Theoretische Grundlagen der Informatik
Tutorium XII & XIII (SR 301)

Tut Nr. 11 – Üb9, Reduktionen

David Münch

Karlsruher Institut für Technologie
Institut für Informatik
IKS Müller-Quade

21. Januar 2010

Karlsruher Institut für Technologie

Inhaltsverzeichnis

1 Auftakt

Inhaltsverzeichnis

1. Auftakt
2. Lernziele

Inhaltsverzeichnis

1. Auftakt
2. Lernziele
3. Themen
   - Übungsblatt 10
   - $\mathcal{NP}$-Vollständigkeit

Inhaltsverzeichnis

1. Auftakt

2. Lernziele

3. Themen
   - Übungsblatt 10
   - $\mathcal{NP}$-Vollständigkeit

4. Abspann

Organisatorisches

Email: muenchdavid@gmail.com

`https://www.stud.uni-karlsruhe.de/~uhbro/`

Tutorium 12: Donnerstags 8:00 Uhr - Raum 301
Tutorium 13: Donnerstags 9:45 Uhr - Raum 301

Übungsblattabgabe Mittwochs 12:00 Uhr.

Organisatorisches

Deckblatt benutzen: `http://www.stud.uni-karlsruhe.de/~unbdh/deckblatt/index.php?course=6`

Gruppenarbeit erwünscht, aber jeder muss handschriftliche Lösung mit Namen aller Gruppenteilnehmer abgeben.

50% der Punkte sind notwendig für den Schein.

Organisatorisches

Nicht abgeholte Übungsblätter können ab sofort bei Nico Döttling R274 abgeholt werden.

`https://puck.iaks.uka.de/eiss/?id=167`

Wichtig!

Es gibt zum neuen Thema ein vorlesungsbegleitendes Skriptum unter: `https://puck.iaks.uka.de/eiss/fileadmin/User/Info_3/skript.pdf`.

Was wollen wir heute erreichen?

Was wollen wir heute erreichen?

- Übungsblatt 9

Was wollen wir heute erreichen?

- Übungsblatt 9
- NP-Vollständigkeits-Beweise

Aufgabe 1

Sei $G = (V, E)$ ein gerichteter Graph. Ein *Kern* von G ist eine Teilmenge $K \subseteq V$, sodass

- Für je zwei $u \neq v$ in K ist weder $(u, v) \in E$ noch $(v, u) \in E$.
- Für jedes $u \notin K$ gibt es ein $v \in K$, sodass $(v, u) \in E$.

Wir erhalten das folgende Problem:

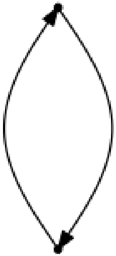
KERNEL

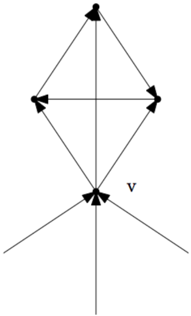
Gegeben: Ein gerichteter Graph $G = (V, E)$

Frage: Hat G einen Kern?

Zeigen Sie: KERNEL ist NP-vollständig. Verwenden Sie dazu die NP-Vollständigkeit von 3SAT.

Hinweis: Verwenden Sie in ihrer Reduktion für die in der 3SAT-Instanz auftauchenden Variablen Teilgraphen wie in der linken Abbildung. Dieser Teilgraph hat genau zwei "Zustände" (oder Belegungen), da immer nur einer der beiden Knoten in einem Kern enthalten sein kann. Verwenden Sie weiter, dass sich für einen Knoten v die Mitgliedschaft im Kern erzwingen läßt, indem man ihn mit einem wie in der rechten Abbildung dargestellten Gadget ausstattet.





Aufgabe 2

Gegeben ist das folgende Problem:

CLIQUE

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und eine Zahl k.
Frage: Besitzt G einen vollständigen Subgraphen mindestens der Größe k (also einen Subgraph $G' = (V', E')$ mit $|V'| \geq k, V' \subseteq V, E' \subseteq E$ und $\forall v_i, v_j \in V' : (v_i, v_j) \in E'$)?

