

Tutorium Woche 9

Dominik Bruhn, Tutorium Nr. 9 + 11

13.01.2010

Agenda

- 1 Organisation
- 2 Komplexitätstheorie
 - Wiederholung
- 3 Aufgabe 1
- 4 Aufgabe 2
- 5 Ende

Komplexitätsklassen (Wiederholung)

Definition (**P**)

Problem $P \in \mathcal{P}$, genau wenn gilt:

Komplexitätsklassen (Wiederholung)

Definition (P)

Problem $P \in \mathcal{P}$, genau wenn gilt:

P kann in **polynominaler** Zeit von einer **deterministischen** Turingmaschine **entscheiden** werden.

Komplexitätsklassen (Wiederholung)

Definition (P)

Problem $P \in \mathcal{P}$, genau wenn gilt:

P kann in **polynomineller** Zeit von einer **deterministischen** Turingmaschine **entscheiden** werden.

Definition (NP)

Problem $P \in \mathcal{NP}$, genau wenn gilt:

Komplexitätsklassen (Wiederholung)

Definition (P)

Problem $P \in \mathcal{P}$, genau wenn gilt:

P kann in **polynomineller** Zeit von einer **deterministischen** Turingmaschine **entscheiden** werden.

Definition (NP)

Problem $P \in \mathcal{NP}$, genau wenn gilt:

P kann in **polynomineller** Zeit von einer **nicht deterministischen** Turingmaschine **entschieden** werden.

Komplexitätstheorie

Definition (NP-Schwere)

Ein Problem F heißt \mathcal{NP} -schwer oder \mathcal{NP} -hart falls gilt:

Komplexitätstheorie

Definition (NP-Schwere)

Ein Problem F heißt \mathcal{NP} -schwer oder \mathcal{NP} -hart falls gilt:
Alle Q aus \mathcal{NP} sind polynomiell reduzierbar auf F .
 F muss selbst nicht $\in \mathcal{NP}$ sein!

Komplexitätstheorie

Definition (NP-Schwere)

Ein Problem F heißt \mathcal{NP} -schwer oder \mathcal{NP} -hart falls gilt:
Alle Q aus \mathcal{NP} sind polynomiell reduzierbar auf F .
 F muss selbst nicht $\in \mathcal{NP}$ sein!

Definition (NP-Vollständigkeit)

Ein Problem F heißt \mathcal{NP} -vollständig falls gilt:

Komplexitätstheorie

Definition (NP-Schwere)

Ein Problem F heißt \mathcal{NP} -schwer oder \mathcal{NP} -hart falls gilt:
Alle Q aus \mathcal{NP} sind polynomiell reduzierbar auf F .
 F muss selbst nicht $\in \mathcal{NP}$ sein!

Definition (NP-Vollständigkeit)

Ein Problem F heißt \mathcal{NP} -vollständig falls gilt:

- F ist in der Klasse \mathcal{NP} , d.h. $F \in \mathcal{NP}$ und
- F ist \mathcal{NP} -schwer

Aufgabe 1.1

4-COLOR (Vierfärbbarkeitsproblem)

Gegeben: Ein ungerichteter Graph $G = (V, E)$

Gesucht: Gibt es eine Färbung der Knoten V , sodass je zwei durch eine Kante aus E miteinander verbundene Knoten unterschiedlich gefärbt sind, wenn nur vier unterschiedliche Farben zur Verfügung stehen?

Aufgabe 1.1

4-COLOR (Vierfärbbarkeitsproblem)

Gegeben: Ein ungerichteter Graph $G = (V, E)$

Gesucht: Gibt es eine Färbung der Knoten V , sodass je zwei durch eine Kante aus E miteinander verbundene Knoten unterschiedlich gefärbt sind, wenn nur vier unterschiedliche Farben zur Verfügung stehen?

Aufgabe

Zeige: 4-COLOR ist \mathcal{NP} -vollständig

Aufgabe 1.1

4-COLOR (Vierfärbbarkeitsproblem)

Gegeben: Ein ungerichteter Graph $G = (V, E)$

Gesucht: Gibt es eine Färbung der Knoten V , sodass je zwei durch eine Kante aus E miteinander verbundene Knoten unterschiedlich gefärbt sind, wenn nur vier unterschiedliche Farben zur Verfügung stehen?

Aufgabe

Zeige: 4-COLOR ist \mathcal{NP} -vollständig

Bekannt

Bekannt: Das Dreifärbbarkeitsproblems 3-COLOR ist \mathcal{NP} -Vollständig.

Aufgabe 1.2

3-CLIQUE

Gegeben: Ein ungerichteter Graph $G = (V, E)$

Gesucht: Gibt es eine Clique (vollständig verbundener Teilgraph) der Größe 3 in G ?

Aufgabe 1.2

3-CLIQUE

Gegeben: Ein ungerichteter Graph $G = (V, E)$

Gesucht: Gibt es eine Clique (vollständig verbundener Teilgraph) der Größe 3 in G ?

Aufgabe

Zeige: 3-CLIQUE ist **nicht** \mathcal{NP} -vollständig.

Aufgabe 2

PARTITION

Gegeben: Natürliche Zahlen $a_1, \dots, a_n \in \mathbb{N}$ ($n \in \mathbb{N}$)

Gesucht: Gibt es eine Teilmenge $J \subseteq \{1, \dots, n\}$ mit

$$\sum_{1 \leq i \leq n, i \in J} a_i = \sum_{1 \leq i \leq n, i \notin J} a_i?$$

BIN PACKING

Gegeben: Eine Behältergröße $b \in \mathbb{N}$, die Anzahl der Behälter k und Objekte a_1, \dots, a_n mit $a_i \in \mathbb{N}$, $a_i \leq b$ für alle $i \in \{1, \dots, n\}$

Gesucht: Können die n Objekte so auf die k Behälter verteilt werden, dass kein Behälter überbeladen ist?

Aufgabe

Zeige: BIN PACKING ist \mathcal{NP} -hart, wobei PARTITION als \mathcal{NP} -vollständig vorausgesetzt werden darf.

Ende

Fragen?

*When your hammer is C++,
everything begins to look like a
thumb.*