2. Klausur zur Vorlesung
Theoretische Grundlagen der Informatik
Wintersemester 2014/2015

Lösung!

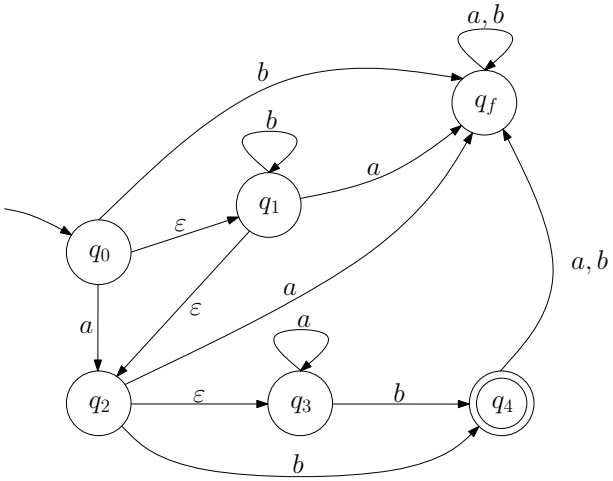
Beachten Sie:

- Bringen Sie den Aufkleber mit Ihrem Namen und Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes Aufgabenblatt mit Ihrem Namen und Matrikelnummer.
- Schreiben Sie die Lösungen auf die Aufgabenblätter und Rückseiten. Zusätzliches Papier erhalten Sie bei Bedarf von der Aufsicht.
- Zum Bestehen der Klausur sind **20** der möglichen **60** Punkte hinreichend.
- Es sind keine Hilfsmittel zugelassen.

Aufgabe	Mögliche Punkte					Erreichte Punkte				
	a	b	c	d	Σ	a	b	c	d	Σ
1	3	4	–	–	7			–	–	
2	2	1	2	2	7					
3	1	6	–	–	7			–	–	
4	1	3	1	–	5				–	
5	1	2	1	4	8					
6	1,5	2,5	3	1	8					
7	1	5	–	–	6			–	–	
8	3	2	–	–	5			–	–	
9	7×1				7					
Σ					60					

Problem 1: Endliche Automaten 3+4=7 Punkte

Gegeben sei der endliche Automat $\mathcal{A} = (Q = \{q_0, q_1, q_2, q_3, q_4, q_f\}, \Sigma = \{a, b\}, \delta, q_0, \{q_4\})$, wobei δ durch folgendes Zustandsdiagramm definiert ist.

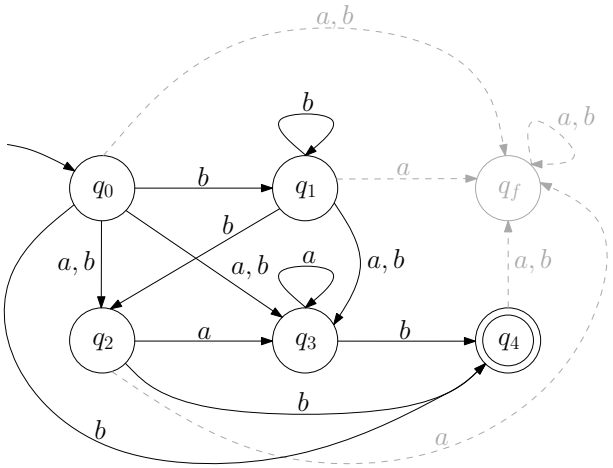


(a) Geben Sie für jeden Zustand dessen ε-Abschluss an und geben Sie für $\mathcal{A}$ einen äquivalenten endlichen nicht-deterministischen Automaten $\mathcal{A}'$ an, der keinen ε-Übergang enthält und vollständig ist. Der Automat $\mathcal{A}'$ soll die gleiche Anzahl an Zuständen wie $\mathcal{A}$ besitzen.

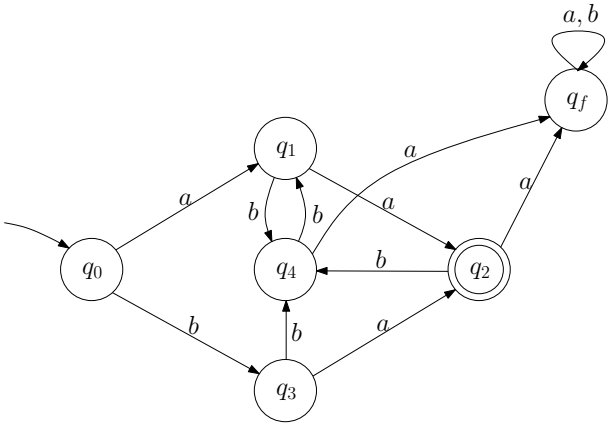
Lösung: Die ε-Abschlüsse für $\mathcal{A}$ lauten:

$$\begin{aligned} E(q_0) &= \{q_0, q_1, q_2, q_3\}, \\ E(q_1) &= \{q_1, q_2, q_3\}, \\ E(q_2) &= \{q_2, q_3\}, \\ E(q_3) &= \{q_3\}, \\ E(q_4) &= \{q_4\}, \\ E(q_f) &= \{q_f\}. \end{aligned}$$

Der zu $\mathcal{A}$ äquivalente NEA sieht wie folgt aus:



(b) Gegeben sei der endliche Automat $\mathcal{A}_d = (Q_d = \{q_0, q_1, q_2, q_3, q_4, q_f\}, \Sigma = \{a, b\}, \delta_d, q_0, \{q_2\})$, wobei δ_d durch folgendes Zustandsdiagramm definiert ist.

