Grundlagen der Künstlichen Intelligenz

3. Klassische Suchprobleme

Malte Helmert

Universität Basel

8. März 2013

Klassische Suchprobleme

Klassische Suchprobleme informell

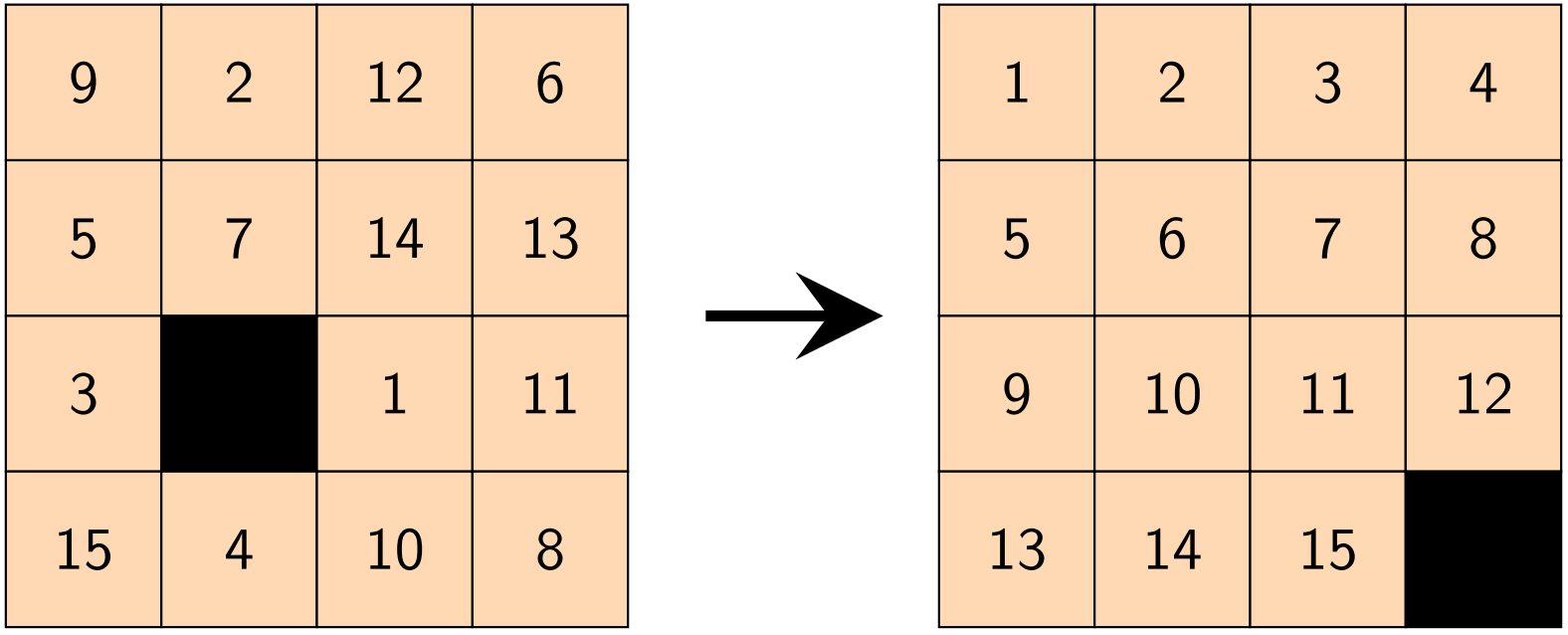
Eine der „einfachsten“ und wichtigsten Klassen von KI-Problemen sind (klassische) Suchprobleme.

Aufgabe des Agenten:

- von gegebenem Anfangszustand
- durch Anwendung von Aktionen
- einen Zielzustand erreichen

Performance-Mass: Aktionskosten minimieren

Motivierendes Beispiel: 15-Puzzle



Klassische Annahmen

„Klassische“ Annahmen:

- einzelner Agent in Umgebung (ein Agent)
- kennt immer genauen Weltzustand (vollständig beobachtbar)
- Zustand ändert sich nur durch den Agenten (statisch)
- endlich viele mögliche Zustände/Aktionen (insbes. diskret)
- Aktionen haben deterministischen Einfluss auf Zustand

$\rightsquigarrow$ können alle verallgemeinert werden
(aber nicht in diesem Kapitel)

Im folgenden lassen wir „klassisch“ der Einfachheit halber weg.

Einordnung

Einordnung:

Klassische Suchprobleme

Umgebung:

- statisch vs. dynamisch
- deterministisch vs. nicht-deterministisch vs. stochastisch
- vollständig vs. partiell vs. nicht beobachtbar
- diskret vs. stetig
- ein Agent vs. mehrere Agenten

Lösungsansatz:

- problemspezifisch vs. allgemein vs. lernend

Beispiele für Suchprobleme

- „Spielzeugprobleme“ (toy problems): kombinatorische Puzzles (Zauberwürfel, 15-Puzzle, Türme von Hanoi, . . .)
- Ablaufplanung (scheduling) in Fertigungsanlagen
- Anfrageoptimierung (query optimization) in Datenbanken
- NPCs in Computerspielen
- Code-Optimierung in Compilern
- Verifikation von Soft- und Hardware
- Sequenzalignment in der Bioinformatik
- Routenplanung (z. B. Google Maps)
- . . .

