Komplexität
Was ist leicht? Was ist schwer? Was geht gar nicht?

Kurt Mehlhorn

17. Dezember 2012

1 Probleme

Definition 1 *Ein Problem heißt* leicht*, wenn es eine Turingmaschine gibt, die es in Polynomzeit löst.*

Was ist ein Problem? Was heißt es, dass eine Turingmaschine ein Problem in Polynomzeit löst?

Ein Problem fasst Fragestellungungen (Instanzen) einen bestimmten Typs zusammen. Eine Instanz ist entweder eine JA-Instanz oder eine NEIN-Instanz. Alternativ, kann man die Menge der NEIN-Instanzen als das Komplement der JA-Instanzen definieren.

Primzahlproblem:
Eingabe/Instanz: eine natürliche Zahl n
Frage: Ist n eine Primzahl?

Kürzeste Wegeproblem:
Eingabe/Instanz: ein gerichteter Graph $G = (V,E)$, eine Längenfunktion $\ell : E \to \mathbb{N}$, zwei Knoten s und t, eine Zahl L.
Frage: gibt es einen Weg von s nach t der Länge $\leq L$?

Maximales Flußproblem:
Eingabe/Instanz: ein gerichteter Graph $G = (V,E)$, eine Längenfunktion $\cap : E \to \mathbb{N}$, zwei Knoten s und t, eine Zahl L.
Frage: gibt es einen Fluss von s nach t der Größe $\geq L$?

Graphendreifärbung:
Eingabe: ein Graph $G = (V,E)$
Frage: kann man die Knoten von V mit drei Farben so färben, daß die beiden Enden einer jeden Kante verschiedene Farben haben?

Hamiltonscher Pfad:
Eingabe: ein Graph $G = (V,E)$
Frage: Sei $n = |V|$. Gibt es eine Rundreise, die jeden Knoten genau einmal besucht, d.h. eine Anordnung $v_1, \ldots, v_n$ der Knoten, so dass $v_i \neq v_j$ for $i \neq j$ und $(v_i, v_{i+1}) \in E$ fur $1 \leq i \leq n$?

Erfüllbarkeit:
Eingabe: Eine boolesche Formel:
Frage: gibt es eine erfüllende Belegung?

Hier sind einige Anwendungen:

- Hamiltonscher Pfad: eine Logistikunternehmen plant die Touren für seine Lastwägen.
- Graphenfärbung: Wir möchten Vorlesungen auf die 10 möglichen Vorlesungsstunden (8:00 bis 17:00) packen.Die Vorlesungen sind die Knoten des Graphen. Zwei Vorlesungen sind durch eine Kante verbunden, wenn es einen Studenten gibt, der beide hören will.
- Maximaler Fluss: man will wissen, ob man ein Fussballstadium in 10 Minuten räumen kann.
- Primzahlproblem: man braucht große Primzahlen für die Kryptographie. Die einfachste Art sie zu finden ist: man würfelt die Zahl Ziffer für Ziffer mit einem Würfel mit 10 Seiten und testet dann, ob die Zahl prim ist.

2 Einfache Probleme

Definition 2 *1. Eine Turingmaschine M löst ein Problem, wenn sie an jeder Instanz des Problems anhält und die richtige Antwort gibt.*

2. Die Laufzeit einer TM ist die Anzahl der ausgeführten Befehle bis zum Halten.

3. Die Größe einer Instanz ist die Anzahl dere Buchstaben, die es braucht, um die Instanz hinzuschreiben. Wir geben Beispiele nach dieser Definition.

4. Eine Turingmaschine hat polynomielle Laufzeit, wenn es ein Polynom $p(x)$ gibt, so dass für jede Instanz e die Laufzeit durch $p($Größe von e$)$ beschränkt ist.

