Technische Universität München
Institut für Informatik
Prof. Dr. Helmut Veith
Dr. Christian Schallhart
Dipl.-Inf. (Univ.) Stefan Kugele

WS 2007/08
Übungsblatt 1
Mittwoch, 14. November 2007

Übung zur Vorlesung Komplexitätstheorie

Hinweis: Die Zentralübung zur Vorlesung "Komplexitätstheorie" findet nach Absprache immer

mittwochs zwischen 10:15 und 11:45 Uhr im HS2

statt. Beachten Sie hierzu bitte auch die Informationen auf der Website zur Vorlesung und Übung unter

`http://www.model.in.tum.de`

Wir wünschen Ihnen für das Wintersemester 2007/08 viel Erfolg.

Präsentation der Lösungen am 5.12.2007

Aufgabe 1 *2-Coloring Solver*

Schreiben Sie ein Programm, welches als Eingabe über stdin eine DIMACS Graph Beschreibung liest und als Ergebnis liefert, ob der Graph 2-färbbar ist oder nicht.
Hinweis: Es existiert eine effiziente Lösung.

Aufgabe 2 *3-Coloring Solver*

Schreiben Sie ein Programm, welches als Eingabe über stdin eine DIMACS Graph Beschreibung liest und als Ergebnis liefert, ob der Graph 3-färbbar ist oder nicht.
Hinweis: Gehen Sie nicht davon aus eine effiziente Lösung zu finden.

Aufgabe 3 *2-SAT Solver*

Schreiben Sie ein Programm, welches als Eingabe über stdin eine DIMACS CNF Beschreibung liest und überprüft, ob die Eingabe-Instanz eine erfüllende 2-SAT Formel ist, d.h. die Eingabe muss eine erfüllende Belegung haben und weiter darf jede Klausel höchstens zwei Literale besitzen.

Aufgabe 4 *Membership von Problemen*

Ziel dieser Aufgabe ist es, Ihnen ein Gefühl für die Probleme, die in „natürlicher Weise" in **NP** liegen, zu vermitteln. Dazu sollen Sie für die folgenden Probleme zeigen, dass sie in **NP** liegen und jeweils eine negative und positive Instanz für das Problem angeben. Im Falle von CircuitSat müssten Sie also einen Algorithmus angeben, der in nichtdeterministischer, polynomieller Zeit läuft, sowie jeweils einen erfüllbaren und einen unerfüllbaren Schaltkreis finden.

(a) 3Sat

(b) VertexCover

(c) DominatingSet

(d) LongestPath

(e) SetCover

(f) Partition

(g) IntegerProgramming

Aufgabe 5 *Beweisen von Zugehörigkeit zu **NP***

Gehen Sie davon aus, dass Ihr Solver nicht nur mit ja/nein antwortet, sondern Ihnen auch eine Lösung ausgibt. Zum Beispiel liefert ein SAT-Solver auch eine erfüllende Belegung, falls es solch eine gibt. Wie schwierig ist es nun zu **überprüfen**, ob die Lösung auch korrekt ist? Geben Sie für Sat, 3Coloring und Clique einen effizienten Test-Algorithmus in Pseudo-Code an.

Aufgabe 6 *Übersetzung von Problemen*

Zeigen Sie, wie eine Instanz von 3Coloring

(a) in eine Instanz von IntegerProgramming übersetzt werden kann.

(b) in eine Instanz von 3Sat übersetzt werden kann.

(c) Und etwas schwieriger: Können Sie eine Übersetzung von 3Sat nach VertexCover finden?

Aufgabe 7 *Reduktionen*

Führen Sie die folgenden Reduktionen durch:

(a) Sat $\leq$ 3Sat

(b) VertexCover $\leq$ DominatingSet

(c) 3Sat $\leq$ SubsetSum

(d) SubsetSum $\leq$ Partition

(e) Partition $\leq$ BinPacking

Bei jeder Reduktion $f(\cdot)$, die $A \leq B$ zeigt, müssen Sie argumentieren (und nicht unbedingt formal beweisen), dass

- für alle x auch $x \in A \Leftrightarrow f(x) \in B$ hält, und dass
- die Transformation $f(\cdot)$ in logarithmischen Platz berechenbar ist.

A DIMACS Formate

A.1 DIMACS Graph Format (.col)

Dateien im DIMACS Standard Graph Format sind ASCII Dateien und zeilenweise aufgebaut:

(a) Zeilen, die einen **Kommentar** enthalten, starten mit einem **c**.

(b) Die **Problembeschreibung** kommt genau einmal in der Datei vor und ist die erste Zeile, die kein Kommentar ist. Sie folgt der Form: **p edge** |**V**| |**E**|, wobei die letzten beiden die Anzahl der Knoten und Kanten darstellen.

(c) **Kanten** werden durch eine Zeile in der Form **e u v**, wobei **u** und **v** die adjazenten Knoten sind, beschrieben. Eine Graph-Datei enthlt niemals gleichzeitig zwei Kanten der Form: **e u v** und **e v u**. u und v sind aus $1 \dots |V|$.

Beachten Sie, dass das erste Zeichen einer Zeile (welches den Typ beschreibt) keine vorangestellten Leerzeichen haben darf

A.2 DIMACS CNF Format (.cnf)

DIMACS CNF Dateien sind wieder zeilenweise aufgebaut und haben folgende Struktur:

(a) **Kommentare** starten mit einem **c**.

(b) Die **Problembeschreibung** kommt genau einmal vor und ist die erste Zeile, die kein Kommentar ist.

