Informatik ↄ: – Blatt 11

Rasmus Diederichsen

20. Juli 2014

Aufgabe 11.1

Der Beweis ist Bullshit. Es wird reduziert von dem in Polynomialzeit lösbaren Problem Eulerkreis auf das Problem Hamiltonkreis. Dies zeigt aber nur, das Hamiltonkreis mindestens so schwer ist wie Eulerkreis, was nicht überrascht, da Hamiltonkreis *NP*-vollständig ist. Um zu zeigen, dass Eulerkreis *NP*-vollständig ist, müsste man aber zeigen, dass sich Hamiltonkreis in polynomiell-deterministischer Zeit auf Eulerkreis reduzieren lässt. Dann wäre bewiesen, dass Eulerkreis mindestens so schwer ist wie Hamiltonkreis.

Aufgabe 11.2

a)

Gegeben einen Zeugen für die Lösbarkeit einer Instanz (also einen Weg, dessen Länge angeblich $\leq k$ ist und der alle Knoten genau einmal enthält), ist es einfach in polynomieller Zeit, nachzuprüfen ob

1. er wirklich kumulierte Kosten $\leq k$ hat (in Linearzeit prüfbar)
2. er alle Knoten enthält (ebenfalls in Linearzeit prüfbar)

b)

Wie wissen (s.o), dass Hamiltonkreis *NP*-vollständig ist. Wir müssen also zeigen, dass sich jede Probleminstanz von Hamiltonkreis auf eine Instanz von TSP reduzieren lässt.

Gegeben einen Graphen $G = (V, E)$, $|V| = n$, für den ein Hamilton-Kreis gefunden werden soll, können wir die Problemstellung umformulieren. Sei $G' = (V', E')$ ein vollständiger Graph mit $V' = V$ und $E' = \{(u, v) \mid u, v \in V', u \neq v\}$. Es sind also alle Vertices miteinander verbunden, auch wenn es zwischen ihnen in G keine Kante gab. Die Kantenkosten seien durch die Funktion

$$d(e = (u, v) \in E') = \begin{cases} 1 & \text{falls } (u, v) \in E \\ 2 & \text{sonst} \end{cases}$$

gegeben. Die Fragestellung lautet nun

> Gibt es einen Weg in G', der jeden Knoten genau einmal besucht und dessen kumulierte Kosten ($\leq$) n sind?

c)

Es bleibt zu zeigen

> G enthält einen Hamiltonkreis $\iff$ G' enthält eine Rundtour $\leq n$.

"$\Rightarrow$" Falls G einen Hamiltonkreis besitzt, dann enthält dieser jeden Knoten genau einmal und derselbe Kreis existiert in G'. Da in G' alle Kanten aus G die Kosten 1 besitzen, hat die Rundtour in G' die Kosten maximal n, da maximal n Kanten zwischen n Knoten besucht werden können.

"$\Leftarrow$" Wenn es in G' eine Rundtour mit Kosten $\leq n$ gibt, so sind die kumulierten Kosten die Summe aus n Kantenkosten (damit ein Kreis entstehen kann). Es muss also jede Kante die Kosten 1 haben, da ansonsten $\sum_{i=1}^{n} c_i \leq n$ nicht erfüllbar ist mit $c_i \in \{1, 2\}$. Dies wiederum bedeutet aber (nach Definition von d oben), dass derselbe Kreis auch in G existiert.

□

Aufgabe 11.3

a)

Das Problem lässt sich in Polynomialzeit per Brute-Force-Search lösen.

Algorithmus für CLIQUE-Lösung

```
1  def find_clique (graph, k):
2     for g in graph.subgraphs(k): # get subgraphs of size <= k
3        found = true
4        for (v1, v2) in [(u,v) for u in g.vertices for v in g.
             vertives if u != v]:
5           if !g.hasEdge(v1, v2):
6              found = false
7              break # skip rest of subgraph
8        if found:
9           return true, g
10    return false
```

b)

Es gibt in einem Graphen mit Maximalgrad k und Knotenzahl n nur n^k (bzw. $\frac{n!}{(n-k-1)!}$, wenn die Clique maximal sein soll) viele Teilgraphen, die theoretisch eine Clique bilden können, da diese nur k Knoten enthalten kann (weil sie vollständig verbunden sein muss). In der Clique muss jeder der k Knoten k Kanten besitzen und diese müssen ihn mit den anderen Knoten in der Clique verbinden. Um die Verbundenheit zu prüfen, muss man also k^2 Kanten betrachten. Der Algorithmus läuft also in $\mathcal{O}\left(n^k k^2\right)$, in diesem Fall $\mathcal{O}\left(n^5\right)$.

