Algorithmen für schwierige Probleme

Dr. Britta Dorn
Prof. Dr. Jacobo Torán

Wintersemester 2011/12

19. Oktober 2011

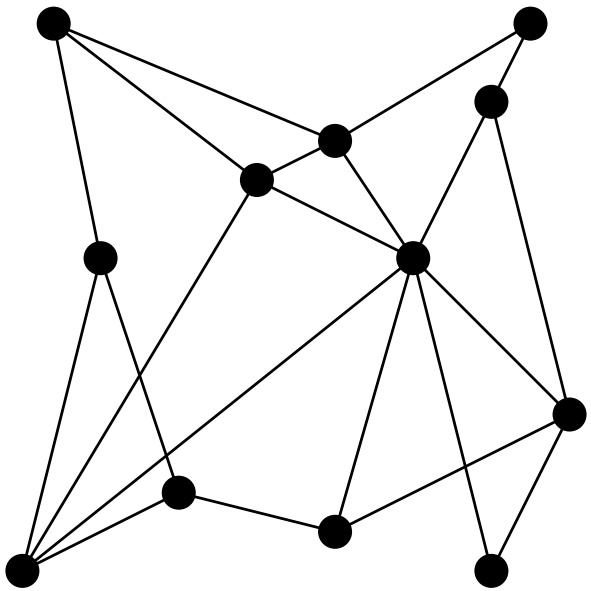
Infos

Vorlesung Mittwoch und Donnerstag 14–16

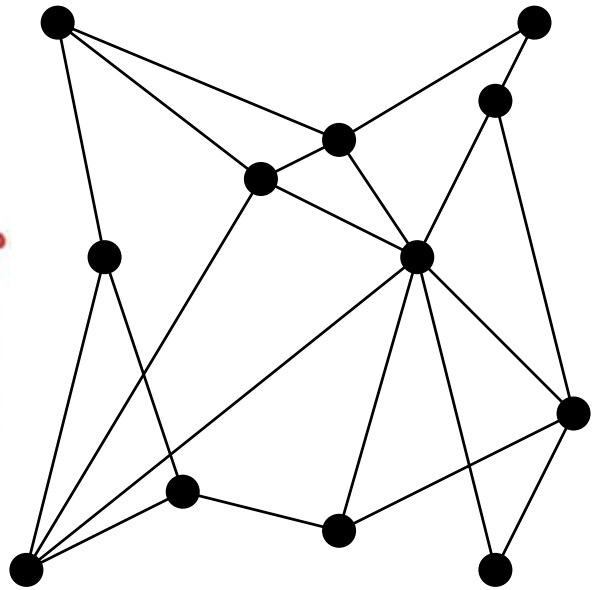
