Algorithmische Konzepte

TECHNISCHE UNIVERSITÄT DARMSTADT

Grundlagen der Informatik II
Sommersemester 2015

Karsten Weihe

Schnitt:???

Schwere algorithmische Probleme

TECHNISCHE UNIVERSITÄT DARMSTADT

Antwort aus der Praxis:

- **Für jeden denkbaren Algo gibt es Beispiele von moderater Größe, auf denen der Algo mehr Laufzeit benötigt als das Universum.**
- **Algorithmische Herausforderung: Algo finden, so dass die realen Anwendungs-beispiele nicht so schlimm sind.**

Zweistufig

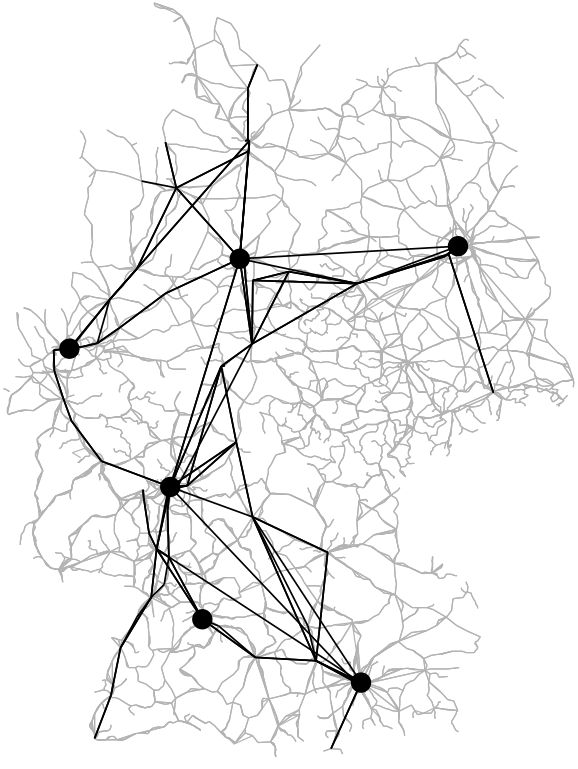
TECHNISCHE UNIVERSITÄT DARMSTADT

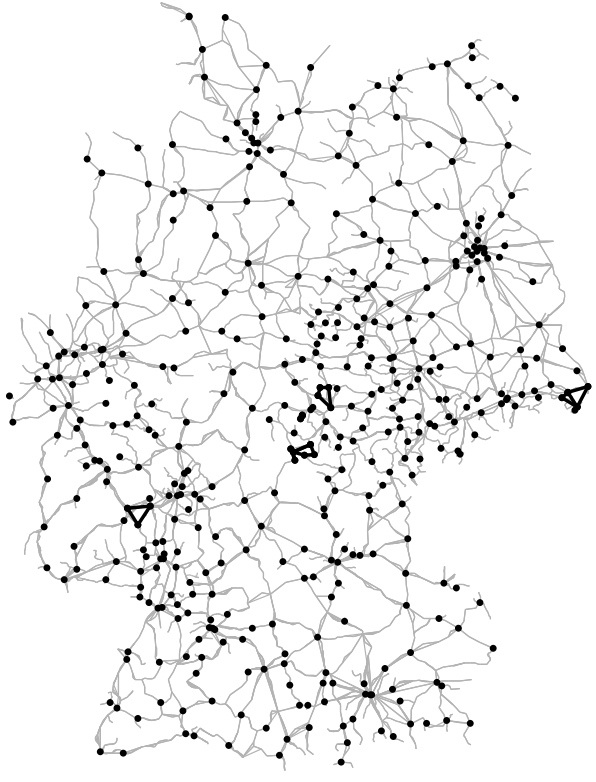
- **Erste Stufe:**
  - ➢ **Datenmenge reduzieren.**
  - ➢ **Kritische Punkte identifizieren und vorab möglichst gut lösen.**
  - ➢ **Lösung zuerst grob bestimmen.**
- **Zweite Stufe: Restproblem lösen.**

Vorab Datenmenge reduzieren

TECHNISCHE UNIVERSITÄT DARMSTADT

Beispiel:





Vorab Datenmenge reduzieren

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Alle Züge, die an Bahnhof A halten, halten auch an Bahnhof B.**
  - ➢ **Bahnhof A kann eliminiert werden.**
- **Alle Bahnhöfe, die Zug A anfährt, fährt auch Zug B an.**
  - ➢ **Zug A kann eliminiert werden.**

Bis keine der beiden Regeln mehr anwendbar ist.

