Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

Übungsblatt 4

Vorlesung Theoretische Grundlagen der Informatik im WS 21/22

Ausgabe: 3. Dezember
Abgabe: 17. Dezember (digital im ILIAS)

Bitte bearbeiten Sie die Aufgaben **handschriftlich** und laden Sie eine eingescannte PDF-Version im Übungsmodul Ihrer ILIAS-Tutoriumsgruppe hoch! Beschriften Sie Ihren handschriftlichen Aufschrieb gut sichtbar mit Name und Matrikelnummer. Nicht handschriftliche oder unbeschriftete Abgaben werden nicht akzeptiert!

Aufgabe 1

(2 + 3 = 5 Punkte)

Gegeben seien eine Menge V von Studierenden und eine Menge $E \subseteq \binom{V}{2}$ von Bekanntschaften. Für $u \neq v \in V$ gilt also $\{u, v\} \in E$ genau dann, wenn u und v einander kennen. Weiterhin sei eine Zahl $k \in \mathbb{N}$ gegeben.

Wir betrachten die folgenden beiden Entscheidungsprobleme:

(a) Wir wollen das Risiko von Abschreiben in der Klausur minimieren. Deswegen ist bei dem Problem Hörsaalverteilung die Frage, ob man die Studierenden so auf k Hörsäle verteilen kann, dass es in keinem Hörsaal zwei Personen gibt, die sich kennen.

(b) Wir sagen, dass eine Gruppe $S \subseteq V$ von Studierenden eine Lerngruppe bildet, falls sich alle Studierenden in S paarweise kennen *(bitte nicht hinterfragen, wie sinnvoll diese Definition ist)*. Bei dem Problem Gruppenräume ist die Frage, ob man die Studierenden so auf k Gruppenräume aufteilen kann, dass jeder Gruppenraum eine Lerngruppe bildet.

Zeigen Sie, dass beide Probleme $\mathcal{NP}$-vollständig sind. Wählen Sie jeweils eines der folgenden drei $\mathcal{NP}$-vollständigen Probleme für die Reduktion aus.

Clique
Gegeben: Ungerichteter Graph $G = (V, E)$, Parameter $k \in \mathbb{N}$
Gibt es eine Teilmenge $U \subseteq V$ mit mindestens k Knoten, sodass für alle $u, v \in U$ gilt: $\{u, v\} \in E$?

Color
Gegeben: Ungerichteter Graph $G = (V, E)$, Parameter $k \in \mathbb{N}$
Gibt es eine Färbung $c\colon V \to \{1, \ldots, k\}$ von G mit k Farben, sodass für alle $\{u, v\} \in E$ gilt: $c(u) \neq c(v)$?

Dominating Set
Gegeben: Ungerichteter Graph $G = (V, E)$, Parameter $k \in \mathbb{N}$
Gibt es eine Teilmenge $U \subseteq V$ mit höchstens k Knoten, sodass für alle $v \in V$ gilt: $v \in U$ oder v ist zu einem Knoten in U benachbart?

Lösung:

(a) Wir zeigen zunächst, dass das Problem Hörsaalverteilung in $\mathcal{NP}$ liegt. Für eine gegebene Aufteilung der Studierenden auf k Hörsäle lässt sich in Zeit $\mathcal{O}(kn^2)$ überprüfen, ob es in keinem Hörsaal zwei Personen gibt, die sich kennen. Dazu werden für jeden Hörsaal alle möglichen Paare überprüft.

Wir reduzieren das Problem Hörsaalverteilung von Color. Sei $I = (G = (V, E), k)$ eine Instanz von Color. Dazu konstruieren wir die Instanz $I' = (V', E', k')$ mit $V' \coloneqq V$, $E' \coloneqq E$ und $k' \coloneqq k$ von Hörsaalverteilung. Diese Konstruktion ist offensichtlich polynomiell.

Wir zeigen: I ist eine Ja-Instanz $\Leftrightarrow$ I' ist eine Ja-Instanz.

- $\Rightarrow$ Sei $c\colon V \to \{1, \dots, k\}$ eine gültige Färbung von I mit k Farben. Dann ist $c'\colon V' \to \{1, \dots, k'\}$ mit $c'(v) \coloneqq c(v)$ auch eine gültige Aufteilung von Studierenden V':
  - Die Studierenden werden auf $k = k'$ Hörsäle aufgeteilt.
  - Für zwei Studierende $u, v \in V'$, die sich kennen (also $uv \in E'$), gilt: u und v sind nicht im gleichen Hörsaal, da $c'(u) = c(u) \neq c(v) = c'(v)$.
- $\Leftarrow$ Sei $c'\colon V' \to \{1, \dots, k'\}$ eine gültige Aufteilung von Studierende auf Hörsäle. Dann ist $c\colon V \to \{1, \dots, k\}$ mit $c(v) \coloneqq c'(v)$ eine gültige Färbung von I:
  - Es werden höchstens $k = k'$ Farben verwendet.
  - Für zwei benachbarte Knoten $u, v \in V$ gilt: u und v sind unterschiedlich gefärbt, da $c(u) = c'(u) \neq c'(v) = c(v)$.

Die Knoten interpretieren wir also als Studierende und eine Kante zwischen zwei Knoten bedeutet, dass sich die zugehörigen Studierenden kennen (und umgekehrt). Dann können wir eine Lösung einer Instanz des einen Problems direkt auf die zugehörige Instanz des anderen Problems übertragen.

Insgesamt gilt Hörsaalverteilung $\in \mathcal{NP}$ und Color $\propto$ Hörsaalverteilung, also ist Hörsaalverteilung $\mathcal{NP}$-vollständig.

(b) Das Problem Gruppenräume ist unter dem Namen Clique Cover bekannt. Die Frage ist: kann man die Knotenmenge (die Studierenden) V in k Mengen (auf k Gruppenräume) so aufteilen, dass jede von diesen Mengen eine Clique in G bildet (in jedem Gruppenraum bilden die Studierenden eine Lerngruppe)? Wir zeigen, dass das Problem Clique Cover $\mathcal{NP}$-vollständig ist.

