsf{b}\}$
(d) $\mathsf{L}_4 = \mathsf{L}((\mathsf{a}^*(\mathsf{b} \mid \mathsf{c}))^*)$
(e) $\mathsf{L}_5 = \{w\mathsf{c}w \mid w \in \{\mathsf{a}, \mathsf{b}\}^*\}$

Erweiterte Lösungsskizze
Sei $\Sigma = \{a, b, c\}$.
Prüfe mithilfe der Äquivalenzklassen der gegebenen Sprachen, ob diese regulär sind.

Wenn es nur endlich viele Äquivalenzklassen gibt, so ist die Sprache regulär.
Wenn man allerdings unendlich viele inäquivalente Wörter finden kann, so ist die Sprache nicht regulär (, da dann der minimale Automat unendlich viele Zustände haben müsste).

(a) $L_1 = \{a^{2i} \mid i \in \mathbb{N}_0\}$ ⇝ Sprache aller Wörter gerader Länge, die nur aus a's bestehen.

Die Äquivalenzklassen zu bestimmen geht hier wohl am einfachsten direkt über den minimalen DFA, der die Sprache L_1 akzeptiert.
Dieser ist so zu konstruieren:



Jeder Zustand steht für eine Äquivalenzklasse.
Eine Äquivalenzklasse wird mit einem Wort bezeichnet, dass in der Klasse liegt (egal welches), d.h. mit einem Wort, das in den entsprechenden Zustand führt.
Die Menge aller Wörter, die in der Äquivalenzklasse eines Zustandes liegen, sind dann alle Wörter, die in den jeweiligen Zustand führen:

$$[\varepsilon]_{L_1} = L\Big((aa)^*\Big)$$ ⇝ Alle Wörter, die in den linken oberen Zustand führen, sind Wörter gerader Länge, die nur aus a's bestehen.

$$[a]_{L_1} = L\Big(a(aa)^*\Big)$$ ⇝ Alle Wörter, die in den rechten oberen Zustand führen, sind Wörter ungerader Länge, die nur aus a's bestehen.

$$[b]_{L_1} = L\Big(\Sigma^* \ (b|c) \ \Sigma^*\Big)$$ ⇝ Alle Wörter, die in den unteren Zustand führen, sind Wörter, die mindestens ein b oder c enthalten.

(b) $L_2 = \{a^i b^i c^i \mid i \in \mathbb{N}_0\}$

Diese Sprache ist bekanntlich nicht regulär.
Um das zu zeigen, reicht es, eine unendliche Familie von Wörtern $u^{(0)}, u^{(1)}, u^{(2)}...$ anzugeben, die paarweise nicht äquivalent sind, d.h. für alle i, j mit $i \neq j$ gilt:

$$u^{(i)} \not\equiv_{L_2} u^{(j)}$$

Beispiel für eine derartige Familie von Wörtern ist $\varepsilon, a, aa, aaa, ...$,
also $u^{(i)} = a^i$.

Beweis: Seien $i, j \in \mathbb{N}_0$ und $i \neq j$:

$$\Longrightarrow \quad a^i b^i c^i \in L_2 \text{ aber } a^j b^i c^i \notin L_2$$

$$\Longrightarrow \quad a^i \not\equiv_{L_2} a^j, \text{ unabhängig davon, was } i \text{ und } j \text{ sind.}$$

Das heißt aber wiederum, dass $[a^i]_{L_2} \neq [a^j]_{L_2}$.
Also hat die Sprache L_2 unendlich viele Äquivalenzklassen und ist damit nicht regulär.

(c) $L_3 = \{w \in \Sigma^* \mid |w|_a = 2 * |w|_b\}$ ⇝ Sprache aller Wörter, die doppelt so viele a's wie b's enthalten.

