Prof. Dr. Jasmin Blanchette
Elisabeth Lempa
Luca Maio

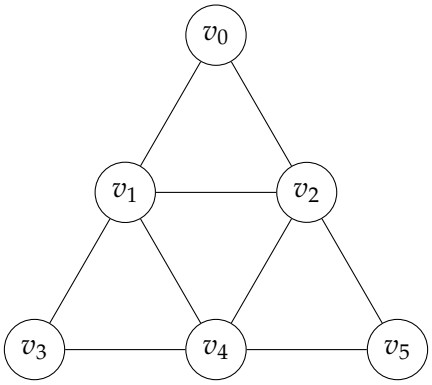
Ludwig-Maximilians-Universität München
Institut für Informatik
Besprechung 17.07.2025 bis 21.07.2025
Abgabe bis 28.07.2025, 10:00 Uhr

Lösungsvorschlag zur Übung 11 zur Vorlesung

Formale Sprachen und Komplexität

FSK11-1 ***$\mathcal{NP}$-Vollständigkeit von Graphenproblemen***

Der Graph G sei:



a) Das Independent-Clique-Problem (ICP) beantwortet für einen Graphen G und eine natürliche Zahl n die Frage, ob es in G eine Clique C der Größe n und eine unabhängige Knotenmenge I der Größe n gibt, sodass $|C \cap I| = 1$ gilt.

Beantworten Sie für alle $n \in \mathbb{N}$ die Frage, ob $(G, n) \in$ ICP gilt. Begründen Sie Ihre Antworten.

LÖSUNGSVORSCHLAG:
Als Bemerkung: Eine Clique und eine unabhängige Knotenmenge können sowieso nur maximal einen Knoten gemeinsam haben.

- Für $n = 1$: Jeder der 6 Knoten allein ist eine Clique und eine unabhängige Knotenmenge. Daher gilt $(G, 1) \in$ ICP.
- Für $n = 2$: Wähle zwei verbundene Knoten für die Clique, einen der beiden Knoten und einen nicht mit ihm verbundenen Knoten für die unabhängige Knotenmenge. Diese gibt es, z.B. $C = \{v_0, v_1\}$, $I = \{v_0, v_3\}$, also ist $(G, 2) \in$ ICP.
- Für $n = 3$: Wähle $C = \{v_0, v_1, v_2\}$ und $I = \{v_0, v_3, v_5\}$, somit $(G, 3) \in$ ICP.

- Für $n \geq 4$: Der Graph hat keine Clique der Größe 4 oder größer. Daher ist $(G, n) \notin$ ICP für alle $n \geq 4$.

b) Das Well-Liked-Clique-Problem (WLCP) beantwortet für einen Graphen G und eine natürliche Zahl n die Frage, ob es in G eine Clique C der Größe n gibt, sodass es für alle Knoten v in $G \setminus C$ einen Knoten u in C gibt, sodass $(u, v) \in E$. (Oder in Worten: Wenn es eine Clique der Größe n gibt, sodass alle Knoten außerhalb der Clique mit mindestens einem Knoten innerhalb der Clique verbunden sind.)

Beantworten Sie für alle $n \in \mathbb{N}$ die Frage, ob $(G, n) \in$ WLCP gilt. Begründen Sie Ihre Antworten.

LÖSUNGSVORSCHLAG:

- Für $n = 1$: Jeder der 6 Knoten allein ist eine Clique. Es gibt aber keinen Knoten, der mit allen anderen Knoten des Graphen verbunden ist. Daher gilt $(G, 1) \notin$ WLCP.
- Für $n = 2$: Wähle zwei Knoten aus $\{v_1, v_2, v_4\}$ als Clique. $(G, 2) \in$ WLCP.
- Für $n = 3$: Wähle $C = \{v_0, v_1, v_2\}$, somit $(G, 3) \in$ WLCP.
- Für $n \geq 4$: Der Graph hat keine Clique der Größe 4 oder größer. Daher ist $(G, n) \notin$ ICP für alle $n \geq 4$.

c) Zeigen Sie, dass ICP $\mathcal{NP}$-vollständig ist.

