Prof. Javier Esparza
Philipp Czerner, Martin Helfrich

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2021 – Übungsblatt 12

- Das Übungsblatt ist in zwei Teile gegliedert: den Vorbereitungsteil, den Sie vor der Übung selbstständig bearbeiten sollen, und den Übungs-/Nachbereitungsteil, der Aufgaben enthält, die in der Übung besprochen werden und von Ihnen anschließend zur Nachbereitung verwendet werden können.
- Das ist nicht das Hausaufgabenblatt! Die Hausaufgaben finden Sie auf einem separaten Blatt.

Vorbereitung ($\rightarrow$ vor der Übung selbständig zu bearbeiten)

Individualaufgabe Ü12.1. (*Wichtige Begriffe*)

Überprüfen Sie, dass Sie die folgenden Begriffe oder Notationen korrekt definieren können.

- $time_M(w)$ und $ntime_M(w)$
- $TIME(f(n))$ und $NTIME(f(n))$
- Landau-Notation (oder auch $\mathcal{O}$-Notation): $\mathcal{O}(n), \mathcal{O}(2^n), ...$
- P und NP
- P $\stackrel{?}{=}$ NP
- NP-hart (oder auch auch: “NP-schwer”)
- NP-vollständig
- Zertifikat und polynomiell beschränkter Verifikator
- polynomielle Reduktion
- $A \leq_p B$ (sprich: *"A ist polynomiell reduzierbar auf B"*)
- Die Probleme[1]: HAMILTON, RUCKSACK, SAT, 3KNF-SAT, FÄRBBARKEIT (COL), 3COL, MENGENÜBERDECKUNG (MÜ), CLIQUE, PARTITION, BIN PACKING, TRAVELLING SALESMAN (TSP)

Individualaufgabe Ü12.2. (*TSP ist NP-vollständig*)

Sie kennen bereits das Problem HAMILTON $:= \{G \mid G \text{ enthält einen Hamiltonkreis}\}$. Dieses Problem ist **NP**-vollständig. Zeigen Sie nun, dass das Travelling-Salesman-Problem (TSP) ebenfalls **NP**-vollständig ist.

TSP:

- Eingabe: $n \times n$ Matrix M mit $M_{i,j} \in \mathbb{N}$ und eine Zahl $k \in \mathbb{N}$.
- Frage: Gibt es eine “Rundreise” der Länge $\leq k$.

[1] Wir lernen viele dieser Probleme erst in den kommenden Vorlesungen kennen. Sie müssen nicht schon zum jetzigen Zeitpunkt alle Probleme kennen / verstehen.

Eine Rundreise ist eine Permutation π von $[1; n]$, also ein einfacher Kreis, der jede Insel einmal enthält. Die Länge der Rundreise ist die Summe der Kosten der einzelnen Reisen, also $\Sigma_{i=1}^{n} M_{\pi_i, \pi_j}$ (dabei ist π_i der i-te Eintrag der Permutation, beginnend bei 0, und π_j ist der Eintrag für die nächste Insel, also $j = (i+1)\%n$).

Lösungsskizze. (Diese Lösung ist sehr ausführlich und wird teilweise auch auf Folie 350 behandelt.) Um zu zeigen, dass TSP **NP**-vollständig ist, müssen wir zeigen, dass TSP $\in$ **NP** und dass TSP **NP**-schwer ist.

- $\in$ **NP**: Wir können eine korrekte Rundreise raten, und dann in polynomieller Zeit verifizieren, dass sie eine Länge $\leq k$ hat, indem wir einmal den Pfad entlang gehen und die Kosten ausrechnen. Das entspricht $|\pi| = n$ Zugriffen auf die Matrix, geht also in linearer Zeit.
- **NP**-schwer: Durch Reduktion HAMILTON $\leq_p$ TSP. Gegeben das Encoding eines Graphen $G = (V, E)$ konstruieren wir eine Matrix wie folgt: Wir nummerieren die Knoten in G von 1 bis n durch. Dann ist

  $$M_{i,j} := \begin{cases} 1 & \text{falls } \{v_i, v_j\} \in E \\ 2 & \text{sonst} \end{cases}$$

  Also sind die Kosten, von i nach j zu kommen 1, wenn eine Kante in G existiert, und 2, wenn nicht. Dann ist (M, n) eine Instanz des TSP und es gilt $G \in$ HAMILTON $\Leftrightarrow$ $f(G) = (M, n) \in$ TSP. (noch zu beweisen)

  Gegeben etwas, das keinen Graphen encodiert, geben wir etwas zurück, das nicht $\in$ TSP ist, also z.B. etwas, das kein Tupel (M, k) encodiert oder eine Matrix mit positiven Kosten und $k = 0$.

