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

SoSe 2025
30.09.2025

Lösung zur Wiederholungsklausur 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 1–12**. 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	28	18	20	18	16	100
Erreicht						

Lösung Aufgabe 1 (Reguläre Sprachen): **(28 Punkte)**

a) Die Sprache L_1 sei definiert als die Menge aller Wörter über dem Alphabet $\{a, b, c\}$, die genau ein a und zwei b's enthalten.

Geben Sie einen regulären Ausdruck an, der L_1 erzeugt. (8 Punkte)

LÖSUNGSVORSCHLAG:

$c^*ac^*bc^*bc^* \mid c^*bc^*ac^*bc^* \mid c^*bc^*bc^*ac^*$.

- 0 Punkte, wenn die Antwort sich nicht als regulärer Ausdruck verstehen lässt
- 4 Punkte: Vollständigkeit
  - 2 Punkte, wenn nur $L(c^*ac^*bc^*bc^*)$ o. Ä. erzeugt wird.
- 4 Punkte: „Soundness“
  - 0 Punkte, wenn nicht erlaubte Wörter erzeugt werden

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

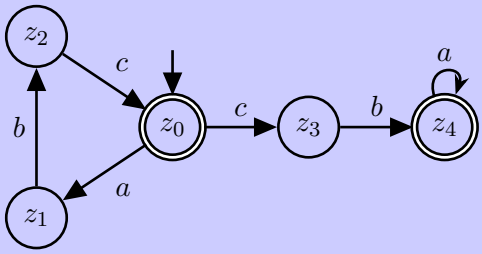
Fortsetzung von Aufgabe 1:

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

$$(abc)^*(cba^* \mid \varepsilon)$$

erzeugt wird. Geben Sie den Zustandsgraphen eines nichtdeterministischen endlichen Automaten (ohne ε-Übergänge) an, der L_2 akzeptiert. (10 Punkte)

LÖSUNGSVORSCHLAG:

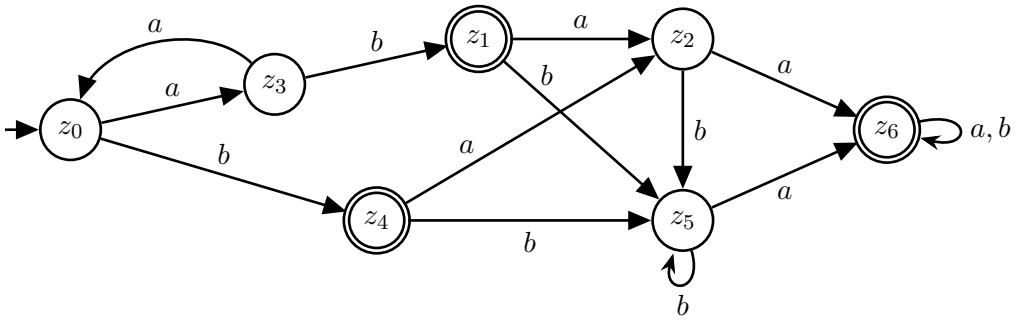


- 10 Punkte, wenn alles stimmt
  - 1 Punkt Abzug bei fehlendem Startzustand
  - 1 Punkt Abzug bei jedem fehlendem Endzustand
  - 1 Punkt Abzug bei jedem falschen (z.B. fehlenden oder zusätzlichen) Übergang oder Zustand oder bei jedem ε-Übergang

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 1:

c) Minimieren Sie den folgenden deterministischen endlichen Automaten (d. h. konstruieren Sie einen deterministischen endlichen Automaten, der die gleiche Sprache akzeptiert und eine minimale Anzahl an Zuständen benutzt). Nutzen Sie dazu das Verfahren aus der Vorlesung. Geben Sie eine Partitionstabelle und den minimierten Automaten als Zustandsgraphen an. (10 Punkte)



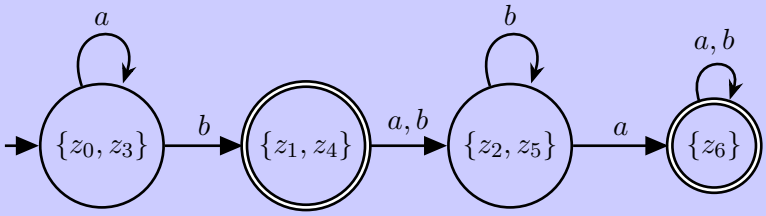
LÖSUNGSVORSCHLAG:

Unerreichbare Zustände: $\emptyset$.