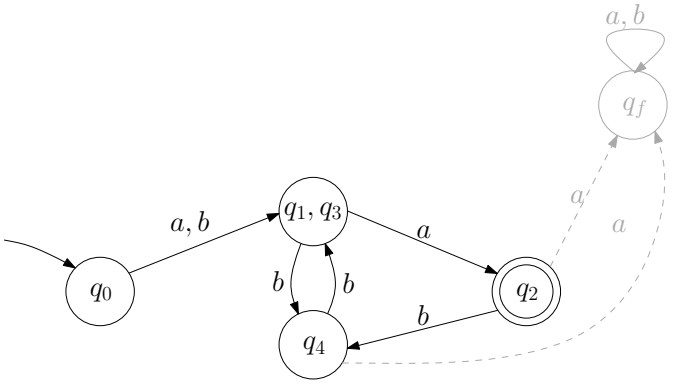


Beweisen Sie, dass $\mathcal{A}_d$ nicht minimal ist. Konstruieren Sie hierfür, mithilfe eines Verfahrens aus der Vorlesung, einen zu $\mathcal{A}_d$ äquivalenten minimalen Automaten. Geben Sie dabei alle Schritte an.

Lösung:

- $\{q_0, q_1, q_2, q_3, q_4, q_f\}$,
- $\varepsilon : \{q_2\}\ \{q_0, q_1, q_3, q_4, q_f\}$,
- $a : \{q_2\}\ \{q_1, q_3\}\ \{q_0, q_4, q_f\}$,
- b : trennt nichts,
- $aa : \{q_2\}\ \{q_1, q_3\}\ \{q_0\}\ \{q_4, q_f\}$,
- ab : trennt nichts,
- $ba : \{q_2\}\ \{q_1, q_3\}\ \{q_0\}\ \{q_4\}\ \{q_f\}$,
- bb : trennt nichts,
- Wörter der Länge 3 trennt nichts.

Der deterministische endliche Automat $\mathcal{A}_d$ ist nicht minimal. Der äquivalente minimale DEA mit 5 Zuständen ist in der folgenden Abbildung angegeben.

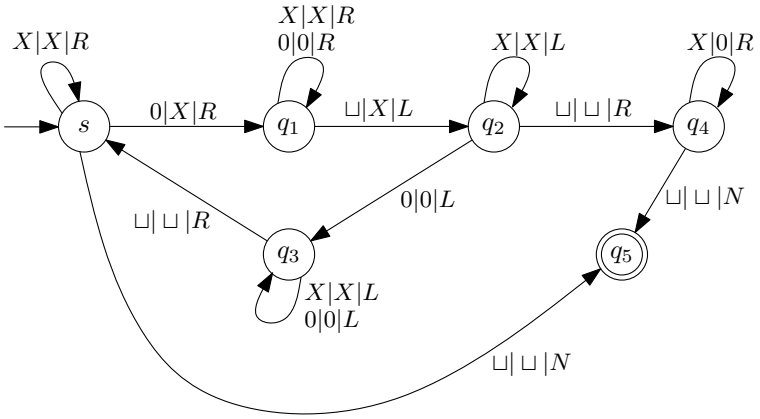


Problem 2: Turingmaschine 2+1+2+2=7 Punkte

Gegeben sei die folgende Turingmaschine

$$\mathcal{M} = (Q = \{s, q_1, q_2, q_3, q_4, q_5\}, \Sigma = \{0\}, \Gamma = \Sigma \cup \{\sqcup, X\}, \delta, s, F = \{q_5\}).$$

Die Übergangsfunktion δ von $\mathcal{M}$ ist durch untenstehendes Zustandsübergangsdiagramm dargestellt.



Für $\mathcal{M}$ gelten folgende Konventionen: $\mathcal{M}$ hält und akzeptiert sobald ein Endzustand erreicht wird. Alle Übergänge, die aus Endzuständen herausführen, werden demnach nicht verwendet. $\mathcal{M}$ hält und **akzeptiert nicht** für nicht dargestellte Zustandsübergänge.

(a) Dokumentieren Sie die Berechnung des Wortes 00 bis zu dem Punkt, an dem die Turingmaschine in den Zustand q_4 gelangt. Geben Sie dazu für jeden Schritt die aktuelle Konfiguration an.

Lösung: $\sqcup(s)00\sqcup$, $\sqcup X(q_1)0\sqcup$, $\sqcup X0(q_1)\sqcup$, $\sqcup X(q_2)0X\sqcup$, $\sqcup(q_3)X0X\sqcup$, $(q_3) \sqcup X0X\sqcup$, $\sqcup(s)X0X\sqcup$, $\sqcup X(s)0X\sqcup$, $\sqcup XX(q_1)X\sqcup$, $\sqcup XXX(q_1)\sqcup$, $\sqcup XX(q_2)XX\sqcup$, $\sqcup X(q_2)XXX\sqcup$, $\sqcup(q_2)XXXX\sqcup$, $(q_2) \sqcup XXXX\sqcup$, $\sqcup(q_4)XXXX\sqcup$

(b) Für welche Eingaben wird der eingezeichnete Zustandsübergang von s nach q_5 benötigt?

Lösung: Mit diesem Übergang wird sicher gestellt, dass auch das leere Wort in $L(\mathcal{M})$ enthalten ist.

(c) Geben Sie die Sprache $L(\mathcal{M}) \subseteq \Sigma^\star$ an, die $\mathcal{M}$ akzeptiert. Geben Sie außerdem die Funktion $f_{\mathcal{M}} \colon \Sigma^\star \to \Gamma^\star$ an, die $\mathcal{M}$ realisiert. Begründen Sie Ihre Antwort, indem Sie die Funktionsweise der Turingmaschine informell beschreiben.