Eine nicht-deterministische Turingmaschine kann Clique Cover wie folgt entscheiden. Das Orakel rät zuerst eine Aufteilung der Knoten in Mengen, z.B. wird jedem Knoten eine Zahl zugewiesen, die der Menge entspricht, in der dieser Knoten landet. Der deterministische Teil überprüft, ob für jeden Knoten diese Zahl aus $\{1, \dots, k\}$ kommt und ob für jede Menge zwischen je zwei Knoten aus dieser Menge eine Kante verläuft. Das ist in $\mathcal{O}(|V|^2)$ möglich. Also existiert eine nicht-deterministische Turingmaschine, die Clique Cover in Polynomialzeit entscheidet und Clique Cover ist somit in $\mathcal{NP}$.

Um zu zeigen, dass Clique Cover auch $\mathcal{NP}$-schwer ist, geben wir eine polynomiale Transformation von Color in Clique Cover an. Sei $I = (G = (V, E), k)$ eine Instanz von Color. Weiter sei

$$E' = \binom{V}{2} \setminus E = \{\{u, v\} \mid u \neq v \in V, \{u, v\} \notin E\}, \tag{1}$$

dann bilden wir I auf die Instanz

$$I' = (G' = (V, E'), k)$$

von Clique Cover ab. Also verläuft in G' eine Kante zwischen zwei Knoten $u \neq v \in V$ genau dann, wenn es in G keine Kante zwischen diesen Knoten gibt. Die Instanz I' kann in $\mathcal{O}(|V|^2)$ (also in Polynomialzeit) berechnet werden.

Sei I eine Ja-Instanz von Color. Dann existiert eine Färbung $\phi : V \to \{1, \ldots, k\}$ so, dass:

$$\forall u, v \in V : \phi(u) = \phi(v) \Rightarrow \{u, v\} \notin E. \tag{2}$$

Für $i \in \{1, \ldots, k\}$ sei

$$V_i = \{v \in V \mid \phi(v) = i\}$$

die Menge aller Knoten in Farbe i. Wegen (2) verläuft keine Kante in E zwischen zwei Knoten in V_i. Nach der Konstruktion von E' (siehe (1)) verläuft also zwischen je zwei unterschiedlichen Knoten in V_i eine Kante in E' und somit ist V_i eine Clique in G'. Da ϕ eine Färbung ist, ist $V_1, \ldots, V_k$ eine Partition der Knotenmenge V, also:

$$V_1 \cup V_2 \cup \cdots \cup V_k = V \text{ und } \forall i \neq j \in \{1, \ldots, k\} : V_i \cap V_j = \emptyset.$$

Also ist $V_1, \ldots, V_k$ eine Partition der Knotenmenge V von G' in k Cliquen und somit ist I' eine Ja-Instanz von Clique Cover.

Sei umgekehrt I' eine Ja-Instanz von Clique Cover. Also existiert eine Partition von V in Mengen $V_1, \ldots, V_k$ so, dass für jedes $i \in \{1, \ldots, k\}$ die Menge V_i eine Clique in G' bildet. Nach Definition von E' verläuft also keine Kante in E zwischen zwei Knoten in V_i. Deswegen definieren wir eine Färbung $\phi : V \to \{1, \ldots, k\}$ mit

$$\forall i \in \{1, \ldots, k\}\ \forall v \in V_i : \phi(v) = i.$$

Da $V_1, \ldots, V_k$ eine Partition von V ist, ist ϕ wohldefiniert. Oben haben wir gezeigt, dass für jedes $i \in \{1, \ldots, k\}$ keine Kante zwischen zwei Knoten in Farbe i verläuft. Also ist ϕ die gewünschte Färbung von V, in der keine Kante in G zwei Endknoten in der gleichen Farbe hat. Somit ist I eine Ja-Instanz von Color.

Also ist die angegebene Abbildung eine polynomiale Transformation von Color in Clique Cover und es gilt:

$$\text{Color} \propto \text{Clique Cover}.$$

Insgesamt ist Clique Cover in $\mathcal{NP}$ und es ist $\mathcal{NP}$-schwer. Also ist Clique Cover $\mathcal{NP}$-vollständig.

Aufgabe 2

(3 + 2 = 5 Punkte)

Für zwei Alphabete Σ_A, Σ_B und zwei Sprachen $A \subseteq \Sigma_A^*$, $B \subseteq \Sigma_B^*$ schreiben wir $A \equiv B$, falls $A \propto B$ und $B \propto A$.

(a) Zeigen Sie, dass $\equiv$ eine Äquivalenzrelation ist. Zeigen Sie hierfür, dass $\equiv$ reflexiv, symmetrisch und transitiv[1] ist.

(b) Zeigen Sie, dass es genau drei Äquivalenzklassen von $\equiv$ auf $\mathcal{P}$ gibt:

[1] Aussage aus der Vorlesung, die hier bewiesen werden muss

- $C_1 = \{\emptyset\}$
- $C_2 = \{\Sigma^* \mid \Sigma \text{ ist ein beliebiges Alphabet}\}$
- $C_3 = \mathcal{P} \setminus (C_1 \cup C_2)$

Lösung:

(a) Um zu zeigen, dass $\equiv$ eine Äquivalenzrelation ist, muss man Reflexivität, Symmetrie und Transitivität von $\equiv$ beweisen.

- **Reflexivität**: Zu zeigen: $A \equiv A$ für alle Sprachen A. Für eine Sprache $A \in \Sigma^*$ betrachte die Identitätsabbildung $f : \Sigma^* \to \Sigma^*$ mit $f(x) = x$ für alle $x \in \Sigma^*$. Da jedes Wort auf sich selbst abgebildet wird, gilt für jedes $x \in \Sigma^*$:

  $$x \in A \Leftrightarrow f(x) = x \in A$$

  Die Identitätsabbildung kann offensichtlich in konstanter Zeit berechnet werden. Also ist die Identitätsabbildung eine polynomiale Transformation der Sprache A in die Sprache A. Also gilt $A \propto A$ und somit $A \equiv A$.
- **Symmetrie**: Folgt direkt aus der Definition von $\equiv$.
- **Transitivität** Zu zeigen: für alle Sprachen A, B, C:

  $$A \equiv B \text{ und } B \equiv C \Rightarrow A \equiv C.$$

  Für Sprachen $A \subset \Sigma_1^*, B \subset \Sigma_2^*, C \subset \Sigma_3^*$ gelte:

  $$A \equiv B, B \equiv C.$$

  Somit gilt:

  $$A \propto B, B \propto A, B \propto C, C \propto B.$$

  Wir zeigen, dass $A \propto B$ und $B \propto C$ impliziert $A \propto C$. Wegen $A \propto B$ existieren eine polynomiale Transformation $f_1 : \Sigma_1^* \to \Sigma_2^*$ der Sprache A in die Sprache B und ein Polynom q_1 so, dass f_1 von einer Turingmaschine $\mathcal{M}_1$ berechenbar ist, deren Laufzeit durch q_1 beschränkt ist (wir dürfen annehmen, dass q_1 auf natürlichen Zahlen nicht-fallend ist). Somit gilt für alle $x \in \Sigma_1^*$:

  $$x \in A \Leftrightarrow f_1(x) \in B. \tag{3}$$

  Analog existieren wegen $B \propto C$ eine polynomiale Transformation $f_2 : \Sigma_2^* \to \Sigma_3^*$ der Sprache B in die Sprache C und ein Polynom q_2 so, dass f_2 von einer Turingmaschine $\mathcal{M}_2$ berechenbar ist, deren Laufzeit durch q_2 beschränkt ist (wir dürfen annehmen, dass q_2 auf natürlichen Zahlen nicht-fallend ist). Und für alle $x \in \Sigma_2^*$:

  $$x \in B \Leftrightarrow f_2(x) \in C \tag{4}$$

  Betrachte $f : \Sigma_1^* \to \Sigma_3^*$ mit $f = f_2 \circ f_1$ (also $f(x) = f_2(f_1(x))$ für alle $x \in \Sigma_1^*$). Zuerst gilt für alle $x \in \Sigma_1^*$:

  $$x \in A \overset{(3)}{\Leftrightarrow} y = f_1(x) \in B \overset{(4)}{\Leftrightarrow} f_2(y) = f_2(f_1(x)) = f(x) \in C$$

  Weiterhin kann f wie folgt berechnet werden. Für eine Eingabe x wird zuerst höchstens in Zeit $q_1(|x|)$ der Wert $y = f_1(x)$ von $\mathcal{M}_1$ berechnet werden. Beobachte, dass

  $$|y| \leq q_1(|x|) \tag{5}$$

gilt, denn in Zeit $q_1(|x|)$ kann $\mathcal{M}_1$ maximal $q_1(|x|)$ Symbole auf das Band schreiben. Danach kann höchstens in Zeit $q_2(|y|)$ der Wert $z = f_2(y)$ von $\mathcal{M}_2$ berechnet werden. Jetzt gilt

$$z = f_2(y) = f_2(f_1(x)) = f(x),$$

somit berechnet die beschriebene Turingmaschine genau die Funktion f. Die Laufzeit dieser Turingmaschine auf Eingabe x ist somit höchstens:

$$q_1(|x|) + q_2(|y|) \overset{(5)}{\leq} q_1(|x|) + q_2(q_1(|x|)) = (q_1 + q_2 \circ q_1)(|x|).$$

Da die Summe und die Verkettung von zwei Polynomen wieder ein Polynom ergibt, ist f in Polynomialzeit berechenbar. Somit ist f eine polynomiale Transformation der Sprache A in die Sprache C und

$$A \propto C.$$

Analog (aus Symmetriegründen) kann man zeigen, dass $C \propto B$ und $B \propto A$ implizieren, dass

$$C \propto A$$

gilt. Somit

$$A \equiv C.$$

(b) Zuerst gilt $\emptyset \in \mathcal{P}$, denn eine Turingmaschine, die die Eingabe ignoriert und direkt in einen ablehnenden Zustand übergeht, entscheidet die Sprache $\emptyset$ in $\mathcal{O}(1)$.

Analog gilt für jedes endliche Alphabet Σ, dass eine Turingmaschine, die die Eingabe ignoriert und direkt in einen akzeptierenden Zustand übergeht, die Sprache Σ^* in $\mathcal{O}(1)$ entscheidet. Also $\Sigma^* \in \mathcal{P}$.

Für jedes $L \in \mathcal{P} \setminus \{\emptyset\}$ gilt, dass mindestens ein Wort $w \in L$ existiert. Eine polynomiale Transformation von L auf $\emptyset$ muss w auf ein Wort $w' \in \emptyset$ abbilden, aber es existiert kein solches w'. Somit gilt

$$L \not\propto \emptyset$$

und dann

$$L \not\equiv \emptyset.$$

Also bildet $\emptyset$ eine eigene Äquivalenzklasse:

$$[\emptyset] = \{\emptyset\}. \tag{6}$$

Für zwei endliche Alphabete Σ_1, Σ_2 betrachte $f : \Sigma_1^* \to \Sigma_2^*$ mit z.B. $f(x) = \varepsilon$ für alle $x \in \Sigma_1^*$. Die Abbildung f kann in Linearzeit berechnet werden, indem die Eingabe gelöscht wird. Sei $L_1 = \Sigma_1^*$, $L_2 = \Sigma_2^*$. Dann gilt für jedes $w \in \Sigma_1^*$: $w \in L_1$ und $\varepsilon \in L_2$, also

$$w \in L_1 \Leftrightarrow f(w) = \varepsilon \in L_2.$$

Somit ist f eine polynomiale Transformation der Sprache $L_1 = \Sigma_1^*$ in die Sprache $L_2 = \Sigma_2^*$ und

$$\Sigma_1^* \propto \Sigma_2^*.$$

Aus Symmetriegründen gilt

$$\Sigma_2^* \propto \Sigma_1^*.$$

Also

$$\Sigma_1^* \equiv \Sigma_2^*$$

für alle endliche Alphabete Σ_1, Σ_2. Somit liegen alle Sprachen der Form Σ^* für ein endliches Alphabet Σ in der gleichen Äquivalenzklasse. Also gilt:

$$C_2 \subseteq [\Sigma_2^*]. \tag{7}$$

Weiter sei $L \in C_3 = \mathcal{P} \setminus (C_1 \cup C_2)$ mit $L \subset \Sigma_1^*$. Also gibt es ein $w \in \Sigma_1^* \setminus L$. Sei Σ_2 ein endliches Alphabet. Dann gibt es keine polynomiale Transformation der Sprache L in die Sprache Σ_2^*: sonst müsste w auf ein Wort $w' \notin \Sigma_2^*$ abgebildet werden und es gibt kein solches w'. Also gilt

$$L \not\propto \Sigma_2^*$$

und dann auch

$$L \not\equiv \Sigma_2^*.$$

Das heißt

$$L \notin [\Sigma_2^*].$$

Das impliziert zusammen mit (7) und (6):

$$C_2 = [\Sigma_2^*]. \tag{8}$$

Schließlich seien $L_1, L_2 \in C_3$ mit $L_1 \subset \Sigma_1^*, L_2 \subset \Sigma_2^*$. Fixiere ein $w_j \in L_2$ und ein $w_n \in \Sigma_2^* \setminus L_2$ (existieren wegen $L_2 \in C_3$). Betrachte $f : \Sigma_1^* \to \Sigma_2^*$ mit

$$f(w) = \begin{cases} w_j & \text{falls } w \in L_1 \\ w_n & \text{falls } w \notin L_1 \end{cases}$$

