Übersicht

Komplexitätsklassen

Entscheidungsproblem ist gegeben durch:

- Menge von Instanzen I
- Teilmenge der positiven Instanzen $Y \subseteq I$

Beispiel: SAT

- I_{SAT} = Formeln in KNF
- Y_{SAT} = erfüllbare Formeln

Formel in KNF	$F = C_1 \wedge \ldots \wedge C_m$
Klausel	$C = a_1 \vee \ldots \vee a_k$
Literal	$a = x$ oder $a = \neg x$.

Komplexitätsklassen

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}$$

NP als Verifikation von Lösungen

Problem P ist in **NP** gdw.

- es gibt Menge W_P, Problem $Q \in \mathbf{P}$ und Polynom $q()$ mit
- I_Q sind Paare (x, w) mit $x \in I_P$, $w \in W_P$ und $|w| \leq q(|x|)$
- $\forall x \in I_P : \quad x \in Y_P \leftrightarrow \exists w \in W_P : (x, w) \in Y_Q$

Beispiel: *SAT*

- W_{SAT} = Bewertung der Variablen
- $(F, \alpha) \in Y_R$ gdw. α erfüllt F

Reduktion

Funktion f reduziert P auf Q, wenn

- $f : I_P \to I_Q$ in polynomieller Zeit berechenbar
- für alle $x \in I_P$ ist $x \in Y_P \leftrightarrow f(x) \in Y_Q$

$P \leq_m Q$ falls es eine Reduktion f von P auf Q gibt.

Eigenschaft:

Ist $Q \in \mathbf{P}$ und $P \leq_m Q$, dann auch $P \in \mathbf{P}$.

Vollständigkeit

Definition:

- Problem P ist **NP**-schwer, falls $Q \leq_m P$ für alle $Q \in \mathbf{NP}$.
- P ist **NP**-vollständig, falls $Q \in \mathbf{NP}$ und **NP**-schwer.

Q **NP**-vollständig $\Rightarrow$ $Q \in \mathbf{P}$ gdw. $\mathbf{P} = \mathbf{NP}$

Satz (Cook)

SAT ist **NP***-vollständig.*

Problem 3-*SAT*: Jede Klausel $C = a_1 \vee a_2 \vee a_3$

Satz (Cook)

3-*SAT ist* **NP***-vollständig.*

Lineare Programmierung

Problem *ILP* (Integer Linear Programming)

Instanz:	Menge I von linearen Ungleichungen über $\mathbb{Z}$ in Variablen $z_1, \ldots, z_n$
Frage:	Gibt es eine Lösung von I in $\mathbb{Z}$ d.h., Zuweisung von Werten $z_1, \ldots, z_n \in \mathbb{Z}$ so dass Ungleichungen I gelten.

Satz

ILP ist **NP***-vollständig.*

Problem *LP*:

Wie ILP, aber gesucht sind Lösungen in $\mathbb{Q}$.

Im Gegensatz zum obigen Satz ist *LP* in **P**.

Graphenprobleme in **NP**

Problem *VC*

Instanz:	Graph $G = (V, E)$, $k \in \mathbb{N}$
Frage:	Gibt es vertex cover U mit $\lvert U \rvert \leq k$?

Vertex cover: $U \subseteq V$ mit $e \cap U \neq \emptyset$ für alle $e \in E$.

Problem CLIQUE

Instanz:	Graph $G = (V, E)$, $k \in \mathbb{N}$
Frage:	Gibt es Clique U mit $\lvert U \rvert \geq k$?

Clique: $U \subseteq V$ mit $\{x, y\} \in E$ für alle $x, y \in U$.

Problem *HC*

Instanz:	Graph $G = (V, E)$
Frage:	Gibt es einen Hamilton-Kreis in G ?

Hamilton-Kreis: Kreis, der jedes $v \in V$ genau einmal besucht.

NP-vollständige Graphenprobleme

Offensichtlich: *VC*, CLIQUE, *HC* sind in **NP**.

Reduktionen (Karp)

- $3\text{-}SAT \leq_m VC$
- $VC \leq_m \text{CLIQUE}$
- $VC \leq_m HC$

Also: *VC*, CLIQUE, *HC* sind **NP**-vollständig.

Approximations-algorithmen

Einführung
Grundlagen
Komplexität
Optimierung
Komplexität von Optimierungsproblemen
Methoden zum Entwurf
Approximationsklassen

Optimierungsprobleme

Ein *Optimierungsproblem* ist gegeben durch:

- Menge I von Instanzen.
- Für jedes $x \in I$ eine Menge $S(x)$ von potentiellen Lösungen.
- Funktion m, die jedem (x, y) mit $x \in I$ und $y \in S(x)$ ein Maß $m(x, y)$ zuordnet.
- Ziel goal $\in \{\min, \max\}$.

