FSK
Zentralübung 8

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik

Stand: 19. Juli 2023
Folien ursprünglich von PD Dr. David Sabel

Komplexitätsklassen $\mathcal{P}$ und $\mathcal{NP}$

Sprache L ist

- in $TIME(f(n))$, wenn es DTM gibt, die L entscheidet und die für jede Eingabe der Länge n nicht mehr als $\leq f(n)$ Schritte benötigt.
- in $NTIME(f(n))$, wenn es NTM gibt, die L entscheidet und die für jede Eingabe der Länge n auf jedem Berechnungspfad nicht mehr als $f(n)$ Schritte benötigt.

Die Klassen $\mathcal{P}$ und $\mathcal{NP}$ sind definiert als

$$\mathcal{P} = \bigcup_{p \text{ Polynom}} TIME(p(n))$$

$$\mathcal{NP} = \bigcup_{p \text{ Polynom}} NTIME(p(n))$$

$\mathcal{NP}$-Vollständigkeit: Wiederholung

Eine Sprache L heißt $\mathcal{NP}$-vollständig, wenn gilt

1. $L \in \mathcal{NP}$ und
2. L ist $\mathcal{NP}$-schwer (manchmal auch $\mathcal{NP}$-hart genannt):
   Für alle $L' \in \mathcal{NP}$ gilt $L' \leq_p L$

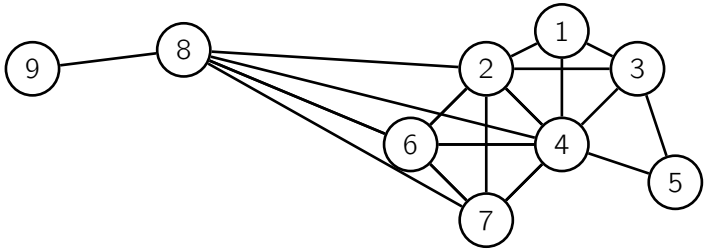
Für den $\mathcal{NP}$-Schwere-Beweis:

- Zeige $L_0 \leq_p L$ für ein bekanntes $\mathcal{NP}$-schweres Problem L_0.
- Dann ist L auch $\mathcal{NP}$-schwer.

Clique

Quiz: Der Graph hat eine

A 1-Clique
B 2-Clique
C 3-Clique
D 4-Clique
E 5-Clique
F 6-Clique
G 7-Clique



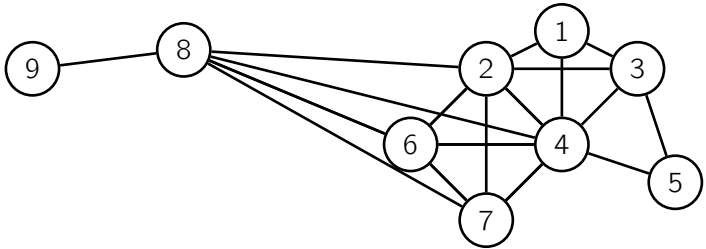
Zur Erinnerung:

Eine k-Clique in einem Graph $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$ mit $|V'| = k$ und $\forall u, v \in V', u \neq v \implies \{u, v\} \in E$

Clique

Quiz: Der Graph hat eine

A	1-Clique	ja
B	2-Clique	ja
C	3-Clique	ja
D	4-Clique	ja
E	5-Clique	ja
F	6-Clique	nein
G	7-Clique	nein



Zur Erinnerung:

Eine k-Clique in einem Graph $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$ mit $|V'| = k$ und $\forall u, v \in V', u \neq v \implies \{u, v\} \in E$

CLIQUE-Problem

CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahl k
Gefragt: Hat G eine k-Clique?

Resultat aus der Vorlesung: CLIQUE ist $\mathcal{NP}$-vollständig.

CLIQUE: Varianten

ONE-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine $|V|$-Clique?

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

TWO-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine 2-Clique?

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Quiz: Unter der Annahme $\mathcal{P} \neq \mathcal{NP}$: Welche der Sprachen sind NP-vollständig?

A ONE-CLIQUE
B DOUBLE-CLIQUE
C TWO-CLIQUE
D TRIPLE-CLIQUE

Zusatzfrage: Warum steht die Annahme in der Frage?

CLIQUE: Varianten

ONE-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine $|V|$-Clique?

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

TWO-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine 2-Clique?

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Quiz: Unter der Annahme $\mathcal{P} \neq \mathcal{NP}$: Welche der Sprachen sind NP-vollständig?