Somit gilt

$$w \in L_1 \Rightarrow f(w) = w_j \in L_2$$

und

$$w \notin L_1 \Rightarrow f(w) = w_n \notin L_2.$$

Also

$$w \in L_1 \Leftrightarrow f(w) \in L_2.$$

Da $L_1 \in \mathcal{P}$ gilt, existiert eine deterministische Turingmaschine $\mathcal{M}_1$, die L_1 entscheidet. Die Funktion f kann wie folgt in Polynomialzeit berechnet werden. Für eine Eingabe w wird zuerst $\mathcal{M}_1$ simuliert, um festzustellen, ob $w \in L_1$ gilt. Falls ja, wird w_j ausgegeben, sonst wird w_n ausgegeben. Da w_j, w_n fest sind, haben sie eine konstante Länge und die Berechung findet tätsächlich in Polynomialzeit statt. Also ist f eine polynomiale Transformation der Sprache L_1 in die Sprache L_2. Somit gilt

$$L_1 \propto L_2.$$

Aus Symmetriegründen gilt auch

$$L_2 \propto L_1.$$

Also

$$L_1 \equiv L_2.$$

Dann gilt

$$C_3 \subseteq [L_1].$$

Zusammen mit (6) und (8) impliziert das

$$[L_1] = C_3.$$

Somit ist die obige Zerlegung in Äquivalenzklassen bewiesen.

Aufgabe 3 (4 Punkte)

Das Entscheidungsproblem 2021-Oft-SAT ist wie folgt definiert. Gegeben sind eine Menge U von Variablen und eine Menge C von Klauseln über U. Die Frage ist: Gibt es mindestens 2021 paarweise unterschiedliche Wahrheitsbelegungen von U, die C erfüllen? Zeigen Sie, dass 2021-Oft-SAT $\mathcal{NP}$-vollständig ist.

Lösung:
Wir zeigen zuerst, dass 2021-Oft-SAT in $\mathcal{NP}$ liegt. Eine nicht-deterministische Turingmaschine kann 2021-Oft-SAT wie folgt entscheiden: Das Orakel rät zuerst 2021 Wahrheitsbelegungen. Der deterministische Teil überprüft, ob je zwei Belegungen sich unterscheiden (also mindestens einer Variable unterschiedliche Werte zuweisen). Danach überprüft er, ob jede von diesen Belegungen C erfüllt. Da 2021 eine Konstante ist, läuft das in $\mathcal{O}(|U| \cdot |C|)$. Also gibt es eine nicht-deterministische Turingmaschine, die 2021-Oft-SAT in Polynomialzeit entscheidet und somit ist 2021-Oft-SAT in $\mathcal{NP}$.

Um zu zeigen, dass 2021-Oft-SAT auch $\mathcal{NP}$-schwer ist, geben wir eine polynomiale Transformation von SAT in 2021-Oft-SAT an. Sei (U, C) eine Instanz von SAT. Wir setzen

$$U' = U \cup \{u_1, u_2, \ldots, u_{11}\}$$

für paarweise unterschiedliche neue Variablen $u_1, u_2, \ldots, u_{11}$. Also:

$$\forall i \neq j \in \{1, \ldots, 11\} : u_i \neq u_j \text{ und } u_i \notin U.$$

Und wir setzen

$$C' = C.$$

Dann bilden wir die Instanz (U, C) von SAT auf die Instanz (U', C') von 2021-Oft-SAT ab. Trivialerweise kann man die Instanz (U', C') in Linearzeit (also Polynomialzeit) berechnen.

Sei (U, C) eine Ja-Instanz von SAT. Dann existiert eine Wahrheitsbelegung $\phi : U \to \{\text{wahr, falsch}\}$, die C erfüllt. Also für alle $c \in C$ gilt:

$$\phi(c) = \text{wahr}. \tag{9}$$

Sei $\psi : \{u_1, \ldots, u_{11}\} \to \{\text{wahr, falsch}\}$ eine beliebige aber feste Wahrheitsbelegung der neuen Variablen. Betrachte die Variablenbelegung $\phi_\psi : U' \to \{\text{wahr, falsch}\}$ definiert als:

$$\phi_\psi(v) = \begin{cases} \phi(v) & \text{falls } v \in U \\ \psi(v) & \text{sonst} \end{cases}.$$

Sei jetzt $c' \in C' = C$ eine beliebige Klausel. Da c' nur Variablen aus U enthält und die Belegung der Variablen in U gleich in ϕ und ϕ_ψ sind, gilt

$$\phi_\psi(c') = \phi(c) \overset{(9)}{=} \text{wahr}.$$

Somit erfüllt ϕ_ψ jede Klausel in C'. Da ψ beliebig aber fest gewählt wurde, erfüllt für jedes $\psi : \{u_1, \ldots, u_{11}\} \to \{\text{wahr, falsch}\}$ die Wahrheitsbelegung ϕ_ψ alle Klauseln in C'. Beobachte, dass zwei unterschiedliche Wahrheitsbelegungen $\psi, \psi' : \{u_1, \ldots, u_{11}\} \to \{\text{wahr, falsch}\}$ sich an der Belegung mindestens einer Variable in $\{u_1, \ldots, u_{11}\}$ unterscheiden, und somit sind dann auch die Wahrheitsbelegungen ϕ_ψ und $\phi_{\psi'}$ unterschiedlich. Da es $2^{11} = 2048$ paarweise unterschiedliche Wahrheitsbelegungen von $\{u_1, \ldots, u_{11}\}$ gibt, gibt es somit auch mindestens 2048 paarweise unterschiedliche Wahrheitsbelegungen von U', die C' erfüllen. Inbesondere gibt es mindestens 2021 paarweise unterschiedliche Wahrheitsbelegungen von U', die C' erfüllen und somit ist (U', C') eine Ja-Instanz von 2021-Oft-SAT.