Übungsblätter

Prüfung

Feuerwehrproblem



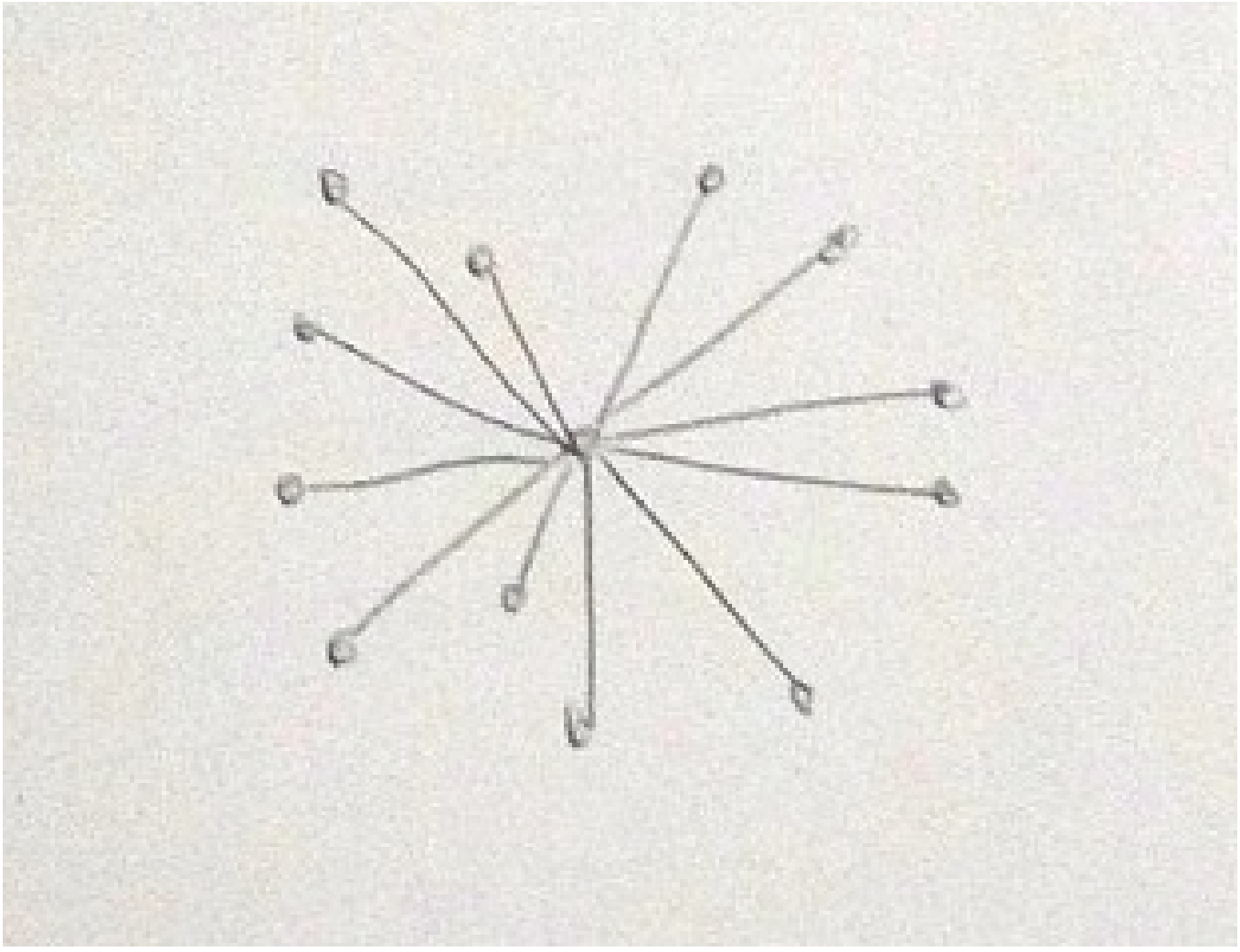
Feuerwehrproblem



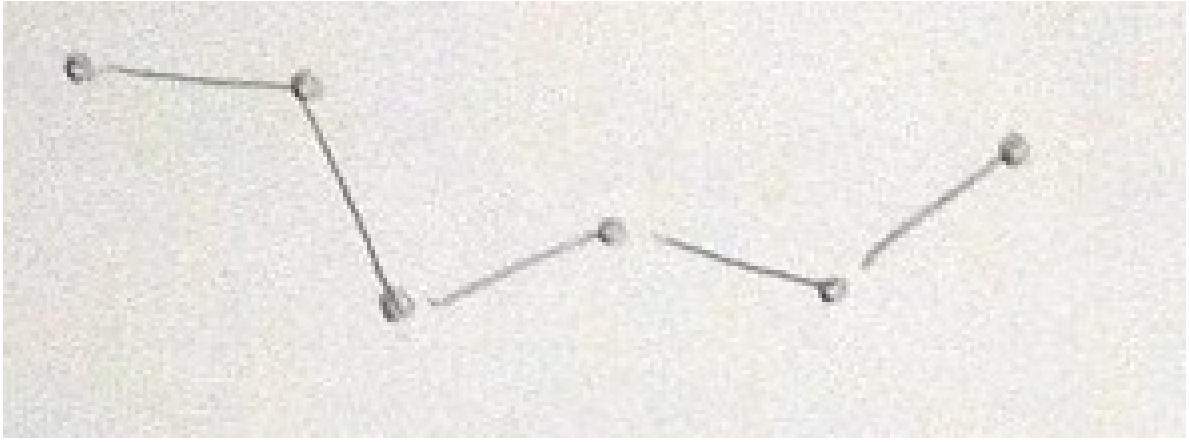
Feuerwehrproblem

Jeder Stadt braucht eine Feuerwehrstation
— oder zumindest eine in einer Nachbarstadt