  Außerdem noch ein Randfall: Gegeben einen Graphen, der weniger als 3 Knoten hat, geben wir etwas zurück, das nicht $\in$ TSP ist, weil ein Graph mit weniger als 3 Knoten niemals einen Hamilton-Kreis enthalten kann; unsere Reduktion könnte aber eventuell ein Element aus TSP zurück geben. Beispiel: $G = (\{1, 2\}, \{\{1, 2\}\})$.

  Korrektheit:

  - Wenn $G \in$ HAMILTON, dann gibt es einen Kreis, der nur Kanten von G verwendet und alle Knoten genau einmal besucht. Folglich gibt es auch eine Rundreise mit Kosten n in M.
  - Wenn $G \notin$ HAMILTON, dann gibt es keinen Kreis, der nur Kanten von G verwendet und alle Knoten genau einmal besucht. Folglich gibt es keine Rundreise mit Kosten $\leq n$ in M, da mindestens einmal eine Kante mit Kosten 2 verwendet werden muss.
  - Gegeben etwas, das keinen Graph encodiert, also nicht in HAMILTON ist, geben wir etwas zurück, das nicht in TSP ist.

  Berechenbarkeit in polynomieller Zeit:

  Die Reduktion definiert eine $n \times n$ Matrix, wobei n die Anzahl der Knoten von G ist. Folglich braucht sie quadratische Zeit in der Anzahl der Knoten.

Übung und Nachbereitung

Fokusaufgabe Ü12.3. (*Stundenplan*)

In dieser Aufgabe betrachten wir das STUNDENPLAN-Problem:

- **Eingabe**: Endliche Mengen S (Studierende), V (Vorlesungen) und T (Termine) und eine Relation $R \subseteq S \times V$. Dabei bedeutet $(s, v) \in R$, dass s die Vorlesung v besuchen möchte.
- **Frage**: Gibt es eine Abbildung $f : V \to T$, so dass alle Studierenden einen überschneidungsfreien Stundenplan haben, also

$$(s, v_1) \in R \wedge (s, v_2) \in R \wedge v_1 \neq v_2 \implies f(v_1) \neq f(v_2) \, .$$

(a) Zeigen Sie, dass STUNDENPLAN in NP liegt.

(b) Zeigen Sie, dass STUNDENPLAN NP-schwer ist, indem sie eine polynomielle Reduktion von 3COL auf STUNDENPLAN angeben. (Wir werden in der Vorlesung noch lernen, dass 3COL NP-schwer ist.)

Übungsaufgabe Ü12.4. (*SAT-Varianten*)

Wir betrachten verschiedene Varianten von SAT, die auch NP-vollständig sind.

In der Vorlesung werden wir das NP-vollständige Problem 3KNF-SAT kennen lernen. Dies führen wir hier vorab schon einmal ein, da es als Einschränkung von SAT die Reduktion leichter macht.
Sei X eine Menge an Variablen. Die Menge der *Literale* ist $L := X \cup \{\neg x : x \in X\}$, enthält also alle Variablen und Negationen von Variablen. Eine *k-Klausel* ist eine Formel der Form $l_1 \vee l_2 \vee ... \vee l_k$ mit $l_1, ..., l_k \in L$, und eine Formel F ist in *k-konjunktiver Normalform (k-KNF)*, wenn $F = F_1 \wedge F_2 \wedge ... \wedge F_n$, wobei $F_1, ..., F_n$ jeweils k-Klauseln sind. Das Problem 3KNF-SAT ist nun

- **Eingabe**: Aussagenlogische Formel F in 3-KNF
- **Frage**: Ist F erfüllbar?

Ein Beispiel für eine Instanz wäre also $(\neg x \vee y \vee z) \wedge (x \vee \neg y \vee z) \wedge (x \vee y \vee \neg z)$.

Wir betrachten zusätzlich zu 3KNF-SAT noch weitere Varianten von SAT, die auch NP-vollständig sind. Zeigen Sie diese NP-Vollständigkeit, indem Sie für jede Variante X eine Reduktion 3KNF-SAT $\leq_p X$ angeben und $X \in$ NP zeigen.

(a) 3-OCC-KNF-SAT:

- **Eingabe**: Eine Formel F in KNF, bei der jede Variable höchstens dreimal auftritt.
- **Frage**: Ist F erfüllbar?

(b) Wir betrachten den ITE-Operator mit der Semantik $\mathsf{ITE}(x, y, z) := (x \to y) \wedge (\neg x \to z)$. Eine ITE-Formel genügt der folgenden Grammatik:

$$F \to \mathsf{ITE}(F, F, F) \mid x \mid \mathsf{true} \mid \mathsf{false} \qquad \text{für Variablen } x \in \mathcal{V}$$

ITE-SAT:

- **Eingabe**: Eine ITE-Formel F.
- **Frage**: Ist F erfüllbar?