LÖSUNGSVORSCHLAG:

- ICP $\in \mathcal{NP}$: Rate nichtdeterministisch eine Knotenmenge C und eine Knotenmenge I von jeweils n Knoten. Verifiziere anschließend, ob C eine Clique, I eine unabhängige Knotenmenge und C und I genau einen gemeinsamen Knoten haben. Wenn ja, akzeptiere, wenn nein, verwirf. Das Verifizieren ist in deterministischer Polynomialzeit möglich.
- ICP ist $\mathcal{NP}$-schwer: Wir reduzieren CLIQUE auf ICP mit einer Polynomialzeitreduktion. Da CLIQUE $\mathcal{NP}$-vollständig ist, ist ICP damit $\mathcal{NP}$-schwer.

  Sei $f((V, E, k)) = (V', E, k)$, wobei $V' = V \cup \{v_1, \ldots, v_{k-1}\}$, sodass $\forall i \in \{1, \ldots, k-1\} : v_i \notin V$ (d.h. $v_1, \ldots, v_{k-1}$ sind neue Knoten).

  Damit ist f total und in Polynomialzeit berechenbar. Wir zeigen die

Korrektheit von f:

- Wenn (V, E, k) eine k-Clique hat, dann hat (V', E, k) ebenfalls eine k-Clique. Sei u ein beliebiger Knoten dieser Clique. Dann ist $\{u, v_1, \ldots, v_{k-1}\}$ eine unabhängige Knotenmenge der Größe k, somit $(V', E, k) \in \text{ICP}$.
- Wenn (V, E, k) keine k-Clique hat, dann hat (V', E, k) ebenfalls keine k-Clique, somit $(V', E, k) \notin \text{ICP}$.

FSK11-2 ***Bunte Graphen*** Ein bunter Graph $G = (V, E, col)$ ist ein ungerichteter Graph (V, E), wobei jeder Knoten $v \in V$ mit einer Farbe $col(v) \in \{\text{blau}, \text{gelb}, \text{rot}\}$ versehen ist. Eine blaue k-Clique ist eine k-Clique deren Knoten alle blau sind, und entsprechend für gelbe und rote k-Cliquen.

Das Bunte-Cliquen-Problem (BCP) ist:

gegeben:	Ein bunter Graph $G = (V, E, col)$ und eine Zahl $k > 0$.
gefragt:	Hat G eine blaue, eine gelbe und eine rote k-Clique?

Zeigen Sie, dass BCP $\mathcal{NP}$-vollständig ist.

LÖSUNGSVORSCHLAG:
Wir zeigen, dass BCP in $\mathcal{NP}$ liegt und anschließend, dass BCP $\mathcal{NP}$-schwer ist.

- BCP $\in \mathcal{NP}$: Rate nichtdeterministisch die drei Knotenmengen der Größe k und verifiziere anschließend, dass die Knotenmengen Cliquen sind und jeweils einheitlich in den drei unterschiedlichen Farben gefärbt sind. Falls nein, verwirf, sonst akzeptiere. Die Verifikation geht in deterministischer Polynomialzeit. Damit kann das Problem in nichtdeterministischer Polynomialzeit entschieden werden.
- BCP ist $\mathcal{NP}$-schwer: Zeige CLIQUE $\leq_{\text{p}}$ BCP: Die Reduktionsfunktion f verdreifacht den Eingabegraph in den drei Farben (und lässt k unverändert): Für $G = (V, E)$ erzeuge $G' = (V \cup V' \cup V'', E \cup E' \cup E'', col)$, wobei V' und V'' umbenannte Kopien von V und dazu passend E' und E'' Kopien von E sind. Es sei

$$col(v) = \begin{cases} \text{blau} & \text{falls } v \in V \\ \text{gelb} & \text{falls } v \in V' \\ \text{rot} & \text{falls } v \in V'' \end{cases}$$

  Die Funktion f ist total und kann in Polynomialzeit von einer DTM berechnet werden, indem diese den Eingabegraph (V, E) verdreifacht und die Funktion col erstellt. Dies geht in Linearzeit in der Größe der Eingabe.

