Institut für Informatik der Ludwig-Maximilians-Universität München
Prof. Dr. Jasmin Blanchette
Elisabeth Lempa
Luca Maio

SoSe 2025
31.07.2025

Lösung zur regulären Klausur zur Vorlesung

Formale Sprachen und Komplexität

Die Bearbeitungszeit beträgt **120 Minuten**. Hilfsmittel sind nicht erlaubt, auch das Mitführen ausgeschalteter elektronischer Geräte (inklusive Smartwatches aber exklusive normaler Uhren) wird als Täuschungsversuch gewertet. Schreiben Sie Ihren vollständigen Namen und Ihre Matrikelnummer deutlich lesbar auf dieses Deckblatt, sowie Ihren Namen in die Kopfzeile auf jedem Blatt der Klausurangabe. Geben Sie alle Blätter ab. Lassen Sie diese zusammengeheftet. Verwenden Sie nur **dokumentenechte Stifte** und **weder** die Farbe **rot noch grün**.

Kontrollieren Sie, ob Sie alle Aufgabenblätter erhalten haben. Aufgabenstellungen befinden sich auf den **Seiten 1–11**. Sie dürfen die Rückseiten für Nebenrechnungen nutzen. Falls Sie die Rückseiten für Antworten nutzen, so markieren Sie klar, was zu welcher Aufgabe gehört und geben Sie in der entsprechenden Aufgabe an, wo alle Teile Ihrer Antwort zu finden sind. Streichen Sie alles durch, was nicht korrigiert werden soll.

Es gibt 5 unterschiedlich gewichtete Aufgaben zu insgesamt 100 Punkten. Mit 50 Punkten haben Sie sicher bestanden. Die Teilaufgaben können unabhängig voneinander bearbeitet werden.

Mit Ihrer Unterschrift bestätigen Sie, dass Sie zu Beginn der Klausur in ausreichend guter gesundheitlicher Verfassung sind und diese Klausurprüfung verbindlich annehmen.

Nachname (in GROSSBUCHSTABEN):

Vorname (in GROSSBUCHSTABEN):

Matrikelnummer:

Studiengang:

Hiermit erkläre ich die Richtigkeit der obigen Angaben: _____

Unterschrift

Die folgende Tabelle nicht ausfüllen:

Aufgabe	1	2	3	4	5	$\sum$
Punkte	30	18	20	22	10	100
Erreicht						

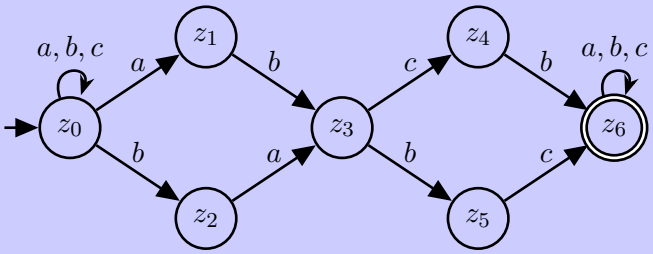
Lösung Aufgabe 1 (Reguläre Sprachen): **(30 Punkte)**

a) Sei L_1 die Sprache über dem Alphabet $\{a, b, c\}$, die vom regulären Ausdruck

$$(a|b|c)^*(ab|ba)(cb|bc)(a|b|c)^*$$

erzeugt wird. Geben Sie den Zustandsgraphen eines nichtdeterministischen endlichen Automaten (ohne ε-Übergänge) an, der L_1 akzeptiert. (10 Punkte)

LÖSUNGSVORSCHLAG:



- 10 Punkte, wenn alles stimmt
  - 1 Punkt Abzug je fehlendem oder falschen Startzustand
  - 1 Punkt Abzug je fehlendem oder falschen Endzustand
  - 1 Punkt Abzug bei jedem falschen (z.B. fehlenden oder zusätzlichen) Übergang oder Zustand oder bei jedem ε-Übergang

