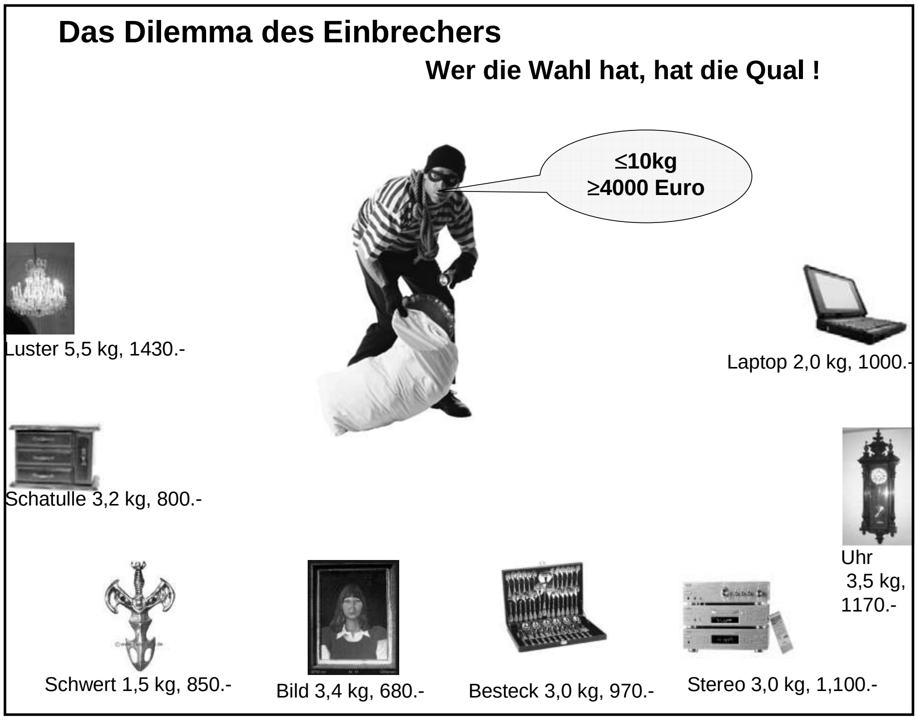
Schwere Probleme sind manchmal leicht NP = P ?

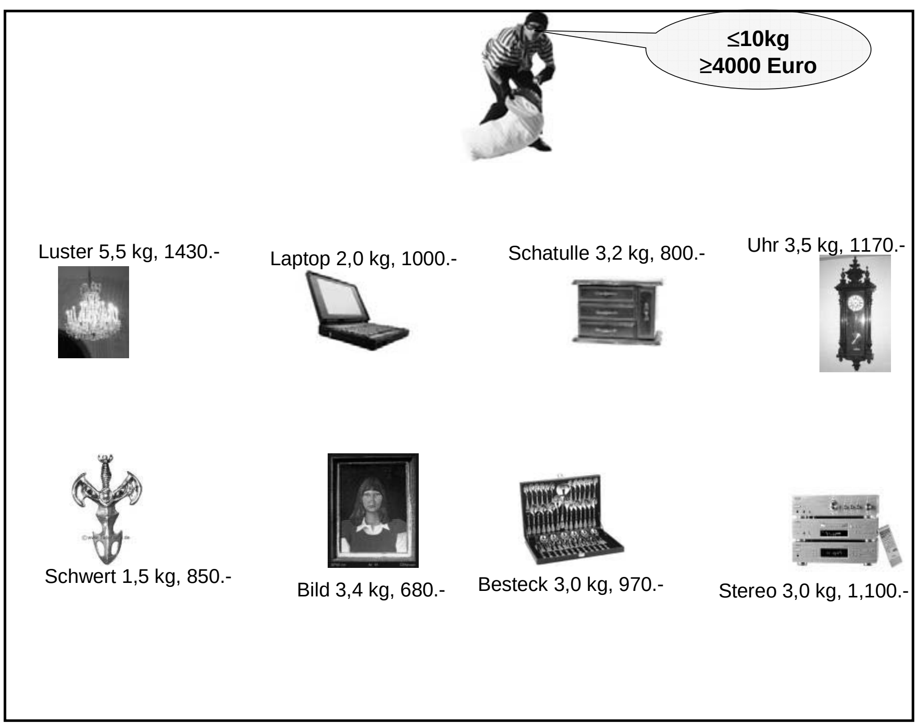
Georg Gottlob
Institut für Informationssysteme
Technische Universität Wien

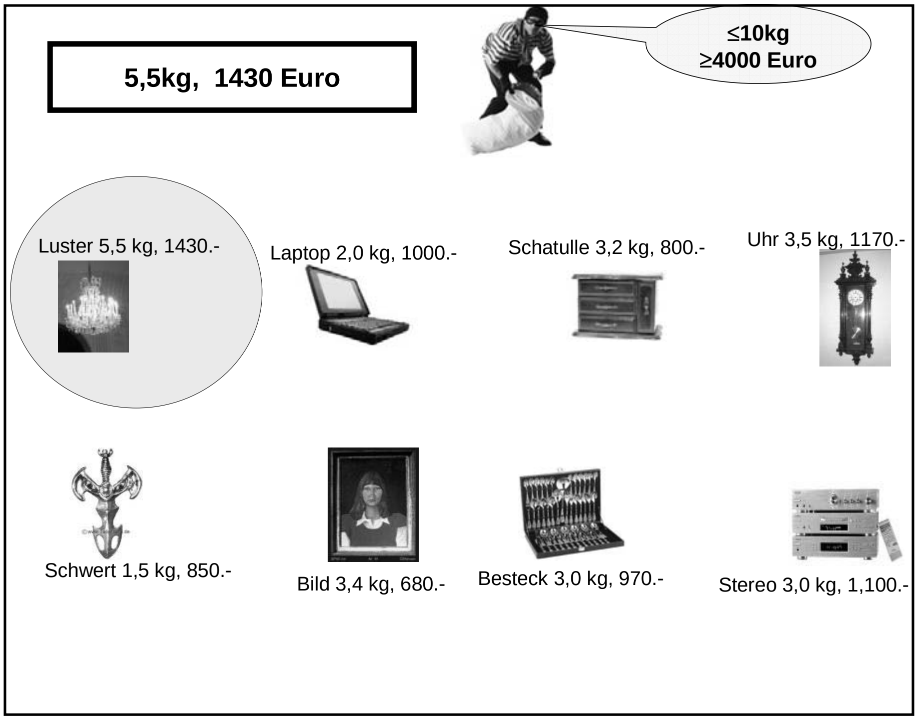
Gliederung

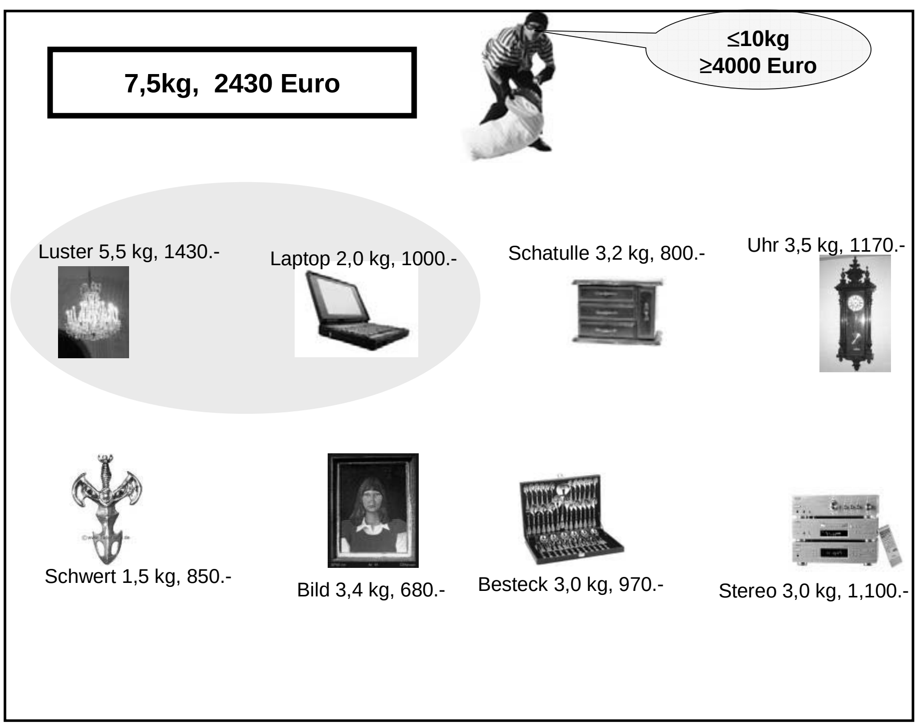
- Was ist ein Entscheidungsproblem / Berechnungsproblem ?
- Komplexität von Algorithmen und Problemen
- NP-vollständige Probleme
- Drei Lösungsansätze:
  - Randomisierte lokale Suche
  - Approximation
  - Identifikation leicht lösbarer Subklassen

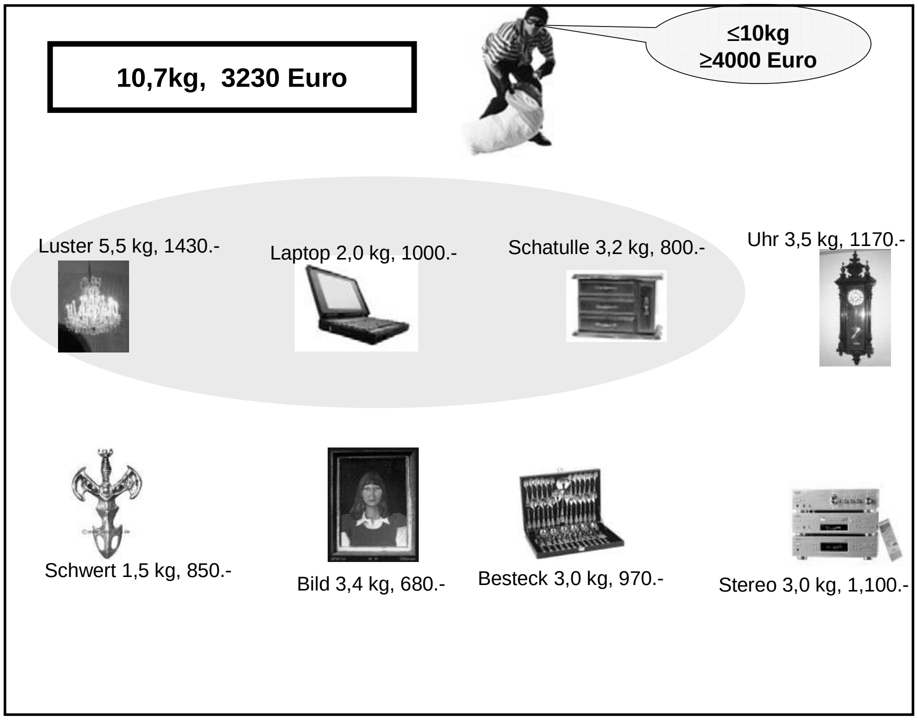
Übersichtsvortrag, keine mathematischen Details.