Kritische Punkte vorab

TECHNISCHE UNIVERSITÄT DARMSTADT

Beispiele in Großbetrieben:

- **Planung von maschinellen Arbeiten**
  - ➢ **Engpassmaschine(n) zuerst**
- **Planung von Betriebszugbewegungen**
  - ➢ **Engpassstrecken/-punkte zuerst**

Kritische Punkte vorab

TECHNISCHE UNIVERSITÄT DARMSTADT

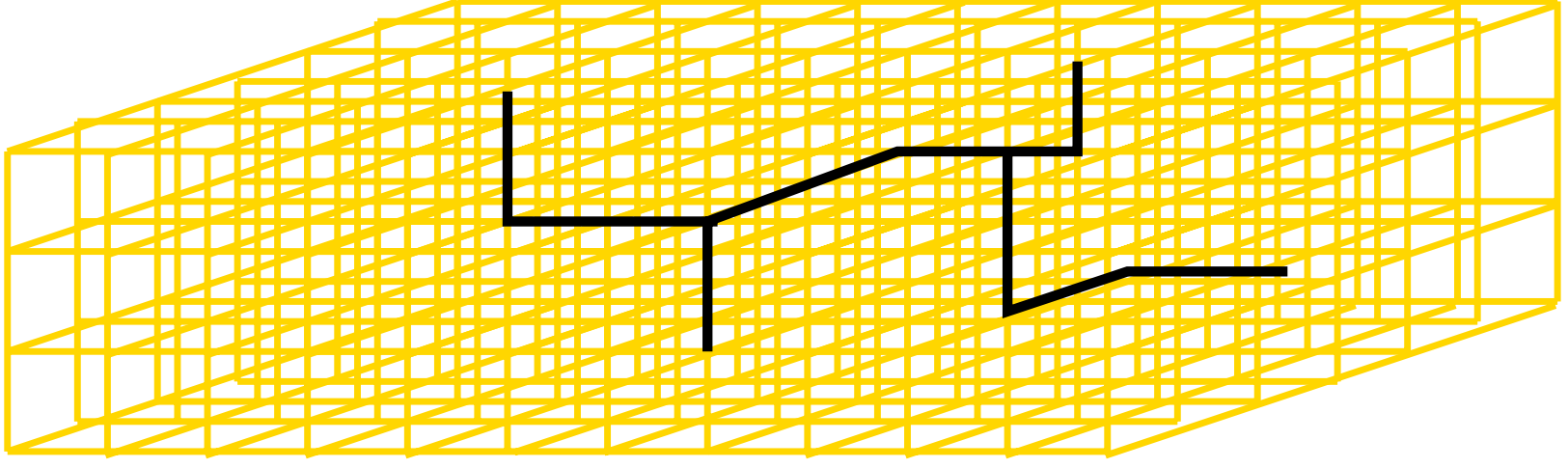
Beispiel Stundenplan:

- **Erst einen Plan nur für die schwierig einzuplanenden LVs erstellen.**
  - ➢ **Großes Publikum, viele Studiengänge, Ausschlüsse mit anderen LVs.**
- **Dann alle anderen LVs „drum herum“ einplanen.**

Kritische Punkte vorab

TECHNISCHE UNIVERSITÄT DARMSTADT

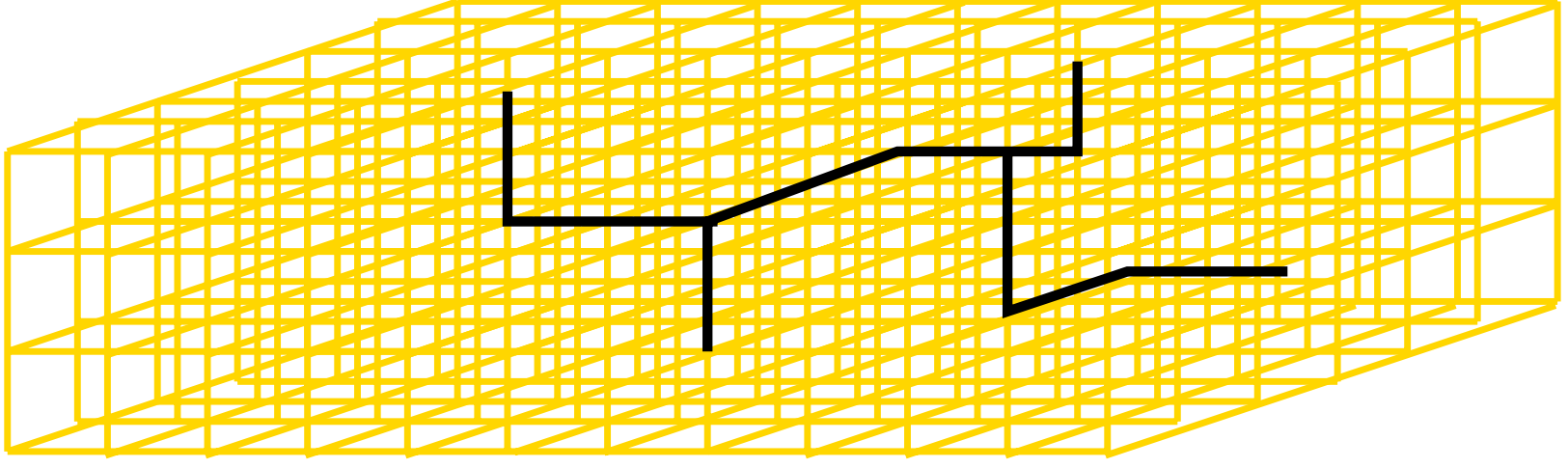
Beispiel Verdrahtung von Mikrochips: kritische Verbindungen zuerst.