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

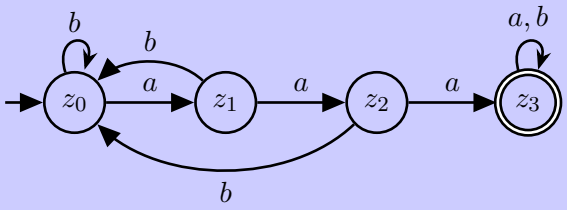
Fortsetzung von Aufgabe 1:

b) Sei L_2 die Sprache über dem Alphabet $\{a, b\}$, die vom regulären Ausdruck

$$(a|b)^*aaa(a|b)^*$$

erzeugt wird. Geben Sie den Zustandsgraphen eines deterministischen endlichen Automaten an, der L_2 akzeptiert. Bitte zeichnen Sie alle Übergänge und Zustände sorgfältig, inklusive des Müllzustands (falls vorhanden). (10 Punkte)

LÖSUNGSVORSCHLAG:

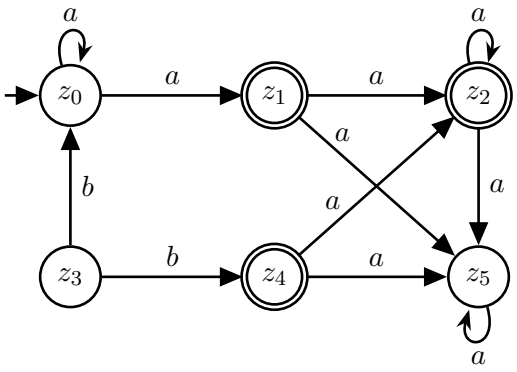


- 10 Punkte, wenn alles stimmt
  - 1 Punkt Abzug je fehlendem oder falschen Startzustand
  - 1 Punkt Abzug je fehlendem oder falschen Endzustand
  - 1 Punkt Abzug bei jedem falschen (z.B. fehlenden oder zusätzlichen) Übergang oder Zustand

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

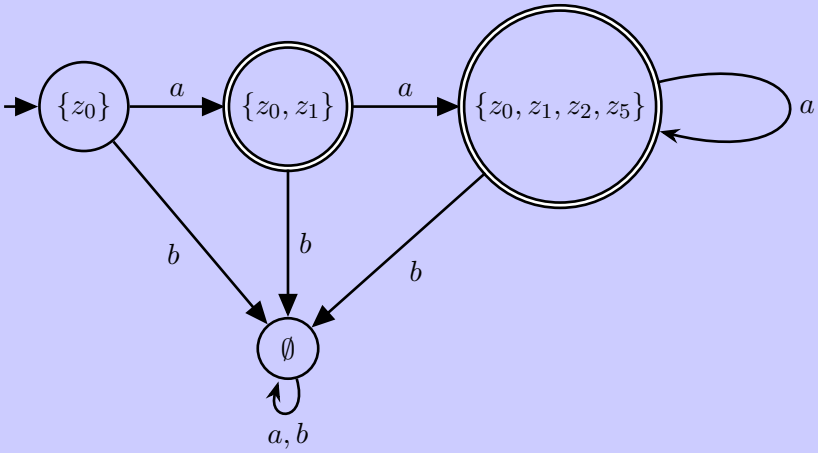
Fortsetzung von Aufgabe 1:

c) Der nichtdeterministische endliche Automat M_1 über dem Alphabet $\{a, b\}$ ist durch den folgenden Zustandsgraphen gegeben:



Berechnen Sie einen deterministischen endlichen Automaten M_1' aus M_1 mit der Potenzmengenkonstruktion. Es ist ausreichend, die erreichbaren Zustände von M_1' anzugeben. Sie müssen keine Begründung angeben, nur das Ergebnis. (10 Punkte)

LÖSUNGSVORSCHLAG:



- 10 Punkte, wenn alles stimmt
  - 1 Punkt Abzug bei fehlendem oder falschen Startzustand
  - 1 Punkt Abzug bei jedem fehlenden oder falschen Endzustand
  - 3 Punkte Abzug, falls $\emptyset$ und die dazugehörigen Übergänge fehlen
  - 1 Punkt Abzug bei jedem sonstigen falschen (z.B. fehlenden oder zusätzlichen) Übergang oder Zustand
  - Höchstens 2 Punkte, falls der angegebene Automat ein NFA und kein DFA ist (Ausnahme: es wurde lediglich der "Müllzustand" vergessen.)

