Lehrstuhl für Informatik 1
Prof. Dr. Gerhard Woeginger
Jan Böker, Tim Hartmann

WS 2018/19

Übung zur Vorlesung
Berechenbarkeit und Komplexität

Lösung Blatt 10

Hausaufgabe 10.1 (3 Punkte)

Zeigen Sie für das Traveling Salesman Problem (TSP), dass, falls die Entscheidungsvariante in P ist, so kann auch die Optimierungsvariante in polynomialer Zeit gelöst werden.

Wir nehmen also an, dass die Entscheidungsvariante von TSP in P ist und konstruieren einen Polynomialzeit-Algorithmus $\mathcal{A}_{\text{OPT}}$, der eine optimale Lösung des TSP berechnet und dabei den Polynomialzeit-Algorithmus für die Entscheidungsvariante $\mathcal{A}_{\text{ENT}}$ als Unterprogramm benutzt.

Als Eingabe erhält der Algorithmus $\mathcal{A}_{\text{OPT}}$ einen gewichteten vollständigen Graphen G mit n Knoten. Wir bezeichnen mit $d_{\max}$ das maximale Kantengewicht in G. Jede Rundreise in G hat höchstens Kosten von $nd_{\max}$. Folglich können wir eine Kante aus G „löschen“, indem wir ihre Kosten auf $nd_{\max} + 1$ setzen.

Wir können durch eine binäre Suche den Wert $b_{opt} \leq nd_{\max}$ der optimalen Lösung bestimmen und damit b_{opt} in Polynomialzeit finden. Die Laufzeit der binären Suche ist $\mathcal{O}(\log_2(n \cdot d_{\max})) = \mathcal{O}(\log_2(N \cdot 2^N)) = \mathcal{O}(N)$ für ursprüngliche Eingabegröße N.

Der Algorithmus $\mathcal{A}_{\text{OPT}}$ zur Berechnung der optimalen Rundreise geht nun wie folgt vor:

- $\mathcal{A}_{\text{OPT}}$ iteriert über alle Kanten e des Graphen G.
- Dabei „löscht“ $\mathcal{A}_{\text{OPT}}$ die Kante e und prüft mit Hilfe von $\mathcal{A}_{\text{ENT}}$, ob es eine Rundreise mit Kosten b_{opt} gibt.
  - Falls ja, dann gibt es eine optimale Rundreise, die nicht über e führt. $\mathcal{A}_{\text{OPT}}$ „löscht“ die Kante e aus dem Graphen und wählt eine andere Kante, bis nur noch n Kanten vorhanden sind.
  - Falls nein, dann führt jede optimale Rundreise über die Kante e und die Kante e wird nicht gelöscht. $\mathcal{A}_{\text{OPT}}$ wählt nun eine andere Kante.

Nachdem über alle Kanten iteriert wurde, sind nur noch die Kanten einer optimalen Rundreise im Graphen vorhanden. $\mathcal{A}_{\text{OPT}}$ gibt diese als optimale Lösung aus.

Jede Kante wird höchstens einmal aus G entfernt. Dabei wird jeweils der Algorithmus $\mathcal{A}_{\text{ENT}}$ einmal aufgerufen mit Eingabegröße $\mathcal{O}(N + \log(n + d_{\max})) = \mathcal{O}(2N)$. Folglich ist die Laufzeit von $\mathcal{A}_{\text{OPT}}$ polynomiell in der Eingabelänge beschränkt.

Hausaufgabe 10.2 **(1+4 Punkte)**

Eine Knotenmenge $R \subseteq V$ spannt eine Kante $\{u, v\} \in E$ auf, falls $u \in R$ und $v \in R$. Wir betrachten folgendes Entscheidungsproblem.

Kanten Aufspannen
Eingabe: Ein Graph $G = (V, E)$; zwei Zahlen r und s.
Frage: Gibt es eine Menge $R \subseteq V$ mit $\|R\| = r$, die mindestens s Kanten aufspannt?