Lösung: $\mathcal{M}$ akzeptiert jede Eingabe, d.h. $L(\mathcal{M}) = \Sigma^\star$. Zudem realisiert $\mathcal{M}$ folgende Funktion

$$f_{\mathcal{M}}(w) = ww$$

Anfangs enthält das Band nur die Eingabe, ein Wort bestehend aus 0en. Es wird die 0 gelesen, die am weitesten links steht und durch ein X ersetzt ($s \to q_1$). Dann geht der Lesekopf an den rechten Rand des aktuellen Wortes und schreibt ein X. Daraufhin läuft der Lesekopf wieder nach links. Stößt der Leskopf dabei auf eine 0, so geht $\mathcal{M}$ in den Zustand q_3 und lässt den Lesekopf bis zum linken Rand laufen. Das Verfahren beginnt nun von vorne. Anderenfalls, falls keine 0 mehr gelesen wird, so kommt $\mathcal{M}$ in den Zustand q_4. Der Lesekopf befindet sich nun ganz links und läuft wieder ganz nach rechts und ersetzt dabei jedes X durch eine 0. Damit befinden sich nun doppelt so viele 0en auf dem Band wie zuvor. Im Fall, dass die Eingabe das leere Wort ist, geht die Turingmaschine direkt in den Endzustand.

(d) Zeigen Sie, dass für zwei entscheidbare Sprachen L_1 und L_2 folgende Aussage gilt. Die Sprache $L_1 \setminus L_2$ ist entscheidbar.

Lösung: Da L_1 und L_2 entscheidbar sind, gibt es Turingmaschinen $\mathcal{M}_1$ und $\mathcal{M}_2$, die L_1 bzw. L_2 erkennen und auf jeder Eingabe halten.

Sei w die Eingabe, die überprüft werden soll. Simuliere mithilfe einer universellen Turingmaschine zuerst $\mathcal{M}_1$ auf w. Falls $w \notin L_1$, so gebe *Nein* aus. Ansonsten simuliere $\mathcal{M}_2$ auf w. Falls $w \in L_2$, dann gebe *Nein* aus und ansonsten *Ja*.

Problem 3: Reguläre und kontextfreie Sprachen 1+6=7 Punkte

(a) Gegeben ist die Sprache $L_1 = \{a^{h(x)} \in \{a\}^\star \mid x \in \mathbb{N} \setminus \{0\}\}$, wobei

$$h(x) = \begin{cases} 0, & \text{falls } x = 0, \\ 1, & \text{falls } x = 1, \\ h(x-2) + h(x-1), & \text{falls } x > 1. \end{cases}$$

Ist die Sprache L_1 regulär?

Lösung:

L_1 ist nicht regulär.

(b) Gegeben ist die Sprache $L_2 = \{a^n b^{m+n} c^n \in \{a,b,c\}^\star \mid m,n \in \mathbb{N}\}$.

(i) Geben Sie die formale Definition für das Pumping-Lemma für kontextfreie Sprachen an.

(ii) Beweisen oder widerlegen Sie, dass L_2 nicht kontextfrei ist.

Lösung:

(i) Definition Pumping-Lemma siehe Skript Seite 100–101.

(ii) Annahme: L_2 ist kontextfrei. Sei $p \in \mathbb{N}$ und $w = a^p b^{m+p} c^p$. Da $w \in L_2$ und $|w| = 3p + w \geq p$ nach Annahme gilt, kann nach dem Pumping Lemma w so zerlegt werden, dass $w = uvxyz$ und folgende Bedigungen gelten:

(1) $uv^i xy^i z \in L_2$ für alle $i \in \mathbb{N}$,

(2) $|vy| > 0$ und

(3) $|vxy| < p$.

Daraus ergeben sich vier Fälle mit der folgenden Fallunterscheidung:

Fall 1: v oder y enthalten mehrere verschiedene Symbole. Wenn v aus a-Zeichen und b-Zeichen besteht, dann ist $uv^2xy^2z \notin L_2$, da b-Zeichen vor a-Zeichen sein werden. Daraus folgt, dass v und y nie a-Zeichen, b-Zeichen oder c-Zeichen sein können (d.h. keine Mischungen).

Fall 2: Sei $v = a^j$, dann ist y entweder a^k oder b^l ($|vxy| \leq p$).
Wenn $y = a^k$, dann $uv^2xy^2z = a^{p+j+k}b^{p+m}c^p \notin L_2$, da $j + k > 0$ und damit die Anzahl der a-Zeichen echt größer der Anzahl der b-Zeichen subtrahiert mit m ($|a| > |b| - m$).
Wenn $y = b^l$, dann $uv^2xy^2z = a^{p+j}b^{p+l+m}c^p \notin L_2$, da entweder die Anzahl der a-Zeichen echt größer der Anzahl der c-Zeichen oder die Anzahl der b-Zeichen subtrahiert mit m echt größer der Anzahl der c-Zeichen ($|a| > |c|$ oder $|b| - m > |c|$).

Fall 3: Sei $v = b^j$, dann ist y entweder b^k oder c^l ($|vxy| \leq p$).
Wenn $y = b^k$, dann $uv^2xy^2z = a^p b^{j+k+p+m}c^p \notin L_2$, da $j + k > 0$ und damit die Anzahl der b-Zeichen subtrahiert mit m echt größer der Anzahl der a-Zeichen bzw. der c-Zeichen ($|b| - m > |a|$ oder $|c|$).
Wenn $y = c^l$, dann $uv^2xy^2z = a^p b^{p+j+m}c^{p+l} \notin L_2$, da $j + l > 0$ und damit entweder die Anzahl der b-Zeichen subtrahiert mit m echt größer der Anzahl der a-Zeichen oder die Anzahl der c-Zeichen echt größer der Anzahl der a-Zeichen ($|c| - m > |a|$ oder $|c| > |a|$).