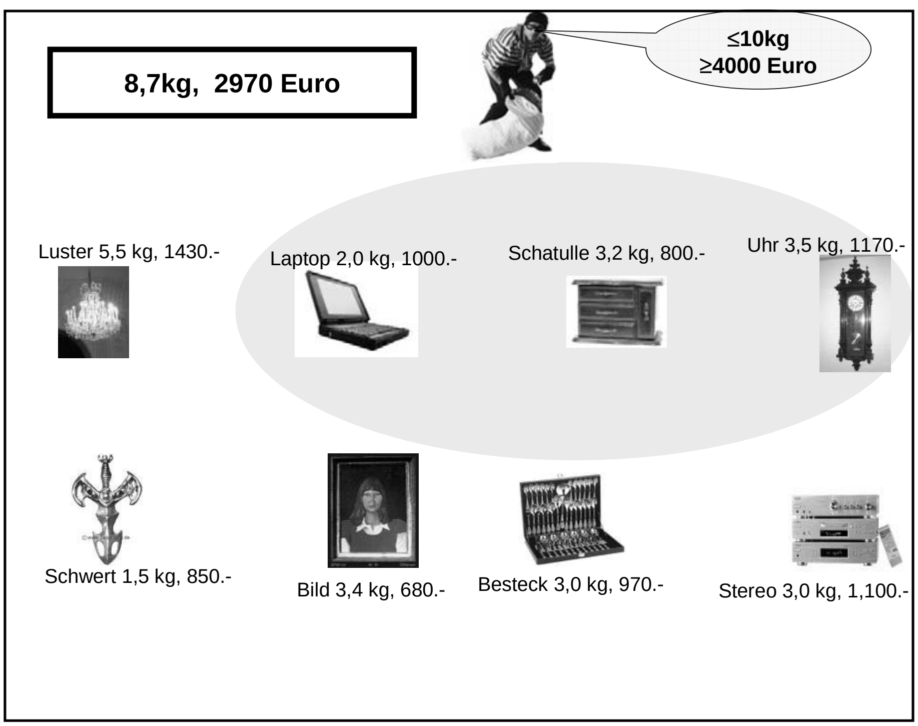

Das Dilemma des Einbrechers
Wer die Wahl hat, hat die Qual !
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr
3,5 kg,
1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