Partitionstabelle:

0. $z_0 \quad z_2 \quad z_3 \quad z_5 \mid z_1 \quad z_4 \quad z_6$
1. $z_0 \quad z_2 \quad z_3 \quad z_5 \mid z_1 \quad z_4 \mid z_6$ mit a
2. $z_0 \quad z_3 \mid z_2 \quad z_5 \mid z_1 \quad z_4 \mid z_6$ mit b

Zustandsgraph:



- 6 Punkte: Partitionstabelle
  - 2 Punkte pro Zeile (inklusive „mit a/b“)
  - 1 Punkt Abzug für falsche unerreichbare Zustände
  - 2 Punkte Abzug, falls eine andere Minimierungsmethode verwendet wurde
- 4 Punkte: minimierter Automat
  - 1 Punkt Abzug, wenn Start- und/oder Endzustand nicht gekennzeichnet sind
  - 1 Punkt Abzug bei jedem falschen (z.B. fehlenden oder zusätzlichen) Übergang oder Zustand

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

a) Zeigen Sie mit dem Pumping-Lemma für reguläre Sprachen, dass die Sprache

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

über dem Alphabet $\{a, b, c, d\}$ nicht regulär ist. (10 Punkte)

Zur Erinnerung:

Eine Sprache L hat die Pumping-Eigenschaft, wenn es eine Zahl $n \in \mathbb{N}_{>0}$ gibt, sodass jedes Wort $z \in L$, welches Mindestlänge n hat, als $z = uvw$ geschrieben werden kann, sodass $|uv| \leq n$, $|v| \geq 1$ und für alle $i \in \mathbb{N}$: $uv^i w \in L$.

Das Pumping-Lemma für reguläre Sprachen besagt, dass jede reguläre Sprache die Pumping-Eigenschaft hat.

LÖSUNGSVORSCHLAG: Um Nichtregularität von L_1 zu zeigen, reicht es durch das Pumping-Lemma zu zeigen, dass L_1 die Pumping-Eigenschaft nicht hat.

Der Beweis ist durch Widerspruch: Wir nehmen an, dass L_1 die Pumping-Eigenschaft hat. Wir wählen $z = a^n b^n c^n d^n$. Damit sind auch $z \in L_1$ und $|z| \geq n$ erfüllt.

Sei $z = uvw$ eine beliebige Zerlegung von z, sodass $|uv| \leq n$, $|v| \geq 1$ und $uv^i w \in L_1$ für jedes $i \in \mathbb{N}$.

Dann ist $u = a^{n_1}$, $v = a^{n_2}$ und $w = a^{n_3} b^n c^n d^n$ mit $n_1 + n_2 + n_3 = n$ und $n_2 > 0$. Wir haben $uv^0 w \notin L_1$, denn $uv^0 w = a^{n_1+n_3} b^n c^n d^n$ und weil $n_1 + n_3 < n$ gibt es weniger a's als b's, c's oder d's. Widerspruch zur Annahme, dass $uv^i w \in L_1$ für jedes $i \in \mathbb{N}$.

- 1 Punkt: $z \in L_1$ wird erwähnt
- 1 Punkt: $|z| \geq n$ wird erwähnt
- 2 Punkte: z ist tatsächlich geeignet
- 3 Punkte: Über alle Zerlegungen u, v, w argumentiert
  - 0 Punkte Abzug, wenn unnötige Fälle betrachtet wurden
- 3 Punkte: i gefunden, sodass $uv^i w \notin L_1$

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 2:

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

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

über dem Alphabet $\{a, b, c, d, e\}$ nicht regulär ist. $\overline{L}$ bezeichnet dabei das Komplement einer Sprache L. Sie dürfen annehmen, dass die Sprache $L_1 = \{a^i b^i c^i d^i \mid i \in \mathbb{N}\}$ 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 $\overline{\overline{\{e^j a^i b^i c^i d^i e^j \mid i, j \in \mathbb{N}\}}} = \{e^j a^i b^i c^i d^i e^j \mid i, j \in \mathbb{N}\}$ auch regulär ist. Somit ist der Schnitt der offensichtlich regulären Sprache $L(a^* b^* c^* d^*) = \{a^i b^j c^k d^\ell \mid i, j, k, \ell \in \mathbb{N}\}$ mit $\{e^j a^i b^i c^i d^i e^j \mid i, j \in \mathbb{N}\}$ auch regulär. Aber dies ist genau die Sprache L_1, die nicht regulär ist. Widerspruch.

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

Lösung Aufgabe 3 (Kontextfreie und kontextsensitive Sprachen): (20 Punkte)

a) Die Sprache L_1 über dem Alphabet $\{a, b\}$ sei definiert als

$$L_1 = \{ab^i ab^j a \mid i, j \in \mathbb{N}, i < j\}$$

Geben Sie eine kontextfreie Grammatik G_1 als 4-Tupel an, die L_1 erzeugt. 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 *ababbba* für Ihre Grammatik an. (4 Punkte)

LÖSUNGSVORSCHLAG: $G_1 = (V_1, \Sigma_1, P_1, S_1)$ mit $V_1 = \{S, T, B\}$, $\Sigma_1 = \{a, b\}$ und

$$P_1 = \{S \to aTa,\ T \to bTb \mid aB,\ B \to BB \mid b\}$$

S erzeugt zunächst das äußere a-Paar mit T dazwischen. Aus T lassen sich null oder mehr b/b-Paare erzeugen, mit aB in der Mitte. Aus B lassen sich ein oder mehr b's erzeugen, die dafür sorgen, dass es im zweiten b-Block mehr b's gibt als im ersten.

Linksableitung:
$S \Rightarrow aTa \Rightarrow abTba \Rightarrow abaBba \Rightarrow abaBBba \Rightarrow ababBba \Rightarrow ababbba$.

- 4 Punkte: Grammatik
  - Höchstens 1 Punkt, wenn die Grammatik nicht kontextfrei ist
  - 1 Punkt Abzug für „off by one“ bzgl. $i \in \mathbb{N}$ oder $j \in \mathbb{N}$
- 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 $G_2 = (V_2, \Sigma_2, P_2, S)$ eine kontextsensitive Grammatik mit $V_2 = \{S, B, A\}$, $\Sigma_2 = \{a, b, c\}$ und

$$P_2 = \{S \to ABSc,\ S \to ABc,\ BA \to AB,\ Bc \to bc,\ Bb \to bb,\ Ab \to ab,\ Aa \to aa\}$$

Um das Wortproblem für G_2 zu entscheiden, berechnen wir Mengen L_i^n mit $i \in \mathbb{N}$, $n \in \mathbb{N}_{>0}$, wobei

$$L_i^n = \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } S \Rightarrow_{G_2}^k w \text{ mit } k \leq i\}$$

Berechnen Sie L_i^5 für alle $i \in \mathbb{N}$. Sie müssen keine Begründung angeben. (8 Punkte)

Geben Sie zusätzlich anhand Ihrer Berechnung ein Wort über Σ_2 an, das in $L(G_2)$ enthalten ist. (2 Punkte)

LÖSUNGSVORSCHLAG:

$L_0^5 = \{S\}$

$L_1^5 = \{S\} \cup \{ABSc, ABc\}$

$L_2^5 = \{S, ABSc, ABc\} \cup \{Abc\}$

$L_3^5 = \{S, ABSc, ABc, Abc\} \cup \{abc\}$

$L_i^5 = L_3^5$ für $i > 3$

Daher $abc \in L(G_2)$.