Fall 4: Sei $v = c^j$, dann ist $y = c^k$. Damit ist $uv^2xy^2z = a^p b^p c^{p+j+k} \notin L_2$, da $j + k > 0$ ist die Anzahlen der c-Zeichen echt größer der Anzahl der a-Zeichen.

Aus dieser Fallunterscheidung folgt, dass es keine Zerlegung $uvxyz$ gibt, sodass $|vy| > 0$, $|vxy| < p$ für alle $i > 0$, sodass $uv^2xy^2z \in L_2$.

Damit haben wir einen Widerspruch zur Annahme und es folgt, dass $L_2 \notin$ Chomsky-Typ 2 (also nicht kontextfrei).

Problem 4: Berechenbarkeit 1+3+1=5 Punkte

(a) In der Vorlesung wurde die Situation, in der sich die Turingmaschine während der Abarbeitung eines Wortes befindet, als *Konfiguration* bezeichnet. Geben Sie die formale Definition einer *Konfiguration* wieder und erläutern Sie diese.

Lösung: Sei $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, , s, F)$ eine Turingmaschine über dem Alphabet Σ. Eine *Konfiguration* ist ein Tupel (w, q, a, v) mit $w, v \in \Gamma^\star$, $q \in Q$ und $a \in \Gamma$.

Bedeutung:

- $\mathcal{M}$ befindet sich gerade im Zustand q.
- Der Lesekopf steht auf dem Zeichen a.
- Links vom Lesekopf steht das Wort w auf dem Rechenband.
- Rechts vom Lesekopf steht das Wort v auf dem Rechenband.

(b) Eine *wiederholungssensitive* Turingmaschine $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, s, F)$, kurz WSTM, hält auf einer Eingabe $w \in \Sigma^\star$ genau dann *nicht*, wenn eine Konfiguration von $\mathcal{M}$ während der Abarbeitung von w mehr als einmal auftritt.

Gegeben sei eine WSTM $\mathcal{M}$. Zeigen Sie, dass die Sprache $L_{\mathcal{M}} = \{w \in \Sigma^\star \mid \mathcal{M} \text{ akzeptiert } w\}$ entscheidbar ist.

Lösung: Simuliere $\mathcal{M}$ auf w mithilfe einer universellen Turingmaschine $\mathcal{M}'$. Man kann wie folgt überprüfen, ob $\mathcal{M}$ auf der Eingabe w nicht hält. Bei jedem Schritt, der von $\mathcal{M}$ ausgeführt wird, speichere die aktuelle Konfiguration C von $\mathcal{M}$ auf einem zusätzlichen Band ohne die bereits gespeicherten Konfiguration zu überschreiben. Überprüfe zudem, ob C zuvor bereits vorhanden ist. Falls ja, so wird $\mathcal{M}$ nicht halten, da sie eine WSTM ist. In endlich vielen Berechnungsschritten kann also bestimmt werden, ob $\mathcal{M}$ nicht auf w hält.

$\mathcal{M}'$ verhält sich während der Abarbeitung von w wie folgt:

- Falls $\mathcal{M}$ das Wort w akzeptiert, so akzeptiert auch $\mathcal{M}'$ das Wort w.
- Falls $\mathcal{M}$ nicht das Wort w akzeptiert, so akzeptiert auch $\mathcal{M}'$ nicht das Wort w.
- Falls $\mathcal{M}$ auf w nicht hält, so akzeptiert $\mathcal{M}'$ das Wort w nicht.

(c) Sei L eine entscheidbare Sprache. Zeigen oder widerlegen Sie folgende Aussage: Jede Teilmenge von L ist ebenfalls entscheidbar.

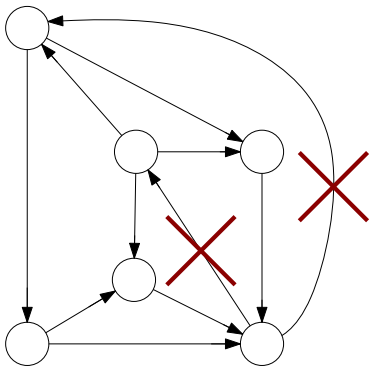
Lösung: Die Aussage gilt nicht. Sei $\Sigma = \{0, 1\}$. Die Sprache $\Sigma^\star$ ist offensichtlich entscheidbar, da sie alle Wörter über dem Alphabet Σ enthält. Allerdings ist das Halteproblem $\mathcal{H} \subseteq \Sigma^\star$ nicht entscheidbar.

Problem 5: NP-Vollständigkeit 1+2+1+4=8 Punkte

Das Entscheidungsproblem FEEDBACKARCSET ist wie folgt definiert:

Problem FEEDBACKARCSET
Gegeben: Gerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.
Frage: Existiert eine Kantenmenge $E' \subseteq E$ mit $|E'| \leq k$, sodass G ohne die Kanten aus E' azyklisch ist?
Hinweis: Ein Graph heißt *azyklisch*, wenn er keinen gerichteten Kreis enthält.

(a) Markieren Sie in dem unten dargestellten Graphen $G = (V, E)$ eine Kantenmenge E', die das Problem FEEDBACKARCSET für $k = 2$ auf G löst.