(c) **Klauseln** bilden bis auf Kommentare den Rest der Datei. Variablen werden von 1 bis $|V|$ nummeriert wobei es nicht notwendig ist, dass jede Nummer in der Datei vorkommt. Nummern werden durch ein Leerzeichen von einander getrennt. Positive Variablen i werden einfach als **i** geschrieben und negativ vorkommende Variablen $\neg i$ als **-i**. Klauseln werden mittels einer **0** beendet.

B Definitionen

Wir werden die folgenden Definitionen zum Teil für dieses Übungsblatt, hauptsächlich aber für spätere Blätter benötigen.

B.1 Satisfiability Variants

- 3Sat is the set of CNF-expressions where each clause contains *at most three* literals, such that there is an assignment which satisfies all clauses simultaneously.
- NaeSat is the set of CNF-expressions where each clause contains *exactly three* literals, such that there is an assignment which does not assign all three literals of any clause the same truth value. I.e., a CNF-expression is not-all-equal satisfiable, if there is an assignment which satisfies at least one literal in each clause but does not satisfy all three literals in any clause simultaneously.
- An instance of Max2Sat consists of a positive integer K and a CNF-expression ϕ where each clause contains at most two literals. The question is whether there is a Boolean assignment which satisfies at least K clauses simultaneously in ϕ.

B.2 Graph Problems

- A VertexCover-instance is consists of an undirected graph $\langle V, E\rangle$ and a positive bound K. The problem is to find a subset $V' \subseteq V$ with $|V'| \leq K$ such that $\{u, v\} \cap V' \neq \emptyset$ for all $\langle u, v\rangle \in E$.

- HyperGraphVertexCover is a generalization of VertexCover. Hypergraphs allow *more than two* vertices per edge, i.e., in a hypergraph $\langle V, H\rangle$ an hyperedge $h \in H$ is subset of V. HyperGraphVertexCover asks for a subset $V' \subseteq V$ with $|V'| \leq K$ such that for all $h \in H$ we have a $v \in V'$ with $v \in h$.

- DominatingSet has directed graphs $\langle V, E\rangle$ together with a bound K as instances. The task is to find a subset $D \subseteq V$ with $|D| \leq K$ such that for each $v \in V - D$ there is a $u \in D$ with $\langle u, v\rangle \in E$.

- A LongestPath-instance contains a directed graph G, two vertices s, t and a positive integer K. The question is whether there is a simple path (i.e., a path that passes a vertex at most once) through G of length K which starts at s and ends at t – or longer?

- In FeedbackVertexSet we are asked to find a subset $V' \subseteq V$ with $|V| \leq K$ of the vertices of a directed graph $G = \langle V, E\rangle$ such that any cycle in G contains at least one vertex in V'.

- In FeedbackEdgeSet we are asked to find a subset $E' \subseteq E$ with $|E| \leq K$ of the edges of a directed graph $G = \langle V, E\rangle$ such that any cycle in G contains at least one edge in E'.

- In 3Coloring we have an undirected graph $\langle V, E\rangle$ as instance and are asked to find an assignment $\tau : V \to \{R, B, G\}$ such that for any two vertices $u, v \in V$ with $\langle u, v\rangle \in E$ $\tau(u) \neq \tau(v)$ holds.

B.3 Set Problems

- SetCover has instances of the form $\langle C, U, K\rangle$ where $c \in C$ are subsets of the universe U and K is a positive integer. The task is to find a subset $C' \subseteq C$ with $|C'| \leq K$ such that $\bigcup_{c \in C'} c = U$.

B.4 Numerical Problems

- The problem: Given a purse with a set of coins, and given an amount of money to pay, can you pay the amount *precisely*?

  Formally: An instance $\langle A, S\rangle$ of SubsetSum is a set of positive integers $A = \{a_1, \ldots, a_n\}$ and a positive integer S. $\langle A, S\rangle$ is a positive instance, iff there is a subset $A' \subseteq A$ such that $\sum_{a \in A'} a = S$.[1]

- A special case of SubsetSum is Partition– given a set of coins, can you divide it into two sets of equal value?

[1] Note that this formulation does not allow to have more than one item with the same weight. Whether multiple items of the same weight are allowed or not is unimportant with respect to the complexity of the problem.

Formally: A PARTITION-instance $\langle A \rangle$ is a positive one, iff there is a subset A' such that $\sum_{a \in A'} a = \sum_{a \in A - A'} a$.

- The Problem: Given a set of items (each with a value and a weight), can you find a subset of maximum value fitting your knapsack?

  Formally: An instance $\langle A, w, v, W, V \rangle$ of KNAPSACK comes with a set of items $A = \{a_1, \ldots, a_n\}$, a weight function $w : A \to \mathcal{N}$, a value function $v : A \to \mathcal{N}$, a positive capacity W, and the minimal value V to be packed into the knapsack. The task is to find a subset $A' \subseteq A$ such that $\sum_{a \in A'} w(a) \leq W$ with $\sum_{a \in A'} v(a) \geq V$.

- The Problem: Given a set of items (each with a certain weight), how many knapsacks do you need to carry them?

  Formally: An instance $\langle A, C, B \rangle$ of BINPACKING is a set of positive integers $A = \{a_1, \ldots, a_n\}$ and two positive integer C and B. Given an instance $\langle A, C, B \rangle$, the task is to find a partition $\langle B_1, \ldots, B_k \rangle$ of A with

  - $k \leq B$
  - $\sum_{a \in B_i} a \leq C$ for $1 \leq i \leq k$
  - $\bigcup_{i=1}^{i=k} B_i = A$.

- In INTEGERPROGRAMMING we have an integer matrix A and two integer vectors b, c and an integer B, determine if there is an integer vector x such that $Ax \leq b$ and $cx \geq B$?