Es gilt:

$$
\begin{array}{ll}
 & (G,k) \in \text{CLIQUE} \\
\text{g.d.w.} & G \text{ hat eine } k\text{-Clique} \\
\text{g.d.w.} & (V,E) \text{ hat eine } k\text{-Clique} \\
\text{g.d.w.} & (V,E),(V',E'),(V'',E'') \text{ haben jeweils dieselbe } k\text{-Clique} \\
\text{g.d.w.} & f((G,k)) \in \text{BCP}
\end{array}
$$

Damit ist f eine polynomielle Reduktionsfunktion, die CLIQUE $\leq_{\text{p}}$ BCP zeigt. Da CLIQUE $\mathcal{NP}$-vollständig ist, folgt die $\mathcal{NP}$-Schwere von BCP.

FSK11-3 ***$\mathcal{NP}$-vollständiges Labyrinth*** (0 Punkte)

Wir betrachten in dieser Aufgabe ein Rätselspiel mit Labyrinthen.

Die Labyrinthe sind an einem Gitter ausgerichtet. Jeweils eine Kachel ist als Start-Kachel gekennzeichnet, eine als Ziel-Kachel. Zwischen je zwei benachbarten Kacheln liegt eine von drei Möglichkeiten vor:

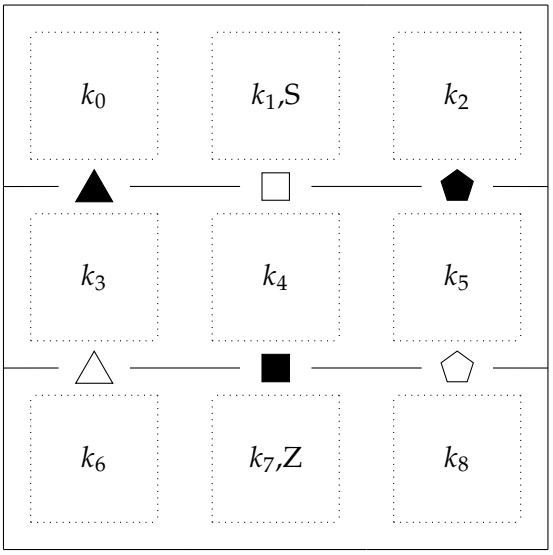
- undurchdringbare Wand
- passierbarer freier Weg
- eine schwarze oder weiße Tür, markiert mit einem Vieleck (drei oder mehr Ecken)

Die Regeln:

- Man muss einen Weg vom Start zum Ziel finden.
- Der Weg besteht aus einer Folge von Kacheln.
- Zwei im Weg aufeinanderfolgende Kacheln müssen benachbart sein (vertikal oder horizontal) und dürfen keine Wand zwischen sich haben.
- Für jedes $k \in \mathbb{N}$ gilt: Wenn im Weg eine schwarze Tür genommen wurde, die mit einem k-Eck markiert ist, dann darf keine weiße mit einem k-Eck markierte Tür auf dem Weg liegen.
- Für jedes $k \in \mathbb{N}$ gilt: Wenn im Weg eine weiße Tür genommen wurde, die mit einem k-Eck markiert ist, dann darf keine schwarze mit einem k-Eck markierte Tür auf dem Weg liegen.

a) Betrachten Sie folgendes Labyrinth, wobei `S` für den Start und `Z` für das Ziel steht.

Schwarze durchgehende Linien sind undurchdringbare Wände. Unterbrochene Linien stehen für die mit dem Symbol markierte Tür. Gepunktete Linien umranden die einzelnen Kacheln.



Bestimmen Sie alle möglichen Wege, in denen keine Kachel mehrmals genommen wird, um in diesem Labyrinth vom Start zum Ziel zu kommen.

Beschreiben Sie jeweils den Weg, indem Sie die Kachelfolge (die Abfolge der $k_.$) angeben.