(b) Gegeben sei ein Problem Π. Erläutern Sie, wie man mithilfe eines NP-vollständigen Problems Π' zeigen kann, dass Π NP-schwer ist.

Lösung: Konstruiere für ein gegebenes NP-vollständiges Problem Π' eine polynomielle Reduktion $\Pi' \propto \Pi$, d.h. man löst Π' mithilfe von Π. Zeige insbesondere folgende Aussagen:

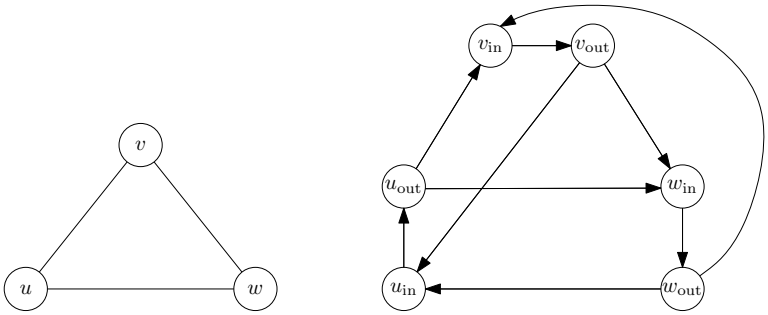
a) $\propto$ ist eine in polynomieller Zeit berechenbare Funktion.

b) Sei I' eine beliebige Instanz von Π' und sei $I \in \Pi$ die Instanz, die man erhält, wenn man I' mithilfe von $\propto$ auf Π' reduziert. Zeige: I' ist eine Ja-Instanz von Π' genau dann, wenn I eine Ja-Instanz von Π ist.

(c) Ein ungerichteter Graph $G = (V, E)$ kann wie folgt in einen gerichteten Graphen $G' = (V', E')$ transformiert werden:

> Für jeden Knoten $u \in V$ besitzt die Menge V' die Knoten u_{in} und u_{out}. Die Menge E' enthält für jeden Knoten $u \in V$ die Kante $(u_{\text{in}}, u_{\text{out}})$. Zudem enthält sie für jede Kante $\{u, v\} \in E$ die Kanten $(u_{\text{out}}, v_{\text{in}})$ und $(v_{\text{out}}, u_{\text{in}})$.

Konstruieren Sie gemäß der beschriebenen Transformation aus dem unten abgebildeten ungerichteten Graphen $G = (V, E)$ einen gerichteten Graphen $G' = (V', E')$.


v
u
w
v_{in}
v_{out}
u_{out}
w_{in}
u_{in}
w_{out}

(d) Gegeben sei das NP-vollständige Problem KNOTENÜBERDECKUNG wie folgt:

Problem KNOTENÜBERDECKUNG

Gegeben:	Ungerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.
Frage:	Gibt es eine Knotenüberdeckung $V' \subseteq V$ mit $\lvert V' \rvert \leq k$?
Hinweis:	$V' \subseteq V$ heißt *Knotenüberdeckung* von G, falls für jede Kante $\{u, v\} \in E$ gilt $u \in V'$ oder $v \in V'$.

Zeigen Sie, dass das Problem FEEDBACKARCSET NP-schwer ist.
Hinweis: Verwenden Sie die in Teilaufgabe (c) beschriebene Transformation.

Lösung:

$\propto$ Sei $(G = (V, E), m)$ eine Instanz von KNOTENÜBERDECKUNG. Wir konstruieren einen gerichteten Graphen $G' = (V', E')$ wie in Teilaufgabe (c) und setzen $k = m$.

Wir zeigen nun, dass gilt: (G, k) ist eine Ja-Instanz von KNOTENÜBERDECKUNG genau dann, wenn (G', k) eine Ja-Instanz von FEEDBACKARCSET ist.

$\Rightarrow$ Sei $S \subseteq V$ eine Knotenüberdeckung von G mit $\lvert S \rvert \leq k$. Entferne für jeden Knoten $v \in V'$ die Kante (v_{in}, v_{out}) aus G'. Der Graph G' wird dadurch azyklisch: Ein Kreis, der einen Knoten v_{in} betritt, kann v_{in} nur über die Kante (v_{in}, v_{out}) verlassen. Gleichermaßen gilt, dass ein Kreis einen Knoten v_{out} nur über die Kante (v_{in}, v_{out}) betreten kann. Ein Kreis, der also eine Kante (u_{out}, v_{in}) verwendet, muss auch die Kante (u_{in}, u_{out}) und die Kante (v_{in}, v_{out}) verwenden. Mindestens einer der beiden Kanten wurde entfernt, da der entsprechende Knoten v oder der entsprechende Knoten u in V' enthalten sein muss.

$\Leftarrow$ Sei $E' \subseteq V$ eine Kantenmenge mit $\lvert E' \rvert \leq k$, sodass G' ohne E' azyklisch ist. O.B.d.A. nehme an, dass jede Kante in E' die Form (u_{in}, u_{out}) hat: Wie oben argumentiert führt jeder Kreis über eine solche Kante. Jede dieser Kanten enspricht einem Knoten u in V, sodass E' eine Knotenmenge S der Größe $\leq k$ induziert. Wir zeigen nun, dass V' eine Knotenüberdeckung von G ist. Angenommen es gibt eine Kante $\{u, v\} \in E'$, sodass $u \notin S$ und $v \notin S$, dann ist der Kreis (u_{in}, u_{out}), (u_{out}, v_{in}), (v_{in}, v_{out}), (v_{out}, u_{in}) in G enthalten.