(a) Zeigen Sie, dass Kanten Aufspannen in NP liegt.

Wir verwenden eine geeignete Kodierung der Knotenmenge $R \subseteq V$ als Zertifikat. Dieses hat polynomielle Größe.

Der Verifizierer arbeitet wie folgt: Prüfe ob $|R| = r$, sonst verwerfe. Zähle die Kanten $\{u, v\} \in E$ mit $u, v \in R$. Akzeptiere, falls mindestens s Kanten gezählt worden sind, sonst verwerfe. Die Laufzeit des Verifizierers ist polynomiell.

(b) Zeigen Sie, dass Clique $\leq_p$ Kanten Aufspannen. Wie bald in der Vorlesung gezeigt wird, gilt SAT $\leq_p$ Clique. Was folgt daraus für Kanten Aufspannen?

Konstruktion: Sei Graph $G = (V, E)$ und natürliche Zahl k eine Clique-Instanz. Setzte $r := k$, $s := \binom{k}{2}$ und übernehme den Graphen G.

Die Konstruktion benötigt polynomielle Zeit.

Korrektheit: Wir zeigen, dass G eine Clique der Größe mindestens k hat gdw. G eine Knotenmenge $R \subseteq V, |R| = k$ hat welche mindestens $\binom{k}{2}$ Kanten aufspannt.

($\Rightarrow$) Sei $W' \subseteq V$ eine Clique der Größe mindestens k in G. Dann ist $W \subseteq W', |W'| = k$ eine k-Clique. Dann spannt W jede Kante $\{u, v\}$ mit $u, v \in W, u \neq w$ auf, also $\binom{k}{2}$ Kanten. Also ist $R := W$ eine Knotenmenge der Größe k, welche $\binom{k}{2}$ Kanten aufspannt.

($\Leftarrow$) Sei $R \subseteq V, |R| = k$ eine Knotenmenge, welche mindestens $\binom{k}{2}$ Knoten aufspannt. Es gibt höchstens $\binom{k}{2}$ viele Kanten mit Knoten aus R. Da R mindestens $\binom{k}{2}$ Knoten aufspannt, gilt für alle $u, v \in R, u \neq v$, dass $\{u, v\} \in E$. Demnach ist R eine Clique der Größe k.

Zusammen mit SAT $\leq_p$ Clique folgt, dass SAT $\leq_p$ Kanten Aufspannen, also Kanten Aufspannen NP-schwer ist. Zusammen mit a) folgt, dass Kanten Aufspannen NP-vollständig ist.

Hausaufgabe 10.3 **(1+4 Punkte)**

Für Vektoren $c, d \in \mathbb{Z}^k$ sei $c \geq d$ falls für alle $i \in \{1, \ldots, k\}$ gilt, dass $c_i \geq d_i$. Wir betrachten folgendes Entscheidungsproblem:

$\{-1, 0, 1\}$ Restricted Integer Programing
Eingabe: Eine Matrix $A \in \{-1, 0, 1\}^{m \times n}$ und ein Vektor $b \in \{-1, 0, 1\}^m$.
Frage: Gibt es einen Vektor $x \in \{0, 1\}^n$ mit $Ax \geq b$?

(a) Zeigen Sie, dass $\{-1, 0, 1\}$ RESTRICTED INTEGER PROGRAMING in NP liegt.

Als Zertifikat verwenden wir $x \in \{0,1\}^n$. Dieses hat polynomielle Größe in der Eingabegröße.

Der Verifzierer prüft ob $Ax \geq b$ in polynomieller Zeit.

(b) Zeigen Sie, dass $\{-1, 0, 1\}$ RESTRICTED INTEGER PROGRAMING NP-schwer ist.

Hinweis: Es bietet sich eine Reduktion von* SAT *an. Außerdem ist hilfreich als Zwischenschritt auch Gleichungen der Art $c+d=1$ zu erlauben. Daraufhin kann man sich überlegen, wie man eine solche Gleichung in Ungleichungen übersetzten kann.