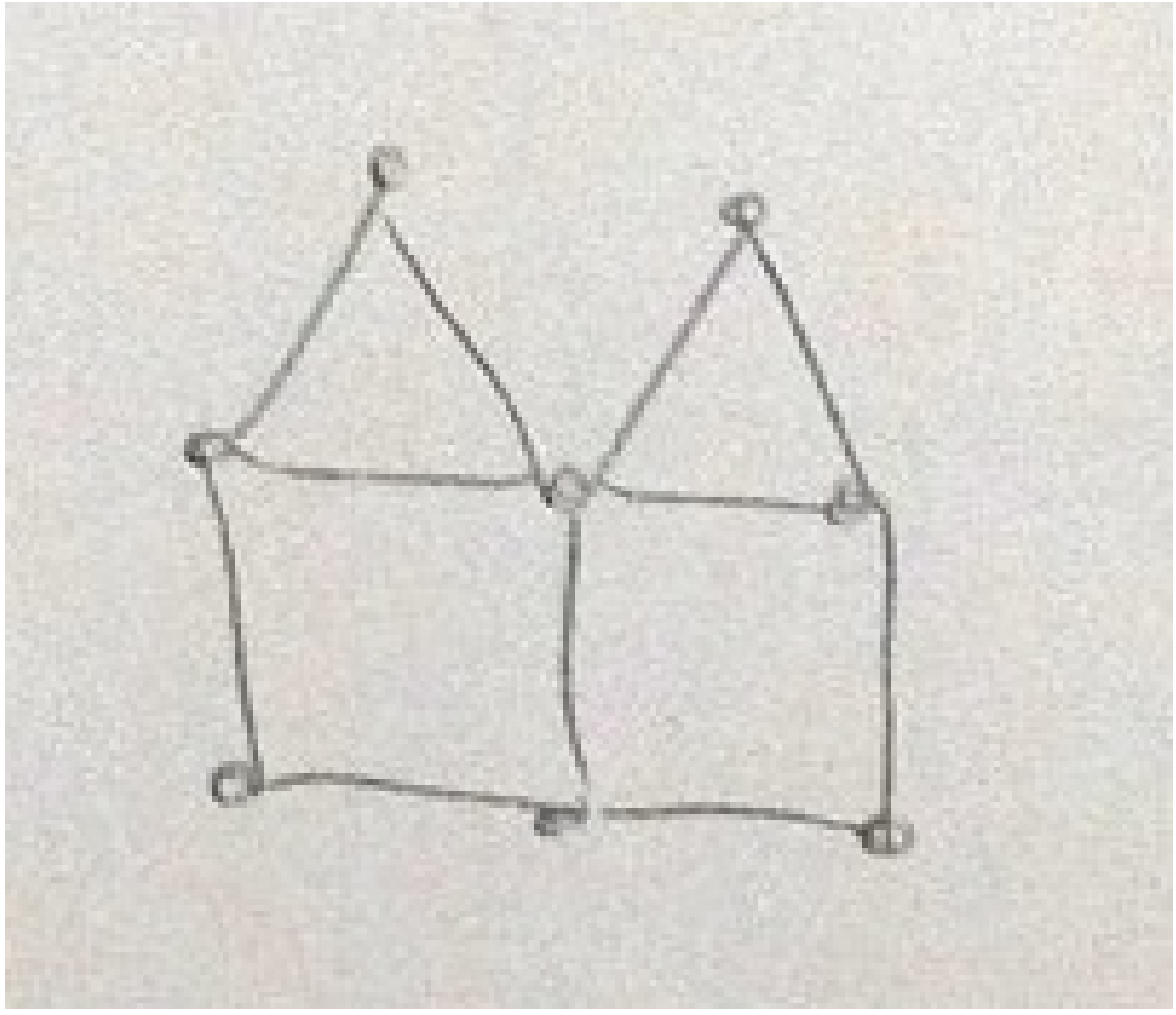
Feuerwehrproblem (Dominating Set)