Lösung Aufgabe 2 (Nicht reguläre Sprachen): (18 Punkte)

a) Zeigen Sie mit dem Satz von Myhill und Nerode, dass die Sprache

$$L_1 = \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

über dem Alphabet $\{a, b, c\}$ nicht regulär ist. Geben Sie für jede Äquivalenzklasse $[u_i]_{\sim_{L_1}}$, die Sie im Beweis nutzen, ein Suffix w_i an, sodass $u_i w_i \in L_1$ aber $u_j w_i \notin L_1$ für jedes $j \neq i$.
(10 Punkte)

LÖSUNGSVORSCHLAG:
$[ab]_{\sim_{L_1}}$ mit Suffix c
$[aabb]_{\sim_{L_1}}$ mit Suffix cc
$[aaabbb]_{\sim_{L_1}}$ mit Suffix ccc
$[aaaabbbb]_{\sim_{L_1}}$ mit Suffix $cccc$
usw.

Daher ist der Index von $\sim_{L_1}$ unendlich und L_1 ist nicht regulär.

- 5 Punkte: Äquivalenzklassen
  - 3 Punkte, wenn die Äquivalenzklassen disjunkt sind
  - 2 Punkte, wenn unendlich viele Äquivalenzklassen angegeben wurden (mit „usw.“ oder Ähnlichem)
- 4 Punkte: korrekte Suffixe
- 1 Punkt: korrekte Schlussfolgerung

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 2:

b) Beweisen Sie mithilfe der Abschlusseigenschaften der regulären Sprachen, dass die Sprache

$$L_2 = \overline{\{a^i b^i c^i d^j \mid i, j \in \mathbb{N}\}}$$

über dem Alphabet $\{a, b, c, d\}$ nicht regulär ist. $\overline{L}$ bezeichnet dabei das Komplement einer Sprache L. Sie dürfen annehmen, dass die Sprache $L_1 = \{a^i b^i c^i \mid i \in \mathbb{N}\}$ aus Teilfrage a) nicht regulär ist.

Zur Erinnerung: Die regulären Sprachen sind unter Vereinigung, Schnitt, Komplement, Produkt und Kleeneschem Abschluss abgeschlossen. (8 Punkte)

LÖSUNGSVORSCHLAG: Wir nehmen an, dass L_2 regulär ist. Das heißt, dass ihr Komplement $\overline{\overline{\{a^i b^i c^i d^j \mid i, j \in \mathbb{N}\}}} = \{a^i b^i c^i d^j \mid i, j \in \mathbb{N}\}$ auch regulär ist. Somit ist der Schnitt der offensichtlich regulären Sprache $L(a^* b^* c^*) = \{a^i b^j c^k \mid i, j, k \in \mathbb{N}\}$ mit $\{a^i b^i c^i d^j \mid i, j \in \mathbb{N}\}$ auch regulär. Aber dies ist genau die Sprache L_1, die nicht regulär ist. Widerspruch.

- 1 Punkt: Annahme, dass L_2 regulär ist.
- 1 Punkt: Komplement ist regulär.
- 3 Punkte: $\{a^i b^j c^k \mid i, j, k \in \mathbb{N}\}$ ist regulär.
- 2 Punkte: Schnitt ist regulär.
- 1 Punkt: Verweis auf Nichtregularität von L_1.