Lösung zuerst grob bestimmen

TECHNISCHE UNIVERSITÄT DARMSTADT

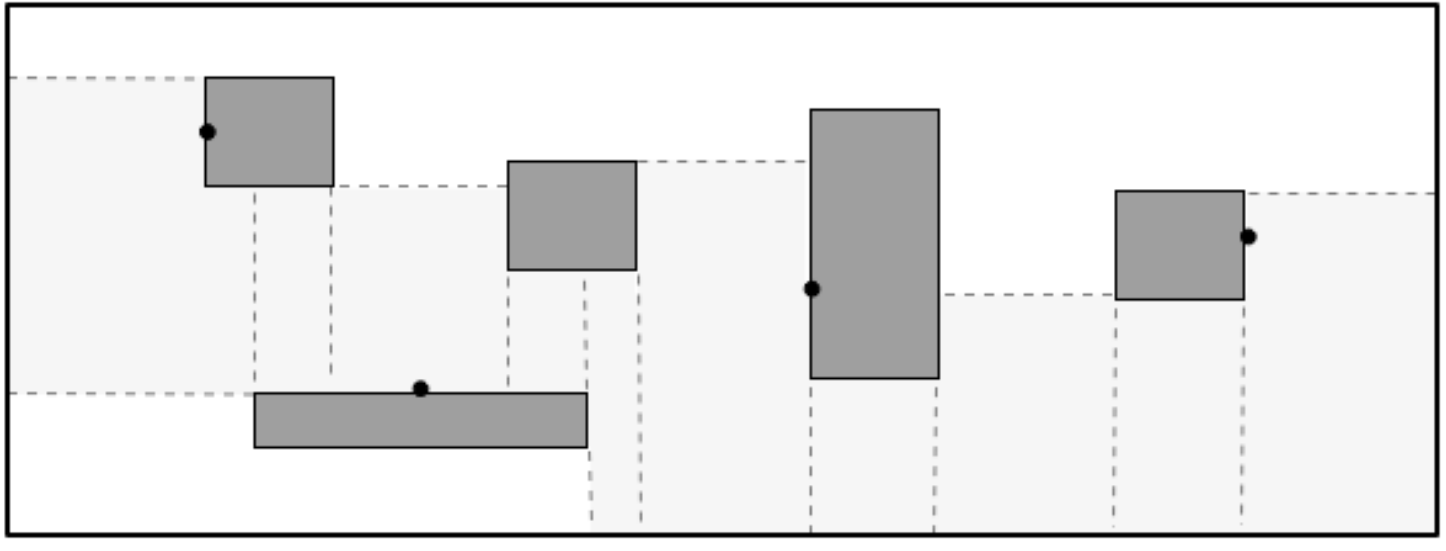
Beispiel Verdrahtung von Mikrochips: zuerst nur ungefähren Verlauf der Verbindungen festlegen.



Lösung zuerst grob bestimmen

TECHNISCHE UNIVERSITÄT DARMSTADT

Beispiel Verdrahtung von Mikrochips: zuerst nur ungefähren Verlauf der Verbindungen festlegen.



Lösung zuerst grob bestimmen

TECHNISCHE UNIVERSITÄT DARMSTADT

Beispiel in Großbetrieben:

- **Zeitliche Einplanung der einzelnen Aufgaben in den Planungshorizont zuerst nur ungefähr.**
- **Und zwar so, dass möglichst überall (vor allem an mglw. kritischen Punkten) ordentlich große Puffer bleiben.**
- **Danach dann Feinplanung mit Hilfe der Puffer.**