Zeigen Sie, dass CLIQUE NP-vollständig ist.

Aufgabe

Zeige, dass TRAVELING SALESMAN $\mathcal{NP}$-vollständig ist.
Reduziere hierbei von HAMILTONIAN CIRCUIT.

Aufgabe

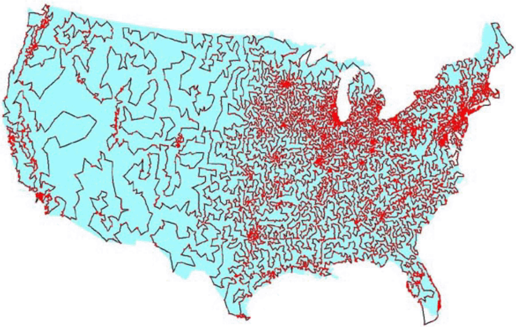
Problem **HAMILTONIAN CIRCUIT**:
Gegeben: Ein Graph $G = (V, E)$
Frage: Existiert ein einfacher Kreis in G, der alle Knoten in V enthält, also eine Folge $(v_1, \ldots, v_n)$ von paarweise verschiedenen Knoten $v_i \in V (i = 1, \ldots, n)$ mit $n := |V|$ und $\{v_j, v_{j+1}, \{v_n, v_1\} \in E(j = 1, \ldots, n-1)\}$?
Hinweis: HAMILTONIAN CIRCUIT ist $\mathcal{NP}$-vollständig.

Aufgabe

Problem **HAMILTONIAN CIRCUIT**:
Gegeben: Ein Graph $G = (V, E)$
Frage: Existiert ein einfacher Kreis in G, der alle Knoten in V enthält, also eine Folge $(v_1, \ldots, v_n)$ von paarweise verschiedenen Knoten $v_i \in V (i = 1, \ldots, n)$ mit $n := |V|$ und $\{v_j, v_{j+1}, \{v_n, v_1\} \in E(j = 1, \ldots, n-1)\}$?
Hinweis: HAMILTONIAN CIRCUIT ist $\mathcal{NP}$-vollständig.

Problem **TSP**:
Gegeben: Ein Graph $G = (V, E)$ und Parameter k
Frage: Existiert ein einfacher Kreis in G, der alle Knoten in V enthält, also eine Folge $(v_1, \ldots, v_n)$ von paarweise verschiedenen Knoten mit Kostenfunktion $c \leq k$?

Beispiel:



Rundreise durch die USA mit Besuch von jedem Ort, mit mehr als 500 Einwohnern.

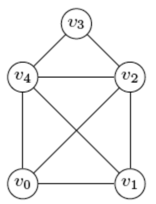
1. Schritt: zu zeigen: **TSP** liegt in $\mathcal{NP}$
Rate Lösung (Folge von Knoten) und überprüfe. $\mathcal{O}(n)$.

1. Schritt: zu zeigen: **TSP** liegt in $\mathcal{NP}$
Rate Lösung (Folge von Knoten) und überprüfe. $\mathcal{O}(n)$.

2. Schritt: **HAMILTONIAN CIRCUIT** $\leq$ **TSP**
Instanz von **HC**: $G = (V, E), V = \{v_1, \ldots, v_n\}$
Die korrespondierende Instanz von **TSP** enthält n Orte, die Gewichte c_{ij} sind 1, wenn $(v_i, v_j) \in E$ und $1 + \alpha$ sonst.
Kostengrenze k ist n.
Offensichtlich ist diese Transformation polynomial.
Damit gilt:
G enthält einen **HC** genau dann, wenn c eine Rundreise mit Kosten n zulässt. Sei $(v_{i_1}, \ldots, v_{i_n}, v_{i_1})$ ein **HC** in G, dann sind dessen Kosten gleich n.
Sei $(v_{i_1}, \ldots, v_{i_n}, v_{i_1})$ eine Rundreise mit Kosten n, dann gilt für $i = 1, \ldots, n-1$, dass $(v_i, v_{i+1}) \in E$ und $(v_n, v_1) \in E$. Damit ist $(v_{i_1}, \ldots, v_{i_n}, v_{i_1})$ ein **HC** in G.