Sei umgekehrt (U', C') eine Ja-Instanz von 2021-Oft-SAT. Also existieren mindestens 2021 paarweise unterschiedliche Wahrheitsbelegungen von U', die alle Klauseln in C' erfüllen. Inbesondere gibt es mindestens eine Wahrheitsbelegung $\phi' : U' \to \{\text{wahr, falsch}\}$, die alle Klauseln in C' erfüllt. Definiere $\phi : U \to \{\text{wahr, falsch}\}$ als

$$\phi(v) = \phi'(v).$$

Da ϕ' alle Klauseln in $C' = C$ erfüllt und alle Variablen in $C' = C$ aus U kommen, erfüllt auch ϕ alle Klauseln in C und somit ist (U, C) eine Ja-Instanz von SAT.

Also ist die angegebene Abbildung tatsächlich eine polynomiale Transformation von SAT in 2021-Oft-SAT und es gilt:

$$\text{SAT} \propto \text{2021-Oft-SAT}.$$

Da SAT $\mathcal{NP}$-schwer ist, ist auch 2021-Oft-SAT $\mathcal{NP}$-schwer. Also ist 2021-Oft-SAT $\mathcal{NP}$-schwer und es liegt in $\mathcal{NP}$, deswegen ist 2021-Oft-SAT $\mathcal{NP}$-vollständig.

Aufgabe 4 (4 Punkte)

Das Problem Vollständiger Pfad ist wie folgt definiert. Gegeben ist ein Graph $G = (V, E)$. Die Frage ist: gibt es einen Pfad in G, der jeden Knoten genau einmal besucht. Also existieren $v_1, \ldots, v_{|V|} \in V$ so, dass für alle $i \neq j \in \{1, \ldots, |V|\}$ gilt $v_i \neq v_j$ und für alle $i \in \{1, \ldots, |V| - 1\}$ gilt $v_i v_{i+1} \in E$?

Das Problem Vollständiger Kreis ist wie folgt definiert. Gegeben ist ein Graph $G = (V, E)$. Die Frage ist: gibt es einen Kreis in G, der jeden Knoten genau einmal besucht. Also existieren $v_1, \ldots, v_{|V|} \in V$ so, dass für alle $i \neq j \in \{1, \ldots, |V|\}$ gilt $v_i \neq v_j$, für alle $i \in \{1, \ldots, |V| - 1\}$ gilt $v_i v_{i+1} \in E$ und es gilt $v_{|V|} v_1 \in E$?

Zeigen Sie, dass das Problem Vollständiger Kreis $\mathcal{NP}$-vollständig ist. Sie dürfen benutzen, dass das Problem Vollständiger Pfad $\mathcal{NP}$-vollständig ist.

Lösung:
Das Problem Vollständiger Pfad ist unter dem Namen Hamiltonian Path bekannt und das Problem Vollständiger Kreis ist unter dem Namen Hamiltonian Cycle bekannt.

Wir zeigen zuerst, dass Vollständiger Kreis in $\mathcal{NP}$ liegt. Eine nicht-deterministische Turingmaschine kann Vollständiger Kreis wie folgt entscheiden: Das Orakel rät eine Folge von Knoten. Der deterministische Teil überprüft zuerst in $\mathcal{O}(|V|^2)$, ob alle Knoten in dieser Folge genau einmal vorkommen (also jeder Knoten kommt vor und keine zwei Knoten sind gleich). Danach überprüft er, ob zwischen je zwei aufeinanderfolgenden Knoten in dieser Folge eine Kante verläuft und ob zwischen dem ersten und dem letzten Knoten auch eine Kante existiert. Das kann in $\mathcal{O}(|V| \cdot |E|)$ gemacht werden. Also gibt es eine nicht-deterministische Turingmaschine, die das Problem Vollständiger Kreis in Polynomialzeit entscheidet und somit ist Vollständiger Kreis in $\mathcal{NP}$.

Um zu zeigen, dass Vollständiger Kreis auch $\mathcal{NP}$-schwer ist, geben wir eine polynomiale Transformation von Vollständiger Pfad in Vollständiger Kreis. Sei $G = (V, E)$ eine Instanz von Vollständiger Pfad. Wir bilden G auf die folgende Instanz von Vollständiger Kreis ab:

$$G' = (V' = V \cup \{w\}, E' = E \cup \{vw \mid v \in V\}),$$

wobei w ein neuer Knoten ist, das heißt $w \notin V$. Klar, dass diese Instanz in $\mathcal{O}(|V| + |E|)$, also in Polynomialzeit, berechnet werden kann.

Sei G eine Ja-Instanz von Vollständiger Pfad. Dann gibt es eine Reihenfolge von Knoten $v_1, v_2, \ldots, v_{|V|} \in V$ so, dass jeder Knoten aus V da genau einmal vorkommt und die Knoten in dieser

Reihenfolge einen Pfad in G bilden, also:

$$\forall i \in \{1, \ldots, |V| - 1\} : v_i v_{i+1} \in E.$$

Da zwischen Knoten aus V die gleichen Kanten in E und E' verlaufen, gilt auch:

$$\forall i \in \{1, \ldots, |V| - 1\} : v_i v_{i+1} \in E'. \tag{10}$$

Betrachte die Reihenfolge $v_1, v_2, \ldots, v_{|V|}, w$ der Knoten in V'. Da $V' = V \cup \{w\}$, kommt in dieser Reihenfolge jeder Knoten aus V' genau einmal vor. Weiterhin gilt (10) und nach Kostruktion von E' gilt auch $v_{|V|}w, wv_1 \in E'$. Somit bilden $v_1, v_2, \ldots, v_{|V|}, w$ einen Kreis in G', in dem jeder Knoten aus V' genau einmal vorkommt. Also ist G' eine Ja-Instanz von Vollständiger Kreis.