Wegbeispiel: k_1, k_4, k_7 ist wegen Benutzung einer weißen, wie auch einer schwarzen Quadrattür kein gültiger Weg. Alle anderen Regeln sind dabei aber eingehalten.

LÖSUNGSVORSCHLAG:
Irgendwie muss man von der oberen Zeile in die mittlere Zeile kommen. Wir können die Wege also anhand der zuerst genommenen Tür gruppieren; da jede Form in der Türenreihe dazwischen nur einmal vorkommt, reicht es dazu die Form zum Gruppieren zu benennen.

Ferner kann der Weg nicht in die oberste Reihe zurückkehren, da der Weg keine Kachel mehrmals nehmen kann und in dem Fall alle drei Türen benutzt werden müssten – dann wäre der Weg in die unterste Reihe komplett versperrt.

Darum sind die folgenden Wege tatsächlich alle Wege, die es gibt:

- Wir verwenden in der obersten Türenreihe die Tür mit dem Dreieck

  k_1, k_0, k_3, k_4, k_7

  $k_1, k_0, k_3, k_4, k_5, k_8, k_7$
- Wir verwenden in der obersten Türenreihe die Tür mit dem Quadrat

  k_1, k_4, k_3, k_6, k_7

  k_1, k_4, k_5, k_8, k_7

- Wir verwenden in der obersten Türenreihe die Tür mit dem Fünfeck

  k_1, k_2, k_5, k_4, k_7

  $k_1, k_2, k_5, k_4, k_3, k_6, k_7$

b) Ein Labyrinth nennt man horizontal wandfrei, wenn es zwischen zwei horizontal benachbarten Kacheln immer einen passierbaren freien Weg gibt.

Zeigen Sie, dass die Frage, ob es in einem horizontal wandfreien Labyrinth einen Weg vom Start zum Ziel gibt, $\mathcal{NP}$-vollständig ist.

Hinweis: 3-CNF-SAT ist ein geeignetes Problem für die Reduktion.

LÖSUNGSVORSCHLAG:
Wir zeigen zunächst, dass jedes Labyrinth (und damit auch horizontal wandfreie) in nichtdeterministisch polynomieller Zeit gelöst werden kann.

Wenn es einen Weg vom Start zum Ziel gibt, dann gibt es auch einen Weg, in dem jede Kachel höchstens einmal vorkommt (also keine Kachel doppelt vorkommt), da Zyklen einfach entfernt werden können.

Wir raten deshalb nichtdeterministisch einen Weg (also eine Kachelabfolge), in dem jede Kachel höchstens einmal vorkommt. Dies benötigt linear viele Auswahlschritte, also insgesamt polynomiell viel Zeit.

Wir überprüfen, ob dies tatsächlich ein Weg vom Start zum Ziel ist, also am Start anfängt, die Kacheln des Weges benachbart sind und nicht durch Wände gehen (nochmal linear viele Schritte, also ein polynomieller Aufwand).

Danach gehen wir in einer Schleife über alle vorkommenden Formen. Für jede Form überprüfen wir, ob sie auf dem gewählten Weg in höchstens einer Farbe vorkommt, also ob nur schwarz, oder nur weiß.

Damit ist das Problem also in $\mathcal{NP}$.

Nun zeigen wir, dass das Problem auch $\mathcal{NP}$-schwer ist.

Wir kodieren dazu eine 3-CNF-SAT-Instanz I in einem linear großen Labyrinth, welches genau dann eine Lösung hat, wenn I eine Lösung hat.

Wir benennen alle Variablen in $x_3, x_4, \ldots$ um.

Wir wählen eine beliebige Reihenfolge der Klauseln in I, wir nennen sie $(C_1, \ldots, C_n)$. (Nicht zu verwechseln mit einer nichtdeterministischen Wahl, denn hier funktioniert jede Auswahl.)

Wir konstruieren ein horizontal wandfreies Labyrinth L der Breite 3 und der Höhe $n + 1$.