- 8 Punkte: Schritte
  - 1 Punkt: L_0^5
  - 2 Punkte pro Schritt L_1^5, L_2^5, L_3^5
  - 1 Punkt: L_i^5 für $i > 3$
- 2 Punkte: Wort
  - 1 Punkt: Das Wort (oder die Satzform) ist in einer der berechneten Mengen L_i^5
  - 1 Punkt: Das Wort ist wirklich ein Wort und keine Satzform und hat Länge ≤ 5

Lösung Aufgabe 4 (Berechenbarkeit und Entscheidbarkeit): (18 Punkte)

a) Für eine Funktion f steht μf für die Anwendung des μ-Operators auf f. Berechnen Sie μg_i für folgende Funktionen g_i. (8 Punkte)

(i) $g_1 : \mathbb{N} \to \mathbb{N},\ g_1(x) = 50$

LÖSUNGSVORSCHLAG: μg_1 ist undefiniert.

(ii) $g_2 : \mathbb{N} \to \mathbb{N},\ g_2(x) = 3x$

LÖSUNGSVORSCHLAG: $\mu g_2 = 0$.

(iii) $g_3 : \mathbb{N} \to \mathbb{N},\ g_3(x) = 5x + 1$

LÖSUNGSVORSCHLAG: μg_3 ist undefiniert.

(iv) $g_4 : \mathbb{N} \to \mathbb{N},\ g_4(x) = x^2$

LÖSUNGSVORSCHLAG: $\mu g_4 = 0$.

- 2 Punkte pro richtige Antwort

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 4:

b) 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 jede Eingabe die Zahl 36 berechnet. (5 Punkte)

LÖSUNGSVORSCHLAG: Der Satz ist anwendbar.
Sei $\mathcal{S}$ die Menge aller (partiellen oder totalen) Funktionen f, sodass gilt $f(i) = 36$ für alle i.
$\mathcal{S}$ ist nicht leer. Z. B. ist $(i \mapsto 36) \in \mathcal{S}$.
$\mathcal{S}$ 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:

$$\begin{aligned} C(\mathcal{S}) &= \{w \mid \text{die von der deterministischen Turingmaschine } M_w \text{ berechnete Funktion liegt in } \mathcal{S}\} \\ &= \{w \mid \text{die Turingmaschine } M_w \text{ berechnet eine Funktion } f, \text{ welche für jede Eingabe die Zahl 36 berechnet}\} \end{aligned}$$

- 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 jede Eingabe $i \in \mathbb{N}$ eine Zahl $j > i$ berechnet. (5 Punkte)

LÖSUNGSVORSCHLAG: Der Satz ist anwendbar.
Sei $\mathcal{S}$ die Menge aller (partiellen oder totalen) Funktionen f, sodass gilt $f(i) > i$ für alle i.
$\mathcal{S}$ 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:

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

- 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

Lösung Aufgabe 5 (Komplexität): **(16 Punkte)**

a) Wir erinnern zunächst an die Definition des CLIQUE-Problems:

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

Ein anderes, weniger bekanntes Problem ist FRIENDS:

gegeben:	eine endliche Menge von Personen P, eine endliche Menge von Mengen $\mathbf{K} \subseteq \mathcal{P}(P)$, sodass jede Menge $K \in \mathbf{K}$ einen Freundeskreis darstellt und eine Zahl $m \in \mathbb{N}$
gefragt:	Existiert einen Freundeskreis $K \in \mathbf{K}$, sodass $\lvert K\rvert \geq m$?

Der folgende Beweis ist falsch. Finden Sie und erklären Sie den Fehler. (6 Punkte)

Satz: $\mathsf{P} = \mathsf{NP}$.

Beweis: Das FRIENDS-Problem ist in $\mathcal{P}$, denn ein Polynomialzeitalgorithmus könnte einfach $\mathbf{K}$ durchlaufen und für jede Menge $K \in \mathbf{K}$ prüfen, ob $|K| \geq m$.