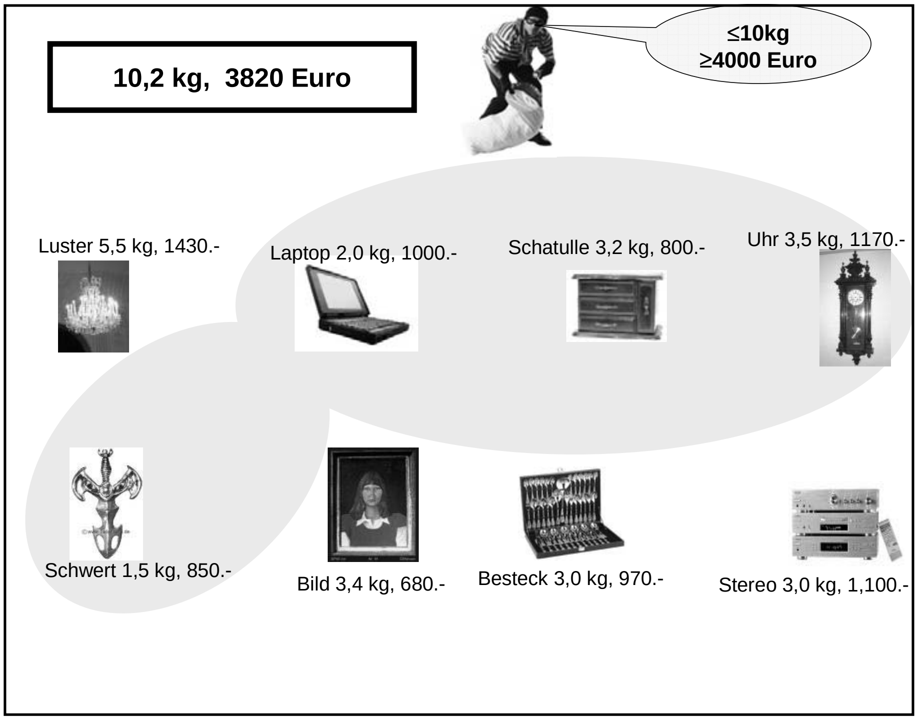

≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

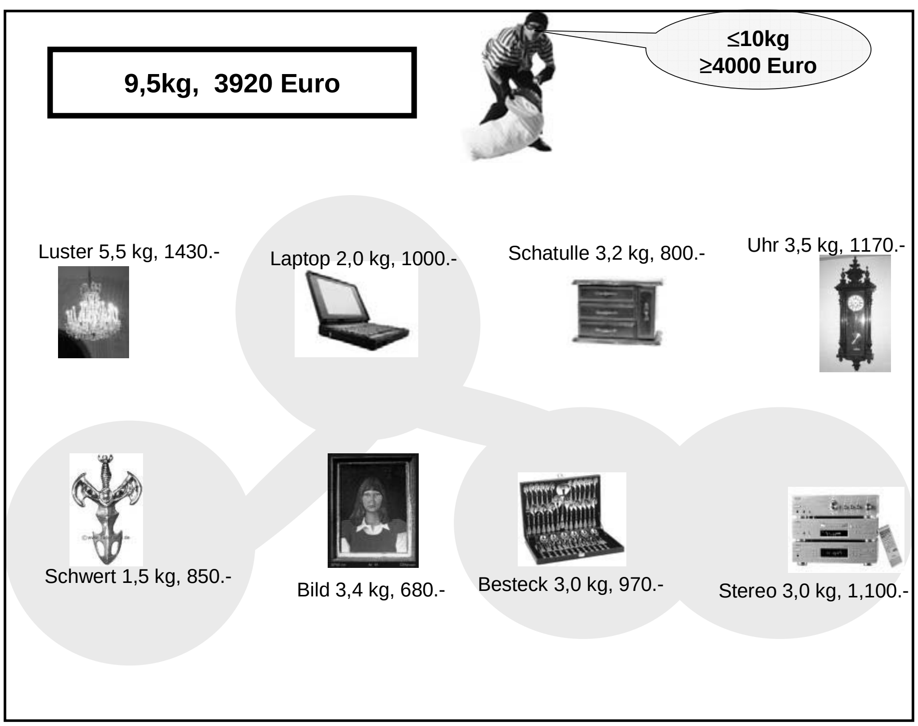

5,5kg, 1430 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

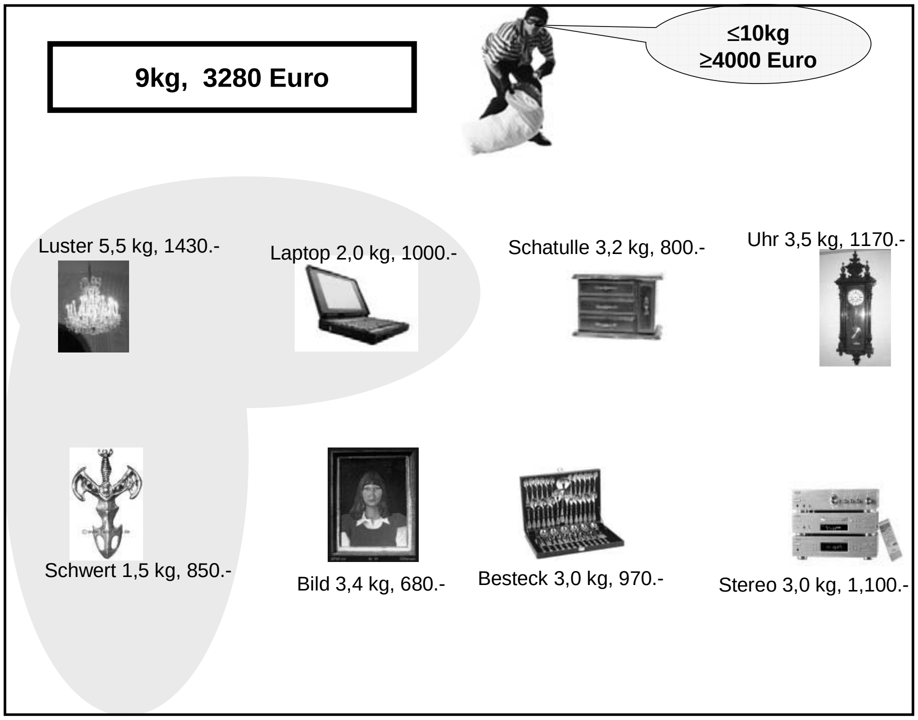

7,5kg, 2430 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-


10,7kg, 3230 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

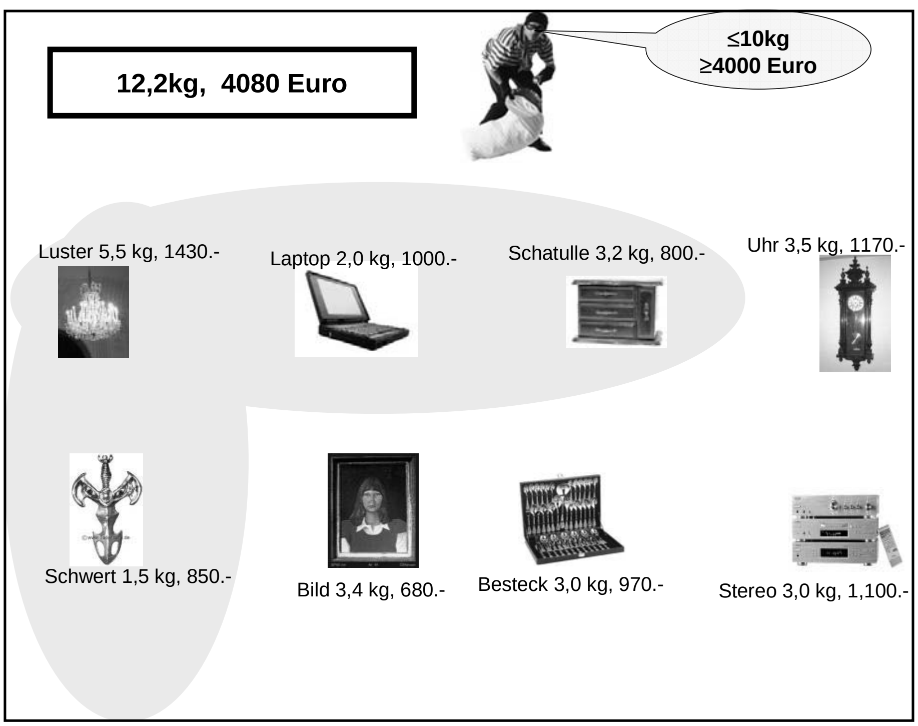

8,7kg, 2970 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-


10,2 kg, 3820 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

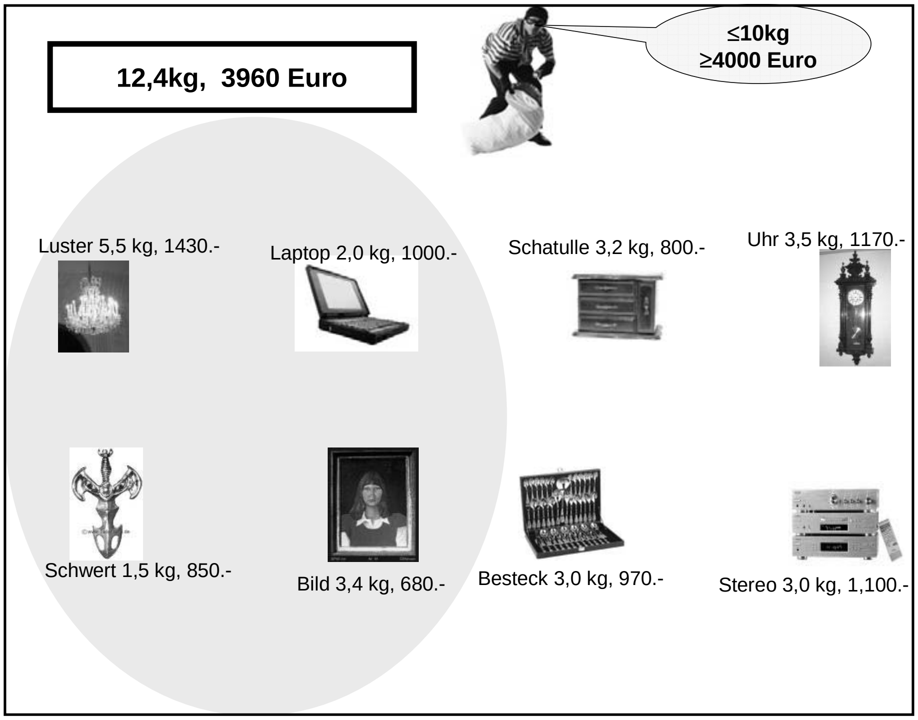

9,5kg, 3920 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

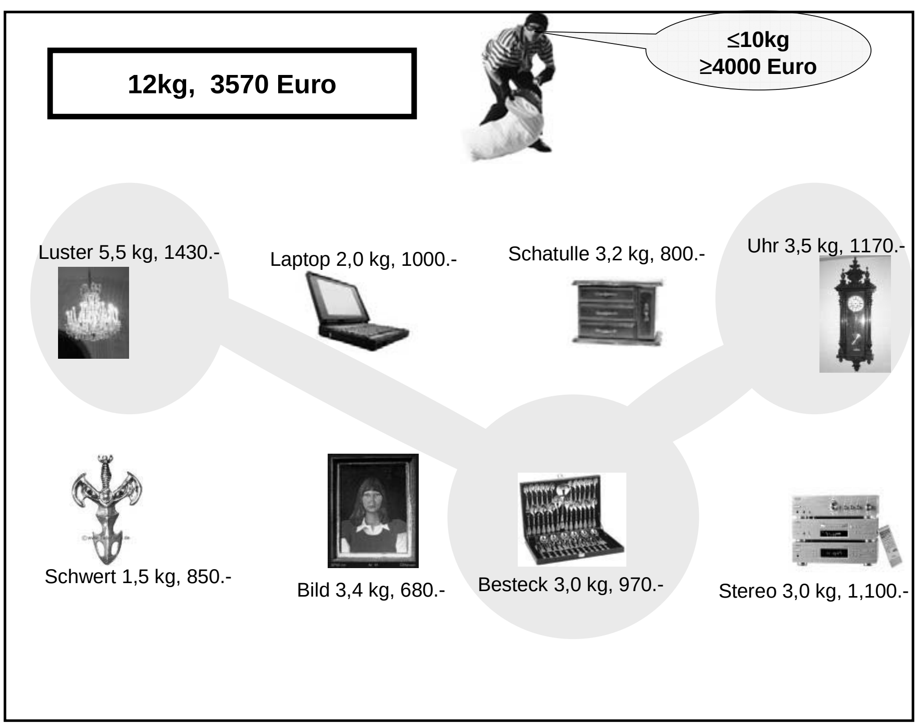


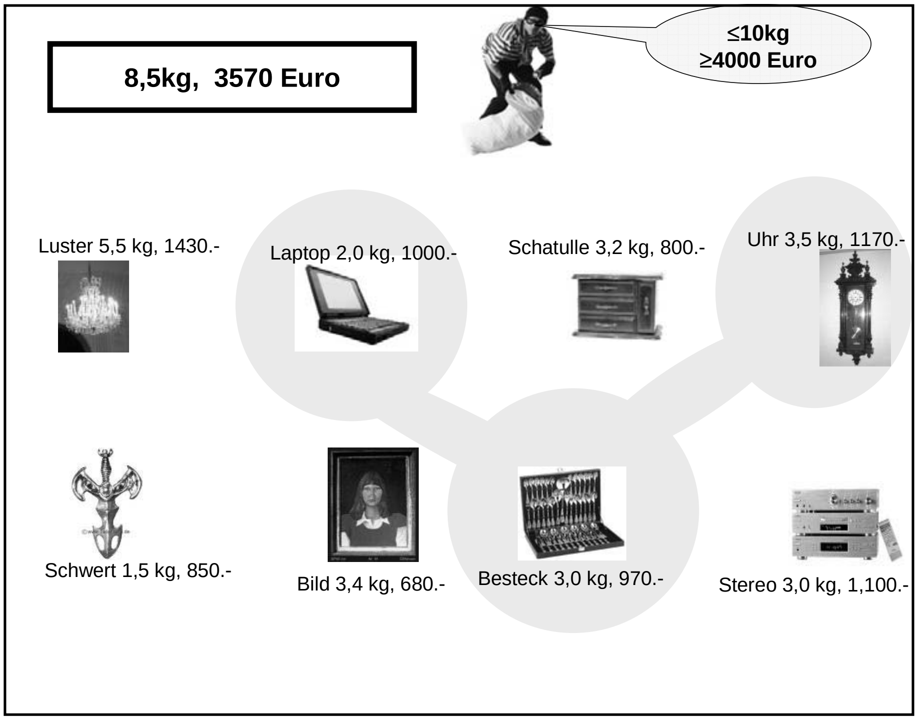
9kg, 3280 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