Die i-te Türreihe entsteht dabei immer aus der Klausel C_i, für alle $i \in \{1, \ldots, n\}$; z. B. etwa die Türen zwischen der obersten und zweitobersten Reihe aus der Klausel C_1.

Dabei wird aus der Klausel C_i die i-te Türreihe, indem für jedes in C_i enthaltene Literal l eine Tür wie folgt platziert wird:

- Ist l ein positives Literal, also $l = x_t$, so wird die Tür mit einem schwarzen t-Eck beschriftet (darum die Variablenbenennung ab x_3).
- Ist l ein negatives Literal, also $l = \neg x_t$, so wird die Tür mit einem weißen t-Eck beschriftet.

An den übrigen Plätzen wird eine undurchdringbare Wand platziert (z. B. wird bei $C_i = x_t \vee \neg x_s$ links eine Tür mit schwarzem t-Eck, in der Mitte eine Tür mit weißem s-Eck und rechts eine undurchdringbare Wand platziert).

Ferner setzen wir den Start nach ganz oben und das Ziel nach ganz unten.

Nun gibt es in I eine erfüllende Variablenbelegung genau dann, wenn es in L einen Weg vom Start zum Ziel gibt, denn:

Gibt es eine erfüllende Variablenbelegung von I, so gibt es auch einen Weg durch L, denn die erfüllende Variablenbelegung macht in jeder Klausel mindestens einen Wert wahr; diese Tür können wir in der entsprechenden Türreihe wählen, ohne dass wir uns irgendwann den Weg versperren.

Gibt es umgekehrt aber einen Weg durch L, so nimmt dieser in jeder Türreihe mindestens eine Türe. Wenn wir die zugehörige Variable entsprechend setzen – falls die Tür ein schwarzes k-Eck hat, setzen wir x_k auf 1, bei einem weißen k-Eck setzen wir x_k auf 0 – haben wir in der zugehörigen Klausel ein Literal wahr gemacht. Dadurch ist jede Klausel erfüllt. Da die Türen der anderen Farbe nicht mehr verwendet werden können, können wir nicht gleichzeitig eine Variable auf die Werte 1 und 0 setzen. Wir haben also eine erfüllende Variablenbelegung I konstruiert, wobei wir nicht benutzte Variablen beliebig setzen können.

Damit das Problem $\mathcal{NP}$-schwer und also insgesamt $\mathcal{NP}$-vollständig.

c) Zeigen Sie, dass die Frage, ob es in einem beliebigen Labyrinth (das nicht horizontal wandfrei sein muss) einen Weg vom Start zum Ziel gibt, ebenfalls $\mathcal{NP}$-vollständig ist.

LÖSUNGSVORSCHLAG:
In Teilaufgabe FSK11-3b) haben wir schon gezeigt, dass das Problem (auch für nicht wandfreie Labyrinthe) in $\mathcal{NP}$ liegt.

Wir reduzieren nun das Problem für horizontal wandfreie Labyrinthe auf das Problem für allgemeine Labyrinthe. Definiere $f(L) = L$. Die Funktion f ist total und in Polynomialzeit berechenbar. Sie reduziert horizontal wandfreie Labyrinthe auf Labyrinthe, sodass auch Labyrinthe $\mathcal{NP}$-schwer sind.

Damit ist das Labyrinth-Problem $\mathcal{NP}$-vollständig.

FSK11-4 ***Beweise Prüfen II***

In den folgenden Teilaufgaben betrachten wir jeweils einen Beweis, der einen Fehler enthält. Identifizieren Sie diesen Fehler (mit kurzer Begründung).

a) Wir werden beweisen, dass 3-CNF-SAT $\mathcal{NP}$-vollständig ist.

Da SAT in $\mathcal{NP}$ ist muss auch 3-CNF-SAT in $\mathcal{NP}$ sein. Wir zeigen SAT $\leq_\text{p}$ 3-CNF-SAT. Sei F eine aussagenlogische Formel. Wir transformieren F in eine Formel in 3-CNF. Dazu ersetzen wir zuerst alle Formeln $A \Leftrightarrow B$ durch $(A \Rightarrow B) \wedge (B \Rightarrow A)$. Dann ersetzen wir alle Formeln $A \Rightarrow B$ durch $\neg A \vee B$.