Einige Beispiele zur Größe einer Instanz:

- Zahl: Anzahl der Ziffern. Das ist nach oben beschränkt durch $\lceil \log(n+1) \rceil$.
- Graph: zuerst schreibt man n für die Anzahl der Knoten. Die Knotenmenge ist dann implizit definiert als $V = \{1, \ldots, n\}$. Dann die Kanten als (Anfangsknoten,Endknoten,Attribut). Dann ...
- siehe Übungen für weitere Beispiele

Die ersten drei Probleme sind leicht: das weiß man für kürzeste Wege und das Flussproblem seit den 60ern und für das Primzahlproblem seit 2004. Um von einem Problem zu zeigen, dass es leicht ist, muss man einen Algorithmus mit polynomieller Laufzeit angeben. Dazu braucht es typisch eine tiefe Einsicht in das Problem.

Abbildung 1: Manindra Agrawal: one of the authors of "Primes are in P".

3 Ein Beispiel aus meiner Forschung: Gleichgewichtspreise für Arrow-Debreu Märkte.

Der Ausgleich zwischen Angebot und Nachfrage geschieht durch Preisbildung. Für ein einzelnes Gut ist die Frage des Gleichgewichtspreises leicht zu klären. Die Nachfrage ist eine fallende Funktion des Preises und das Angebot ist eine steigende Funktion des Preises. Daher gibt es immer einen Gleichgewichtspreis, bei dem sich Angebot und Nachfrage ausgleichen. Wir steht es nun in einer Wirtschaft, in der es viele Güter, viele Anbieter und viele Kunden gibt?

Fischer (1890) und Walras (1875) haben schon im 19ten Jahrhundert mathematische Modelle für Märkte eingeführt. Das Modell von Fischer ist wie folgt: Es gibt Käufer 1 bis n und Güter 1 bis m. Käufer haben ein gewisses Geldbudget (Käufer i habe b_i Euro) und Präferenzen über die Güter, Güter sind in einer bestimmten Menge vorhanden (ohne Beschränkung der Allgemeinheit, genau eine Einheit). Die Präferenz sagt aus, wieviel Nutzen ein Käufer aus einer gewissen Menge eines bestimmten Gutes zieht. Im einfachsten Fall der linearen Präferenzen ist der Nutzen eine lineare Funktion der Menge. Wenn i den Bruchteil x_{ij} des Gutes j bekommt, so ist sein Nutzen $u_{ij}x_{ij}$. Dabei ist u_{ij} ein Teil der Eingabe. Nutzen ist additiv, d.h., der Gesamtnutzen von Käufer i ist $\sum_j u_{ij}x_{ij}$.

Nehmen wir nun weiter an, dass Güter Preise haben: eine Einheit des Gutes X hat einen Preis pX. Dann gibt es für jeden Käufer und jedes Gut die Größe Nutzen pro Einheit Geld. Bei einem Preis von 2 Euro pro Flasche Champagner und 1 Euro pro Flasche Bier ist für viele Käufer der Nutzen pro Einheit Geld bei Champagner höher als bei Bier. Fischer postuliert nun, dass jeder Käufer nur solche Güter kauft, die den Nutzen pro Einheit Geld für ihn maximieren. Fischer fragt, ob es immer Preise gibt, so dass alle Käufer ihr Budget vollständig ausgeben und alle Güter vollständig verkauft werden?

Wir formulieren nun das Problem präziser. Eingabe: Nichtnegative ganze Zahlen b_i, $1 \leq i \leq n$, und u_{ij}, $1 \leq i \leq n$, $1 \leq j \leq m$.
Frage: gibt es Preise p_j und eine Zuteilung x_{ij}, so dass alle Güter vollständig verkauft werden, alle Käufer ihr Budget voll ausschöpfen, und jeder Käufer bei den gegebenen Preisen seinen Nutzen maximiert.