Sei jetzt umgekehrt G' eine Ja-Instanz von Vollständiger Kreis. Also gibt es eine Reihenfolge von Knoten $v_1, v_2, \ldots, v_{|V|+1}$ so, dass jeder Knoten aus V' in dieser Reihenfolge genau einmal vorkommt und die Knoten in dieser Reihenfolge einen Kreis in G' bilden, also:

$$\forall i \in \{1, \ldots, |V|\} : v_i v_{i+1} \in E' \text{ und } v_{|V|+1} v_1 \in E'. \tag{11}$$

Da es sich um einen Kreis handelt, der alle Knoten aus V' enthält, dürfen wir ohne Beschränkung der Allgemeinheit annehmen, dass $v_1 = w$ gilt. Betrachte die Reihenfolge $v_2, \ldots, v_{|V|+1}$ der Knoten. Da wir nur w weggelassen haben, kommt jeder Knoten aus V in dieser Reihenfolge genau einmal vor. Weiterhin gilt (11). Da zwischen Knoten in V die gleichen Kanten in E und in E' verlaufen, gilt auch:

$$\forall i \in \{2, \ldots, |V|\} : v_i v_{i+1} \in E.$$

Also bilden die Knoten $v_2, \ldots, v_{|V|+1}$ einen Pfad in G, in dem jeder Knoten aus V genau einmal vorkommt. Somit ist G eine Ja-Instanz von Vollständiger Pfad.

Also ist die angegebene Abbildung tatsächlich eine polynomiale Transformation von Vollständiger Pfad in Vollständiger Kreis und es gilt:

$$\text{Vollständiger Pfad} \propto \text{Vollständiger Kreis}.$$

Da Vollständiger Pfad laut dem Hinweis $\mathcal{NP}$-schwer ist, ist auch Vollständiger Kreis $\mathcal{NP}$-schwer. Insgesamt ist Vollständiger Kreis in $\mathcal{NP}$ und es ist $\mathcal{NP}$-schwer. Somit ist Vollständiger Kreis $\mathcal{NP}$-vollständig.

Aufgabe 5 (4 Punkte)

Nachdem der Pizza-Lieferdienst letztes Jahr so erfolgreich war, möchte Dr. Meta nun sein Unternehmen weiter expandieren und eine Mensa mit insgesamt p Linien (die aber alle das gleiche Essen kochen) für die Menge der Studierenden S eröffnen. Da sein Budget noch relativ begrenzt ist, kann jede Linie höchstens $q \in \mathbb{N}_0$ Portionen kochen. Die Studierenden können unterschiedlich viele Portionen essen. Dabei gibt die Funktion $g\colon S \to \mathbb{N}_0$ für jede Person $s \in S$ an, wie viele Portionen s essen möchte. Kann Dr. Meta die Studierenden auf die p Linien verteilen, sodass jede Person s sich die gewünschte Anzahl $g(s)$ an Portionen kaufen kann, aber keine Linie mehr als q Portionen kochen muss?

Zeigen Sie, dass Dr. Metas Problem $\mathcal{NP}$-vollständig ist.

Hinweis: Verwenden Sie, dass das Problem Partition aus der Vorlesung $\mathcal{NP}$-vollständig ist.

Lösung:
Das Problem ist unter dem Namen Bin Packing bekannt. Gegeben ist eine Menge von Elementen mit nicht-negativen Gewichten, die Frage ist: kann man die Elemente auf p Behälter so aufteilen, dass in jedem Behälter das Gesamgewicht der Elemente maximal q beträgt.

Eine nicht-deterministische Turingmaschine kann Bin Packing wie folgt entscheiden. Der Orakel rät zuerst eine Aufteilung der Menge S in p Mengen, z.B. wird jedem Element aus S eine Zahl zugewiesen. Der deterministische Teil überprüft dann, ob jedem Element eine Zahl aus $\{1, \dots, p\}$ zugewiesen wird. Weiterhin bildet er für jede von p Mengen die Summe der Gewichte der Elemente in dieser Menge und überprüft, ob diese Summe maximal q beträgt. Das ist in Linearzeit möglich. Also existiert eine nicht-deterministische Turingmaschine, die Bin Packing in Polynomialzeit entscheidet und somit ist Bin Packing in $\mathcal{NP}$.

Um zu zeigen, dass Bin Packing auch $\mathcal{NP}$-schwer ist, geben wir eine polynomiale Transformation von Partition in Bin Packing an. Sei $I = (M, w : M \to \mathbb{N}_0)$ eine Instanz von Partition. Und sei

$$W = \sum_{m \in M} w(m)$$

das Gesamtgewicht der Elemente in M. Wir bilden I auf die Instanz

$$I' = (S = M, g = w, p = 2, q = \lfloor \frac{W}{2} \rfloor)$$

von Bin Packing ab. Die Instanz I' kann in Linearzeit konstruiert werden.

Sei I eine Ja-Instanz von Partition. Dann existiert eine Teilmenge $M' \subseteq M$ von M so, dass

$$\sum_{m \in M'} w(m) = \sum_{m \in M \setminus M'} w(m)$$

gilt. Da jedes Element von M entweder in M' oder in $M \setminus M'$ liegt, gilt somit

$$\sum_{m \in M'} w(m) = \sum_{m \in M \setminus M'} w(m) = \frac{W}{2}.$$

Da jedes Element $m \in M'$ ein ganzzahliges Gewicht hat, ist somit auch $\frac{W}{2}$ ganzzahlig, also gilt

$$\frac{W}{2} = \lfloor \frac{W}{2} \rfloor.$$

Wir zeigen nun, dass I' ebenfalls eine Ja-Instanz ist. Betrachte die Funktion $f : M \to \{1, 2\}$ definiert als

$$f(m) = \begin{cases} 1 & \text{falls } m \in M' \\ 2 & \text{sonst} \end{cases}.$$

Es gilt:

$$\sum_{\substack{m \in M: \\ f(m)=1}} w(m) = \sum_{\substack{m \in M: \\ m \in M'}} w(m) = \frac{W}{2} = \lfloor \frac{W}{2} \rfloor \leq \lfloor \frac{W}{2} \rfloor = q$$

und

$$\sum_{\substack{m \in M: \\ f(m)=2}} w(m) = \sum_{\substack{m \in M: \\ m \notin M'}} w(m) = \frac{W}{2} = \lfloor \frac{W}{2} \rfloor \leq \lfloor \frac{W}{2} \rfloor = q.$$

Also ist f die gewünschte Aufteilung von M und somit ist I' eine Ja-Instanz von Bin Packing.