Lösung Aufgabe 3 (Kontextfreie Sprachen): **(20 Punkte)**

a) Die Sprache L_1 über dem Alphabet $\Sigma_1 = \{a, b, c, d, e\}$ sei definiert als

$$L_1 = \{a^m b^n c d^n e^m c \mid m, n \in \mathbb{N}\}$$

Geben Sie eine kontextfreie Grammatik G_1, die L_1 erzeugt, als 4-Tupel an. Die Grammatik darf keine ε-Produktionen enthalten. Erläutern Sie, warum G_1 die Sprache L_1 erzeugt. Beschreiben Sie beispielsweise, welche „Aufgabe“ die einzelnen Nichtterminale bei der Erzeugung übernehmen. (6 Punkte)

Geben Sie zusätzlich eine Linksableitung für das Wort *abbcddec* für Ihre Grammatik an. (4 Punkte)

LÖSUNGSVORSCHLAG: $G_1 = (V_1, \Sigma_1, P_1, S)$ mit $V_1 = \{S, T, U\}$ und

$$P_1 = \{S \to Tc,\ T \to aTe \mid U,\ U \to bUd \mid c\}$$

S erzeugt zunächst das letzte c, mit T vorne. Aus T lassen sich null oder mehr a/e-Paare erzeugen, mit U in der Mitte. Aus U lassen sich null oder mehr b/d-Paare erzeugen, mit c in der Mitte.

Linksableitung:
$S \Rightarrow Tc \Rightarrow aTec \Rightarrow aUec \Rightarrow abUdec \Rightarrow abbUddec \Rightarrow abbcddec.$

- 4 Punkte: Grammatik
  - Höchstens 1 Punkt, wenn die Grammatik nicht kontextfrei ist
  - 1 Punkt Abzug für „off by one“ bzgl. n oder m
- 2 Punkte: Erläuterung
  - Höchstens 1 Punkt bei falscher Grammatik
- 4 Punkte: Linksableitung
  - 1 Punkt, falls keine Linksableitung oder falls Syntaxbaum

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 3:

b) Sei $G_2 = (V_2, \Sigma_2, P_2, S)$ mit $V_2 = \{S, A, B\}$, $\Sigma_2 = \{a, b\}$ und

$$P_2 = \{S \to AB \mid BA,\ A \to AA \mid AB \mid a,\ B \to BA \mid a \mid b\}$$

eine kontextfreie Grammatik in Chomsky-Normalform.

Entscheiden Sie, ob $bbabab \in L(G_2)$ ist, indem Sie den Algorithmus von Cocke, Younger und Kasami (CYK-Algorithmus) ausführen. Geben Sie die dabei entstehende Tabelle und die Ausgabe des Algorithmus an. (10 Punkte)

LÖSUNGSVORSCHLAG: Tabelle:

Wort	b	b	a	b	a	b
$j\backslash i$	1	2	3	4	5	6
1	B	B	A, B	B	A, B	B
2		S, B	S, A	S, B	S, A	
3		S, B	S, A	S, B		
4		S, B	S, A			
5		S, B				
6						

Da $S \notin V(1, 6)$ ist, ist $bbabab \notin L(G_2)$.

- 1 Punkt: richtige Antwort (nein)
- 9 Punkte: Tabelle
  - 1 Punkt Abzug pro falsche Zelle (wobei Folgefehler berücksichtigt werden)

Lösung Aufgabe 4 (Berechenbarkeit und Entscheidbarkeit): (22 Punkte)

a) Geben Sie eine primitiv rekursive Funktion f an, die $n \mapsto n!$ berechnet. Zum Beispiel soll $f(4) = 4! = 24$ gelten. Sie können die Tatsachen nutzen, dass

$$0! = 1 \qquad \text{und} \qquad n! = n \cdot (n-1)! \quad \text{für } n > 0$$

gelten. Erläutern Sie, warum f primitiv rekursiv ist. Sie dürfen dabei annehmen, dass Multiplikation $(\cdot)$ primitiv rekursiv ist. (8 Punkte)

Zur Erinnerung:

Eine Funktion $f : \mathbb{N}^k \to \mathbb{N}$ ist *primitiv rekursiv*, wenn sie der folgenden Definition genügt:

- Jede *konstante Funktion* $f(x_1, \ldots, x_k) = c$ mit $c \in \mathbb{N}$ ist primitiv rekursiv.
- Die *Projektionsfunktionen* $\pi_i^k(x_1, \ldots, x_k) = x_i$ sind primitiv rekursiv.
- Die *Nachfolgerfunktion* $succ(x) = x + 1$ ist primitiv rekursiv.
- *Komposition*: Wenn die Funktionen $g : \mathbb{N}^m \to \mathbb{N}$ und $h_i : \mathbb{N}^k \to \mathbb{N}$ für $i = 1, \ldots, m$ primitiv rekursiv sind, dann ist auch f mit $f(x_1, \ldots, x_k) = g(h_1(x_1, \ldots, x_k), \ldots, h_m(x_1, \ldots, x_k))$ primitiv rekursiv.
- *Rekursion*: Wenn $g : \mathbb{N}^{k-1} \to \mathbb{N}$ und $h : \mathbb{N}^{k+1} \to \mathbb{N}$ primitiv rekursiv sind, dann ist auch folgende Funktion primitiv rekursiv:

$$f(x_1, \ldots, x_k) = \begin{cases} g(x_2, \ldots, x_k) & \text{falls } x_1 = 0 \\ h(f(x_1 - 1, x_2, \ldots, x_k), x_1 - 1, x_2, \ldots, x_k) & \text{sonst} \end{cases}$$

LÖSUNGSVORSCHLAG:

- Rekursion:
$$f(n) = \begin{cases} g() & \text{falls } n = 0 \\ h(f(n-1), n-1) & \text{sonst} \end{cases}$$
- Konstante Funktion: $g() = 1 \in \mathbb{N}$
- Kompositionen, Projektionen, Multiplikation: $h(m, n) = (n + 1) \cdot m$
- Zusammen:
$$f(n) = \begin{cases} 0 & \text{falls } n = 0 \\ (n - 1 + 1) \cdot f(n-1) & \text{sonst} \end{cases}$$

- 2 Punkte: Die Funktion berechnet die Fakultät richtig
- 2 Punkte: Die Funktion ist primitiv rekursiv
- 4 Punkte: Erklärung
  - 1 Punkt: Rekursion
  - 1 Punkt: Basisfall (konstante Funktion)
  - 2 Punkte: rekursiver Fall (Kompositionen, Projektionen und Multiplikation)

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 4:

b) Das Postsche Korrespondenzproblem (PCP) ist bekanntlich unentscheidbar. Gegeben sei die folgende Variante des Problems:

> Sei Σ ein Alphabet mit $|\Sigma| > 1$. Eine Instanz des *gespiegelten Postschen Korrespondenzproblems* (*GPCP*) besteht aus einer endlichen Folge $(x_1, y_1), \ldots, (x_k, y_k)$ von Wortpaaren mit $x_i, y_i \in \Sigma^+$. Das Entscheidungsproblem ist die Frage, ob es eine Folge von Indizes $i_1, \ldots, i_m$ mit $i_j \in \{1, \ldots, k\}$ und $m > 0$ gibt, sodass $x_{i_1} \cdots x_{i_m} = \overline{y_{i_1} \cdots y_{i_m}}$ gilt, wobei $\overline{a_1 \cdots a_n}$ als $a_n \cdots a_1$ definiert ist.

Der folgende Beweis ist falsch. Lokalisieren Sie den Fehler und begründen Sie, wieso der Beweis falsch ist. (6 Punkte)

> **Satz:** GPCP ist unentscheidbar.
>
> *Beweis:* Wir zeigen PCP $\leq$ GPCP.
>
> Sei $f((x_1, y_1), \ldots, (x_k, y_k)) = (x_1, \overline{y_1}), \ldots, (x_k, \overline{y_k})$.
>
> Die Funktion f ist offensichtlich total und berechenbar.
>
> Korrektheit:
>
> $(x_1, y_1), \ldots, (x_k, y_k) \in$ PCP
>
> g.d.w. es gibt $i_1, \ldots, i_m$, sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdot \ldots \cdot y_{i_m}$
>
> g.d.w. es gibt $i_1, \ldots, i_m$, sodass $x_{i_1} \cdots x_{i_m} = \overline{\overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}}$
>
> g.d.w. es gibt $i_1, \ldots, i_m$, sodass $x_{i_1} \cdots x_{i_m} = \overline{\overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}}$
>
> g.d.w. $(x_1, \overline{y_1}), \ldots, (x_k, \overline{y_k}) \in$ GPCP.
>
> Wenn GPCP entscheidbar wäre, könnten wir obige Konstruktion verwenden, um PCP zu entscheiden. Dies wäre dann ein Widerspruch zur Unentscheidbarkeit von PCP. Daher muss auch GPCP unentscheidbar sein.