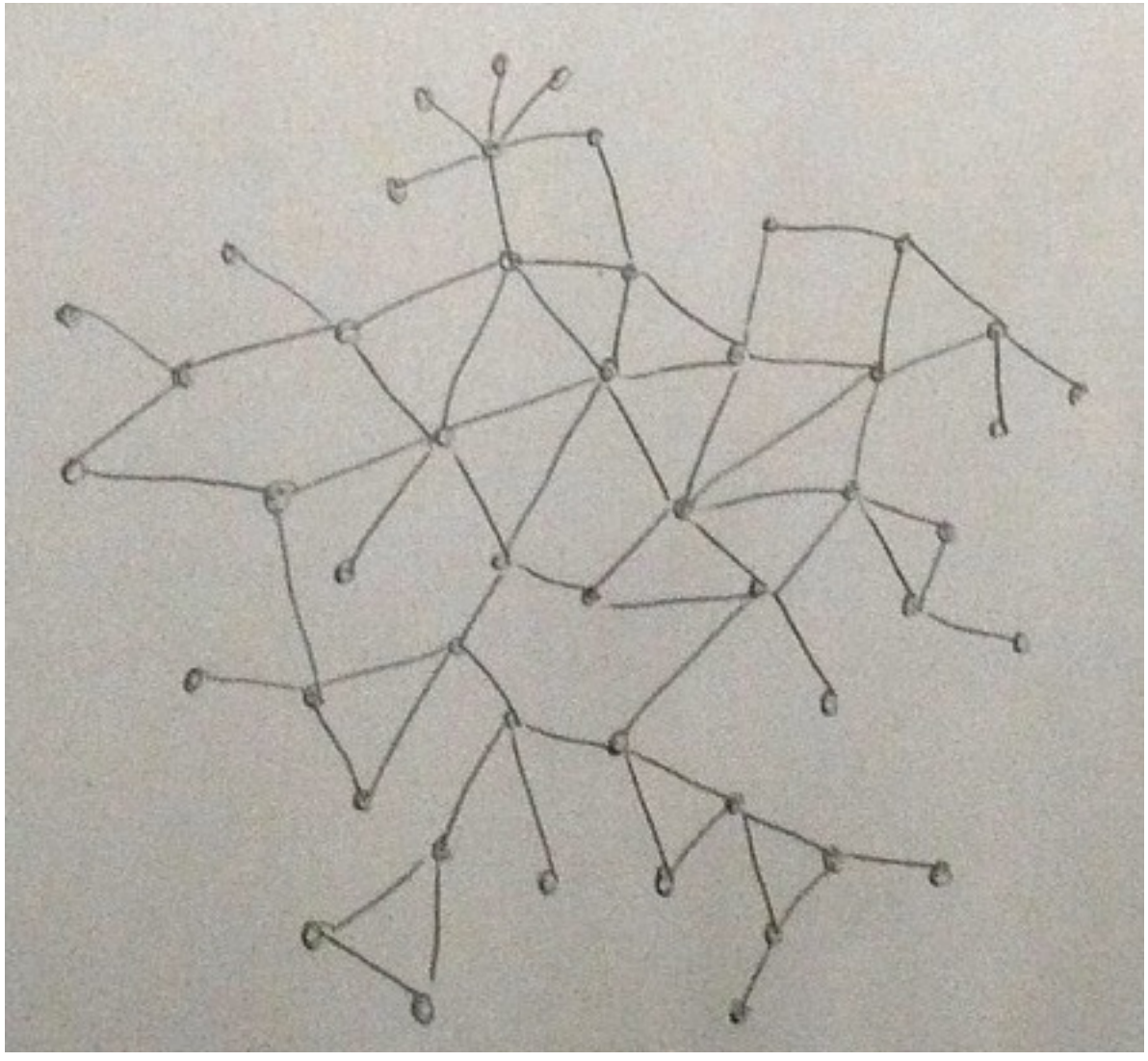
Feuerwehrproblem (Dominating Set)



Feuerwehrproblem (Dominating Set)



Feuerwehrproblem (Dominating Set)



Feuerwehrproblem (Dominating Set)

$n =$ Anzahl der Städte, $k \leq n$. Reichen k Feuerwehrstationen aus?
Ist das Problem einfach/schwierig?