Problem 6: Approximation – Maximum Matching 1,5 + 2,5 + 3 + 1 = 8 Punkte

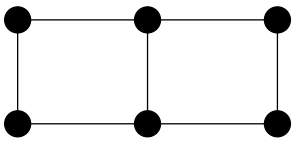
In einem einfachen, ungerichteten Graphen $G = (V, E)$ ist ein *Matching* eine Kantenmenge $M \subseteq E$, sodass keine zwei Kanten aus M zu einem gemeinsamen Knoten inzident sind. Ein Matching M heißt

- **inklusions***maximal*, wenn es keine echte Teilmenge eines anderen Matchings M' ist, d.h. es existiert kein Matching M', sodass $M \subsetneq M'$.
- **kardinalitäts***maximal*, wenn es kein Matching M' mit echt größerer Kantenanzahl gibt, d.h. es existiert kein Matching M', so dass $|M| < |M'|$.

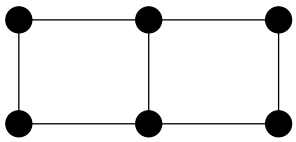
(a) Zeichnen Sie in die untenstehenden Graphen Folgendes ein:

- in (i) ein **kardinalitäts**maximales Matching.
- in (ii) ein **inklusions**maximales Matching, das nicht **kardinalitäts**maximal ist.
- in (iii) ein nicht-leeres, **nicht inklusions**maximales Matching.

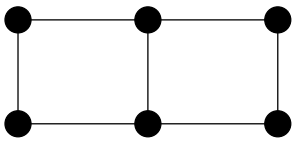
Zeichnen Sie hierzu die Kanten der Matchings fett ein.



(i) **kardinalitäts**maximal.

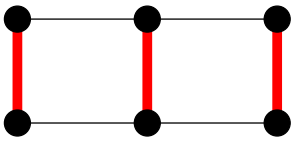


(ii) **inklusions**maximal, nicht kardinalitätsmaximal.

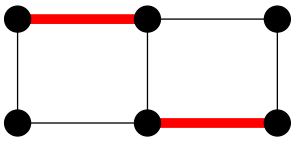


(iii) nicht leer, **nicht** inklusionsmaximal.

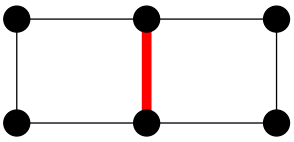
Lösung:



(i) **kardinalitäts**maximal.



(ii) **inklusions**maximal, nicht kardinalitätsmaximal.



(iii) nicht leer, **nicht** inklusionsmaximal.

Eine Knotenmenge $V' \subseteq V$ heißt *Überdeckung* der Kantenmenge E, wenn für alle $e \in E$ gilt:
$\exists u \in V'$ mit u inzident zu e.

(b) Sei M ein beliebiges Matching in $G = (V, E)$ und sei V' eine beliebige Überdeckung von E.

Zeigen Sie: $|M| \leq |V'|$.

Lösung: Jeder Knoten in V kann maximal eine Kante aus M überdecken. Mit $M \subseteq E$ braucht man also mindestens $|M|$ Knoten um M und damit ganz E zu überdecken.

(c) Sei $M_{\max}$ ein **inklusions**maximales Matching in $G = (V, E)$ und sei $V'_{\min}$ eine Überdeckung von E mit minimaler Kardinalität.

Zeigen Sie: $|V'_{\min}| \leq 2\,|M_{\max}|$.

Lösung: Jede Kante $e \notin M_{\max}$ hat mindestens einen Knoten mit einer Kante $e' \in M_{\max}$ gemeinsam, da sonst das Matching erweiterbar wäre. Damit kann jede Kante $e \notin M_{\max}$ von einem Knoten überdeckt werden, der inzident zu einer Kante $e' \in M_{\max}$ ist, was für die jeweilige Kante e' ohnehin gilt. Insgesamt gibt es $2|M_{\max}|$ Knoten, die zu einer Kante $e' \in M_{\max}$ inzident sind. Diese reichen aus um ganz E zu überdecken.

(d) Das Problem MAXIMUM-MATCHING ist wie folgt definiert:

Problem MAXIMUM MATCHING
Gegeben: Einfacher, ungerichteter Graph $G = (V, E)$.
Gesucht: **Kardinalitäts**maximales Matching M_M.

Sei I eine Instanz des Matchingproblems, sei M_M ein **kardinalitäts**maximales Matching bzgl. I und sei $M_{\max}$ ein **inklusions**maximales Matching bzgl. I.

Zeigen Sie: $\mathrm{OPT}(I) \leq 2 \cdot \mathcal{A}(I)$, mit $\mathrm{OPT}(I) := |M_M|$, $\mathcal{A}(I) := |M_{\max}|$.

Hinweis: Man kann die Teilaufgaben (b) und (c) nutzen.

Lösung: Zu zeigen: $|M_M| \leq 2\,|M_{\max}|$.
Sei also $\hat{V}$ eine Überdeckung von E mit minimaler Kardinalität.
Mit (b) folgt $|M_M| \leq |\hat{V}|$.
Mit (c) folgt $|\hat{V}| \leq 2\,|M_{\max}|$.
Insgesamt folgt $|M_M| \leq |\hat{V}| \leq 2\,|M_{\max}|$.

Problem 7: Grammatiken und Chomsky-Hierarchien 1+5=6 Punkte

Gegeben sei die Grammatik $G = \{\Sigma, V, S, R\}$ mit Terminalen $\Sigma = \{a, b, c\}$, Nichtterminalen $V = \{S, A, B, C, D\}$ und Produktionen

$$R = \left\{ \begin{array}{lcl} S & \to & aAb, \\ A & \to & B \mid c, \\ B & \to & BCB \mid \varepsilon, \\ C & \to & aA \mid cc, \\ D & \to & SB \end{array} \right\}$$

(a) Enthält diese Grammatik G nutzlose Variablen? Falls dies zutrifft, entfernen Sie diese. Begründen Sie in beiden Fällen Ihre Entscheidung.