Formal: Für jeden Käufer i definiere $\alpha_i = \max_j(u_{ij}/p_j)$; α_i ist der maximale Nutzen pro

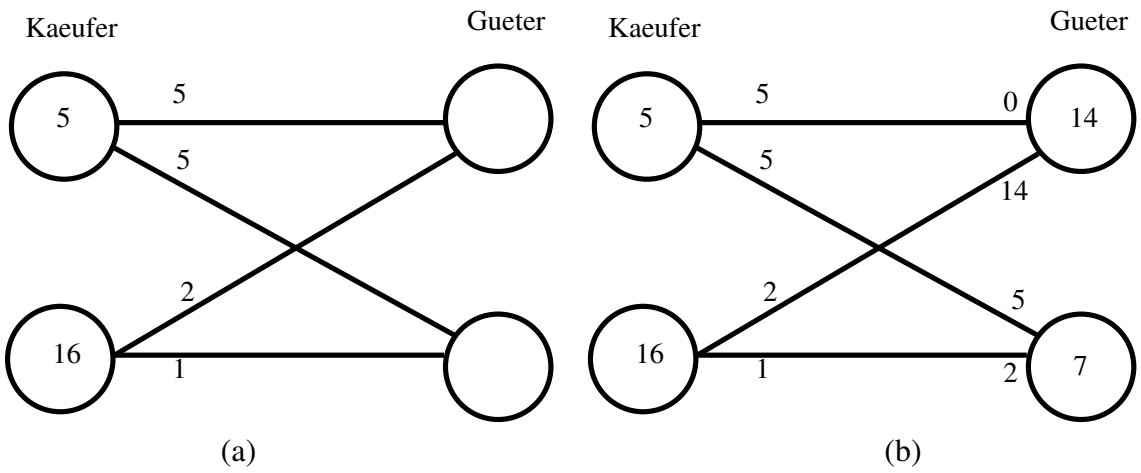

Abbildung 2: (a) Der obere Käufer hat ein Budget von 10, der untere ein Budget von 20. Der obere Käufer zieht gleichen Nutzen aus gleichen Mengen der beiden Güter. Der untere schätzt das obere Gut doppelt so hoch als das untere. Formal: $b_1 = 5$, $b_2 = 16$, $u_{11} = u_{12} = 5$, $u_{21} = 2$, $u_{22} = 1$.
Die Preise $p_1 = p_2 = 10.5$ sind keine Gleichgewichtspreise. Denn dann ist $\alpha_1 = 5/10.5 = 10/21$ und $\alpha_2 = \max(2/10.5, 1/10.5) = 4/21$. Der Käufer 2 wird also nur Geld für das Gut 1 ausgeben. Dann kann er aber nicht sein ganzes Geld ausgeben.
(b) Die Preise $p_1 = 14$ und $p_2 = 7$ sind Gleichgewichtspreise. Dann ist $\alpha_1 = \max(5/14, 5/7) = 5/7$ und Käufer 1 ist nur an Gut zwei interessiert. Dann ist $\alpha_2 = \max(2/14, 1/7) = 1/7$ und Käufer 2 ist an beiden Gütern gleich interessiert. Käufer 1 gibt 5 Euro für Gut 2 aus und keinen Euro für Gut 1, Käufer 2 gibt 14 Euro für Gut 1 aus und 2 Euro für Gut 2. Käufer 1 bekommt $5/7$ des Gutes 2.

Euro, den i erzielen kann. Dann muss gelten:

$$\sum_i x_{ij} = 1 \quad \text{für alle } j$$

$$\sum_j x_{ij} p_j = b_i \quad \text{für alle } i$$

$$x_{ij} > 0 \text{ implies } \frac{u_{ij}}{p_j} = \alpha_i$$

Abbildung 2 zeigt ein Beispiel.

Das Modell von Walras ist etwas anders: Er nimmt nicht an, dass Käufer von vorn herein über eine gewisse Menge Geld verfügen. Vielmehr Käufer auch Verkäufer. Sie besitzen Güter und erhalten Geld nur durch den Verkauf von Gütern. Die Frage ist wieder die gleiche: Gibt es Preise, so das alle Käufer ihr durch den Verkauf ihrer Güter erhaltenes Budget vollständig ausgeben und alle Güter vollständig verkauft werden?