Nun negieren wir die Formel und nutzen die de-morganschen Gesetze und Double-Negation-Elimination so lange bis wir nur noch Literale, Konjunktion und Disjunktion vorhanden sind. Mithilfe der Distributivitäts-Gesetze formen wir nun die Formel so um, dass sie in disjunktiver Normalform ist, sie also eine mit Disjunktionen verbundene Menge von beliebig großen Konjunktionen ist.

Nun negieren wir die Formel nochmal, wodurch wir insgesamt $\neg\neg F$ erhalten, was nach den Gesetzen der klassischen Logik äquivalent ist zu F. Die äußere Negation können wir mithilfe zweischichtiger Anwendung der de-morganschen Gesetze so umformen, dass die Formel jetzt in konjunktiver Normalform, also CNF, ist. Um den letzten Schritt zu 3-CNF zu machen führen wir für jede Klausel $x_1 \vee \cdots \vee x_k$ mit $k > 3$ Literalen $k - 3$ neue Variablen $z_1, \ldots, z_{k-3}$ ein und ersetzen die Klausel durch

$$(x_1 \vee x_2 \vee z_1) \wedge (\neg z_1 \vee x_3 \vee z_2) \wedge \cdots \wedge (\neg z_{k-3} \vee x_{k-1} \vee x_k)$$

Damit haben wir nur durch Anwenden von Äquivalenzumformungen F in eine logisch äquivalente Formel F' in 3-CNF umgeformt. Da sie logisch äquivalent sind, ist F erfüllbar g.d.w. F' erfüllbar ist, somit gilt SAT $\leq_\text{p}$ 3-CNF-SAT.

LÖSUNGSVORSCHLAG:
Im zweiten Schritt bei der Umformung in disjunktive Normalfom erzeugen wir potentiell exponentiell viele Klauseln, weshalb die Transformation nicht in Polynomialzeit berechenbar ist. Somit ist das kein korrekter Beweis

für SAT $\leq_p$ 3-CNF-SAT. Weiterhin ist eine Umformung in Polynomialzeit trotzdem möglich, siehe Vorlesung.

FSK11-5 ***SAT-Varianten in $\mathcal{P}$ und $\mathcal{NP}$*** (0 Punkte)

a) Sei UNSAT = $\{F \mid F$ ist eine widersprüchliche Formel$\}$. Nehmen Sie an, dass UNSAT in $\mathcal{NP}$ ist.

Wir betrachten folgende Reduktionsfunktion von SAT auf UNSAT: Teste alle möglichen Variablenbelegungen der Formel. Wenn eine erfüllende Variablenbelegung gefunden wurde, gib $x \wedge \neg x$ zurück, ansonsten x.

Ist diese Reduktionsfunktion geeignet, um zu zeigen, dass UNSAT $\mathcal{NP}$-vollständig ist? Begründen Sie Ihre Antwort.

LÖSUNGSVORSCHLAG:
Nein, denn diese Reduktionsfunktion hat keine polynomielle, sondern exponentielle Laufzeit, da sie exponentiell viele Schritte braucht um alle möglichen Variablenbelegungen zu testen.

b) Sei 3mal-3SAT = $\left\{ F \;\middle|\; \begin{array}{l} F \text{ ist eine 3-CNF und hat mindestens} \\ 3 \text{ verschiedene erfüllende Belegungen} \end{array} \right\}$.

Zeigen Sie, dass 3mal-3SAT $\mathcal{NP}$-vollständig ist, wobei Sie für den Nachweis der $\mathcal{NP}$-Schwere eine Reduktion von 3-CNF-SAT auf 3mal-3SAT durchführen.

Hinweis: Was könnten Sie in der benötigten Reduktionsfunktion f hinzufügen, um aus einer erfüllenden Belegung drei erfüllende Belegungen zu erzeugen?