Für jede Konstante k läuft der Algorithmus immer in Polynomialzeit.

□

Aufgabe 11.4

Wir können zumindest die Vermutung anstellen, dass der Beweis prinzipiell nicht funktionieren kann. Damit NonSat in *NP* ist, muss ein Zeuge für die Nichterfüllbarkeit einer Formel in polynomiell-deterministischer Zeit verifizierbar sein. Die Frage ist, wie so ein Zeuge überhaupt aussehen kann. Meines Erachtens ist die einzige Art von Zeuge, die die Nichterfüllbarkeit beweist, eine Enumeration sämtlicher möglicher Variablenbelegungen mit dem Wert, zu dem die Formel mit der Belegung evaluiert (nämlich `false`). Dieser Zeuge müsste bei n Variablen die Länge $k \cdot 2^n$ haben. Ein exponetieller Zeuge ist nicht in polynomieller Zeit verifizierbar. Das Problem kann also nicht in *NP* sein (es sei denn, es gibt irgendeinen schlaueren Weg, einen Zeugen zu spezifizieren).

Aufgabe 11.5

a)

Sei $F = \{f_1, \ldots, f_m\}$ die Menge aller Farben. Seien $H_1, \ldots, H_n \subseteq F$ Mengen mit je höchstens drei Elementen. Gibt es eine Teilmenge $F_{\text{schön}} \subseteq F$ der Farben, sodass $\forall i \leq n : |F_{\text{schön}} \cap H_i| = 1$?

b)

$$[[x_i \vee x_j \vee x_k]]$$

c)

Asterix ist in *NP*

Offensichtlich ist in polynomialzeit feststellbar, ob jede Farbe maximal einmal pro Haufen vorkommt.

Asterix ist *NP*-vollständig

Wir reduzieren eine beliebige 3-SAT-Instanz $\mathcal{F}$ auf Asterix wie folgt.

1. Für jedes Literal x_i in $\mathcal{F}$, führe Literal x_i' ein und ersetze jedes Vorkommen von $\neg x_i$ durch x_i'. Erzeuge für jede Ersetzung eine Klausel $[[x_i \vee x_i']]$.

2. Für jede Klausel in $\mathcal{F}$ mit 3 Literalen x_i, x_j, x_k, kreiere eine Klausel

$$[[x_i' \vee a \vee b]] \wedge [[x_j \vee b \vee c]] \wedge [[x_k' \vee b \vee d]],$$

   wobei a, b, c, d neue Variablen sind.

3. Für jede Klausel in $\mathcal{F}$ mit zwei Literalen x_i, x_j, führe eine neue Variable e ein, sowie $e' := \neg e$. Schreibe die Klausel um zu

$$(x_i \vee x_j \vee e) \wedge (x_i \vee x_j \vee e').$$

   Diese hat dieselben Wahrheitsbedingungen wie zuvor. Transformiere nun wie oben zu

$$[[x_i' \vee a \vee b]] \wedge [[x_j \vee b \vee c]] \wedge [[e' \vee b \vee d]] \wedge [[e \vee e']],$$

   wobei a, b, c, d neue Variablen sind.

4. Für jede Klausel in $\mathcal{F}$ mit einem Literal, kreiere eine Klausel

$$[[x_i]]$$

5. Verbinde alle Klauseln mit $\wedge$.

Die Umformungen sind allesamt wahrheitserhaltend, daher ist die 3-SAT-Instanz lösbar genau dann wenn, die ASTERIX-Instanz lösbar ist.