Lösung: In unserer Grammatik G ist D eine nutzlose Variable, da keine Ableitung $S \overset{\star}{\to} w$ existiert in der D vorkommt.

(b) Formen Sie die obige Grammatik G nach Entfernen nutzloser Variablen so um, dass diese in Chomsky-Normalform ist. Geben Sie die einzelnen Schritte aus der Vorlesung an und beschreiben Sie Ihr Vorgehen. Sie können dabei auf die topologische Sortierung im letzten Schritt verzichten.

Lösung:

Schritt 1: Alle Regeln enthalten auf der rechten Seite nur Symbole aus V oder nur ein Symbol aus Σ. Ersetze dazu a, b und c mit neuen Variablen bspw. Z_A, Z_B und Z_C.

$$\begin{array}{lcl} S & \to & aAb, \\ A & \to & B \mid c, \\ B & \to & BCB \mid \varepsilon, \\ C & \to & aA \mid cc, \end{array} \qquad \begin{array}{lcl} S & \to & Z_A A Z_B, \\ A & \to & B \mid Z_C, \\ B & \to & BCB \mid \varepsilon, \\ C & \to & Z_A A \mid Z_C Z_C, \\ Z_A & \to & a, \\ Z_B & \to & b, \\ Z_C & \to & c. \end{array}$$

Schritt 2: Alle rechten Seiten haben eine Länge ≤ 2.

$$\begin{array}{lcl} S & \to & Z_A E, \\ A & \to & B \mid Z_C, \\ B & \to & BF \mid \varepsilon, \\ C & \to & Z_A A \mid Z_C Z_C, \\ Z_A & \to & a, \\ Z_B & \to & b, \\ Z_C & \to & c, \\ E & \to & A Z_B, \\ F & \to & CB. \end{array}$$

Schritt 3: Eliminiere alle Regeln $S \to \varepsilon$, $C \to \varepsilon$ und Nachfolgende.

$$\begin{array}{lcl} S & \to & Z_A E, \\ A & \to & B \mid Z_C, \\ B & \to & BF \mid F, \\ C & \to & Z_A A \mid Z_A \mid Z_C Z_C, \\ Z_A & \to & a, \\ Z_B & \to & b, \\ Z_C & \to & c, \\ E & \to & A Z_B \mid Z_B, \\ F & \to & CB \mid C. \end{array}$$

Schritt 4: Eliminiere Kettenregel mit linker Seite A. Eliminiere Kettenregel mit linker Seite Z_B und Z_C.

$$\begin{array}{lcl} S & \to & Z_A E, \\ A & \to & BF \mid CB \mid Z_A A \mid a \mid Z_C Z_C \mid Z_C, \\ B & \to & BF \mid CB \mid Z_A A \mid a \mid Z_C Z_C, \\ C & \to & Z_A A \mid a \mid Z_C Z_C, \\ Z_A & \to & a, \\ Z_B & \to & b, \\ Z_C & \to & c, \\ E & \to & AZ_B \mid Z_B, \\ F & \to & CB \mid Z_A A \mid a \mid Z_C Z_C. \end{array}$$

Eliminiere Kettenregel mit linker Seite Z_B und Z_C:

$$\begin{array}{lcl} S & \to & Z_A E, \\ A & \to & BF \mid CB \mid Z_A A \mid a \mid Z_C Z_C \mid c, \\ B & \to & BF \mid CB \mid Z_A A \mid a \mid Z_C Z_C, \\ C & \to & Z_A A \mid a \mid Z_C Z_C, \\ Z_A & \to & a, \\ Z_B & \to & b, \\ Z_C & \to & c, \\ E & \to & AZ_B \mid b, \\ F & \to & CB \mid Z_A A \mid a \mid Z_C Z_C. \end{array}$$

Problem 8: Kellerautomaten 3+2=5 Punkte

Gegeben sei der Kellerautomat $P = (\{q_0, q_1, q_2, q_3\}, \{a, b, c\}, \{X, Y, Z_0\}, q_0, Z_0, \delta)$, der mit leerem Stack akzeptiert, wobei die Übergangsfunktion δ wie folgt definiert ist:

$$
\begin{array}{lrcl}
1. & \delta(q_0, a, \varepsilon) & = & (q_0, X), \\
2. & \delta(q_0, b, \varepsilon) & = & (q_1, Y), \\
3. & \delta(q_0, c, \varepsilon) & = & (q_2, \varepsilon), \\
4. & \delta(q_1, b, \varepsilon) & = & (q_1, Y), \\
5. & \delta(q_1, c, \varepsilon) & = & (q_2, \varepsilon), \\
6. & \delta(q_2, b, Y) & = & (q_2, \varepsilon), \\
7. & \delta(q_2, a, X) & = & (q_3, \varepsilon), \\
8. & \delta(q_3, a, X) & = & (q_3, \varepsilon), \\
9. & \delta(q_3, \varepsilon, Z_0) & = & (q_3, \varepsilon).
\end{array}
$$

(a) Geben Sie die von dem Kellerautomaten P akzeptierte Sprache L an. Zudem wandeln Sie P in eine Grammatik G um, sodass $L(G) = L$ gilt.