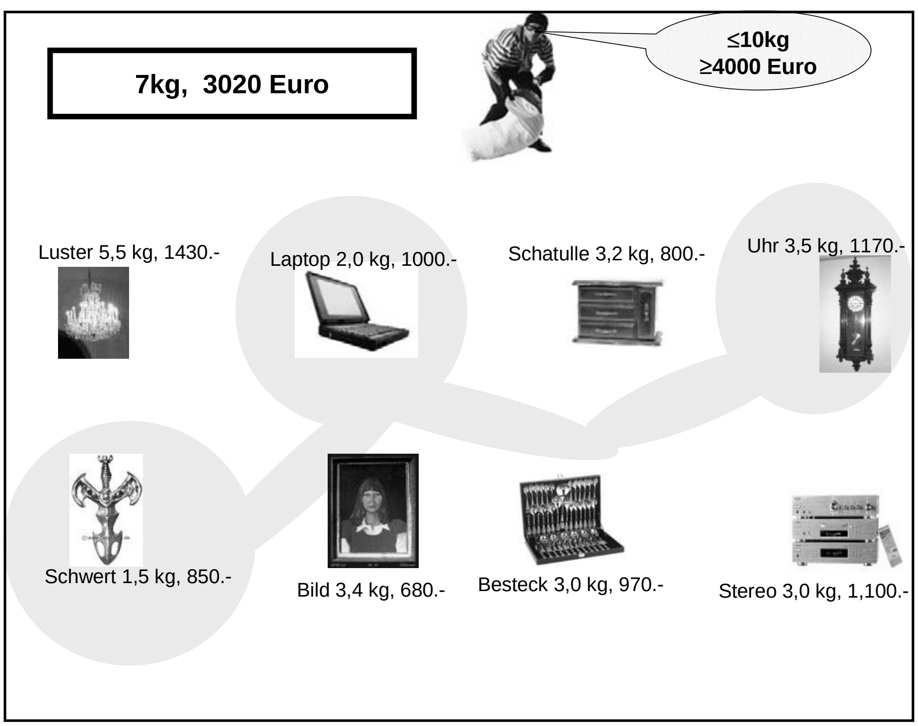


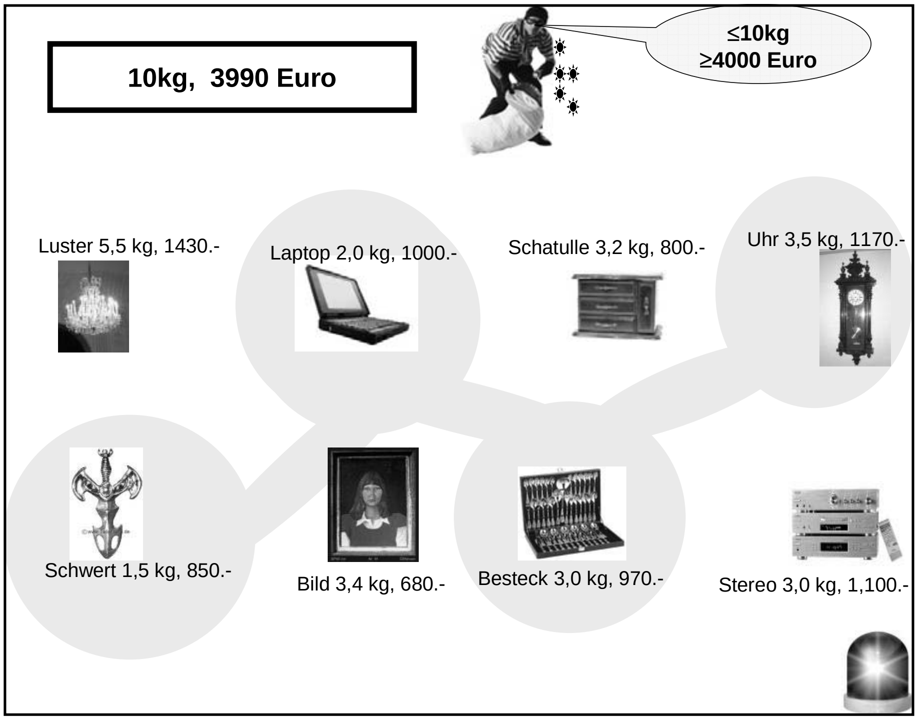
12,2kg, 4080 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

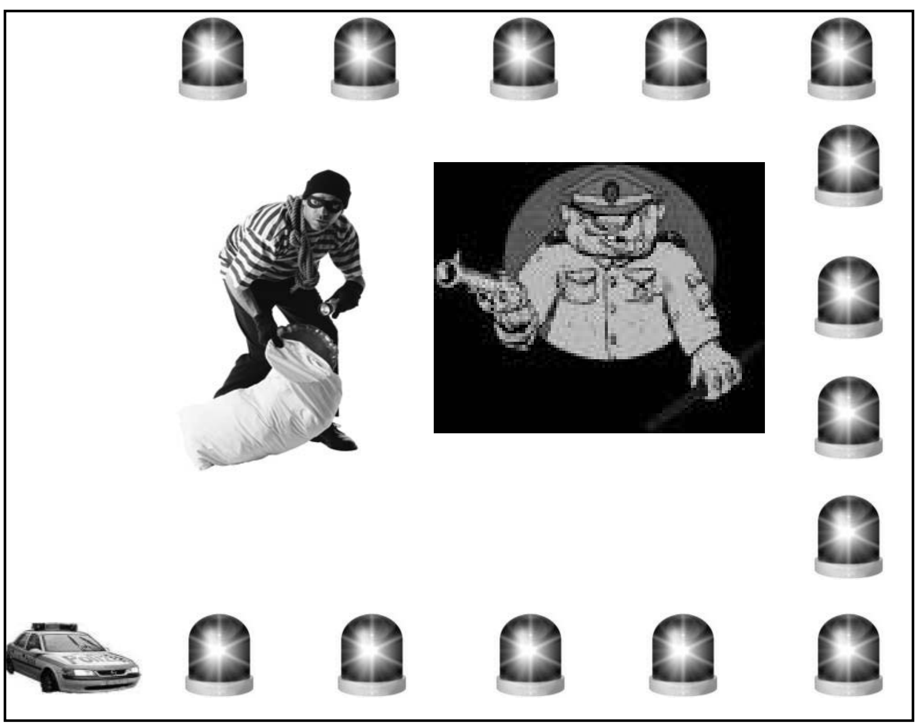

12,4kg, 3960 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-


12kg, 3570 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-


8,5kg, 3570 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-

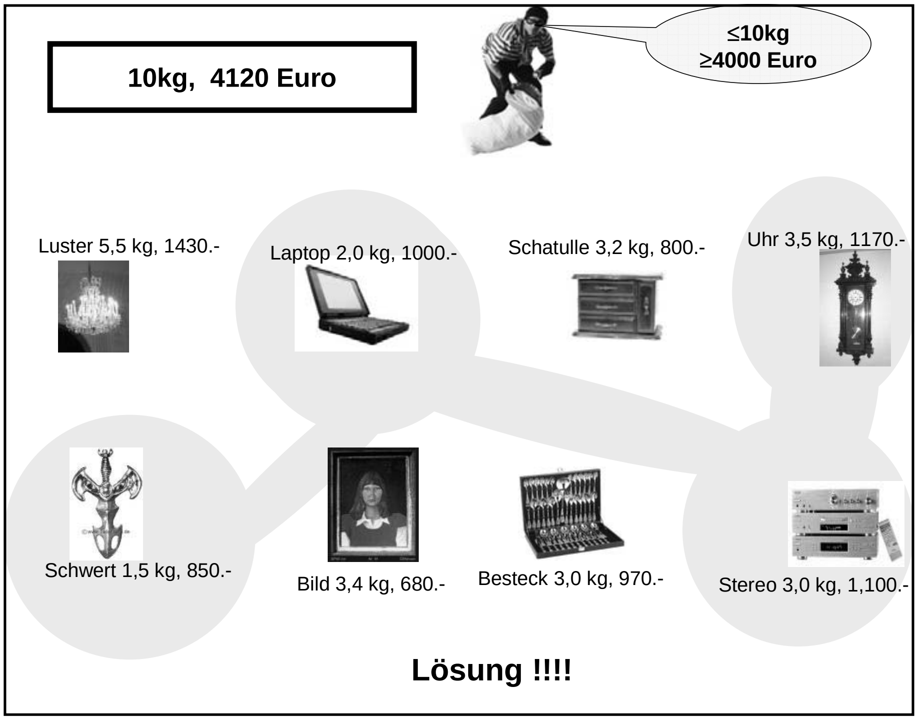

7kg, 3020 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-


10kg, 3990 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-




So, jetzt ab in
den Knast,
du Schuft!


10kg, 3990 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-
Zu spät, aber eine Lösung lag nahe!!!!



Merkmale dieses Problems

- **Sehr viele Lösungskandidaten**

  z.B. {Laptop, Schwert, Besteck},
  {Laptop, Bild, Uhr},
  {Bild, Uhr, Besteck},……..usw…….usw

- **Von jedem solchen Kandidaten lässt sich leicht und schnell feststellen, ob er tatsächlich eine Lösung ist.**

  Es genügt, das Gesamtgewicht G und den Gesamtpreis P des Kandidaten zu berechnen, und zu prüfen, ob G ≥ 10, P ≥ 4000.

- **Wir kennen kein einfaches Verfahren, um Lösung zu finden.**

  Z.B.: “Nimm zuerst die teuersten Gegenstände” funktioniert nicht.

- **Zeit spielt eine wichtige Rolle.**

Kombinatorische Explosion

- **8 Objekte, 2^8 = 256 Teilmengen als Lösungskandidaten. Davon sind grobgeschätzt mindestens 30 "interessant" und man muss sie generieren und überprüfen ("checken").**

Bei n Objekten 2^n Teilmengen als Lösungskandidaten.

n	2^n	n	2^n	n	2^n
0	1	8	256	16	65,536
1	2	9	512	17	131,072
2	4	10	1,024	18	262,144
3	8	11	2,048	19	524,288
4	16	12	4,096	20	1,048,576
5	32	13	8,192	21	2,097,152
6	64	14	16,384	22	4,194,304
7	128	15	32,768	23	8,388,608

Bei n Objekte.

Das Rucksackproblem

GEGENSTAND	GEWICHT	WERT
Luster	5500	Euro 1430.-
Schatulle	3200	Euro 800.-
Schwert	1500	Euro 850.-
Bild	3400	Euro 680.-
...	...	...

Allgemeine Problemformulierung:

Angabe: Gegeben Liste von Einträgen $\langle Gegenstand, Gewicht, Preis \rangle$, sowie Maximales Gewicht G, gewünschter Wert W

Frage: Gibt es eine Menge $S \subseteq$ Gegenstände, sodaß

$$\sum_{x \in S} gewicht(x) \leq G \quad \text{und} \quad \sum_{x \in S} preis(x) \geq W \quad ?$$

Berechnungsproblem

Berechne eine Lösung S, so daß

$$\sum_{x \in S} gewicht(x) \leq G \quad \text{und} \quad \sum_{x \in S} preis(x) \geq W$$

Berechnungsproblem mit Optimierung

Berechne eine Lösung S, so daß

$$\sum_{x \in S} gewicht(x) \leq G \quad \text{und} \quad \sum_{x \in S} preis(x) \text{ maximal}$$

Ähnliche Probleme: Frachtraumplanung, Lagerplanung,
Verschnittoptimierung, Anlagenoptimierung.

Problem/Probleminstanz

Problem: Abstrakte Problemstellung ohne konkrete Daten.

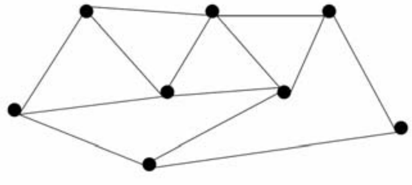
Instanz: Konkrete Angabe.

Zu einem Problem existieren i.a. unendlich viele Instanzen.

Größe einer Instanz: Anzahl der verwendeten Zeichen.