Wir zeigen, dass SAT $\leq_p \{-1,0,1\}$RESTRICTED INTEGER PROGRAMING:

(Idee: Repräsentiere eine Varibele v_i durch eine i-te Spalten für x und eine $(n+i)-te$ Spalte für $\overline{x}$. Wir erzwingen die Belegung von x so, dass $x_i \neq x_{n+i}$, sprich $x \neq \overline{x}$ gilt. Für eine Klausel der Form $v_1 + \overline{v_2} + v_3$ muss dann gelten $x_1 + x_{n+2} + x_3$.)

Konstruktion: Gegeben sei eine 3-SAT-Instanz bestehend aus einer Formel φ in CNF über Variablen $v_1, \ldots, v_{n'}$. Wir konstruieren eine Matrix A mit $2n'$ Spalten und Vektor b welche folgende Ungleichungen abbilden. Für jede Variable $v_i, i \in \{1, \ldots, n'\}$, fügen wir die Ungleichungen $x_i + x_{n'+i} \geq 1$ und $(-x_i) + (-x_{n'+i}) \geq -1$ ein, so dass $x_i + x_{n'+i} = 1$ gilt. Für jede Klausel der Form $v_{i_1} \vee \cdots \vee v_{i_{k'-1}} \vee \overline{v_{i_{k'}}} \vee \cdots \vee \overline{v_{i_k}}$ füge Ungleichung $x_{i_1} + \cdots + x_{i_{k'-1}} + x_{n'+i_{k'}} + \cdots + x_{n'+i_k} \geq 1$.

Die Laufzeit der Konstruktion ist polynomiell.

Korrektheit:

($\Rightarrow$) Sei φ in CNF Form mit erfüllender Belegung $I : \{v_1, \ldots, v_n\} \to \{0,1\}$. Sei $x_i := I(v_i)$ und $x_{n+i} = 1 - I(v_i)$ für alle $i \in \{1, \ldots, n'\}$. Für jede Klausel $v_{i_1} \vee \cdots \vee v_{i_{k'-1}} \vee \overline{v_{i_{k'}}} \vee \cdots \vee \overline{v_{i_k}}$ gibt es ein positives oder ein negatives Literal v_i bzw. $\overline{v_i}$, welches wahr ist mit Belegung I. Dann ist die entsprechende Ungleichung erfüllt, da $x_i = 1$ bzw. $x_{n'+i} = 1$ ist und alle anderen Summanden ≥ 0.

($\Leftarrow$) Sei $x \in \mathbb{N}^{2n'}$, so dass $Ax = b$, also alle Ungleichungen aus der Konstruktion erfüllt sind. Dann gilt, dass $x_i \neq x_{n+i}$, da $x_i + x_{n'+i} = 1$. Wir zeigen, dass die Variablenbelegung $I : \{v_1, \ldots, v_{n'}\} \to \{0,1\}, v_i \mapsto x_i$ eine erfüllende Belegung für die ursprüngliche Formel φ ist. Für jede Klausel $v_{i_1} \vee \cdots \vee v_{i_{k'-1}} \vee \overline{v_{i_{k'}}} \vee \cdots \vee \overline{v_{i_k}}$ gilt, dass $x_{i_1} + \cdots + x_{i_{k'-1}} + x_{n'+i_{k'}} + \cdots + x_{n'+i_k} \geq 1$ erfüllt ist. Es gilt also, dass $v_j = 1$ für ein $j \in \{1, \ldots, k'-1\}$ oder $v_{n+j'} = 1$ für ein $j' \in \{k', \ldots, k\}$. Dann ist $I(v_j) = 1$ bzw., da $x_i \neq x_{n'+i}$, ist $I(v_{j'}) = 0$, und daher ist die Klausel erfüllt.