A ONE-CLIQUE — nein, ist in $\mathcal{P}$
B DOUBLE-CLIQUE — ja
C TWO-CLIQUE — nein, ist in $\mathcal{P}$
D TRIPLE-CLIQUE — ja

Zusatzfrage: Warum steht die Annahme in der Frage?

ONE-CLIQUE

ONE-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine $|V|$-Clique?

ONE-CLIQUE

ONE-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine $|V|$-Clique?

Ist in $\mathcal{P}$ (prüfe in deterministischer Polynomialzeit, ob der Graph vollständig ist) und damit unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ nicht $\mathcal{NP}$-vollständig.

TWO-CLIQUE

TWO-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine 2-Clique?

TWO-CLIQUE

TWO-CLIQUE

Gegeben: Graph $G = (V, E)$
Gefragt: Hat G eine 2-Clique?

Ist in $\mathcal{P}$ (prüfe in deterministischer Polynomialzeit, ob $E \neq \emptyset$) und damit unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ nicht $\mathcal{NP}$-vollständig.

TRIPLE-CLIQUE

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Ist eigentlich das gleiche wie CLIQUE, da man $\max\{k_1, k_2, k_3\}$ testen muss, aber trotzdem ein $\mathcal{NP}$-Vollständigkeitsbeweis:

- TRIPLE-CLIQUE $\in \mathcal{NP}$:

TRIPLE-CLIQUE

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Ist eigentlich das gleiche wie CLIQUE, da man $\max\{k_1, k_2, k_3\}$ testen muss, aber trotzdem ein $\mathcal{NP}$-Vollständigkeitsbeweis:

- TRIPLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch Knotenmengen V_1, V_2, V_3 mit $|V_i| = k_i$ und $V_i \subseteq V$ und verifiziert in determistischer Polynomialzeit, ob alle V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).

TRIPLE-CLIQUE

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Ist eigentlich das gleiche wie CLIQUE, da man $\max\{k_1, k_2, k_3\}$ testen muss, aber trotzdem ein $\mathcal{NP}$-Vollständigkeitsbeweis:

- TRIPLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch Knotenmengen V_1, V_2, V_3 mit $|V_i| = k_i$ und $V_i \subseteq V$ und verifiziert in determistischer Polynomialzeit, ob alle V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).
- TRIPLE-CLIQUE ist $\mathcal{NP}$-schwer: Zeige CLIQUE $\leq_p$ TRIPLE-CLIQUE:

TRIPLE-CLIQUE

TRIPLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k_1, k_2, k_3
Gefragt: Hat G je eine k_1-, k_2- und k_3-Clique?

Ist eigentlich das gleiche wie CLIQUE, da man $\max\{k_1, k_2, k_3\}$ testen muss, aber trotzdem ein $\mathcal{NP}$-Vollständigkeitsbeweis:

- TRIPLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch Knotenmengen V_1, V_2, V_3 mit $|V_i| = k_i$ und $V_i \subseteq V$ und verifiziert in determistischer Polynomialzeit, ob alle V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).
- TRIPLE-CLIQUE ist $\mathcal{NP}$-schwer: Zeige CLIQUE $\leq_p$ TRIPLE-CLIQUE:
  Sei $f((V, E), k)) = (V, E, k, k, k)$.
  - f ist total und in deterministischer Polynomialzeit berechenbar.
  - f ist korrekt: $((V, E), k) \in$ CLIQUE g.d.w. (V, E) hat k-Clique
    g.d.w. (V, E) hat k-, k- und k-Clique
    g.d.w. $((V, E), k, k, k) \in$ TRIPLE-CLIQUE

DOUBLE-CLIQUE

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

$\mathcal{NP}$-Vollständigkeitsbeweis:

- DOUBLE-CLIQUE $\in \mathcal{NP}$:

DOUBLE-CLIQUE

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

$\mathcal{NP}$-Vollständigkeitsbeweis:

- DOUBLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch disjunkte Knotenmengen V_1, V_2 mit $|V_i| = k$ und $V_i \subseteq V$ und verifiziert in deterministischer Polynomialzeit, ob beide V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).

DOUBLE-CLIQUE

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

$\mathcal{NP}$-Vollständigkeitsbeweis:

- DOUBLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch disjunkte Knotenmengen V_1, V_2 mit $|V_i| = k$ und $V_i \subseteq V$ und verifiziert in deterministischer Polynomialzeit, ob beide V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).
- DOUBLE-CLIQUE ist $\mathcal{NP}$-schwer: Zeige CLIQUE $\leq_p$ DOUBLE-CLIQUE:

DOUBLE-CLIQUE

DOUBLE-CLIQUE

Gegeben: Graph $G = (V, E)$ und Zahlen k
Gefragt: Hat G zwei Knotendisjunkte k-Cliquen?