Divide and Conquer

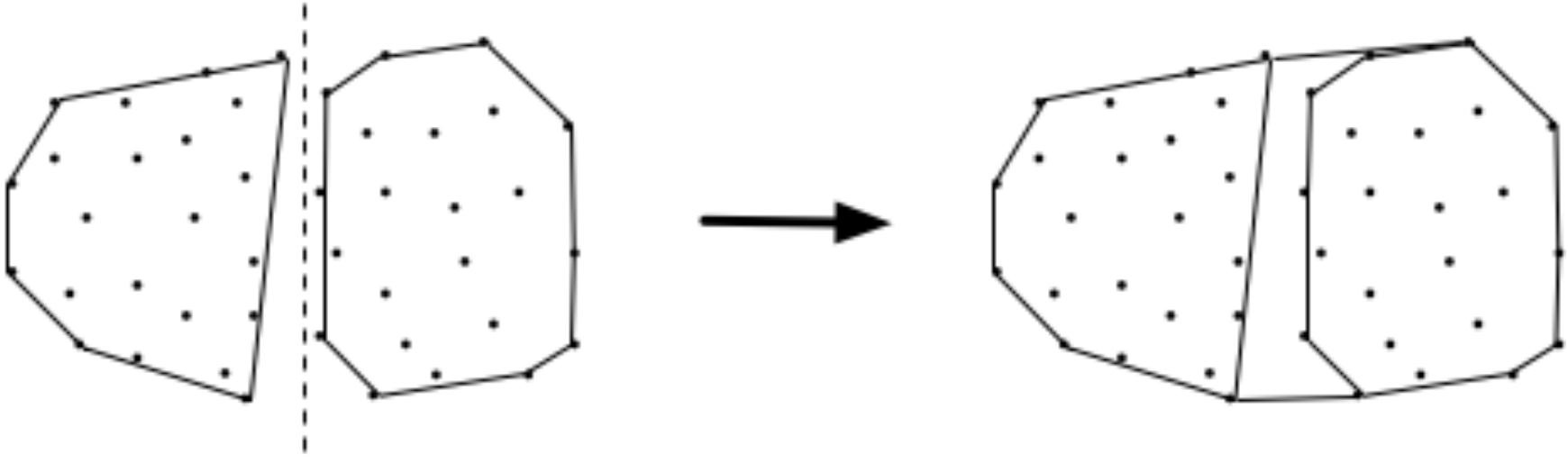
TECHNISCHE UNIVERSITÄT DARMSTADT

- **Passt auf jedes algorithmische Problem, das sich in mehrere *voneinander unabhängige* Teilprobleme zerlegen lässt.**
- ***Bisherige Beispiele*: Mergesort und Quicksort.**
- ***Weiteres Beispie*l: konvexe Hülle.**

➢ **Nächste Folie.**

Divide and Conquer

TECHNISCHE UNIVERSITÄT DARMSTADT



Dynamische Programmierung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Passt auf viele Probleme, die durch eine Rekursionsgleichung beschreibbar sind.
  - ➢ Rekursion wird in Iteration umgewandelt.
- Einfache Beispiele:
  - ➢ Fibonacci-Zahlen
  - ➢ Binomialkoeffizienten

Dynamische Programmierung

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Fibonacci-Zahlen:**
  - fib(0) = fib(1) = 0
  - fib(n) = fib(n-1) + fib (n-2) **für** n>1
- **Binomialkoeffizient:**
  - $\binom{n}{0} = \binom{n}{n} = 1$
  - $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ **für** 0<k<n

Fibonacci-Zahlen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Falls n = 1 oder n = 2: gib 1 aus und Ende
- f1 := f2 := 1
- Für i = 2 → n:
  - f := f1 + f2
  - f1 := f2
  - f2 := f
- Gib f aus und Ende

Binomialkoeffizienten

TECHNISCHE UNIVERSITÄT DARMSTADT

- Falls k = 0 oder k = n: gib 1 aus und Ende
- M: Matrix mit Indexbereich 0 ... n und 0 ... k
- Für i = 0 → n: M[i,0] := 1
- Für j = 1 → k: M[j,j] := 1
- Für i = 2 → n und j = 1 → min{k,i-1}:

  M[i,j] := M[i-1,j-1] + M[i-1,j]
- Gib M[n,k] aus und Ende

Binomialkoeffizienten

TECHNISCHE UNIVERSITÄT DARMSTADT

- Falls k = 1 oder k = n: gib 1 aus und Ende
- Falls k > floor(n/2): k := n-k // => *k <= floor(n/2)*
- A: Array mit Indexbereich 0 ... k
- Für i = 0 → k: A[i] := 1
- Für i = 1 → n-k und j = 1 → k:
  A[j] := A[j-1] + A[j]
- Gib A[k] aus und Ende

Greedy-Ansatz

TECHNISCHE UNIVERSITÄT DARMSTADT

- Passt auf jedes Problem, in dem Lösungen (möglichst gute) Auswahlen sind.
- Generelles Vorgehen:
  - Starte mit der leeren Auswahl.
  - Füge das profitabelste Element ein, so dass die bisherige Auswahl zusammen mit diesem Element weiterhin zu einer zulässigen Lösung erweiterbar ist.

Greedy-Ansatz

TECHNISCHE UNIVERSITÄT DARMSTADT

- Rucksackproblem
- TSP
- Steinerbaum
- Mengenpackung und Mengenüberdeckung
- Cliquen und unabhängige Mengen

Greedy-Ansatz

TECHNISCHE UNIVERSITÄT DARMSTADT

Falls Profite der Elemente konstant:

- **Alle Elemente vorab absteigend nach ihrem jeweiligen Profit sortieren**
- **In dieser Reihenfolge alle Elemente in der Schleife durchgehen.**
- **Element in aktuelle Auswahl einfügen, falls die bisherige Auswahl zusammen mit diesem Element weiterhin zu einer zulässigen Lösung erweiterbar ist.**

Greedy-Ansatz

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Liefert bei vielen algorithmischen Problem-stellungen eine recht gute Lösung.**
- **Algorithmus von Kruskal:**
  - **Beim MST liefert Greedy garantiert immer eine *optimale* Lösung.**
- **Bei den meisten anderen Probleme kann Greedy *beliebig schlechte* Lösungen liefern.**
- **Unterschied: Matroid-Struktur**