Aufgabe

Gegeben sei folgender Graph:



Gibt es einen Hamiltonkreis? Wandeln Sie hierzu das Problem in ein TSP um und finden Sie eine optimale Rundtour.

Lösung:

Eine optimale Rundtour ist z.B.: $(v_0, v_1, v_2, v_3, v_4, v_0)$ mit Kosten 5, also ist diese optimale Rundtour (da ihre Kosten nicht grösser als 5 sind) auch ein Hamiltonkreis.

Aufgabe

Betrachte das Problem $\mathrm{HALFCLIQUE}$:
Gegeben: Graph $G = (V, E)$ mit $|V|$ gerade.
Frage: Gibt es eine Clique mindestens der Grösse $|V|/2$ in G?
Zeige, dass $\mathrm{HALFCLIQUE}$ $\mathcal{NP}$-vollständig ist.

Aufgabe

Betrachte das Problem HALFCLIQUE:
Gegeben: Graph $G = (V, E)$ mit $|V|$ gerade.
Frage: Gibt es eine Clique mindestens der Grösse $|V|/2$ in G?
Zeige, dass HALFCLIQUE $\mathcal{NP}$-vollständig ist.

Problem **CLIQUE**:
Gegeben: Ein Graph $G = (V, E)$ und einen Parameter $k \leq |V|$
Frage: Gibt es in G eine Clique der Grösse mindestens k? Hinweis: CLIQUE ist $\mathcal{NP}$-vollständig.

Aufgabe

The problem is NP-complete.

To show that it is in NP, we give a non-deterministic polynomial-time algorithm for recognizing "yes" instances. This algorithm non-deterministically guesses a set of $n/2$ vertices, and then checks whether it is a clique.

To show that it is NP-hard, we give a reduction from the NP-complete problem **Clique**. Let (G, j) be an instance of **Clique**. (Check your lecture notes to remind yourself about this problem.) We will construct an instance G' of **HalfClique** such that (G, j) is a "yes" instance of **Clique** iff G' is a "yes" instance of **HalfClique**.

We do the construction in two cases. Let $V(G)$ be the set of vertices of G, so $|V(G)|$ is the number of vertices of G.

Aufgabe

Case 1: If $j \leq |V(G)|/2$ then we will construct a graph G' with $2|V(G)| - 2j$ vertices by adding $|V(G)| - 2j$ vertices to G. Connect each of these new vertices to every vertex in G' (except itself). We claim that G has a clique of size j iff G' has a clique of size $|V(G)| - j$. So (G, j) is a "yes" instance of **Clique** iff G' is a "yes" instance of **HalfClique**. Here is why the claim is true.

1. If G' has a clique of size $|V(G)| - j$ then at least j of these are from G and these are a clique.

2. If G has a clique of j then these plus the new vertices make a clique of size $|V(G)| - j$ in G'.

Case 2: If $j > |V(G)|/2$ then construct a graph G' with $2j$ vertices by adding $2j - |V(G)|$ new vertices and no new edges. We claim that G has a clique of size j iff G' has a clique of size j. So (G, j) is a "yes" instance of **Clique** iff G' is a "yes" instance of **HalfClique**.

Aufgabe

Zeigen Sie, dass die polynomielle Reduzierbarkeit transitiv ist.
Aus $A \leq_p B$ und $B \leq_p C$ folgt $A \leq_p C$.