Wir formulieren nun das Problem präziser. Eingabe: Nichtnegative ganze Zahlen u_{ij}, $1 \leq i \leq n$, $1 \leq j \leq m$.

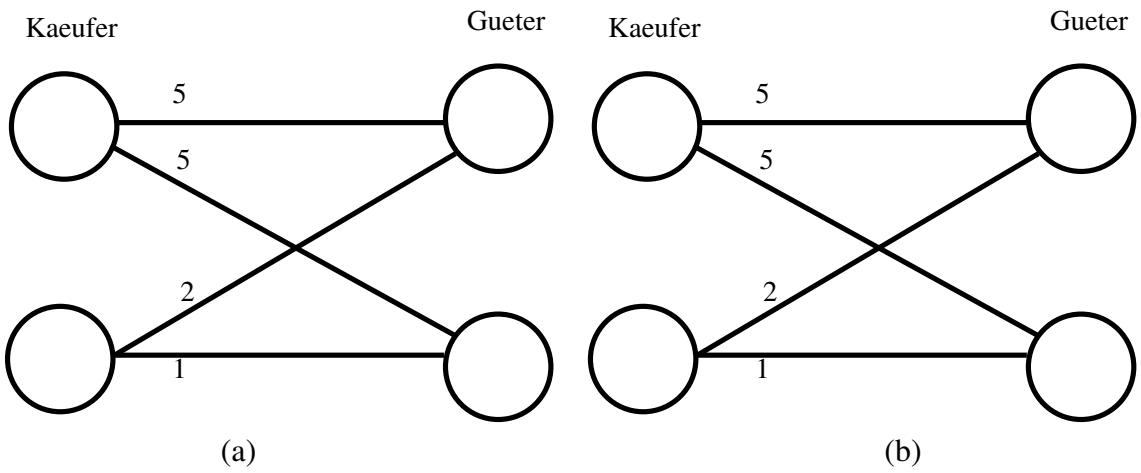

Abbildung 3: (a) Der obere Käufer zieht gleichen Nutzen aus gleichen Mengen der beiden Güter. Der untere schätzt das obere Gut doppelt so hoch als das untere. Formal: $u_{11} = u_{12} = 5$, $u_{21} = 2$, $u_{22} = 1$.
Die Preise $p_1 = 14$ und $p_2 = 7$ sind keine Gleichgewichtspreise. Dann ist $\alpha_1 = \max(5/14, 5/7) = 5/7$ und Käufer 1 ist nur an Gut zwei interessiert. Er hat aber ein Budget von 14 und kann daher sein Geld nicht vollständig ausgeben.
(b) Die Preise $p_1 = p_2 = 15$ sind Gleichgewichtspreise. Denn dann ist $\alpha_1 = 5/15 = 1/3$ und $\alpha_2 = \max(2/15, 1/15) = 2/15$. Der Käufer 2 kauft das ganze Gut 1 und der Käufer 1 das ganze Gut 2.

Frage: gibt es Preise p_j und eine Zuteilung x_{ij}, so dass alle Güter vollständig verkauft werden, alle Käufer ihr Budget voll ausschöpfen, und jeder Käufer bei den gegebenen Preisen seinen Nutzen maximiert.

Formal: Für jeden Käufer i definiere $\alpha_i = \max_j(u_{ij}/p_j)$; α_i ist der maximale Nutzen pro Euro, den i erzielen kann. Dann muss gelten:

$$\sum_i x_{ij} = 1 \quad \text{für alle } j$$

$$\sum_j x_{ij} p_j = p_i \quad \text{für alle } i$$

$$x_{ij} > 0 \text{ implies } \frac{u_{ij}}{p_j} = \alpha_i$$

Abbildung 3 zeigt ein Beispiel.

Im Modell von Fischer hat Geld einen Wert an sich, im Modell von Walras dient es nur dazu, Güter zu vergleichen. Das Modell von Fischer ist ein Spezialfall des Modells von Walras.