Greedy verallgemeinert

TECHNISCHE UNIVERSITÄT DARMSTADT

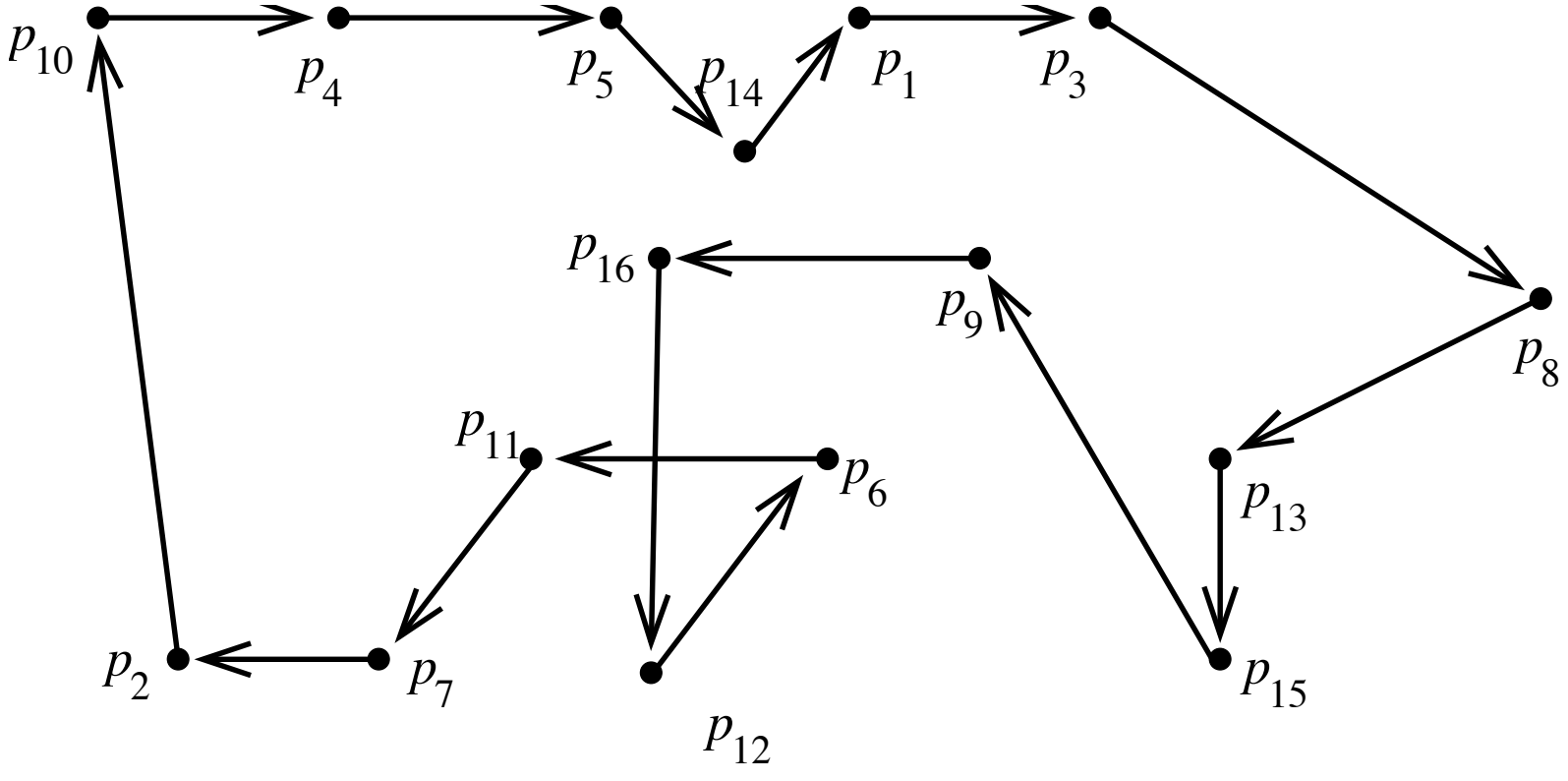
- **Anwendbar auf Einplanungsprobleme, Platzierungsprobleme u.ä.**
- ***Gemeinsamkeit*: Eine Lösung kommt durch eine Folge von Entscheidungen zustande.**
- **Greedy: Bei jeder Einzelentscheidung wird die am Besten erscheinende Option gewählt.**
  - **Ohne Berücksichtigung von Konsequenzen auf die später noch zu treffenden Entscheidungen.**

Backtracking

- Wenn die Entscheidungsfindung akut oder absehbar in eine Sackgasse läuft:
  - Ursachen für die Sackgasse identifizieren.
  - Letzte Entscheidungen rückgängig machen, bis mindestens eine Ursache entfernt ist.
  - Die nächste Entscheidung von diesem Zustand aus anders als zuvor treffen.
- Auch beim zweistufigen Vorgehen sinnvoll.