Lösung:

- $L(G) = \{a^n b^m c b^m a^n \in \{a, b, c\} \mid m \in \mathbb{N}_0, n \in \mathbb{N} \setminus \{0\}\}$
- Grammatik $G = (\Sigma, V, S, R)$ mit Terminalen $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$ und den Produktionen R, die wie folgt gegeben sind:

$$
\begin{array}{lcl}
S & \to & aSa \mid aAa, \\
A & \to & bAb \mid B, \\
B & \to & c.
\end{array}
$$

(b) Geben Sie an, ob der Kellerautomat P deterministisch ist. Begründen Sie Ihre Antwort.

Lösung: Der Kellerautomat ist deterministisch, da dieser in keiner Situation für $\delta(q, a, X)$ mehr als ein Paar enthält und Folgendes gilt:

Ein DPDA ist genau dann deterministisch, wenn folgende Bedingungen gelten:

a) $\delta(q, a, X)$ besitzt für jedes $q \in Q$, $a \in \Sigma$ und $a = \varepsilon$ und $X \in \Gamma$ höchstens ein Element.

b) Wenn $\delta(q, a, X)$ für ein $a \in \Sigma$ nicht leer ist, dann muss $\delta(q, \varepsilon, X)$ leer sein.

Problem 9: Verschiedenes 7 Punkte

Jeder der folgenden Aussagenblöcke umfasst mehrere Einzelaussagen. Die sich unterscheidenden Aussagenbausteine sind durch Kästchen gekennzeichnet. Kreuzen Sie genau jene Bausteine an, die in einer wahren Einzelaussage enthalten sind. Jeder Aussagenblock enthält mindestens eine wahre Einzelaussage. Unvollständig oder falsch angekreuzte Aussagenblöcke werden mit null Punkten bewertet. Sie erhalten einen Punkt für jeden Aussagenblock, für den Sie genau die richtige Menge an Aussagenbausteinen angekreuzt haben.

Sei $G = (\Sigma, V, S, R)$ eine Grammatik.

- ☒ Falls G kontextfrei ist, dann haben alle Ableitungsregeln in R die Form $A \to v$ mit $A \in V$ und $v \in (V \cup \Sigma)^\star$.
- ☐ Falls G kontextsensitiv ist, darf jedes Nichtterminal auf das leere Wort abgeleitet werden.
- ☐ Falls G eine rekursiv-aufzählbare Grammatik ist, dann gilt für jede Ableitungsregel, dass die linke Seite weniger Nichterminale als die rechte Seite besitzt.
- ☐ Falls G eine rechtslineare Grammatik ist, dann besteht die rechte Seite jeder Ableitungsregel aus einem Terminal gefolgt von einem Nichtterminal.

Der Cocke-Younger-Kasami Algorithmus

- ☒ überprüft, ob für eine gegebene Chomsky-2-Grammatik G in Chomsky-Normalform ein Wort w in $L(G)$ liegt.
- ☐ überprüft, ob für eine gegebene Chomsky-1-Grammatik G ein Wort w in $L(G)$ liegt.
- ☒ besitzt eine polynomielle Laufzeit.

Sei L eine Sprache über dem Alphabet Σ und sei $L^c = \Sigma^\star \setminus L$ das Komplement zu L, sodass L und L^c semientscheibar sind.

- ☐ Weder L noch L^c ist entscheidbar.
- ☐ Entweder L oder L^c ist entscheidbar.
- ☒ L und L^c sind entscheidbar.
- ☒ Die Sprache $L \cup L^c$ ist entscheidbar.

Kreuzen Sie an, welche der folgenden Aussagen wahr ist.

- ☒ Die Klasse der von Chomsky-1-Grammatiken erzeugten Sprachen stimmt mit der Klasse NTAPE(n) überein, wobei n die Eingabelänge ist.
- ☐ Für jede Sprache, die durch einen nichtdeterministischen Kellerautomaten akzeptiert wird, gibt es einen deterministischen Kellerautomaten, der dieselbe Sprache akzeptiert.
- ☒ Zu jedem nichtdeterministischen endlichen Automaten gibt es einen äquivalenten deterministischen endlichen Automaten.
- ☒ Jede Turingmaschine mit mehreren Bändern kann von einer Turingmaschine mit einem Band simuliert werden.

Die maximale Entropie für 42 Buchstaben ist auf zwei Nachkommastellen gerundet (unter Verwendung des Logarithmus zur Basis 2)

- ☐ $4,43$
- ☐ $7,81$
- ☒ $5,39$
- ☐ $3,14$

Kreuzen Sie an, welche der folgenden Aussagen wahr ist.

- ☒ Für einen Präfix-Code gilt, dass kein Codewort Anfang eines anderen Codeworts ist.
- ☐ Die Huffman-Codierung setzt nicht voraus, dass die Wahrscheinlichkeiten der Zeichen im Voraus bekannt sind.
- ☐ Die Datenkompression erhöht die Redundanz und reduziert damit die Fehleranfälligkeit.
- ☒ Der Huffman-Algorithmus berechnet einen Codierungsbaum mit minimaler mittlerer Codewortlänge.

Kreuzen Sie an, welche der folgenden Aussagen wahr ist.

- ☒ Ein Block-Code betrachtet Codewörter fester Länge, wobei aufeinanderfolgende Blöcke unabhängig voneinander kodiert werden.
- ☐ Die mittlere Codewortlänge der Shannon-Fano Codierung ist optimal.
- ☐ In einem Faltungs-Code können Codewörter beliebige Länge besitzen, wobei die Zeichen vom Vorgeschehen unabhängig sind.
- ☒ Jeder Paritätscode erkennt Einzelfehler.