Viele interessante Probleme beziehen sich auf Graphen:

Knoten,
Kanten

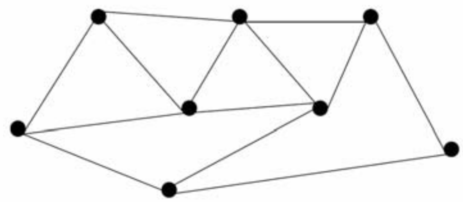


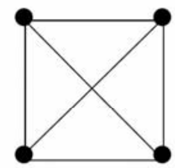
Graph-Dreifärbbarkeit

Angabe: Ein Graph G.

Frage: Ist G dreifärbbar?

Beispiele für Instanzen:





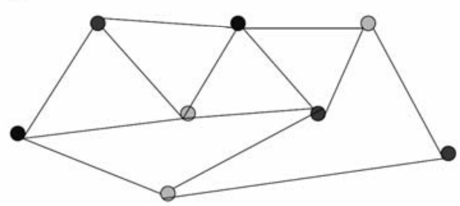
Berechnungsproblem: Berechne eine korrekte 3-Färbung.

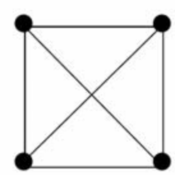
Graph-Dreifärbbarkeit

Angabe: Ein Graph G.

Frage: Ist G dreifärbbar?

Beispiele für Instanzen:



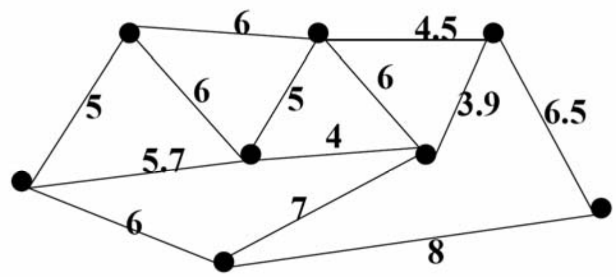


Berechnungsproblem: Berechne eine korrekte 3-Färbung.

Das Traveling Salesperson Problem (TSP)

Angabe: Ein Straßennetz G mit Distanzen, Zahl M.

Frage: Gibt es eine "Tour" mit Gesamtlänge $\leq M$?



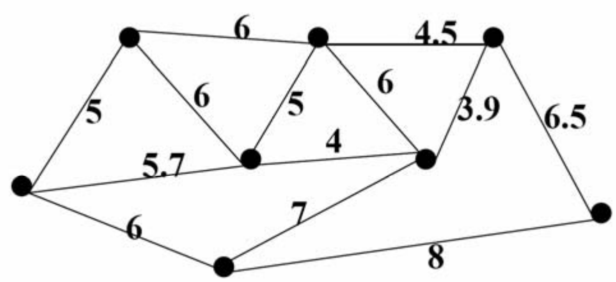
Berechnungsproblem: Berechne optimale Tour.

7

Das Traveling Salesperson Problem (TSP)

Angabe: Ein Straßennetz G mit Distanzen, Zahl M.

Frage: Gibt es eine "Tour" mit Gesamtlänge $\leq M$?

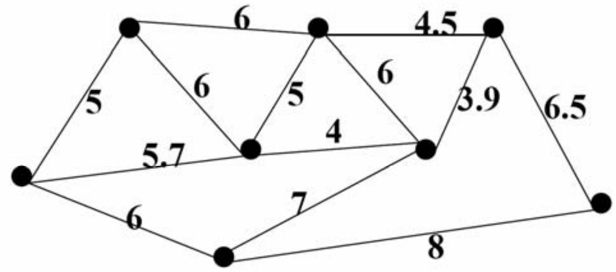


Berechnungsproblem: Berechne optimale Tour.

Das Traveling Salesperson Problem (TSP)

Angabe: Ein Straßennetz G mit Distanzen, Zahl M.

Frage: Gibt es eine "Tour" mit Gesamtlänge $\leq M$?

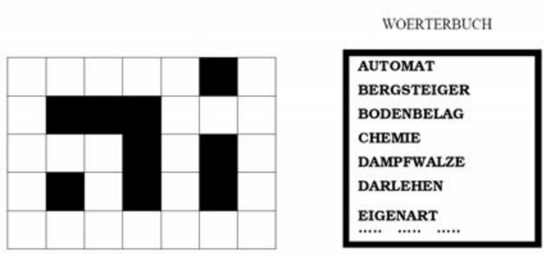


Berechnungsproblem: Berechne optimale Tour.

Kombinatorisches Kreuzworträtsel

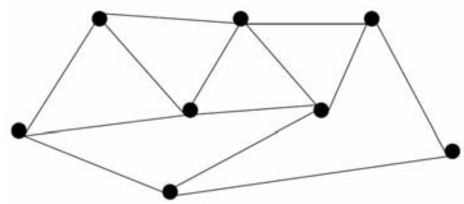
Angabe: Raster für Kreuzworträtsel, Wörterbuch.

Frage: Kann ich das Raster füllen?



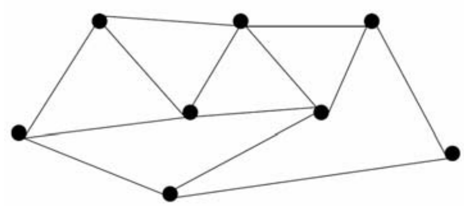
Hamiltonscher Kreis

Gegeben ein Graph G.
Gibt es einen geschlossenen Linienzug, der alle Knoten erfasst
Aber jeden Knoten nur ein mal berührt?



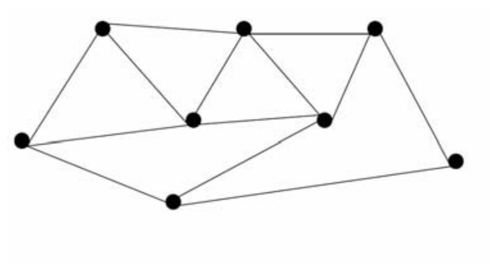
Hamiltonscher Kreis

Gegeben ein Graph G.
Gibt es einen geschlossenen Linienzug, der alle Knoten erfasst
Aber jeden Knoten nur ein mal berührt?



Hamiltonscher Kreis

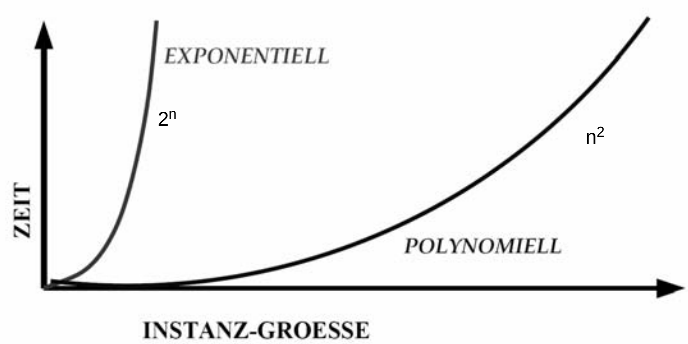
Gegeben ein Graph G.
Gibt es einen geschlossenen Linienzug, der alle Knoten erfasst
Aber jeden Knoten nur ein mal berührt?



9

Zeitkomplexität von Algorithmen

Anzahl der Rechenschritte, die ein Algorithmus zur Lösung eines Problems braucht (worst case).



10

Inhärente Problemkomplexität

Probleme *entscheidbar* oder *unentscheidbar*.

Wir betrachten hier nur entscheidbare Probleme.

Ein Problem ist so komplex wie der bestmögliche Algorithmus zu seiner Lösung.

11

BEWEISBAR EXPONENTIELL

Logische Theorie der Reellen Zahlen

Domino Probleme

PROBL.

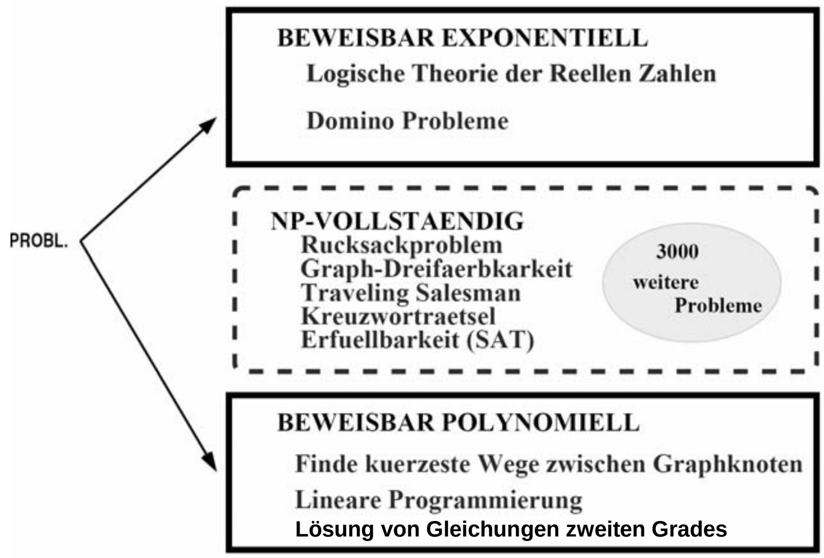
BEWEISBAR POLYNOMIELL

Finde kuerzeste Wege zwischen Graphknoten

Lineare Programmierung

Lösung von Gleichungen zweiten Grades

12

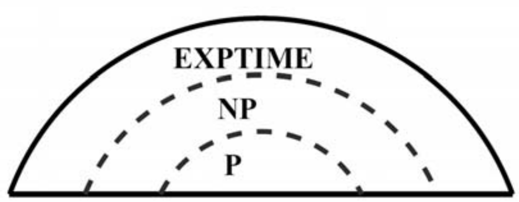


13

$\boxed{\text{NP}}$

NP: Nichtdeterministisch polynomiell

Paradigma: Guess and Check

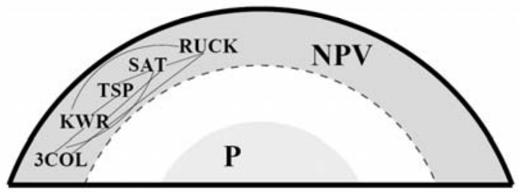


Kein Problem in NP konnte bisher als nichtpolynomiell bewiesen werden.

Für viele Probleme in NP kennt man aber keinen polynomiellen Algorithmus.