Sei umgekehrt I' eine Ja-Instanz von Bin Packing. Dann existiert eine Abbildung $f : M \to \{1, 2\}$ so, dass für jedes $i \in \{1, 2\}$ gilt:

$$\sum_{\substack{m \in M: \\ f(m)=i}} w(m) \leq q = \lfloor \frac{W}{2} \rfloor.$$

Angenommen für ein $i_0 \in \{1, 2\}$ gilt:

$$\sum_{\substack{m \in M: \\ f(m)=i}} w(m) < \lfloor \frac{W}{2} \rfloor.$$

Dann gilt

$$\sum_{m \in M} w(m) = \sum_{\substack{m \in M: \\ f(m)=i_0}} w(m) + \sum_{\substack{m \in M: \\ f(m) \neq i_0}} w(m) < \lfloor \frac{W}{2} \rfloor + \sum_{\substack{m \in M: \\ f(m) \neq i_0}} w(m) \leq \lfloor \frac{W}{2} \rfloor + \lfloor \frac{W}{2} \rfloor \leq W$$

Also

$$\sum_{m \in M} w(m) < W,$$

das ist ein Widerspruch zur Definition von W. Also gilt

$$\sum_{\substack{m \in M: \\ f(m)=1}} w(m) = \sum_{\substack{m \in M: \\ f(m)=2}} w(m) = \frac{W}{2} \tag{12}$$

Sei $M' \subseteq M$ definiert als

$$M' = \{m \in M \mid f(m) = 1\}.$$

Dann gilt

$$M \setminus M' = \{m \in M \mid f(m) = 2\}$$

und somit

$$\sum_{m \in M'} w(m) = \sum_{\substack{m \in M: \\ f(m)=1}} w(m) \overset{(12)}{=} \sum_{\substack{m \in M: \\ f(m)=2}} w(m) = \sum_{m \in M \setminus M'} w(m)$$

Also

$$\sum_{m \in M'} w(m) = \sum_{m \in M \setminus M'} w(m)$$

und M' ist die gewünschte Teilmenge. Somit ist I eine Ja-Instanz von Partition.

Die angegebene Abbildung ist somit eine polynomiale Transformation von Partition in Bin Packing und

$$\text{Partition} \propto \text{Bin Packing}.$$

Da Partition $\mathcal{NP}$-schwer ist, ist auch Bin Packing $\mathcal{NP}$-schwer. Insgesamt ist Bin Packing in $\mathcal{NP}$ und es ist $\mathcal{NP}$-schwer, also ist Bin Packing $\mathcal{NP}$-vollständig.

Aufgabe 6 (4 Punkte)

Sei Π ein $\mathcal{NP}$-vollständiges Problem. Zeigen Sie: Falls Π in co-$\mathcal{NP}$ liegt, dann gilt $\mathcal{NP} = \text{co-}\mathcal{NP}$.

Lösung:
Angenommen, Π liegt in co-$\mathcal{NP}$. Dann liegt Π^c in $\mathcal{NP}$, also existiert eine NTM $\mathcal{M}$, die Π^c in Polynomialzeit entscheidet.

Sei Π' ein beliebiges Problem in $\mathcal{NP}$ (also ist Π'^c ein beliebiges Problem in co-$\mathcal{NP}$). Wir zeigen, dass $\Pi' \in \text{co-}\mathcal{NP}$ ist (also auch, dass $\Pi'^c \in \mathcal{NP}$ gilt). Da Π $\mathcal{NP}$-vollständig ist und Π' in $\mathcal{NP}$ liegt, existiert eine polynomiale Transformation ϕ von Π' nach Π. Insbesondere kann ϕ in Polynomialzeit berechnet werden und für jede Instanz I von Π' gilt:

$$I \text{ ist eine Ja-Instanz von } \Pi' \Leftrightarrow \phi(I) \text{ ist eine Ja-Instanz von } \Pi. \tag{13}$$

Dann ist ϕ in Polynomialzeit berechenbar und es gilt für jede Instanz I von Π'^c:

$$\begin{aligned} & I \text{ ist eine Ja-Instanz von } \Pi'^c \\ \Leftrightarrow & I \text{ ist eine Nein-Instanz von } \Pi' \\ \overset{(13)}{\Leftrightarrow} & \phi(I) \text{ ist eine Nein-Instanz von } \Pi \\ \Leftrightarrow & \phi(I) \text{ ist eine Ja-Instanz von } \Pi^c. \end{aligned}$$

Also ist ϕ auch eine polynomiale Transformation von Π'^c nach Π^c. Jetzt kann Π'^c wie folgt von einer NTM entschieden werden. Sei I eine Instanz von Π'^c. Zuerst wird deterministisch $\phi(I)$ berechnet, danach wird $\mathcal{M}$ auf $\phi(I)$ simuliert und die Antwort von $\mathcal{M}$ wird übernommen. Da ϕ in Polynomialzeit berechnet werden kann und $\mathcal{M}$ in Polynomialzeit läuft, läuft auch diese neue NTM in Polynomialzeit. Weiterhin akzeptiert $\mathcal{M}$ genau die Ja-Instanzen von Π^c und da ϕ eine polynomiale Transformation von Π'^c nach Π^c ist, akzeptiert die angegebene NTM genau die Ja-Instanzen von Π'^c. Also entscheidet sie Π'^c in Polynomialzeit. Somit gilt $\Pi'^c \in \mathcal{NP}$ und somit $\Pi' \in \text{co-}\mathcal{NP}$. Da $\Pi'^c \in \text{co-}\mathcal{NP}$ beliebig war, gilt

$$\text{co-}\mathcal{NP} \subseteq \mathcal{NP}.$$

Weiterhin da $\Pi' \in \mathcal{NP}$ beliebig war, gilt auch

$$\mathcal{NP} \subseteq \text{co-}\mathcal{NP}.$$

Insgesamt gilt also

$$\mathcal{NP} = \text{co-}\mathcal{NP}.$$

Achtung: Wir haben $\mathcal{NP} = \text{co-}\mathcal{NP}$ **nur** unter der Annahme bewiesen, dass es ein $\mathcal{NP}$-vollständiges Problem in co-$\mathcal{NP}$ existiert. Es ist eine wichtige offene Frage, ob die Gleichheit $\mathcal{NP} = \text{co-}\mathcal{NP}$ gilt.