Existenz von Gleichgewichtspreisen Es hat bis 1954 gedauert, bis die Existenz von Gleichgewichtspreisen von Arrow und Debreu mathematisch stringent bewiesen wurde. Arrow and Debreu wurden dafür und für andere Leistungen mit dem Nobelpreis für Wirtschaftswissenschaften

ausgezeichnet. Der Beweis von Arrow und Debreu ist ein reiner Existenzbeweis und liefert kein Verfahren zur Berechnung von Gleichgewichtspreisen.

Berechnung von Gleichgewichtspreisen Kann man Gleichgewichtspreise effizient berechnen? Es hat mehrere Jahrzehnte gedauert, bis für das Modell von Walras die Antwort gefunden wurde. Für das Modell von Fischer wurde die Frage schon früher beantwortet. Erst 2007 fand K. Jain einen Algorithmus mit polynomieller Laufzeit, der allerdings wegen der Benutzung der Ellipsoidmethode noch unbefriedigend ist. Ran Duan und Kurt Mehlhorn fanden in 2012 einen relativ einfachen kombinatorischen Algorithmus.

4 Beweisbar Schwere Probleme

Halteproblem:
Eingabe: die Beschreibung M einer Turingmaschine und ein Wort x
Frage: hält M an der Eingabe x?

eingeschränktes Halteproblem:
Eingabe: die Beschreibung M einer Turingmaschine und ein Wort x diese Maschine
Frage: hält M an der Eingabe x in höchstens $2^{|x|}$ Schritten? Dabei ist $|x|$ die Anzahl der Buchstaben in x.

Das Halteproblem ist unentscheidbar Es gibt keine Turingmaschine, die das Halteproblem löst. Bevor wir diese Aussage beweisen, geben wir ein Plausibilitätsargument.

Die Vermutung von Fermat besagt, dass es fürr $n \geq 3$ keine ganzzahligen positive Lösungen der Gleichung

$$x^n + y^n = z^n$$

gibt. Die Vermutung wurde vor 350 Jahren von Fermat aufgestellt (er fügte die Bemerkung hinzu: leider ist hier nicht genügend Platz auch den Beweis aufzuschreiben) und erst vor kurzem von Wiles bewiesen.

Es ist aber leicht ein Programm zu schreiben, dass die Vermutung überprüft.

```
for N = 3,4,5,... do
    for all x,y, z,n ≤ N do
        if x^n + y^n = z^n then
            HALT;
        end if
    end for
end for
```

Die Maschine hält an, wenn es ein Gegenbeispiel für die Vermutung von Fermat gibt. Wenn es kein Gegenbeispiel gibt, dann läuft die Maschine für immer. Wenn nun das Halteproblem lösbar wäre, dann könnten wie sie benutzen, um die Vermutung von Fermat zu entscheiden. Es gilt sogar, dass wir jede mathematische Vermutung dann automatisch entscheiden könnten. Es

ist also nicht plausibel, dass man das Halteproblem entscheiden kann. Turing bewies, dass das Problem unentscheidbar ist.

Theorem 1 *Es gibt keine Turingmaschine, die das Halteproblem löst.*

Beweis (indirekt). Wir nehmen an, dass es eine Turingmaschine H gibt, die das Halteproblem löst. Betrachte folgende Maschine C:

– bei Eingabe M lassen wir zunächst $H(M,M)$ laufen. Diese Berechnung ist immer endlich.
– falls Antwort "hält an", dann gehen wir in eine unendliche Schleife
– falls Antwort "hält nicht an", dann halten wir an.

Frage: was macht C an der Eingabe C?

– lässt zun¸''ahst $H(C,C)$ laufen. Diese Berechnung ist endlich und stoppt entweder mit "hält an" oder mit "hält nicht an".
– falls mit "hält an", dann hält C nicht
– falls mit "hält nicht an", dann hält C.