Lösungsskizze.

(a)
- **NP-schwer**: Sei F eine Formel in 3-KNF. Somit $F = \bigwedge_{i=1}^{n} K_i$ und $K_i = \bigvee_{j=1}^{m} L_{ij}$ mit $m \leq 3$. O.b.d.A. kommen keine Variablen doppelt in einer Klausel vor. Sei v die Anzahl der in F verwendeten Variablen. Dann ersetzten wir in jeder Klausel die Variablen mit für diese Klausel spezifische Variablen und fügen eine Bedingung ein, die erzwingt, dass alle Kopien einer Variable in der Eingabe den gleichen Wert zugewiesen bekommen. Diese Bedingung hat eine Ringstruktur, um Variablenverwendungen zu sparen: $(x_{k,1} \to x_{k,2}) \wedge (x_{k,2} \to x_{k,3}) \wedge ...(x_{k,n} \to x_{k,1})$.

$$f(F) = \bigwedge_{i=1}^{n} f(K_i) \wedge \bigwedge_{k=1}^{v} \bigwedge_{i=1}^{n} (\neg x_{k,i} \vee x_{k,(i \bmod n)+1})$$
$$f(K_i) = \bigvee_{j=1}^{m} f(L_{ij})$$
$$f(L_{ij}) = \begin{cases} x_{k,i} & \text{falls } L_{ij} = x_k \\ \neg x_{k,i} & \text{falls } L_{ij} = \neg x_k \end{cases}$$

Diese Reduktion ist in polynomieller Zeit berechenbar, da nur die Eingabe einmal kopiert werden muss und ein zusätzlicher Term der Größe $m \cdot v$ erzeugt wird.

Korrektheit: Aus jeder erfüllenden Belegung σ für F können wir eine erfüllende Belegung σ' für $f(F)$ konstruieren mit $\sigma'(x_{k,i}) = \sigma(x_k)$. Auch können wir aus jeder erfüllenden Belegung σ für $f(F)$ ein Belegung für F konstruieren. Da $\sigma(x_{k,i}) = \sigma(x_{k,j})$ für alle i, j gelten muss, können wir einfach σ' über $\sigma'(x_k) = \sigma(x_{k,1})$ definieren. Somit ist die Konstruktion erfüllbarkeitserhaltend und die Reduktion korrekt.

- $\in$ **NP**: Eine erfüllende Belegung ist ein geeignetes Zertifikat, das es maximal genauso groß ist wie die Formel F und in polynomieller Zeit geprüft werden kann.

(b)
- **NP-schwer**: Sei F eine Formel in 3-KNF. Wir wandeln F in polynomieller Zeit unter der Verwendung der folgenden semantischen Äquivalenzen in eine ITE-Formel:

$$\neg x \equiv \mathsf{ITE}(x, \mathsf{false}, \mathsf{true}) \qquad x \wedge y \equiv \mathsf{ITE}(x, y, \mathsf{false}) \qquad x \vee y \equiv \mathsf{ITE}(x, \mathsf{true}, y)$$

Eine Klausel mit drei Literalen, z.B. $\neg x \vee \neg y \vee \neg z$ lässt sich damit umformen

zu

$$
\begin{aligned}
\neg x \vee \neg y \vee \neg z \equiv \mathsf{ITE}(&\\
&\mathsf{ITE}(x, \mathsf{false}, \mathsf{true}),\\
&\mathsf{true},\\
&\mathsf{ITE}(\\
&\quad \mathsf{ITE}(y, \mathsf{false}, \mathsf{true}),\\
&\quad \mathsf{true},\\
&\quad \mathsf{ITE}(z, \mathsf{false}, \mathsf{true})\\
&)\\
)&
\end{aligned}
$$

Jede Klausel wird in eine ITE-Formel übersetzt, welche nur um einen konstanten Faktor größer ist. Entsprechend übersetzt sich jede 3KNF-Formel in eine semantisch äquivalente ITE-Formel, die nur um einen konstanten Faktor größer ist. Die Korrektheit folgt aus der Verwendung semantisch-äquivalenten Umformungen.

- $\in$ **NP**: Eine erfüllende Belegung ist ein geeignetes Zertifikat, das es maximal genauso groß ist wie die Formel F und in polynomieller Zeit geprüft werden kann.