LÖSUNGSVORSCHLAG: Der Fehler liegt im Schritt „es gibt $i_1, \ldots, i_m$, sodass $x_{i_1} \cdots x_{i_m} = \overline{\overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}}$ g.d.w. es gibt $i_1, \ldots, i_m$, sodass $x_{i_1} \cdots x_{i_m} = \overline{\overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}}$“ Im Allgemeinen gilt $\overline{\overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}} = \overline{y_{i_m}} \cdot \ldots \cdot \overline{y_{i_1}} \neq \overline{y_{i_1}} \cdot \ldots \cdot \overline{y_{i_m}}$ per Definition von $\overline{\ }$.

- 4 Punkte: Ort des Fehlers korrekt identifiziert
- 2 Punkte: kurze Begründung

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 4:

c) Der Satz von Rice besagt:

> Sei $\mathcal{R}$ die Klasse aller turingberechenbaren Funktionen. Sei $\mathcal{S}$ eine nichtleere echte Teilmenge von $\mathcal{R}$. Dann ist folgende Sprache unentscheidbar:
>
> $$C(\mathcal{S}) = \{w \mid \text{die von der deterministischen Turingmaschine } M_w \text{ berechnete Funktion liegt in } \mathcal{S}\}$$
>
> wobei M_w die Turingmaschine mit der Gödelnummer w bezeichnet.

Wenden Sie für jede der beiden Aussagen unten den Satz von Rice an, um sie zu beweisen, oder erklären Sie, warum der Satz nicht anwendbar ist.

(i) Es ist unentscheidbar, ob eine gegebene deterministische Turingmaschine für keine Eingabe $i \in \mathbb{N}$ die Zahl i berechnet. (5 Punkte)

LÖSUNGSVORSCHLAG: Der Satz ist anwendbar.
Sei $\mathcal{S}$ die Menge aller (partiellen oder totalen) Funktionen f, sodass es kein i mit $f(i) = i$ gibt.
$\mathcal{S}$ ist nicht leer. Z. B. ist $(i \mapsto i + 1) \in \mathcal{S}$.
$\mathcal{S}$ ist auch nicht gleich $\mathcal{R}$, weil es z. B. die Funktion $(i \mapsto i) \in \mathcal{R} \setminus \mathcal{S}$ gibt.
Dann ist folgende Sprache unentscheidbar:

$$\begin{aligned} C(\mathcal{S}) &= \{w \mid \text{die von der deterministischen Turingmaschine } M_w \text{ berechnete Funktion liegt in } \mathcal{S}\} \\ &= \{w \mid \text{die Turingmaschine } M_w \text{ berechnet eine Funktion } f, \text{ welche für keine Eingabe } i \in \mathbb{N} \text{ die Zahl } i \text{ berechnet}\} \end{aligned}$$

- 1 Punkt: richtige Antwort (anwendbar) bzw. es wird versucht, den Satz anzuwenden
- 1 Punkt: Definition von $\mathcal{S}$
- 1 Punkt: „$\mathcal{S}$ nicht leer“-Beweis
- 1 Punkt: „$\mathcal{S}$ nicht gleich $\mathcal{R}$“-Beweis
- 1 Punkt: „$C(\mathcal{S}) = \cdots$“-Beweis

(ii) Es ist unentscheidbar, ob eine gegebene deterministische Turingmaschine für alle Eingaben erst nach 1000 Schritten anhält. (3 Punkte)

LÖSUNGSVORSCHLAG: Der Satz ist nicht anwendbar, weil die Eigenschaft eine der Turingmaschine und nicht ihrer Sprache ist.