"Einfach": Polynomiell

Brute Force: Probiere alle Teilmengen von k Städten aus.
Problem: 2^n viele Teilmengen bei n Städten. **Nicht** polynomiell!

"'Schwierig": Bisher nichts wesentlich Besseres als Brute Force, nur exponentielle Laufzeiten oder schlimmer

P und NP-schwer

Ein paar Probleme

Gegeben:

- Gruppe von Studenten (Knoten)
- bekannt, wer mit wem befreundet ist (Kanten)

1. Matching: Studenten in 2er-Gruppen einteilen (nur befreundete zusammen)
2. Partition into Triangles: 3er-Gruppen
3. Clique: größere Gruppen (jeder mag jeden)
4. Hamiltonian cycle: Studenten sitzen um runden Tisch, keiner soll neben jemandem sitzen, den er nicht leiden kann.

schwer?/einfach?

Ein paar Probleme

1. Matching: Studenten in 2er-Gruppen einteilen (nur befreundete zusammen)
2. Partition into Triangles: 3er-Gruppen
3. Clique: größere Gruppen (jeder mag jeden)
4. Hamiltonian cycle: Studenten sitzen um runden Tisch, keiner soll neben jemandem sitzen, den er nicht leiden kann.

schwer?/einfach?

Alle Probleme haben gemeinsam:
Falls jemand Lösung vorschlägt (z.B. Sitzplan für runden Tisch), dann kann man "einfach und schnell" nachprüfen, ob sie stimmt.

Laufzeitvergleiche

	polynomiell n^2		exponentiell 2^n	
	n^2	Dauer (10^8 Ops/s)	2^n	Dauer (10^8 Ops/s)
$n = 2$	4	0,04 μs	4	0,04 μs
$n = 10$	100	1 μs	1024	10 μs
$n = 50$	2 500	25 μs	10^{15}	ca. 116 Tage
$n = 100$	10 000	0,1 ms	10^{30}	ca. $3 \cdot 10^{14}$ Jahre

Laufzeitvergleiche

Vergleich der Funktionswerte verschiedener Polynom- und Exponentialfunktionen für $n = 50$

polynomiell	
Komplexität	Dauer (10^8 Ops/s)
n^2	25 μs
n^3	1 ms
n^5	3 s
n^8	4,5 Tage
n^{10}	31 Jahre

exponentiell	
Komplexität	Dauer (10^8 Ops/s)
1.25^n	700μs
1.5^n	6 min
1.75^n	4 h
2^n	130 Tage
3^n	230 Mio Jahre

Grad des Polynoms bzw. Basis der Exponentialfunktion entscheidend!!

NP-schwere Probleme lösen

Zwei Möglichkeiten:

nicht optimal (aber schnell)

- Approximation
- Randomisiert
- Heuristik

optimal (aber nicht so schnell)

- Guter exponentieller Algorithmus (kleine Basis)
- Parametrisierte Algorithmen

Feuerwehrproblem (auf planaren Graphen)

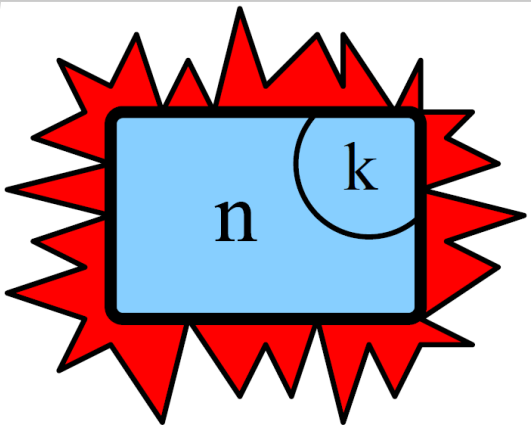
Für das Feuerwehrproblem gilt:

- NP-schwer

  kein Algorithmus bekannt, der das Problem in Polynomzeit löst
- **Aber:** falls die Lösungsmenge (k) klein ist (falls nur wenige Feuerwehrstationen nötig sind), lässt sich das Problem schnell lösen.