Übungsaufgabe Ü12.5. (*ZOLP*)

Das *Zero-One-Linear-Program* (ZOLP) Problem sei wie folgt definiert:

- **Eingabe**: Ein System von linearen Ungleichungen

$$
\begin{aligned}
b_1 &\leq a_{1,1}y_1 + \ldots + a_{1,n}y_n\\
b_2 &\leq a_{2,1}y_1 + \ldots + a_{2,n}y_n\\
&\vdots\\
b_m &\leq a_{m,1}y_1 + \ldots + a_{m,n}y_n
\end{aligned}
$$

  mit $a_{i,j}, b_i \in \mathbb{Z}$ und $m, n > 0$.

- **Frage**: Gibt es für die Variablen $y_1, \ldots, y_n$ Werte aus $\{0, 1\}$, sodass alle Ungleichungen gleichzeitig erfüllt sind?

Zeigen Sie, dass das Entscheidungsproblem ZOLP NP-schwer ist. Geben Sie hierzu eine geeignete Reduktion von 3KNF-SAT auf ZOLP an und beschreiben Sie **zusätzlich** das Vorgehen anhand folgender Formel:

$$(\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_3 \vee x_4)$$

Den Korrektheitsbeweis müssen Sie nicht ausformulieren.

Lösungsskizze.

- *Möglichkeit 1: Für jede Variable im Ausgangsproblem zwei Variablen im Zielproblem.* Sei $F := \bigwedge_{i=1}^{k} C_i$ und $C_i := \bigvee_{j=1}^{3} L_{i,j}$ mit z Variablen die Eingabe für das 3KNF-SAT Problem. Die Reduktion f modelliert jede Variable des SAT-Problems

mit zwei Variablen des ZOLP-Problems. Sei x_i eine Variable dann encodiert y_{2i}, ob das Literal x_i wahr ist, und y_{2i-1}, ob das Literal $\neg x_i$ wahr ist. Somit haben wir

$$f(x_i) \mapsto y_{2i} \qquad f(\neg x_i) \mapsto y_{2i-1}$$

Für das ZOLP wählen wir $m = k + 2z$ und $n = 2z$. Die Zeile $i \in [1, k]$ ist dann definiert in Abhängigkeit von C_i als

$$f(L_{i,1} \vee L_{i,2} \vee L_{i,3}) \qquad \mapsto \qquad 1 \leq f(L_{i,1}) + f(L_{i,2}) + f(L_{i,3})$$

Zusätzlich fügen wir folgende Konsistenzgleichungen ein, damit y_{2i} und $y_{2i} - 1$ unterschiedliche Werte erhalten für alle $i \in [1, z]$

$$\begin{aligned} 1 &\leq y_{2i} + y_{2i-1} \\ -1 &\leq -y_{2i} - y_{2i-1} \end{aligned}$$

Die Reduktion ist polynomiell in der Eingabegröße, da für jede Klausel eine Ungleichung und für jede Variable 2 Ungleichungen erstellt werden.

Anwendung auf Beispiel $(\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_3 \vee x_4)$:

$$\begin{aligned} 1 &\leq y_1 + y_4 + y_6 \\ 1 &\leq y_2 + y_3 + y_8 \\ 1 &\leq y_1 + y_2 \\ -1 &\leq -y_1 - y_2 \\ 1 &\leq y_3 + y_4 \\ -1 &\leq -y_3 - y_4 \\ 1 &\leq y_5 + y_6 \\ -1 &\leq -y_5 - y_6 \\ 1 &\leq y_7 + y_8 \\ -1 &\leq -y_7 - y_8 \end{aligned}$$

- *Möglichkeit 2: Für jede Variable im Ausgangsproblem eine Variable im Zielproblem. Skizze:* Wir kodieren jede Klausel als eine Ungleichung im Zielproblem. Disjunktionen ersetzen wir mit Addition. Literale x_i werden in einen Summanden y_i übersetzt und Literale $\neg x_i$ in einen Summanden $1 - y_i$. Damit ist ein Literal l erfüllt genau dann wenn der entsprechende Summand den Wert 1 annimmt. Um sicherzustellen, dass jede Klausel erfüllt ist, steht auf der linken Seite jeder Ungleichung 1. Offensichtlich kann eine jede Ungleichung dieser Form in die von ZOLP geforderte Form gebracht werden.

  Anwendung auf Beispiel $(\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_3 \vee x_4)$:

$$\begin{aligned} 1 &\leq (1 - y_1) + y_2 + y_3 \\ 1 &\leq y_1 + (1 - y_3) + y_4 \end{aligned}$$

  Umformen:

$$\begin{aligned} 0 &\leq -y_1 + y_2 + y_3 \\ 0 &\leq y_1 - y_3 + y_4 \end{aligned}$$