LÖSUNGSVORSCHLAG:
Für die $\mathcal{NP}$-Vollständigkeit von 3mal-3SAT ist zu zeigen:

- 3mal-3SAT ist in $\mathcal{NP}$: Betrachte die folgende NTM M:
  - M berechnet aus F die Menge der in F vorkommenden aussagenlogischen Variablen.
  - M rät nichtdeterministisch ein Tupel von Belegungen (I_1, I_2, I_3) der Variablen von F. Das geht in nichtdeterministischer Polynomialzeit, indem man I_1 für jede Variable x in die Möglichkeiten $x = 0$ und $x = 1$ nichtdeterministisch verzweigt, für I_2 nochmal dasselbe durchführt, und für I_3 auch nochmal dasselbe.
  - Falls $I_1 = I_2$, $I_1 = I_3$ oder $I_2 = I_3$ verwirf diese nichtdeterministische Berechnung (der Test kann in Polynomialzeit durchgeführt werden).

– Sonst prüfe, ob $I_1(F) = 1$, $I_2(F) = 1$ und $I_3(F) = 1$. Wenn ja, dann akzeptiere, anderenfalls verwirf diese nichtdeterministische Berechnung (die Tests können dabei in Polynomialzeit durchgeführt werden).

M entscheidet 3mal-3SAT (denn es werden 3 verschiedene Lösungen gesucht) und M benötigt nur polynomiell viele Schritte (denn es handelt sich um eine Hintereinanderausführung von endlich vielen jeweils polynomiell langen Abfolgen von Schritten). Somit ist 3mal-3SAT in $\mathcal{NP}$.

- 3mal-3SAT ist $\mathcal{NP}$-schwer. Wir reduzieren 3-CNF-SAT $\leq_p$ 3mal-3SAT. Dazu definieren wir zunächst die Reduktionsfunktion f:

$$f(F) = F \wedge (x \vee y \vee z)$$

Dabei sind x, y und z neue Variablen. Die Funktion f ist total und in Polynomialzeit berechenbar.

Wir zeigen: F ist erfüllbar g.d.w. $f(F)$ dreimal erfüllbar ist:

– „$\Rightarrow$“: Wenn $I(F) = 1$, dann machen $I_1 = I \cup \{x \mapsto 1, y \mapsto 0, z \mapsto 0\}$, $I_2 = I \cup \{x \mapsto 0, y \mapsto 1, z \mapsto 0\}$ und $I_3 = I \cup \{x \mapsto 0, y \mapsto 0, z \mapsto 1\}$ alle F wahr.

– „$\Leftarrow$“: Wenn $I_i(F \wedge (x \vee y \vee z)) = 1$ für $i = 1, 2, 3$, dann sei $I(v) = I_1(v)$ für alle Variablen v aus F. Dann gilt $I(F) = 1$.

Damit folgt 3-CNF-SAT $\leq_p$ 3mal-3SAT.

c) Sei Pos-3-SAT = $\left\{ F \;\middle|\; \begin{array}{l} F \text{ ist eine erfüllbare 3-CNF, in der ausschließlich} \\ \text{positive Literale vorkommen} \end{array} \right\}$.

i) Liegt Pos-3-SAT in $\mathcal{P}$?

LÖSUNGSVORSCHLAG:
Eine Pos-3-SAT-Formel ist bereits erfüllbar, wenn jede Klausel mindestens eine Variable enthält: Wenn man alle Variablen auf 1 setzt, sind alle Klauseln erfüllt. Das ist in Polynomialzeit überprüfbar.

Ferner muss noch überprüft werden, ob das Wort eine gültige Kodierung einer Formel ist; auch das ist in Polynomialzeit überprüfbar.

Also liegt Pos-3-SAT in $\mathcal{P}$.

ii) Liegt Pos-3-SAT in $\mathcal{NP}$?

LÖSUNGSVORSCHLAG: $\mathcal{P} \subseteq \mathcal{NP}$, somit liegt Pos-3-SAT auch in $\mathcal{NP}$.

Begründen Sie jeweils Ihre Antworten.