□

Aufgabe 11.6

a)

Definition von SetCover

Das SETCOVER-Problem formuliert als Entscheidungsproblem ist wie folgt definiert:

> Gegeben ein Universum $\mathcal{U} = \{1, 2, \ldots, m\}$, eine Menge von Mengen $\mathcal{S} = \{S_1, \ldots, S_n\}$ mit $\bigcup_{i=1}^{n} S_i = \mathcal{U}$ und ein $k \in \mathbb{N}$, gibt es eine Teilmenge $\mathcal{C} = \{C_1, \ldots, C_k\} \subseteq \mathcal{S}$ mit $\bigcup_{i=1}^{k} C_i = \mathcal{U}$ und $|\mathcal{C}| \leq k$?

Komplexität von SetCover

Wir zeigen, dass SETCOVER *NP*-vollständig ist.

1. Gegeben ein Zeuge $\mathcal{Z} = \{S_1, \ldots, S_k\} \subseteq \mathcal{S}$, für jedes S_i und für jedes $l \in S_i$ entferne l aus $\mathcal{U}$. Am Ende prüfe, ob $\mathcal{U} = \emptyset$. Da die n S_i jeweils maximal m Elemente enthalten können und die Suche in $\mathcal{U}$ linearen Aufwand bedeutet, sind die Gesamtkosten im Worst Case $\mathcal{O}(n \cdot m^2)$, also polynomiell.

2. Wir zeigen die Härte durch Reduktion von VertexCover. Sei $VC = (G = (V, E), k)$ eine beliebige VertexCover-Instanz. Erstelle eine SetCover-Instanz $SC = (\mathcal{U}, \mathcal{S}, k)$ mit $\mathcal{U} = \{\{u, v\} \mid (u, v) \in E\}$ und $\mathcal{S} = \{S_i \mid S_i = \{\{u, v\} \mid V_u = V_i \vee V_v = V_i\}\}$. $\mathcal{S}$ ist also eine Menge, die für jeden Knoten die Menge der von ihm ausgehenden Kanten enthält. Diese Reduktion ist offensichtlich schlimmstenfalls quadratisch in $|V|$.

   Zu zeigen ist:

   SetCover ist erfüllbar $\Leftrightarrow$ VertexCover erfüllbar.

   "$\Rightarrow$" Sei $\mathcal{C}$ eine Lösung der SetCover-Instanz für k. Dann enthält $\mathcal{C} \subseteq \mathcal{S}$ alle Kanten von G, da $\mathcal{U}$ alle Kanten enthält und von $\mathcal{C}$ vollständig abgedeckt wird.

   Jedes S_i entspricht genau einem $V_i \in V$. Die Menge $\{V_i \mid S_i \in \mathcal{C}\}$ ist also ein Vertex Cover für G mit k Knoten.

   "$\Leftarrow$" Sei $K \subseteq V$ ein Vertex Cover für G mit $|K| = k$. Dann ist jede Kante in G inzident zu einem Knoten in K. Konstruieren wir also für jeden Knoten in K die Menge seiner inzidenten Kanten und vereinigen diese Kantenmengen, erhalten wir gerade E, was äquivalent zu $\mathcal{U}$ ist (s.o.). Die Menge $\mathcal{K} = \{S_i \mid S_i = \{\{u, v\} \mid (u, v) \text{ inzident zu } v_i \in K\}\}$ ist also gerade ein Set Cover für $\mathcal{U}$ mit $|\mathcal{K}| = k$.

□

b)

Das Kgb-Problem lässt sich leicht in ein SetCover-Problem umformulieren. Es sei

- $\mathcal{U}$ die Menge aller Studierenden, nummeriert von 1 bis n,
- $\mathcal{S} = \{S_i = \{s \mid \text{ Student/in } s \text{ hat Zeit an Termin } i\}\}$ die Menge von Mengen, die jeweils die Studierenden umfassen, die sich für einen gegebenen Termin eingetragen haben.

Offensichtlich ist das Problem genau dann gelöst, wenn es eine Teilmenge $\mathcal{C} \subseteq \mathcal{S}$ gibt, sodass alle Studierenden aus $\mathcal{U}$ abgedeckt sind durch die Auswahl der in $\mathcal{C}$ enthaltenen Termine und $|\mathcal{C}| \leq k$.

□