14

NP-vollständige Probleme



Feinstruktur der Klasse NP

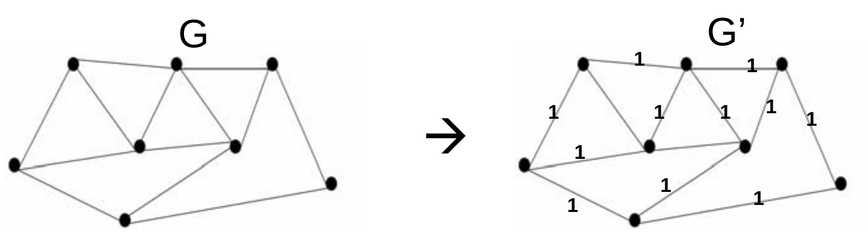
NPV: Die schwersten Probleme innerhalb von NP.

Alle polynomiell ineinander überführbar

Eines polynomiell ⇒ alle polynomiell, d.h. NP=P.

Problemreduktion A→B

Beispiel: Hamiltonscher Kreis → TSP

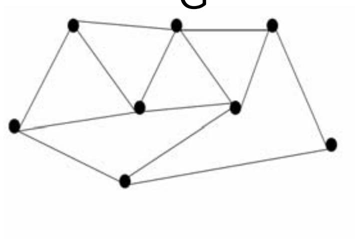


M=8 (=Knotenanzahl)

Problemreduktion A→B

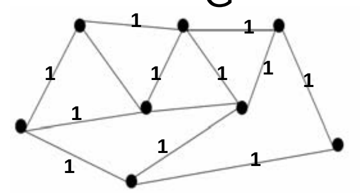
Beispiel: Hamiltonscher Kreis → TSP

G



→

G'



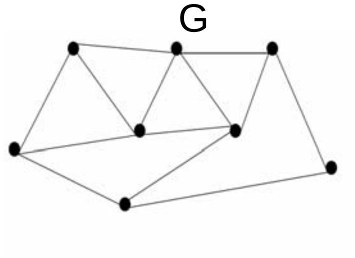
M=8 (=Knotenanzahl)

G hat Hamiltonschen Kreis ⇔ G' hat Tour der Länge 8

Problemreduktion A→B

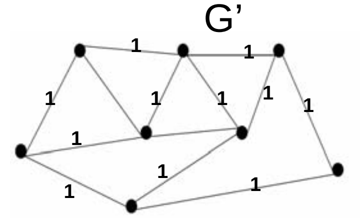
Beispiel: Hamiltonscher Kreis → TSP

G



→

G'



M=8 (=Knotenanzahl)

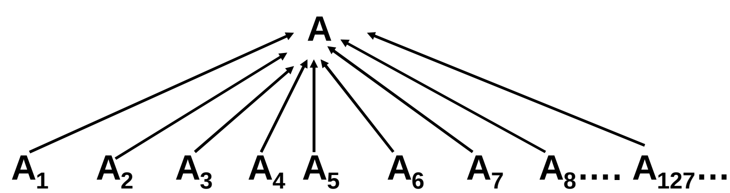
G hat Hamiltonschen Kreis ⇔ G' hat Tour der Länge 8

NP-Vollständigkeit

Ein Problem A ist NP-vollständig, wenn:

1.) A in der Klasse NP liegt:
Lösungskandidaten können in Polynomialzeit "gecheckt" werden

2.) Alle anderen Probleme aus NP sich in Polynomialzeit in A transformieren lassen.

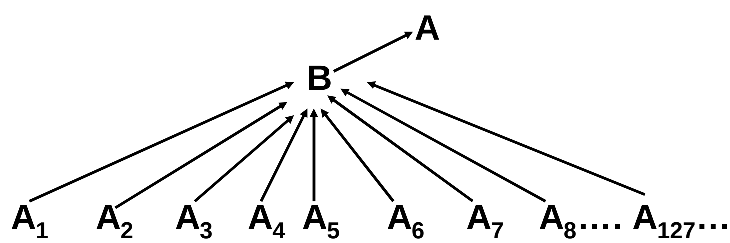


Wie zeigt man, dass ein Problem NP-vollständig ist ???

Wie zeigt man, dass ein Problem A NP-vollständig ist ???

Man zeigt zuerst, dass A überhaupt in NP liegt.

Dann sucht man ein geeignetes NP-vollständiges Problem B und reduziertes polynomiell auf A.



Wie zeigt man, dass ein Problem A NP-vollständig ist ???

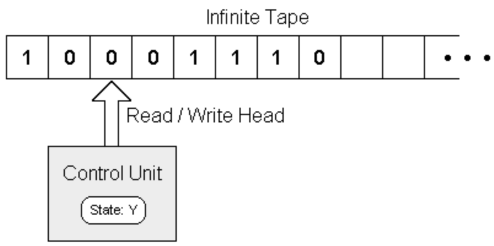
Man zeigt zuerst, dass A überhaupt in NP liegt.

Dann sucht man ein geeignetes NP-vollständiges Problem B und reduziertes polynomiell auf A.

Ja, aber wie war das dann beim ersten NPV-Problem???

Alan Turing

1936 *On computable numbers, with an application to the Entscheidungsproblem*



State	Symbol	New State	New Symbol	Move
1	0	1	1	R
1	1	1	0	R
1	b	2	b	R

Start State: 1
Halt State: 2

Stephen Cook

1971 *The Complexity of Theorem Proving Procedures*

1.) Jedes Problem in NP kann durch eine <u>nichtdeterministische</u> Turing Maschine in polynomieller Zeit gelöst werden.

State	Symbol	New State	New Symbol	Move
1	0	1	1	R
1	1	1	0	R
1	1	2	1	L

2.) Jede solche Turing Maschine plus Eingabe kann in eine Instanz des <u>SATISFIABILITY</u>-Problems (SAT) transgformiert werden (in polynomieller Zeit).

SATISFIABILITY (SAT)

Gegeben: Eine Menge von logischen Klauseln:

$$(X1 \text{ or } X2 \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } X2 \text{ or } X3)$$

Frage: Gibt es eine Wahrheitswertbelegung fuer die Variablem Xi (Xi=Wahr oder Xi=Falsch) durch die jede Klausel wahr wird ?

SATISFIABILITY (SAT)

Gegeben: Eine Menge von logischen Klauseln:

$$(X1 \text{ or } X2 \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } X2 \text{ or } X3)$$

Versuch:
X1=W
X2=W
X3=F

Frage: Gibt es eine Wahrheitswertbelegung fuer die Variablem Xi (Xi=Wahr oder Xi=Falsch) durch die jede Klausel wahr wird ?

SATISFIABILITY (SAT)

Gegeben: Eine Menge von logischen Klauseln:

$(\boxed{X1} \text{ or } \boxed{X2} \text{ or } \boxed{\overline{X3}})$ ✓

$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$ —

$(\overline{X1} \text{ or } \overline{X2} \text{ or } \boxed{\overline{X3}})$ ✓

$(\overline{X1} \text{ or } \boxed{X2} \text{ or } X3)$ ✓

Versuch:
X1=W
X2=W
X3=F

Frage: Gibt es eine Wahrheitswertbelegung fuer die Variablem Xi (Xi=Wahr oder Xi=Falsch) durch die jede Klausel wahr wird ?

SATISFIABILITY (SAT)

Gegeben: Eine Menge von logischen Klauseln:

$(X1 \text{ or } X2 \text{ or } \overline{X3})$

$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$

$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$

$(\overline{X1} \text{ or } X2 \text{ or } X3)$

2. Versuch:
X1=W
X2=F
X3=W

Frage: Gibt es eine Wahrheitswertbelegung fuer die Variablem Xi (Xi=Wahr oder Xi=Falsch) durch die jede Klausel wahr wird ?

SATISFIABILITY (SAT)

Gegeben: Eine Menge von logischen Klauseln:

$(\boxed{X1} \text{ or } X2 \text{ or } \overline{X3})$ ✓

$(\overline{X1} \text{ or } \boxed{\overline{X2}} \text{ or } \boxed{X3})$ ✓

$(\overline{X1} \text{ or } \boxed{\overline{X2}} \text{ or } \overline{X3})$ ✓

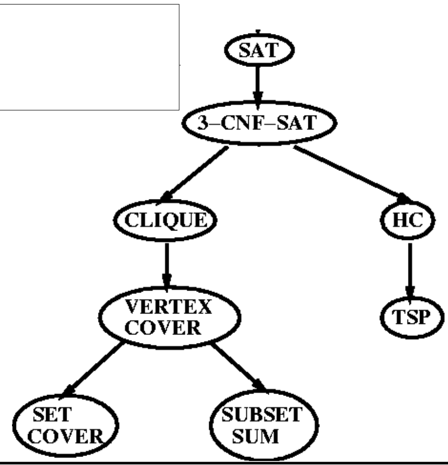
$(\overline{X1} \text{ or } X2 \text{ or } \boxed{X3})$ ✓

2. Versuch:
X1=W
X2=F
X3=W
Lösung !
Klauselmenge erfüllbar!

Frage: Gibt es eine Wahrheitswertbelegung fuer die Variablem Xi (Xi=Wahr oder Xi=Falsch) durch die jede Klausel wahr wird ?

Satz von Cook: **SAT ist NP-vollständig**.

SAT war das erste Problem, welches als NP-vollständig bewiesen wurde.

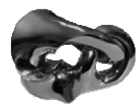


Tausende weitere Probleme

NP=P ?

Das wichtigste offene Problem der Theoretischen Informatik!

Clay Mathematical Institute



$1.000.000 fuer die Lösung

Zur vollständigen Lösung von NPV-Problemen kennt man keine polynomiellen Algorithmen.

Alle bekannten Algorithmen zählen exponentiell viele Lösungskandidaten auf!

15

WAS TUN ?

Lösungsansätze

NP-vollständige Probleme treten in der Praxis häufig auf.

Sie müssen mit akzeptablen Methoden gelöst werden.