Diese Sprache ist ebenfalls nicht regulär.
Um das zu zeigen, reicht es, eine unendliche Familie von Wörtern $u^{(0)}, u^{(1)}, u^{(2)} \ldots$ anzugeben, die paarweise nicht äquivalenz sind, d.h. für alle i, j mit $i \neq j$ gilt:

$$u^{(i)} \not\equiv_{L_2} u^{(j)}$$

Beispiel für eine derartige Familie von Wörtern ist $\varepsilon, b, bb, bbb, \ldots$,
also $u^{(i)} = b^i$.

Beweis: Seien $i, j \in \mathbb{N}_0$ und $i \neq j$:

$$\Longrightarrow \quad b^i a^{2i} \in L_3 \text{ aber } b^j a^{2i} \notin L_3$$

$$\Longrightarrow \quad b^i \not\equiv_{L_3} b^j, \text{ unabhängig davon, was } i \text{ und } j \text{ sind.}$$

Das heißt aber wiederum, dass $[b^i]_{L_3} \neq [b^j]_{L_3}$.
Also hat die Sprache L_3 unendlich viele Äquivalenzklassen und ist damit nicht regulär.

(d) $L_4 = L\Big((a^* \, (b|c))^*\Big)$ ⇝ Sprache aller Wörter, die auf b oder c enden.

Betrachtet man diese Sprache, so gibt es eigentlich nur zwei Fälle, die zu unterscheiden sind:

- Das bisherige Wort endet auf a bzw.
- das bisherige Wort endet auf b oder c (oder ist das leere Wort).

Wenn ein Wort v auf b oder auf c endet, dann liegt es in der Sprache. D.h. man kann ε als Suffix anhängen und das entstandene Wort $v\varepsilon$ liegt immer noch in der Sprache. Außerdem kann man, da es ja in L_4 nur um das letzte Zeichen geht, jedes beliebige Wort $w \in L_4$ an v dranhängen und vw wird in L_4 liegen.

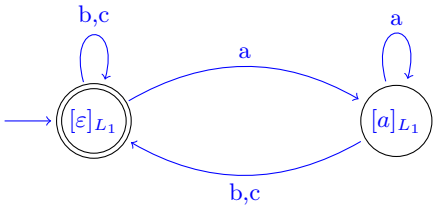
Wenn ein Wort v aber auf a endet, so kann man auch ein beliebiges Wort $w \in L_4$ an v dranhängen und vw wird in L_4 liegen. Nur mit $w = \varepsilon$ geht das jetzt nicht mehr.

Daraus folgt, dass es exakt zwei Äquivalenzklassen gibt:

$$[\varepsilon]_{L_4} = L_4$$ ⇝ Alle Wörter in L_4 sind in einer Äquivalenzklasse, da man an diese noch genau ε oder ein auf b oder c endendes Wort hängen darf.

$$[a]_{L_4} = L_4\{a\}^+ = \Sigma^* \setminus L_4$$ ⇝ Alle Wörter außerhalb L_4 sind in einer Äquivalenzklasse, da man an diese noch genau ein auf b oder c endendes Wort hängen darf.

Der kanonische Minimalautomat hat dann entsprechend einen Zustand für jede Äquivalenzklasse:



(e) $L_5 = \{wcw \mid w \in \{a,b\}^*\}$

Diese Sprache ist nicht regulär.
Um das zu zeigen, reicht es, eine unendliche Familie von Wörtern $u^{(0)}, u^{(1)}, u^{(2)}...$ anzugeben, die paarweise nicht äquivalenz sind, d.h. für alle i, j mit $i \neq j$ gilt:

$$u^{(i)} \not\equiv_{L_5} u^{(j)}$$

Beispiel für eine derartige Familie von Wörtern ist wieder $\varepsilon, a, aa, aaa, ...$,
also $u^{(i)} = a^i$.

Beweis: Seien $i, j \in \mathbb{N}_0$ und $i \neq j$:

$\Longrightarrow$ $a^i c a^i \in L_5$ aber
$a^j c a^i \notin L_5$

$\Longrightarrow$ $a^i \not\equiv_{L_5} a^j$, unabhängig davon, was i und j sind.