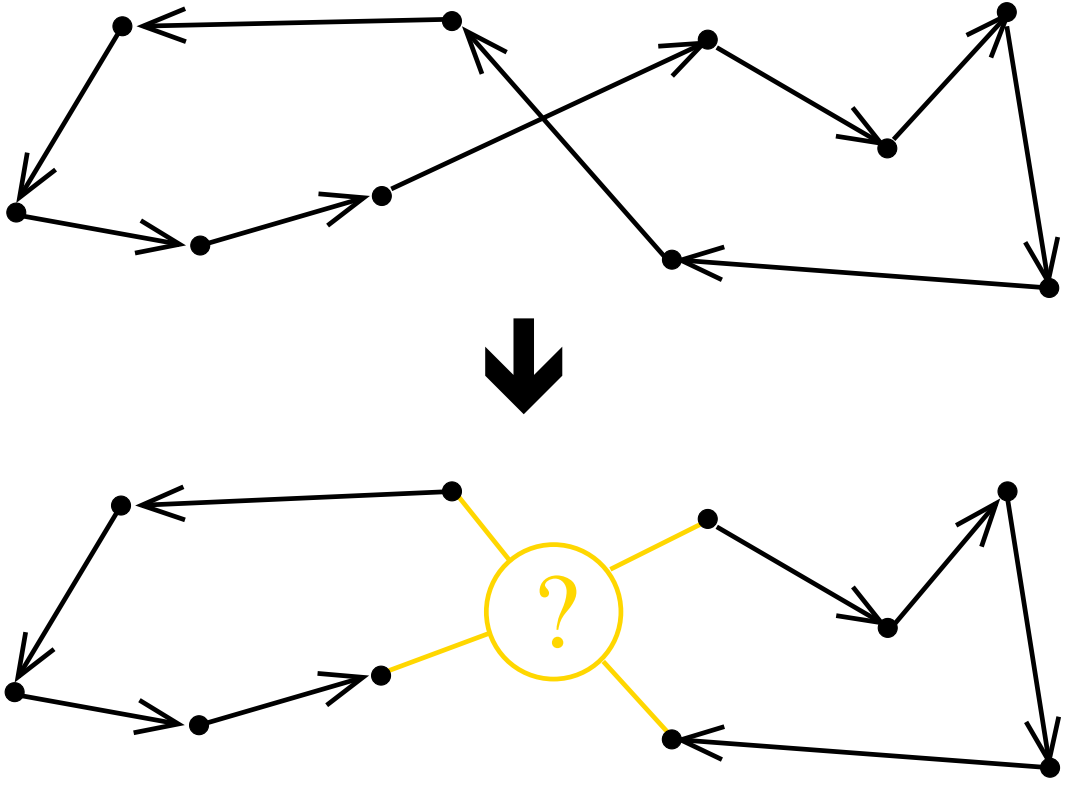
Lokale Suche: Beispiel TSP

TECHNISCHE UNIVERSITÄT DARMSTADT



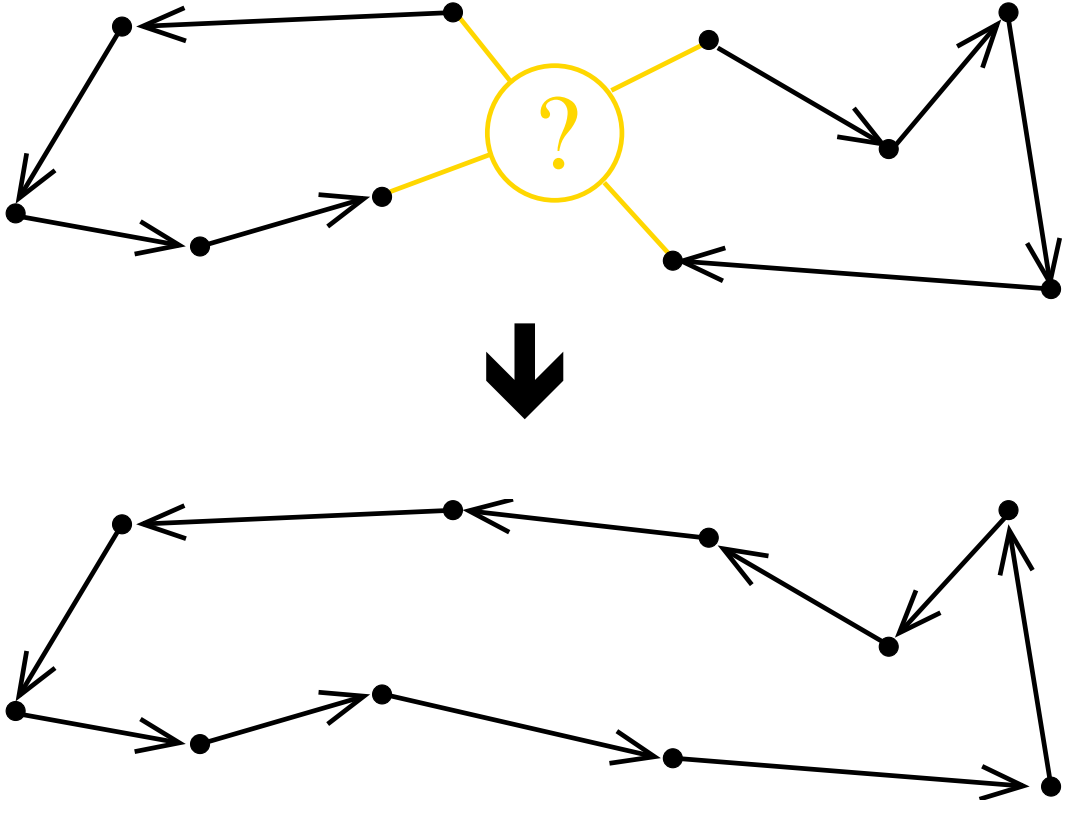
Lokale Suche: Beispiel TSP

TECHNISCHE UNIVERSITÄT DARMSTADT



Lokale Suche: Beispiel TSP

TECHNISCHE UNIVERSITÄT DARMSTADT



Lokale Suche: Beispiel Rucksack

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Schlechte Nachbarschaft: Ein Objekt wird durch ein anderes ersetzt.**
  - **Die Anzahl der ausgewählten Objekte kann nicht verändert werden.**
- **Bessere Nachbarschaft:**
  - **Ein Objekt kommt zusätzlich in die Auswahl oder eines wird gelöscht (oder beides).**

Generische lokale Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

- Konstruiere eine beliebige Startlösung *S*.
- Führe wiederholt aus:
  - Suche nach einer zu *S* benachbarten Lösung, die besser als *S* ist.
  - Falls keine gefunden: terminiere und gib *S* zurück.
  - Falls doch: Lass *S* nun eine solche sein.