Eine Lösung $y^* \in S(x)$ heißt optimal, wenn

$$m(x, y^*) = m^*(x) := \text{goal}\{\, m(x, y)\,;\; y \in S(x)\,\}$$

$$S^*(x) := \{\, y^* \in S(x)\,;\; m(x, y^*) = m^*(x)\,\}$$

Beispiele

Approximations-algorithmen

Einführung
Grundlagen
Komplexität
Optimierung
Komplexität von Optimierungsproblemen
Methoden zum Entwurf
Approximationsklassen

Minimum Path

Instanz:	ungerichteter Graph $G = (V, E)$, Ecken s, t
Lösung:	Weg von s nach t
Maß:	Länge des Weges

Minimum Vertex Cover

Instanz:	ungerichteter Graph $G = (V, E)$
Lösung:	vertex cover $U \subseteq V$
Maß:	$\lvert U \rvert$

Maximum Clique

Instanz:	ungerichteter Graph $G = (V, E)$
Lösung:	Clique $U \subseteq V$
Maß:	$\lvert U \rvert$

Problemvarianten

Zu einem Optimierungsproblem P sind definiert:

konstruktives Problem P_C: Gegeben $x \in I$, finde $y \in S^*(x)$.

Auswertungsproblem P_E: Gegeben $x \in I$, berechne $m^*(x)$.

Entscheidungsproblem P_D: Gegeben $x \in I$ und $k \in \mathbb{N}$,
ist $\text{goal}(m^*(x), k) = m^*(x)$?

Beispiele:

$P =$ Minimum Vertex Cover	$P_D =$ VC.
$P =$ Maximum Clique	$P_D =$ Clique

Die Klassen **NPO** und **PO**

Ein Optimierungsproblem ist in der Klasse **NPO**, falls

1. I ist in **P**.
2. für $x \in I$ und $y \in S(x)$ ist $|y| \leq q(|x|)$
3. für x, y mit $|y| \leq q(|x|)$ ist die Frage "$y \in S(x)$?" in **P**
4. m ist in polynomieller Zeit berechenbar.

Eigenschaft: P in **NPO** $\implies$ P_D in **NP**

$P \in$ **NPO** ist in der Klasse **PO**, falls
das konstruktive Problem P_C in polynomieller Zeit lösbar ist.

Analog: P in **PO** $\implies$ P_D in **P**

Turing-Reduktion

Eine Turing-Reduktion von P auf Q ist

- Algorithmus, der P löst
- benutzt Subroutine zur Lösung von Q
- Laufzeit des Algorithmus ohne Subroutine ist polynomiell

$P \leq_T Q$ falls eine Turing-Reduktion von P auf Q gibt.

Eigenschaft:

Ist $Q \in \mathbf{P}$ und $P \leq_T Q$, dann auch $P \in \mathbf{P}$.

Relationen zwischen Problemvarianten

Offensichtlich: $P_D \leq_T P_E \leq_T P_C$

Fakt

*Für alle Optimierungsprobleme $P \in$ **NPO** ist $P_E \leq_T P_D$.*

Beweis:

Da $|m(x,y)| \leq p(|x|)$ für alle $y \in S(x)$, ist $m^*(x) < 2^{p(|x|)}$

$\rightsquigarrow$ binäre Suche findet $m^*(x)$ in Zeit $O(p(|x|))$.

Relationen zwischen Problemvarianten

Beispiel

Für $P = \textsc{Maximum Clique}$ ist $P_C \leq_T P_E$.

Für $G = (V, E)$ definiere

- $N(v) := \{ u \in V ; \{u, v\} \in E \}$
- $G(v)$: induzierter Teilgraph auf $\{v\} \cup N(v)$
- $G^-(v)$: induzierter Teilgraph auf $N(v)$

Algorithmus:

MaxClique(G)
  $k := \textsc{Maximum Clique}_E(G)$
  `if` $k = 1$ `then return any` $v \in V$
  `finde` $v \in V$ `mit` $\textsc{Maximum Clique}_E(G(v)) = k$
  `return` $\{v\} \cup$ *MaxClique*$(G^-(v))$

Relationen zwischen Problemvarianten

Approximations-algorithmen

Einführung
Grundlagen
Komplexität
Optimierung
Komplexität von Optimierungsproblemen
Methoden zum Entwurf
Approximationsklassen

Ob $P_C \leq_T P_D$ allgemein gilt, ist unbekannt.

Satz

$P_C \leq_T P_D$ gilt, falls P_D **NP***-vollständig ist.*

NP-schwere Optimierungsprobleme

Definition

Problem P in **NPO** ist **NP**-schwer, falls $Q \leq_T P_C$ für alle $Q \in$ **NP**.

Eigenschaft:

$$P_D \text{ NP-vollständig} \implies P \text{ ist NP-schwer.}$$

$$\mathbf{P} \neq \mathbf{NP} \implies \mathbf{PO} \neq \mathbf{NPO}$$

Ein **NP**-schweres Problem

MINIMUM TRAVELING SALESPERSON

Instanz: n Städte $c_1, \ldots, c_n$,
Distanzmatrix $D \in \mathbb{N}^{n \times n}$

Lösung: *Tour*: Permutation $c_{i_1}, \ldots, c_{i_n}$ der Städte

Maß: Kosten $\sum_{k=1}^{n-1} D(i_k, i_{k+1}) + D(i_n, i_1)$

Satz: MINIMUM TRAVELING SALESPERSON ist **NP**-schwer.