Drei Ansätze:

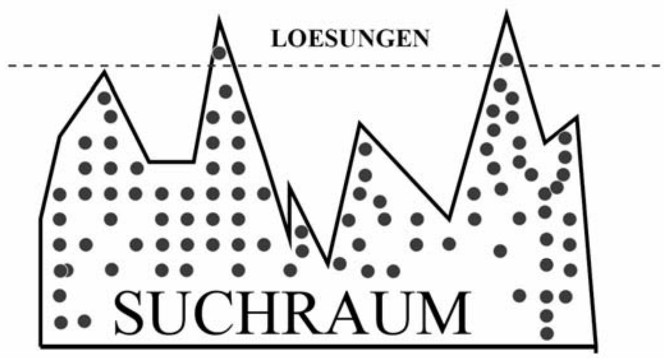
- Randomisierte lokale Suche
- Approximation
- Identifikation leicht lösbarer Subklassen

Randomisierte lokale Suche

Vorgegeben: Bewertungsfunktion für Lösungskandidaten; Zeitlimit.

Verfahren:

1. Erzeuge zufälligen Lösungskandidaten.
2. Führe solange wie möglich lokale Verbesserungen durch.
3. Wenn Lösung gefunden: Ausgeben und Programm beenden.
4. Wenn Zeitlimit erreicht, Abbruch: "Keine Lösung gefunden"
5. Gehe zu Schritt 1.



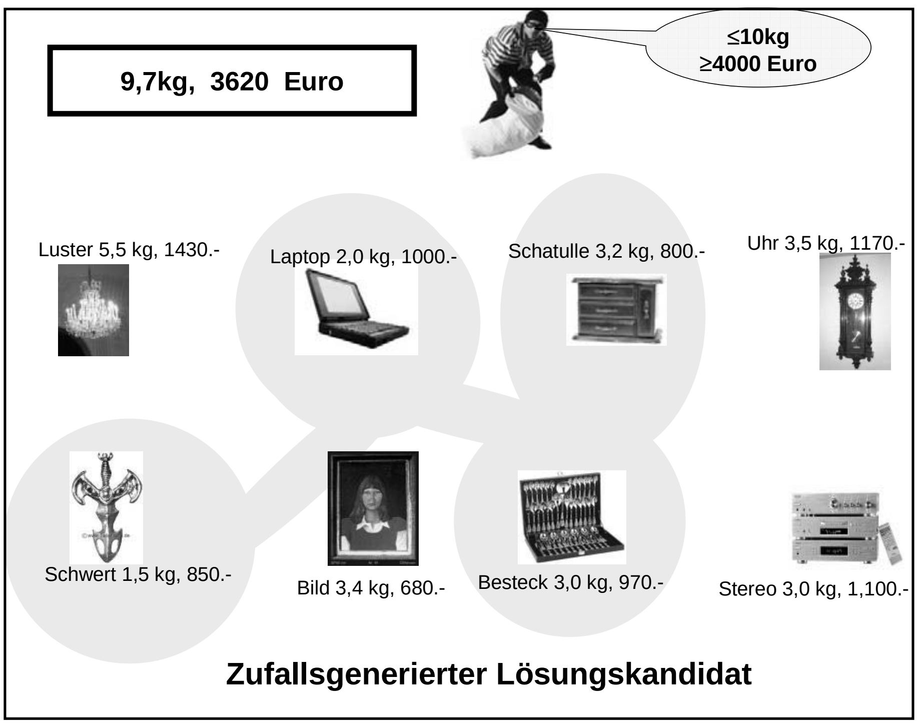
Beispiel für Bewertungsfunktion (bei Dreifärbbarkeit): Anzahl der korrekt gefärbten Kanten.

Verfeinerungen: Gradientensuche, Tabusuche, Simuliertes Ausglühen, genetische Algorithmen, usw.

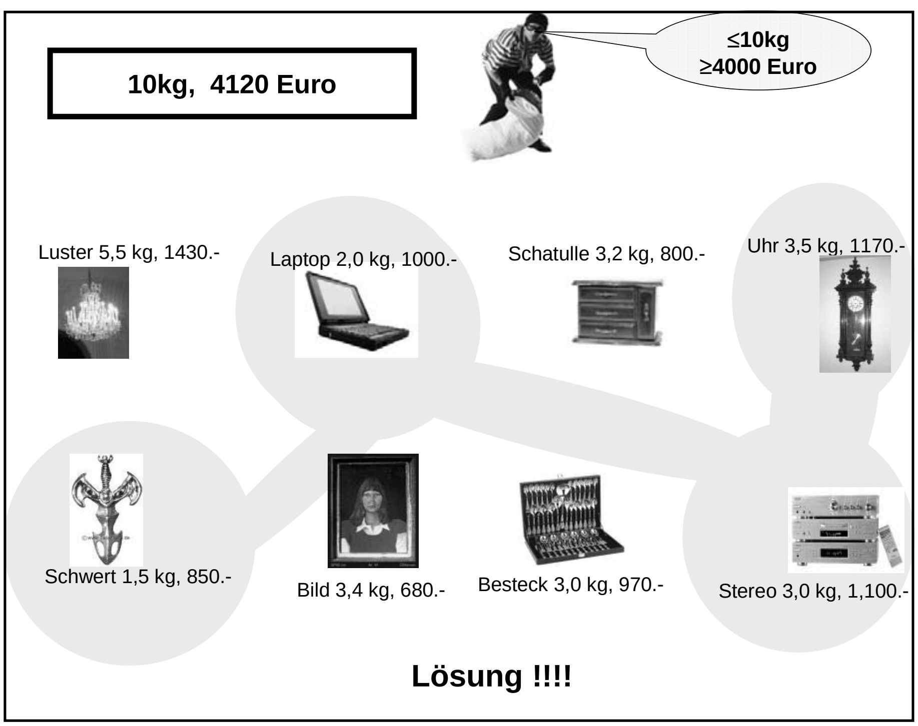
Rucksackproblem:

Lokale Verbesserungen können sein:

- **Streichen eines Objekts**
- **Hinzufügen eines Objekts**
- **Austausch von 2 Objekten**




10kg, 3990 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-
Zu spät, aber eine Lösung lag nahe!!!!

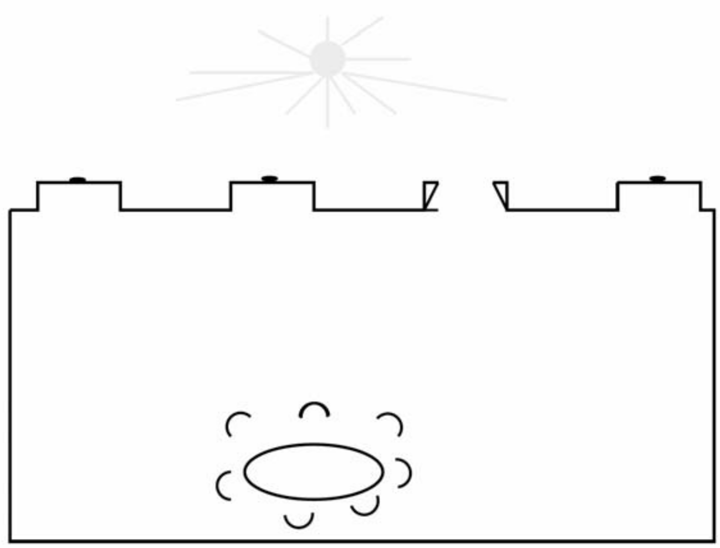

10kg, 4120 Euro
≤10kg
≥4000 Euro
Luster 5,5 kg, 1430.-
Laptop 2,0 kg, 1000.-
Schatulle 3,2 kg, 800.-
Uhr 3,5 kg, 1170.-
Schwert 1,5 kg, 850.-
Bild 3,4 kg, 680.-
Besteck 3,0 kg, 970.-
Stereo 3,0 kg, 1,100.-
Lösung !!!!

Vorteile der randomisierten lokalen Suche:

- Funktioniert erstaunlich gut für viele Probleme von Praktischer Relevanz.
- Führt dann mit hoher Wahrscheinlichkeit zum Erfolg.
- Liefert vollständige Lösungen.
- Ist ein natürliches Verfahren.

Nachteile:

- Funktioniert schlecht, falls keine Lösung existiert.
- Liefert keinen "Unlösbarkeitsbeweis"
- Funktioniert schlecht bei Instanzen mit wenigen Lösungen.
- Funktioniert nur mit Bewertungsfunktion. (Z.B. nicht geeignet zum Knacken von codes.)



SAT

$(X1 \text{ or } X2 \text{ or } \overline{X3})$

$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$

$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$

$(\overline{X1} \text{ or } X2 \text{ or } X3)$

Approximation

Oft genügt suboptimale “Lösung”

Nicht zu weit vom wahren Optimum entfernt…

Approximation für das Rucksackproblem

Luster 5,5 kg, 1430.-

281 Euro/kg

Bild 3,4 kg, 680.-

200 Euro/kg

Besteck 3,0 kg, 970.-

323,3 Euro/kg

Methode “Greedy+”

1.) Ordne Gegenstände absteigend nach Preis/Kilo
2.) Fülle eine Lösungsmenge in dieser Reihenfolge auf
3.) Führe einige “Bereinigungen” durch

Man kann zeigen:
Methode Greedy+ fuehrt zu einer Lösung L mit:

½ Opt ≤ L ≤ Opt

21

Beispiele

- γ(RUCKSACK) $= 0$; beliebig approximierbar.
- γ(TSP) $= 1$ außer P=NP; überhaupt nicht approximierbar.
- γ(TSP/EUCLID.) $\leq \frac{1}{3}$.

$\gamma(A) = 0$ bedeutet nicht, daß A polynomiell lösbar!

22

Vorteil der Approximationsmethode:

- Approximation in der Praxis oft ausreichend (brauche keine exakte Lösung).
- Auch auf manche beweisbar exponentielle Probleme anwendbar.

Nachteile:

- Wenige praktische Probleme approximierbar.
- Nicht auf Entscheidungsprobleme anwendbar.

Approximationstheorie: Eleganter und reichhaltiger Zweig der theoretischen Informatik. (Doz. Wöginger, TU Graz).

23

Identifikation von Polynomiellen Subklassen

Hohe Komplexität oft nur in "worst cases" gegeben.

Vertrackte Struktur der worst case Probleminstanzen.