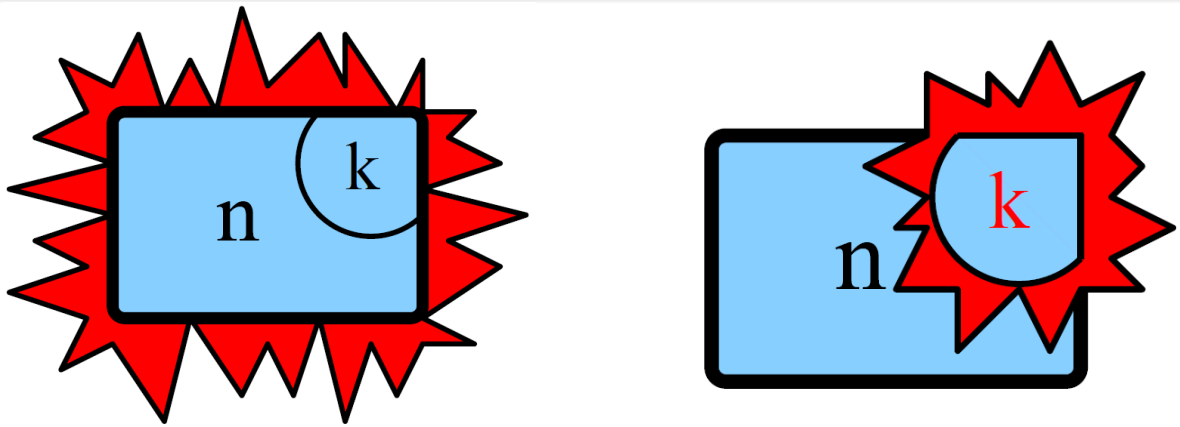
Fixed-parameter tractability

Grund: Laufzeit ist exponentiell



Fixed-parameter tractability

Grund: Laufzeit ist exponentiell, aber verantwortlich für den exponentiellen Teil ist im Wesentlichen das k (Parameter)



Falls in bestimmen Fällen die Werte dieser Parameter klein sind: schnelle Lösung ist möglich!
Ziel: Finde Parameter, die Schuld an der schlimmen Laufzeit sind!

Beispiele für Parameter

Parameter

- Größe der Lösungsmenge (z.B.: Anzahl der Feuerwehrstationen, Größe der Gruppe, in der alle sich leiden könnnen, . . .)
- Strukturelle Parameter (z.B.: Maximaler Verzweigungsgrad, durchschnittliche Anzahl Freunde eines Studenten, . . .)
- . . .

Running Example: Vertex Cover

Vertex Cover

gegeben: $G = (V, E)$, ungerichteter Graph
$k \in \mathbb{N}$

Frage: Gibt es eine Teilmenge $C \subseteq V$ von Knoten mit $|C| \leq k$ so, dass jede Kante aus E mindestens einen Endpunkt in C hat?

Parametrisierte Algorithmen

1. Einführung
   - Motivation/Einordnung
   - Basics: Komplexität, Probleme (Graphen, Wahlsysteme, SAT)
   - Grundlagen der Parametrisierten Algorithmen
2. Algorithmische Methoden
   - Datenreduktion und Problemkerne
   - Suchbäume
   - Dynamisches Programmieren
   - Baumzerlegung von Graphen
   - Weitere Techniken
3. Komplexitätstheorie dazu
   - Parametrisierte Reduktion
   - Parametrisierte Komplexitätsklassen, vollständige Probleme
   - Verbindungen zu klassischer Komplexitätstheorie und Approximation