$\mathcal{NP}$-Vollständigkeitsbeweis:

- DOUBLE-CLIQUE $\in \mathcal{NP}$: NTM M rät nichtdeterministisch disjunkte Knotenmengen V_1, V_2 mit $|V_i| = k$ und $V_i \subseteq V$ und verifiziert in deterministischer Polynomialzeit, ob beide V_i Cliquen sind (prüfe $u, v \in V_i \implies \{u, v\} \in E$).
- DOUBLE-CLIQUE ist $\mathcal{NP}$-schwer: Zeige CLIQUE $\leq_p$ DOUBLE-CLIQUE:
  Sei $f((V, E), k)) = (V \cup V', E \cup E', k)$ (und bei syntaktisch falscher CLIQUE-Eingabe erzeuge f eine syntaktisch falsche DOUBLE-CLIQUE-Eingabe) wobei $V' = \{v' \mid v \in V\}$ und $E' = \{\{u', v'\} \mid \{u, v\} \in E\}$.
  - f ist total und in deterministischer Polynomialzeit berechenbar.
  - f ist korrekt: $((V, E), k) \in$ CLIQUE g.d.w. (V, E) hat k-Clique
    g.d.w. (V, E) hat k-, und (V', E') hat k-Clique
    g.d.w. $(V \cup V', E \cup E', k) \in$ DOUBLE-CLIQUE

GRAPH-COLORING

GRAPH-COLORING

Gegeben: Graph $G = (V, E)$ und Zahl k
Gefragt: Hat G eine k-Färbung: Eine Zuordnung $f : V \to \{1, \ldots, k\}$ mit $f(v) \neq f(u)$ für alle $\{u, v\} \in E$?

Letzte Vorlesung: GRAPH-COLORING ist $\mathcal{NP}$-vollständig.

ASSIGN-EXAM (1)

ASSIGN-EXAM	
Gegeben:	Eine Menge von (Studierende, Klausur)-Paaren, die erfassen, welche Studierenden welche Klausur schreiben und eine Menge von Terminen
Gefragt:	Gibt es eine Zuordnung aller Klausuren auf Termine, sodass kein Student zur gleichen Zeit mehrere Klausuren schreiben muss?

ASSIGN-EXAM ist in $\mathcal{NP}$: Rate nichtdeterm. Zuordnung Klausur zu Termin und prüfe anschließend pro Studierende, ob ein Konflikt vorliegt.

ASSIGN-EXAM (2)

ASSIGN-EXAM

Gegeben: Eine Menge von (Studierende, Klausur)-Paaren, die erfassen, welche Studierenden welche Klausur schreiben und eine Menge von Terminen

Gefragt: Gibt es eine Zuordnung aller Klausuren auf Termine, sodass kein Student zur gleichen Zeit mehrere Klausuren schreiben muss?

ASSIGN-EXAM ist $\mathcal{NP}$-schwer: GRAPH-COLORING $\leq_p$ ASSIGN-EXAM

Welche der beiden folgenden Anweisungen ergibt eine korrekte Reduktionsfunktion f?

A Pro Klausur erzeuge Knoten, pro Studierende erzeuge Knoten.
Pro (Studierende, Klausur)-Paar erzeuge eine Kante im Graph.
Pro Termin erzeuge eine Farbe (Anzahl Farben = Anzahl Termine).

B Pro Farbe, erzeuge einen Termin (Anzahl Termine = Anzahl Farben).
Pro Knoten v, erzeuge eine Klausur v.
Pro Kante $e = \{u, v\}$ erzeuge Studierende e, die/der Klausur u und Klausur v schreibt, d.h. füge Paare (e, u) und (e, v) hinzu.

Das Rucksack-Problem

Definition (KNAPSACK-Problem)

Das KNAPSACK-Problem lässt sich in der gegeben/gefragt-Notation wie folgt formulieren:

gegeben: k Gegenstände mit Gewichten $w_1, \ldots, w_k \in \mathbb{N}$ und
Nutzenwerten $n_1, \ldots, n_k \in \mathbb{N}$,
sowie zwei Zahlen $s_w, s_n \in \mathbb{N}$

gefragt: Gibt es Teilmenge $I \subseteq \{1, \ldots, k\}$, sodass $\sum_{i \in I} w_i \leq s_w$ und $\sum_{i \in I} n_i \geq s_n$?