Wiederholte lokale Suche

TECHNISCHE UNIVERSITÄT DARMSTADT

Führe wiederholt (sehr häufig) aus:

- **Konstruiere eine zufällige Lösung für das algorithmische Problem.**
  - ➢ **De facto keine Duplikate.**
  - ➢ **Kleine Verbesserung: adaptiv**
- **Starte eine lokale Suche von dieser Lösung.**

Spezielles Datenprofil ausnutzen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Granularität
- Spezielle Struktur
- Zusätzliche Informationen

Granularität: Rucksackproblem

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Gegeben:**
  - **Eine obere Gewichtsschranke *G*.**
  - **Eine Zahl *n* von Elementen.**
  - **Zu jedem Element *i* ein Gewicht g_i und Profit p_i.**
- **Gesucht: eine Auswahl von Elementen, so dass**
  - **das Gesamtgewicht *G* nicht überschritten und**
  - **der Gesamtprofit maximiert wird.**

Beispiel Rucksackproblem

TECHNISCHE UNIVERSITÄT DARMSTADT

Rekursionsgleichung: Die beste Auswahl aus 1... *n* bei maximalem Gesamtgewicht *G* ist

- ***entweder*** **gleich der besten Auswahl aus 1 ... *n*-1 bei maximalem Gesamtgewicht *G***
- ***oder*** **gleich**
  - **Element Nr. *n* plus**
  - **der besten Auswahl aus 1 ... *n*-1 für maximales Gesamtgewicht $G - g_i$**

Beispiel Rucksackproblem

TECHNISCHE UNIVERSITÄT DARMSTADT

- M: Matrix mit Indexbereich 0 ... n und 0 ... G (!!!)
- Für i = 1 → n: M[i,0] := 0
- Für j = 1 → G: M[0,j] := 0
- Für i = 1 → n und j = 1 → G:
  - M[i,j] := M[i-1,j]
  - Falls g_i >= j: M[i,j] := min { M[i,j], M[i-1,j-g_i] + p_i }
- Gib M[n,G] aus und Ende

Spezielle Strukturen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Beispiel: hierarchische Netzwerke.
- Konkrete Beispiele: Straßennetz, Zugverbindungen.
- Bitonische Verbindungen: Kategorien
  - zuerst ansteigend,
  - dann wieder absteigend.
- (Seltene) Ausnahmen separat behandeln.

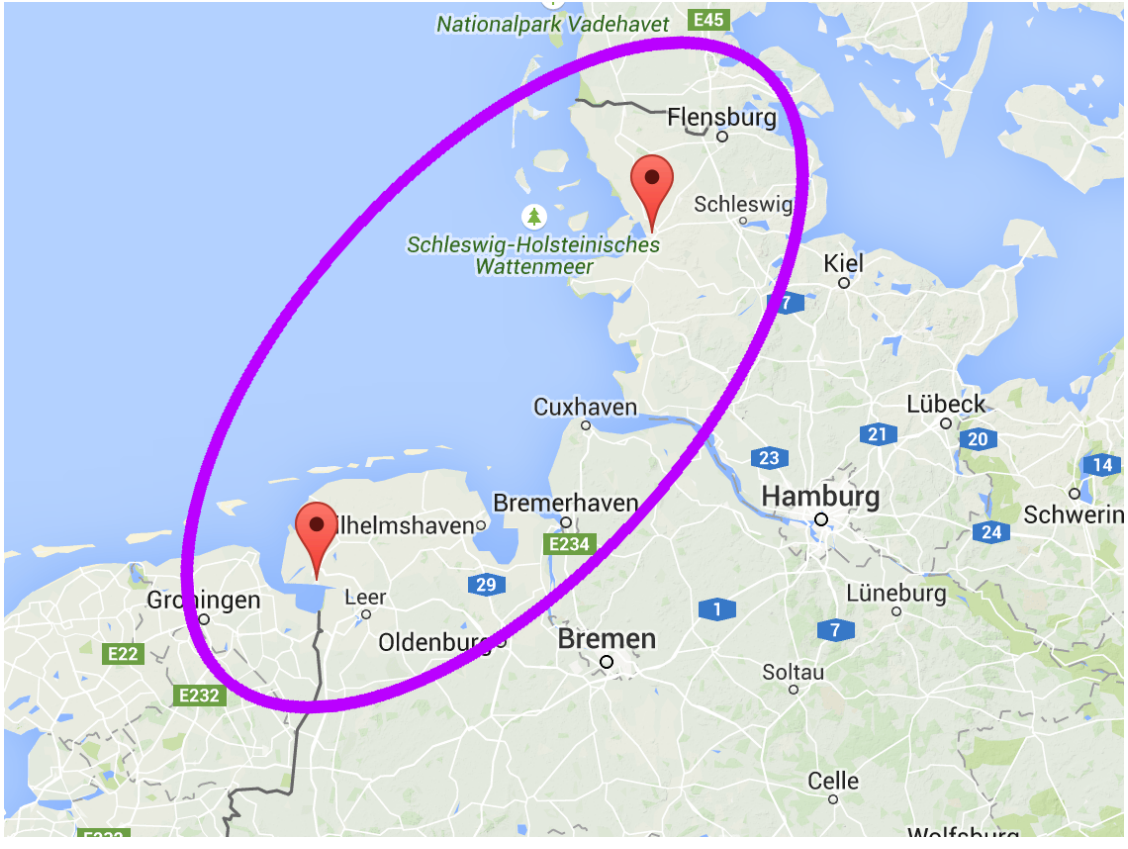
Spezielle Strukturen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Beispiel: (fast) planare Netzwerke.
- Konkretes Beispiel: weiträumige Netzwerke mglw. mit Brücken und Tunneln.
- Koordinaten ausnutzen: Suche beschleunigen durch Präferierung von Kanten in Richtung Ziel.