Aufgabe 7 (5 Punkte)

Das Entscheidungsproblem Seminar-Bewerbung ist wie folgt definiert. Gegeben seien eine Menge A von Studierenden, die sich für ein Seminar beworben haben, wobei $|A|$ gerade ist, und eine Menge $B \subseteq \binom{A}{2}$ von Paaren von Studierenden. Für $x \neq y \in A$ bedeutet $\{x, y\} \in B$, dass x und y in der Lage sind, zusammen produktiv zu arbeiten. Eine Betreuerin fragt sich bei diesem Problem, ob es möglich ist, die Hälfte der Bewerber:innen auszuwählen, sodass diese zusammen im Seminar produktiv arbeiten können. Formal ist die Frage wie folgt definiert: Gibt es eine Teilmenge $C \subseteq A$ der Größe $|A|/2$, sodass für alle $p \neq q \in C$ gilt, dass $\{p, q\} \in B$? Zeigen Sie, dass Seminar-Bewerbung $\mathcal{NP}$-vollständig ist.

Hinweis: Verwenden Sie das $\mathcal{NP}$-vollständige Problem Clique aus der Vorlesung für die Reduktion.

Lösung:
Dieses Problem ist unter dem Namen Half-Clique bekannt: Für einen Graphen $G = (A, B)$ (mit $|A|$ gerade) ist zu entscheiden, ob er eine Clique enthält, die aus genau der Hälfte der Knoten besteht.

Half-Clique $\in \mathcal{NP}$: Das Orakel rät eine Teilmenge T von A. Danach überprüft der deterministische Teil, ob $|T| = |A|/2$ und ob je zwei unterschiedliche Knoten in T durch eine Kante verbunden sind. Das ist in Zeit $\mathcal{O}(|A|^2|B|)$ möglich. Also gilt Half-Clique $\in \mathcal{NP}$.

Wir zeigen, dass Half-Clique $\mathcal{NP}$-schwer ist, indem wir eine polynomiale Transformation von Clique in Half-Clique angeben.

Sei $I = (G = (V, E), k)$ eine Instanz von Clique. Sei $n = |V|$. Seien $v_1, \ldots, v_n$ neue Knoten, also

$$\forall i \in \{1, \ldots, n\} : v_i \notin V$$

und

$$\forall i \neq j \in \{1, \ldots, n\} : v_i \neq v_j.$$

Wir setzen

$$A = V \cup \{v_1, \ldots, v_n\}.$$

Beobachte, dass

$$|A| = |V| + |\{v_1, \ldots, v_n\}| = n + n = 2n. \tag{14}$$

Jetzt ist die Kantenmenge zu definieren. Da wir eine Clique in G finden wollen, behalten wir die Kanten aus E, führen aber einige weitere Kanten hinzu. Dafür machen wir die Knoten $v_1, \ldots, v_{n-k}$ zu einer Clique und verbinden jeden von diesen Knoten mit jedem Knoten aus V. Die Knoten $v_{n-k+1}, \ldots, v_n$ lassen wir isoliert. Formal setzen wir:

$$B = E \cup \{\{v_i v_j\} \mid i \neq j \in \{1, \ldots, n-k\}\} \cup \{\{v_i v\} \mid i \in \{1, \ldots, n-k\}, v \in V\}.$$

Wir bilden I auf die Instanz $I' = (A, B)$ von Half-Clique ab. Das Hinzufügen der Knoten benötigt $\mathcal{O}(n)$ Zeit, das Hinzufügen der Kanten $\mathcal{O}(n^2)$, also kann diese Instanz in Polynomialzeit konstruiert werden.

Sei I eine Ja-Instanz von Clique. Also gibt es eine Teilmenge $U \subseteq V$ so, dass $|U| = k$ und U eine Clique in G bildet. Betrachte die Knotenmenge

$$U' = U \cup \{v_1, \ldots, v_{n-k}\}.$$

Wir behaupten, dass U' eine Clique in (A, B) bildet. Da $E \subseteq B$, bildet U eine Clique in (A, B). Weiterhin sind je zwei Knoten in $\{v_1, \ldots, v_{n-k}\}$ nach Konstruktion von B durch eine Kante verbunden sind. Schließlich ist jedes $v \in U \subseteq V$ mit jedem v_i ($i \in \{1, \ldots, n-k\}$) durch eine Kante in B verbunden. Somit ist U' tatsächlich eine Clique in (A, B). Und es gilt

$$|U'| = |U| + |\{v_1, \ldots, v_{(n-k)}\}| = k + (n-k) = n \overset{(14)}{=} |A|/2.$$

Also ist I' eine Ja-Instanz von Half-Clique.

Sei I' eine Ja-Instanz von Half-Clique. Also gibt es eine Clique $U' \subseteq A$ in (A, B) mit

$$|U'| = |A|/2 \overset{(14)}{=} n$$

Gilt $n = 1$ so gilt $k \in \{0, 1\}$ und I ist trivialerweise eine Ja-Instanz. Also sei ab jetzt $n > 1$. Sei $U = U' \setminus \{v_1, \ldots, v_n\}$. Dann ist $U \subseteq U'$ auch eine Clique in (A, B). Es gilt $U \subseteq V$. Weiterhin gilt für jedes $i \in \{n - k + 1, \ldots, n\}$: $v_i \notin U'$, da v_i keine inzidenten Kanten in B hat und kann in keiner Clique der Größe mindestens 2 enthalten sein. Also

$$U = U' \setminus \{v_1, \ldots, v_n\} = U' \setminus \{v_1, \ldots, v_{n-k}\}$$

und

$$|U| = |U' \setminus \{v_1, \ldots, v_{n-k}\}| \geq |U'| - \{v_1, \ldots, v_{n-k}\} = n - (n - k) = k.$$

Die Menge U bildet eine Clique in (A, B) und es gilt

$$B \cap \binom{V}{2} = E,$$

also verlaufen die gleichen Kanten in E und in B zwischen Knoten in V. Also bildet U eine Clique der Größe mindestens k auch in $G = (V, E)$. Somit ist $I = (G, k)$ eine Ja-Instanz von Clique.

Die angegebene Abbildung ist somit eine polynomiale Transformation von Clique in Half-Clique und

$$\text{Clique} \propto \text{Half-Clique}.$$

Da Clique $\mathcal{NP}$-schwer ist, ist auch Half-Clique $\mathcal{NP}$-schwer. Somit gilt Half-Clique $\in \mathcal{NP}$ und Half-Clique ist $\mathcal{NP}$-schwer, also ist Half-Clique $\mathcal{NP}$-vollständig.