- 1 Punkt: richtige Antwort (nicht anwendbar)
- 2 Punkte: kurze Begründung

Lösung Aufgabe 5 (Komplexität): **(10 Punkte)**

In der Vorlesung wurde das VERTEX-COVER-Problem vorgestellt:

gegeben:	ein ungerichteter Graph $G = (V, E)$ und eine Zahl $k \in \mathbb{N}$
gefragt:	Besitzt G eine überdeckende Knotenmenge der Größe höchstens k?

Für einen Graphen $G = (V, E)$ ist $V' \subseteq V$ eine *überdeckende Knotenmenge*, wenn jede Kante aus E mindestens einen Knoten in V' hat, d. h. für alle Knoten $u, v \in V$: $\{u, v\} \in E$ impliziert $u \in V' \vee v \in V'$.

Sie dürfen als bekannt annehmen, dass VERTEX-COVER $\mathcal{NP}$-vollständig ist.

Nun führen wir folgendes HYPERCOVER-Problem ein:

gegeben:	eine endliche Grundmenge X, eine Menge von Mengen $\mathbf{M} \subseteq \mathcal{P}(X)$ und eine Zahl $\ell \in \mathbb{N}$
gefragt:	Gibt es eine Menge $X' \subseteq X$, sodass $\lvert X' \rvert \leq \ell$ und jede Menge aus $\mathbf{M}$ mindestens ein Element in X' hat, d. h. für jede Menge $M \in \mathbf{M}$ gilt $M \cap X' \neq \emptyset$?

Beweisen Sie mithilfe einer Polynomialzeit-Reduktion, dass HYPERCOVER $\mathcal{NP}$-schwer ist.
(10 Punkte)

LÖSUNGSVORSCHLAG: Wir zeigen VERTEX-COVER $\leq_p$ HYPERCOVER.

Sei $((V, E), k)$ eine VERTEX-COVER-Instanz. Wir setzen $f((V, E), k) = (X, \mathbf{M}, \ell)$ mit $X = V$, $\mathbf{M} = E$ und $\ell = k$.

Die Funktion f ist offensichtlich total und lässt sich in Polynomialzeit berechnen.

Korrektheit:

$((V, E), k) \in$ VERTEX-COVER
g.d.w. es existiert $V' \subseteq V$ mit $\lvert V' \rvert \leq k$, sodass für alle $\{u, v\} \in E$ gilt $u \in V' \vee v \in V'$
g.d.w. es existiert $V' \subseteq V$ mit $\lvert V' \rvert \leq k$, sodass für jede Menge $\{u, v\} \in E$ gilt $\{u, v\} \cap V' \neq \emptyset$
g.d.w. es existiert $X' \subseteq X$ mit $\lvert X' \rvert \leq \ell$, sodass für jede Menge $M \in \mathbf{M}$ gilt $M \cap X' \neq \emptyset$
g.d.w. $(X, \mathbf{M}, \ell) \in$ HYPERCOVER.

- 2 Punkte: „VERTEX-COVER $\leq_p$ HYPERCOVER“
- 3 Punkte: Definition von f
  - 1 Punkt: X
  - 1 Punkt: $\mathbf{M}$
  - 1 Punkt: ℓ
- 1 Punkt: f ist (total und) polynomiell
- 4 Punkte: „g.d.w.“-Beweis
  - 2 Punkte: Sinnvolle Beweisstruktur gewählt ($((V, E), k) \in$ VERTEX-COVER g.d.w. $(X, \mathbf{M}, \ell) \in$ HYPERCOVER oder Kontraposition oder Biimplikation.)

  - 2 Punkte: Ausfalten der Definitionen von VERTEX-COVER und HYPERCOVER
  - Falls Beweis fehlt oder falsch ist: 1 Punkt für "Zu zeigen ist $x \in$ VERTEX-COVER g.d.w. $f(x) \in$ HYPERCOVER" o.ä.
- Insgesamt höchstens 2 Punkte, falls die Reduktion falsch herum ist