Für Eingaben einfacherer Struktur würde polynomieller Algorithmus existieren.

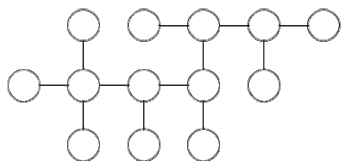
In der Praxis sind viele Eingabeinstanzen einfach.

Daher:

- Definiere geeignete polynomiell lösbare Subklassen v. Instanzen.
- Zeige, daß Zugehörigkeitstest polynomiell ist.
- Entwickle gute polynomielle Algorithmen für diese Klassen.

Beispiel: Geringe Graph-Zyklizität

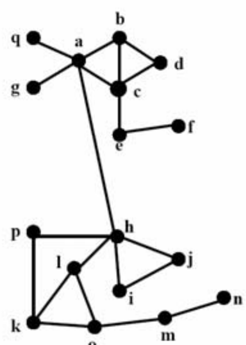
Jeder azyklische Graph ist 3-färbbar



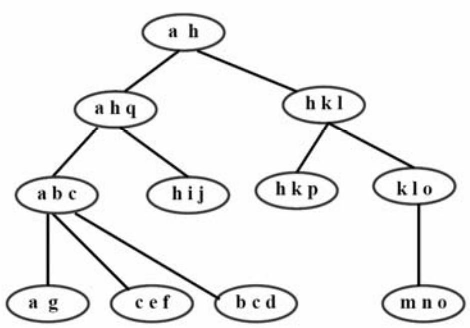
Kein azyklische Graph enthält Hamiltonschen Kreis

Für Graphen mit wenig Kreisen bzw. “geringer Zyklizität” sind fast alle NPV-Probleme leicht (= polynomiell) lösbar.

25

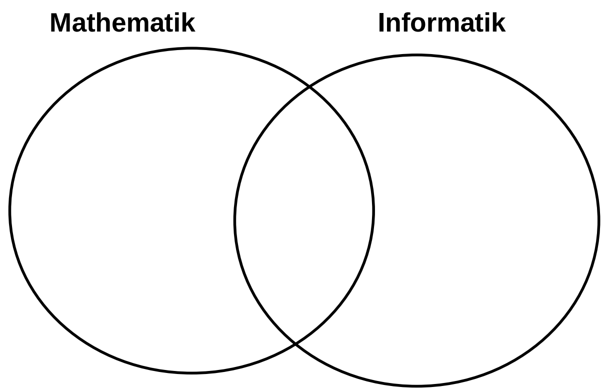


Graph G

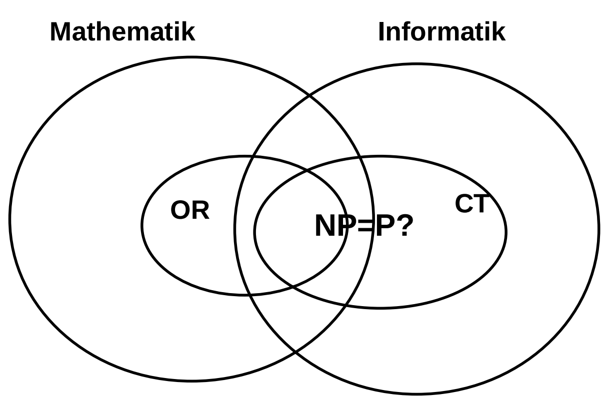


Tree decomposition of width 2 of G

Schluss



Schluss



OR= Operations Research CT= Complexity Theory

24

Beschränkte Baumweite (bounded treewidth)

Bei graphbasierten Problemen ist hohe Komplexität meist durch Zyklizität bedingt.

Für *azyklische* Graphen sind Probleme oft trivial lösbar. (→3COL)

Viele Graphenprobleme sind auch für Instanzen *niederer Zyklizität* lösbar.

Maß für den Zyklizitätsgrad eines Graphen?

Baumweite (Robertson und Seymour)

Basiert auf dem Konzept einer Baumdekomposition.

26

Baumweite eines Graphen: Kleinstmögliche Weite von Baumzerlegung.

$bw(G)$

Satz. (Bodlaender): Für fixe Konstante k kann man in linearer Zeit feststellen, ob $bw(G) \leq k$.

Viele NP-vollständige Graphprobleme sind für Graphen mit konstant beschränkter Baumweite in linearer Zeit lösbar.

Bsp. Dreifärbbarkeit (3COL).

27

Verbindung zur Logik

Satz. (Fagin): Jedes NP-Problem über Graphen läßt sich durch eine existentielle Formel der Prädikatenlogik zweiter Stufe (SO) darstellen.

NP=ESO

Monadisches SO (MSO): Teilklasse von SO, nur Mengenvariablen, aber keine mehrstlligen Relationenvariablen zugelassen.

Dreifärbbarkeit $\in$ MSO.

Satz. (Courcelle): Alle Graphprobleme, die sich in MSO darstellen lassen, sind polynomiell für Graphen mit konstant beschränkter Baumweite.

28

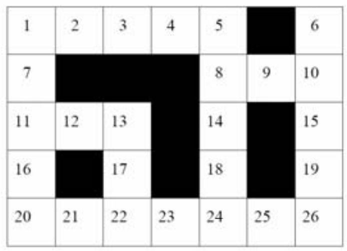
Dreifärbbarkeit als MSO-Formel

Der Graph wird als Struktur $\langle U, E \rangle$ angesehen.

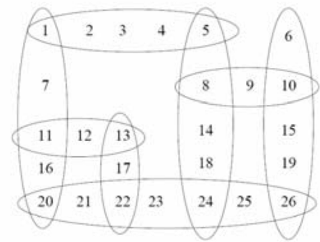
U: Universum, Menge der Knoten; E: Kanten, 2stellige Relation.

$$
\begin{array}{lll}
(\exists R, G, B) & [& (\forall x\, (R(x) \vee G(x) \vee B(x))) \\
& \wedge & (\forall x (R(x) \Rightarrow (\neg G(x) \wedge \neg B(x)))) \\
& \wedge & \ldots \\
& \wedge & \ldots \\
& \wedge & (\forall x, y (E(x,y) \Rightarrow (R(x) \Rightarrow (G(x) \vee B(y))))) \\
& \wedge & (\forall x, y (E(x,y) \Rightarrow (G(x) \Rightarrow (R(x) \vee B(y))))) \\
& \wedge & (\forall x, y (E(x,y) \Rightarrow (B(x) \Rightarrow (R(x) \vee G(y)))))\,]
\end{array}
$$

Viele Probleme lassen sich besser durch *Hypergraphen* als durch Graphen beschreiben.



CSP instance I



Hypergraph $\mathcal{H}_I$

Dekomposition von Hypergraphen

Dzt. intensives Forschungsgebiet. Vorschläge:

- *Hypergraph-Dekomposition/-Weite* Robertson und Seymour.
- *Query-Dekomposition/-Weite* Chekuri und Rajaraman.

Satz. (Chekuri und Rajaraman): Hypergraphbasierte CSP- Probleme sind polynomiell, wenn Hypergraph-Weite konstant beschränkt.

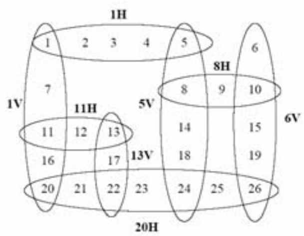
Frage: Kann man Hypergraphen von beschr. Weite leicht erkennen?

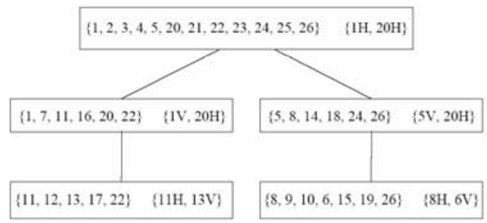
Antwort: nein!

Satz. (G., Leone, Scarcello): Für $k > 3$ ist dieses Erkennungsproblem NP-vollständig.

Hypertree Decompositions

Neuer Ansatz zur Hypergraphendekomposition.





Hypergraph $\mathcal{H}_I$ A hypertree decomposition of width 2

Fakten zur Hypertree-Weite

Satz. Hypergraphbasierte CSP-Probleme sind polynomiell, wenn Hypertree-Weite konstant beschränkt.

Satz. Für jede fixe Konstante k kann in polynomieller Zeit festgestellt werden, ob ein Hypergraph durch k beschränkte Hypertree-Weite hat.

Satz. $\forall G$: Hypertree-Weite(G) $\leq$ Query-Weite(G).

Daher ist unser Ansatz allgemeiner und umschließt mehr Instanzen.

Vorteile der Verwendung solcher polynomieller Subklassen

- In vielen Anwendungsbereichen haben fast alle "natürlichen" Eingaben Hypertree-Weite ≤ 5.
- Auch negative Instanzen werden erkannt.
- Hoher Parallelisierungsgrad der Subklassen (LOGCFL).

Nachteile:

- Sagt nichts über Instanzen von großer Weite.
- Bei hypergraphbasierten Methoden steigt Polynomgrad mit Baumweite.

Schluß

- Was ist ein Entscheidungsproblem / Berechnungsproblem ?
- Komplexität von Algorithmen und Problemen
- NP-vollständige Probleme
- Drei Lösungsansätze:
  - Randomisierte lokale Suche
  - Approximation
  - Identifikation leicht lösbarer Subklassen

In allen Teilbereichen offene Probleme. Aktive Forschung.

Quantencomputer: Nicht zur polynomiellen Lösung von NPV-Probleme geeignet; aber für Subklasse RP von NP.

Approximation von Berechnungsproblemen

Ziel: Finde annähernd optimale Lösung bzgl. Bewertungsfunktion w.

Ein Maximierungsproblem ist ϵ-approximierbar, wenn es es einen Algorithmus T mit polynomieller Laufzeit gibt, sodaß:

Für alle Instanzen x

$$\frac{max w(x) - w(T(x))}{max w(x)} \leq \epsilon$$

Approximationsgrad $\gamma(A)$ eines Maximierungsproblems A: Größte untere Schranke für ϵ-Approximierbarkeit.

$$\gamma(A) = \inf\{\epsilon \mid A \text{ ist } \epsilon - \text{approximierbar}\}.$$