Aus $A \leq_p B$ folgt, dass es eine Abbildung $f : A \to B$ gibt, die

$$w \in A \text{ genau dann, wenn } f(w) \in B$$

erfüllt. Der Aufwand, $f(w)$ zu ermitteln, sei durch das Polynom p beschränkt. Ebenso erhalten wir aus $B \leq_p C$, dass es eine Funktion $g : B \to C$ gibt, die

$$x \in B \text{ genau dann, wenn } g(x) \in C$$

erfüllt. Der Aufwand, $g(x)$ zu ermitteln, sei durch das Polynom q beschränkt. Aus den beiden Abbildungeigenschaften erhalten wir

$$w \in A \text{ geanu dann, wenn } g(f(w)) \in C$$

Bei der Aufwandabschätzung von $g \circ f$ beachten wir, dass der Aufwand aus der Summe des Aufwands für die Anwendung der beiden Abbildungen ist. Er wird beschränkt durch $p(|w|) + q(|f(w)|)$. Da die Summe zweier Polynome wieder ein Polynom ist, ist A polynomiell auf C reduzierbar.

Aufgabe

Finden Sie den Fehler im folgenden "Beweis" für **P** $\neq$ **NP**!
Betrachten Sie folgenden Algorithmus für SAT:

- Durchlaufe für die gegebene Formel ϕ alle möglichen Belegungen der Variablen mit den Wahrheitswerten
- Akzeptiere ϕ, wenn eine der durchlaufenen Belegungen ϕ erfüllt

Dieser Algorithmus hat eine mit der Anzahl der Variablen exponentiell wachsende Laufzeit. Daher hat das Problem SAT einen exponentiellen Aufwand und kann nicht in **P** liegen. Weil aber SAT in **NP** liegt, muß also **P** $\neq$ **NP** gelten.

Lösung:

Die Vorgabe eines Algorithmus mit exponentiellem Aufwand zur Lösung eines gegebenen Problems bedeutet nicht, dass das Problem eine exponentielle Komplexität besitzt und damit in **NP** liegt. Der Denkfehler besteht darin, dass aus der Tatsache, dass man nicht auf triviale Weise einen effizienten (polynomiellen) Algorithmus findet, geschlossen wird, dass ein solcher effizienter (polynomieller) Algorithmus auch nicht existieren kann.

Aufgabe

Zeigen Sie, dass es unter der Voraussetzung $\mathbf{P} = \mathbf{NP}$ möglich ist, für eine aussagenlogische Formel ϕ in polynomieller Zeit eine erfüllende Belegung der Variablen zu finden, falls eine solche Belegung existiert!

Lösung:

Es ist bekannt, dass SAT $\in$ **NP** gilt. Wegen der Voraussetzung **P** $=$ **NP** gilt damit auch SAT $\in$ **P**. Für eine gegebene aussagenlogische Formel ϕ seien die Variablen mit $X_1, ..., X_n$ für ein geeignetes $n \in \mathbb{N}$ bezeichnet.
Es wird nun folgender Algorithmus betrachtet:

Lösung:

1. Initialisiere i mit 1
2. Ersetze X_i in ϕ mit TRUE und prüfe, ob $\phi \in$ SAT gilt
- Falls ja: Gehe zu 4.
- Falls nein: Gehe zu 3.
3. Ersetze X_i in ϕ mit FALSE und prüfe, ob $\phi \in$ SAT gilt
- Falls ja: Gehe zu 4.
- Falls nein: Gib aus "ϕ ist nicht erfüllbar"
4. Prüfe, ob $i < n$ gilt
- Falls ja: Erhöhe i um 1 und gehe zu 2.
- Falls nein: Gib die aktuelle Belegung für $X_1, ..., X_n$ als Lösung aus

Der Aufwand des Algorithmus ist im wesentlichen das $2n$-fache des Aufwandes von SAT. Da SAT nach der Voraussetzung dieser Aufgabe einen polynomiellen Aufwand in n hat, so hat auch der obige Algorithmus einen polynomiellen Aufwand in n.

Reflexion

Was haben wir heute gelernt?

Reflexion

Was haben wir heute gelernt?

- Polynomiale many-one Transformierbarkeit

Noch Fragen?

Vorschau

Vorschau

- .

Vorschau

- .
- .

Bis zum nächsten Mal