In der Vorlesung: KNAPSACK ist $\mathcal{NP}$-Vollständigkeit.

TASKS

TASKS

Gegeben: Zielwert W und n Aufgaben mit
- t_i ist Bearbeitungszeit (in Std) von Aufgabe i
- d_i ist Deadline (in Std) von Aufgabe i
- val_i ist Wert von Aufgabe i

Es gibt eine Maschine, die immer eine Aufgabe vollständig (ohne Unterbrechung) bearbeiten kann, und dann die nächste usw. Nur wenn eine Aufgabe vor der Deadline fertigstellt wird, wird der Wert val_i erzielt, ansonsten wird für diese Aufgabe kein Wert erzielt.

Gefragt: Gibt es eine Abarbeitungsreihenfolge der Aufgaben, sodass der Wert W mindestens erreicht wird?

Definition (KNAPSACK-Problem)

Das KNAPSACK-Problem lässt sich in der gegeben/gefragt-Notation wie folgt formulieren:

gegeben: k Gegenstände mit Gewichten $w_1, \ldots, w_k \in \mathbb{N}$ und Nutzenwerten $n_1, \ldots, n_k \in \mathbb{N}$, sowie zwei Zahlen $s_w, s_n \in \mathbb{N}$.

gefragt: Gibt es Teilmenge $I \subseteq \{1, \ldots, k\}$, sodass $\sum_{i \in I} w_i \leq s_w$ und $\sum_{i \in I} n_i \geq s_n$?

TASKS

Gegeben: Zielwert W und n Aufgaben mit
- t_i ist Bearbeitungszeit (in Std) von Aufgabe i
- d_i ist Deadline (in Std) von Aufgabe i
- val_i ist Wert von Aufgabe i

Es gibt eine Maschine, die immer eine Aufgabe vollständig (ohne Unterbrechung) bearbeiten kann, und dann die nächste usw. Nur wenn eine Aufgabe vor der Deadline fertigstellt wird, wird der Wert val_i erzielt, ansonsten wird für diese Aufgabe kein Wert erzielt.

Gefragt: Gibt es eine Abarbeitungsreihenfolge der Aufgaben, sodass der Wert W mindestens erreicht wird?

Frage: Wie kann man KNAPSACK auf TASKS reduzieren?

$f(w_1, \ldots, w_k, n_1, \ldots, n_k, s_w, s_n) = (t_1, \ldots, t_k, d_1, \ldots, d_k, val_1, \ldots, val_k, W)$ mit $t_i = ?, d_i = ?, val_i = ?, W = ?$

Definition (KNAPSACK-Problem)

Das KNAPSACK-Problem lässt sich in der gegeben/gefragt-Notation wie folgt formulieren:

gegeben: k Gegenstände mit Gewichten $w_1, \ldots, w_k \in \mathbb{N}$ und Nutzenwerten $n_1, \ldots, n_k \in \mathbb{N}$, sowie zwei Zahlen $s_w, s_n \in \mathbb{N}$.

gefragt: Gibt es Teilmenge $I \subseteq \{1, \ldots, k\}$, sodass $\sum_{i \in I} w_i \leq s_w$ und $\sum_{i \in I} n_i \geq s_n$?

TASKS

Gegeben: Zielwert W und n Aufgaben mit
- t_i ist Bearbeitungszeit (in Std) von Aufgabe i
- d_i ist Deadline (in Std) von Aufgabe i
- val_i ist Wert von Aufgabe i

Es gibt eine Maschine, die immer eine Aufgabe vollständig (ohne Unterbrechung) bearbeiten kann, und dann die nächste usw. Nur wenn eine Aufgabe vor der Deadline fertigstellt wird, wird der Wert val_i erzielt, ansonsten wird für diese Aufgabe kein Wert erzielt.

Gefragt: Gibt es eine Abarbeitungsreihenfolge der Aufgaben, sodass der Wert W mindestens erreicht wird?

Frage: Wie kann man KNAPSACK auf TASKS reduzieren?

$f(w_1, \ldots, w_k, n_1, \ldots, n_k, s_w, s_n) = (t_1, \ldots, t_k, d_1, \ldots, d_k, val_1, \ldots, val_k, W)$ mit $t_i = ?, d_i = ?, val_i = ?, W = ?$

$$
\begin{aligned}
t_i &= w_i \\
val_i &= n_i \\
d_i &= s_w \text{ (für alle } i \text{ gleich!)} \\
W &= s_n
\end{aligned}
$$