Das heißt aber wiederum, dass $[a^i]_{L_5} \neq [a^j]_{L_5}$.
Also hat die Sprache L_5 unendlich viele Äquivalenzklassen und ist damit nicht regulär.

Definition (Monoid)

Ein Monoid $\langle M, \circ, 1\rangle$ besteht aus einer (Träger-)Menge M, einer assoziativen Abbildung $\circ\colon M \times M \to M$ und einem bzgl. $\circ$ neutralen Element $1 \in M$ (d.h. $\forall m \in M.\ m \circ 1 = m = 1 \circ m$). Ist $\circ$ kommutativ, dann wird $\langle M, \circ, 1\rangle$ als kommutatives Monoid bezeichnet. Gilt $\forall m \in M.\ m \circ m = m$, so wird $\langle M, \circ, 1\rangle$ als **idempotentes** Monoid bezeichnet.

Definition (Kleene Algebra)

Eine *Kleene Algebra* $\langle K, +, \cdot, ^*, 0, 1\rangle$ besteht aus einer Trägermenge K, den binären Operationen $+\colon K \times K \to K$ (Addition), $\cdot\colon K \times K \to K$ (Multiplikation), der unären Operation $^*\colon K \to K$ (Stern) und zwei Konstanten $0, 1 \in K$. Mittels der Addition definiert man die binäre Relation $\sqsubseteq$ auf K durch

$$a \sqsubseteq b \overset{\text{Def}}{\longleftrightarrow} a + b = b$$

Wie üblich schreibt man kurz ab für $a \cdot b$. Um Klammern zu sparen, gilt "Stern vor Punkt vor Strich".
Eine Kleene Algebra erfüllt folgende Eigenschaften für alle $a, b, c, x \in K$:

Ax1: $\langle K, +, 0\rangle$ ist ein kommutatives und idempotentes Monoid.

Ax2: $\langle K, \cdot, 1\rangle$ ist ein Monoid.

Ax3: $a(b + c) = ab + ac$ und $(a + b)c = ac + bc$.

Ax4: $a0 = 0 = 0a$.

Ax5: $1 + aa^* \sqsubseteq a^*$ und $1 + a^*a \sqsubseteq a^*$.

Ax6: $b + ax \sqsubseteq x \to a^*b \sqsubseteq x$ und $b + xa \sqsubseteq x \to ba^* \sqsubseteq x$.

AUFGABE 5.5. Stufe E

In dieser Aufgabe abstrahieren wir von der konkreten Interpretation von regulären Ausdrücken als Konstruktionsbeschreibungen regulärer Sprachen. Ziel ist es den Zusammenhang mit Pfadproblemen im Bereich der Informatik und dem Lösen linearer Gleichungssysteme zu verdeutlichen.

(a) Sei Σ ein Alphabet. Beweisen Sie, dass $\langle 2^{\Sigma^*}, \cup, \circ, ^*, \emptyset, \{\varepsilon\}\rangle$ eine Kleene Algebra ist für

$$LL' := L \circ L' := \{ww' \mid w \in L, w' \in L'\} \qquad \text{und} \qquad L^* := \bigcup_{k \in \mathbb{N}_0} L^k$$

(b) Die Addition und das Minimum auf $\mathbb{R}$ seien auf $[-\infty, \infty] = \mathbb{R} \cup \{\pm\infty\}$ wie folgt erweitert:

$$\infty + a = \infty \quad -\infty + a = -\infty \quad -\infty + \infty = \infty$$

$$\min(a, \infty) = a \quad \min(a, -\infty) = -\infty \quad \min(-\infty, \infty) = -\infty$$

Weiterhin gelte:

$$a^* := \begin{cases} -\infty & \text{falls } a \in [-\infty, 0) \\ 0 & \text{falls } a \in [0, \infty] \end{cases}$$

Beweisen Sie, dass $\langle \mathbb{R} \cup \{\pm\infty\}, \min, +, ^*, \infty, 0\rangle$ eine Kleene Algebra ist.