Dennoch ist FRIENDS auch $\mathcal{NP}$-schwer. Der Beweis erfolgt mithilfe einer Polynomialzeit-Reduktion vom $\mathcal{NP}$-schweren Problem CLIQUE auf FRIENDS. Die Reduktionsfunktion ist definiert wie folgt:

$$f((V,E),k) = \big(\underbrace{V}_{P :=}, \underbrace{\{V' \mid V' \text{ ist eine Clique von } (V,E)\}}_{\mathbf{K} :=}, \underbrace{k}_{m :=}\big)$$

Diese Funktion ist offensichtlich total und berechenbar. Sie ist auch korrekt:

$((V,E),k) \in \text{CLIQUE}$
g.d.w. (V,E) besitzt eine Clique der Größe k
g.d.w. es existiert $K \in \{V' \mid V' \text{ ist eine Clique von } (V,E)\}$, sodass $|K| \geq m$
g.d.w. es existiert $K \in \mathbf{K}$, sodass $|K| \geq m$
g.d.w. $(P, \mathbf{K}, m) \in \text{FRIENDS}$.

FRIENDS ist sowohl NP-schwer als auch in P. Dies ist nur dann möglich, wenn $\mathsf{P} = \mathsf{NP}$.

LÖSUNGSVORSCHLAG: Der Fehler liegt darin, dass f nicht polynomiell ist. Im Beweis sollte „lässt sich in Polynomialzeit berechnen“ statt „berechenbar“ stehen, aber das wäre auch nicht richtig, weil die Berechnung von $\mathbf{K}$ exponentiell ist.

- 3 Punkte: Ort des Fehlers korrekt identifiziert
- 3 Punkte: kurze Begründung

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 5:

b) In der Vorlesung wurde das INDEPENDENT-SET-Problem vorgestellt:

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

Für einen ungerichteten Graphen $G = (V, E)$ ist $V' \subseteq V$ eine *unabhängige Knotenmenge*, wenn keine zwei Knoten aus V' über eine Kante verbunden sind, d. h. $u, v \in V'$ impliziert $\{u, v\} \notin E$.

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

Auch für gerichtete Graphen $G = (V, E)$ lassen sich unabhängige Knotenmengen definieren. Für einen gerichteten Graphen $G = (V, E)$ ist $V' \subseteq V$ eine *unabhängige Knotenmenge*, wenn keine zwei Knoten aus V' über eine Kante verbunden sind, d. h. $u, v \in V'$ impliziert $(u, v) \notin E$ und $(v, u) \notin E$.

Nun führen wir folgendes DIRECTED-INDEPENDENT-SET-Problem ein:

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

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

LÖSUNGSVORSCHLAG: Wir zeigen INDEPENDENT-SET $\leq_p$ DIRECTED-INDEPENDENT-SET.

Sei $((V, E), k)$ eine INDEPENDENT-SET-Instanz. Wir setzen $f((V, E), k) = ((V, E'), k)$ mit $E' = \{(x, y) \mid \{x, y\} \in E\}$.

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

Korrektheit:

$((V, E), k) \in$ INDEPENDENT-SET
g.d.w. (V, E) besitzt eine unabhängige Knotenmenge V' der Größe k
g.d.w. es existiert V', sodass: $u, v \in V'$ impliziert $\{u, v\} \notin E$
g.d.w. es existiert V', sodass: $u, v \in V'$ impliziert $(u, v) \notin \{(x, y) \mid \{x, y\} \in E\}$
g.d.w. es existiert V', sodass: $u, v \in V'$ impliziert $(u, v) \notin E'$
g.d.w. (V, E') besitzt eine unabhängige Knotenmenge V' der Größe k
g.d.w. $((V, E'), k) \in$ DIRECTED-INDEPENDENT-SET

- 2 Punkte: „INDEPENDENT-SET $\leq_p$ DIRECTED-INDEPENDENT-SET“
- 4 Punkte: Definition von f
  - 1 Punkt: V
  - 2 Punkte: E'
    - 0 Punkte, falls E' einfach als E definiert oder die ungerichtete Natur von E sonst nicht berücksichtigt
  - 1 Punkt: k
- 1 Punkt: f ist (total und) polynomiell
- 3 Punkte: „g.d.w.“-Beweis
  - 1 Punkt: erster und letzter Schritt

  - 2 Punkte: Ausfalten der Definitionen von INDEPENDENT-SET und DIRECTED-INDEPENDENT-SET
- Insgesamt höchstens 2 Punkte, falls die Reduktion falsch herum ist