C straft H Lügen. Daher kann es H nicht geben.

Das eingeschränkte Halteproblem ist entscheidbar aber schwer Wir zeigen zunächst, dass es entscheidbar ist. Betrachte den folgenden Algorithmus. Llasse M an x laufen und zähle die Anzahl der ausgeführten Schritte mit. Halte an, wenn M an x anhält oder wenn die Anzahl $2^{|x|}$ erreicht. Im ersten Fall gib die Antwort "hält an" im letzteren Fall gib die Antwort "hält nicht an".

Theorem 2 *Sei H eine TM, die das eingeschränkte Halteproblem entscheidet. Dann gibt es eine Eingabe (M,x), an der H mindestens Laufzeit $2^{(|M|+|x|)/2}-1$ hat.*

Proof: Wir nehmen an, dass es ein H gibt, das immer in $2^{(|M|+|x|)/2}-1$ anhält.

Betrachte folgende Turingmaschine C:

– bei Eingabe M lassen wir zunächst $H(M,M)$ laufen. Diese Berechnung hält in $2^{(|M|+|M|)/2} = 2^{|M|}-1$ Schritten.
– falls Antwort "hält an", dann gehen wir in eine unendliche Schleife
– falls Antwort "hält nicht an", dann halten wir an.

Falls C anhält, dann innerhalb von $2^{|M|}-1+1$ Schritten. Falls also C an einer Eingabe nicht innerhalb von $2^{|M|}$ Schritten anhält, dann hält es gar nicht an dieser Eingabe an.

Frage: was macht C an der Eingabe C?

– lässt zunächst $H(C,C)$ laufen. Diese Berechnung ist endlich und stoppt entweder mit "hält an" oder mit "hält nicht an".
– falls mit "hält an", dann hält C nicht
– falls mit "hält nicht an", dann hält C.

C straft H Lügen. Daher kann es H nicht geben. ∎

5 NP-Completeness

Es gibt eine große Klasse von Problemen, bei denen es sehr unwahrscheinlich ist, dass sie je als einfach klassifiziert werden. Es ist aber nicht bewiesen, dass sie nicht einfach sind. Es gilt als eines der schwersten Probleme der Mathematik, zu beweisen, dass diese Probleme nicht einfach sind. Das ist die

$$\mathrm{P} = \mathrm{NP}?$$

Frage.

Diese Probleme sind alle äquivalent: entweder sind alle einfach oder keines.

informelle Definition der Klasse: für die Ja-Antwort gibt es stets einen kurze Begründung, die man in Polynomzeit überprüfen kann.

formale Definition

Ja-Instanzen = $\{x$; es gibt ein y mit $|y| \leq p(|x|)$ und $M(x,y) = \mathrm{JA}\}$ dabei ist p ein Polynom und M ein polynomial beschränkte Turingmaschine.

Äquivalenzbeweis führt man durch Reduktion. Zum Beispiel: Graphfärbung $\leq$ Erfüllbarkeit: Wir zeigen, dass jede JA-Instanz des Graphenfärbungsproblem auf eine JA-Instanz des Erfüllbarkeitsproblems abgebildet werden kann und jede NEIN-Instanz auf eine NEIN-Instanz.

Wir haben für jeden Knoten v drei Variablen b_v, g_v und r_v. Sie stehen für die drei Farben. $b_v = \mathrm{WAHR}$ steht für v is blau gefärbt.

Die Formel sieht so aus

$$\wedge_{v \in V} \mathit{GenauEine}(v) \wedge \wedge_{e=uv \in E} \mathit{Verschieden}(u,v)$$

Dabei ist

$$\begin{aligned} \mathit{GenauEine}(v) &= (b_v \vee g_v \vee r_v) \wedge \neg(b_v g_v \vee b_v r_v \vee g_v r_v) \\ \mathit{Verschieden}(u,v) &= \neg(b_u b_v \vee g_u g_v \vee r_u r_v) \end{aligned}$$