(c) Zeigen Sie, dass in jeder Kleene Algebra $\langle K, +, \cdot, ^*, 0, 1\rangle$ für beliebige $a, b, c, d, e, f \in K$ gilt:

(i) $\sqsubseteq$ ist eine partielle Ordnung auf K, die monoton bzgl. Addition, Multiplikation und Stern ist, d.h.:

$$a \sqsubseteq b \to (ca \sqsubseteq cb \wedge ac \sqsubseteq bc \wedge a + c \sqsubseteq b + c \wedge a^* \sqsubseteq b^*)$$

(ii) a^*b ist die bzgl. $\sqsubseteq$ kleinste Lösung der linearen Ungleichung $b + aX \sqsubseteq X$ in K (X Variable), genauer:

$$b + a(a^*b) \sqsubseteq a^*b \qquad \wedge \qquad \forall x \in K\colon b + ax \sqsubseteq x \to a^*b \sqsubseteq x$$

Entsprechend ist ba^* die kleinste Lösung in K von $b + Xa \sqsubseteq X$.

Man kann zeigen, dass jedes lineare Ungleichungssystem

$$\begin{aligned} a_{1,1}X_1 + a_{1,2}X_2 + \ldots a_{1,n}X_n + b_1 &\quad\sqsubseteq\quad X_1 \\ &\vdots \\ a_{n,1}X_1 + a_{n,2}X_2 + \ldots a_{n,n}X_n + b_n &\quad\sqsubseteq\quad X_n \end{aligned}$$

mit $a_{i,j}, b_i \in K$ und $X_1, \ldots, X_n$ Variablen stets eine eindeutige $\sqsubseteq$-kleinste Lösung in K hat, d.h. dass es konkrete Elemente $x_1, \ldots, x_n \in K$ gibt, so dass für $X_1 = x_1, \ldots, X_n = x_n$ alle obigen Ungleichungen erfüllt sind, und für

jede weitere Lösung $X_1 = y_1, \ldots, X_n = y_n \in K$ stets $x_i \sqsubseteq y_i$ gilt. Insbesondere gilt

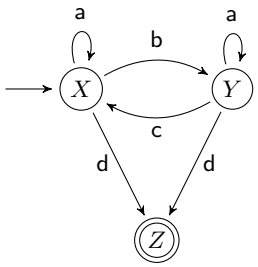
$$x_1 = a_{1,1}^*(a_{1,2}x_2 + \ldots + a_{1,n}x_n + b_1)$$

Somit kann das Gauß-Verfahren zur Bestimmung von $x_1, \ldots, x_n$ verwendet werden.

(d) Bestimmen Sie die kleinste Lösung x_1, x_2, x_3, x_4 für folgendes System:

$$\begin{array}{rcl} aX + bY + eZ & \sqsubseteq & X \\ cX + dY + fZ & \sqsubseteq & Y \\ 1 & \sqsubseteq & Z \end{array}$$

(e) Bestimmen Sie den regulären Ausdruck für den Zustand X. Durch welche Werte muss man a, b, c, d, e, f konkret ersetzen, damit sich die in (d) berechneten Terme in $\langle 2^{\Sigma^*}, \cup, \circ, ^*, \emptyset, \{\varepsilon\}\rangle$ zu dem gewünschten Ausdruck auswerten?



(f) Bestimmen Sie die Länge eines kürzesten Pfades von jedem Knoten zum Knoten Z in folgendem gewichteten Graphen. Durch welche Werte muss man a, b, c, d, e, f konkret ersetzen, damit sich die in (d) berechneten Terme in $\langle \mathbb{R} \cup \{\pm\infty\}, \min, +, \infty, 0\rangle$ zu den gesuchten Pfadlängen auswerten?