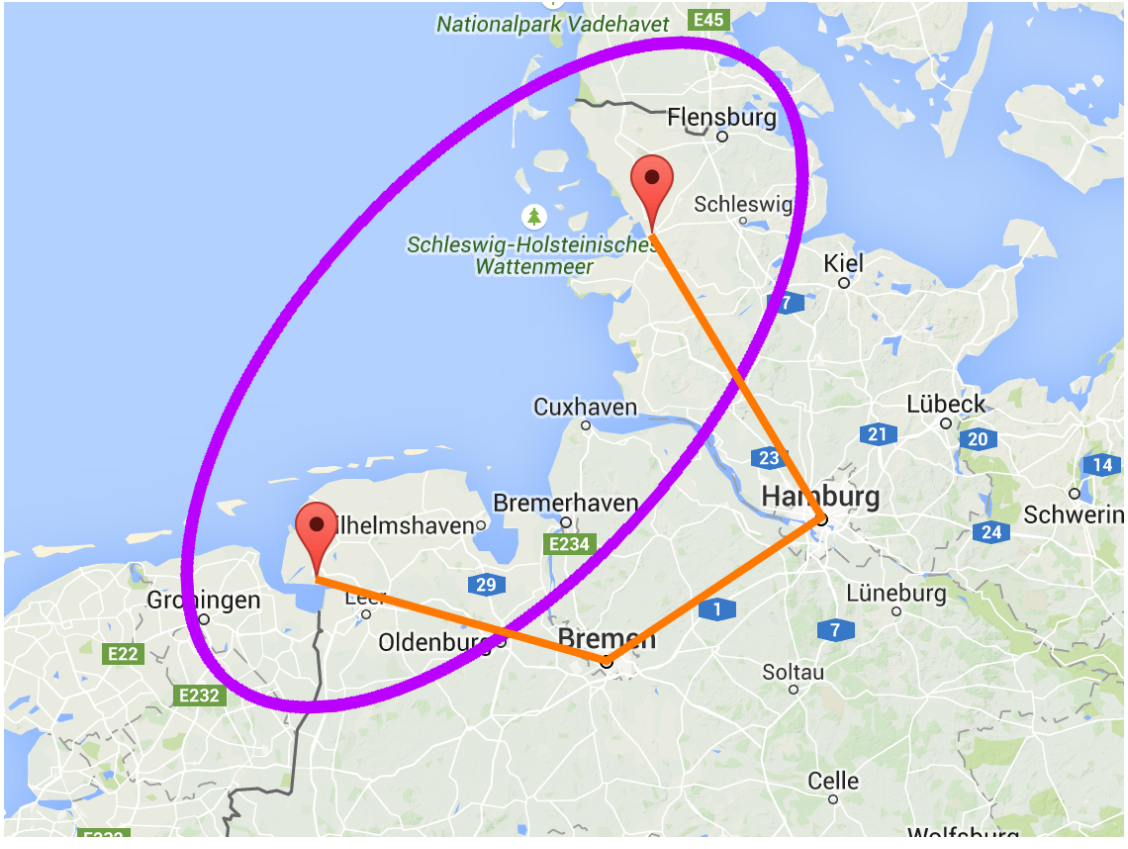
Spezielle Strukturen

TECHNISCHE UNIVERSITÄT DARMSTADT



Spezielle Strukturen

TECHNISCHE UNIVERSITÄT DARMSTADT



Zusätzliche Informationen

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Häufiger Fall: einige Fakten zum Ergebnis sind vorgeschrieben / bereits bekannt.**
- **Zwei Möglichkeiten:**
  - ***Vorgeschrieben/bekannt*: Algorithmus so erweitern, dass die zusätzlichen Informationen integriert werden können.**
  - ***Bekannt*: Algorithmus durch die zusätzlichen Informationen justieren.**