Tausende von praktischen Beispielen!

Suchprobleme: Überblick

Kapitelüberblick:

- Formalisierung von Suchproblemen
  $\rightsquigarrow$ dieses Kapitel
- blinde Suchverfahren
- informierte Suchverfahren
  $\rightsquigarrow$ folgende Kapitel

Formalisierung

Formalisierung

Vorbemerkungen:

- Um Suchprobleme sauber algorithmisch fassen zu können, benötigen wir eine formale Definition.
- grundlegendes semantisches Konzept: Zustandsräume
- Zustandsräume sind (annotierte) Graphen
- Pfade zu Zielzuständen entsprechen Lösungen
- kürzeste Pfade entsprechen optimalen Lösungen

Zustandsräume

Definition (Zustandsraum)

Ein Zustandsraum ist ein 6-Tupel $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ mit

- S endliche Menge von Zuständen
- A endliche Menge von Aktionen
- $cost : A \to \mathbb{R}_0^+$ Aktionskosten
- $T \subseteq S \times A \times S$ Transitionsrelation oder Übergangsrelation; deterministisch in $\langle s, a \rangle$ (siehe nächste Folie)
- $s_0 \in S$ Anfangszustand
- $S_\star \subseteq S$ Menge der Zielzustände

- auch: Transitionssystem (transition system)
- englisch: state space, state, action, action costs, transition relation, initial state, goal states

Zustandsräume: Transitionen, Determinismus

Definition (Transition, deterministisch)

Sei $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ ein Zustandsraum.

Die Tripel $\langle s, a, s' \rangle \in T$ heissen Transitionen/Zustandsübergänge.

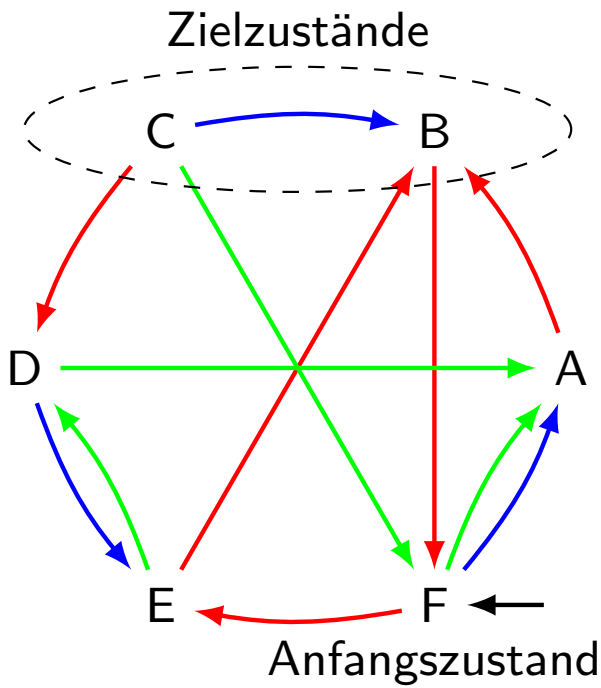
Wir sagen $\mathcal{S}$ hat die Transition $\langle s, a, s' \rangle$, falls $\langle s, a, s' \rangle \in T$ und schreiben dafür $s \xrightarrow{a} s'$ sowie $s \to s'$, wenn a nicht interessiert.

Transitionen sind deterministisch in $\langle s, a \rangle$: $s \xrightarrow{a} s_1$ und $s \xrightarrow{a} s_2$ mit $s_1 \neq s_2$ ist nicht erlaubt.

Zustandsraum: Beispiel

Zustandsräume werden oft als gerichtete Graphen dargestellt.

- Zustände: Graphknoten
- Transitionen: beschriftete Kanten
  (hier: Färbung statt Beschriftung)
- Anfangszustand: eingehender Pfeil
- Zielzustände: markiert
  (hier: durch Ellipse)
- Aktionen: die Kantenbeschriftungen
- Aktionskosten: separat anzugeben
  (oder implizit $= 1$)



Zustandsräume: Begriffe

Wir verwenden übliche Begriffe aus der Graphentheorie.

Definition (Vorgänger, Nachfolger, anwendbare Aktionen)

Sei $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ ein Zustandsraum.

Seien $s, s' \in S$ Zustände mit $s \to s'$.

- s ist Vorgänger von s'
- s' ist Nachfolger von s

Wenn $s \xrightarrow{a} s'$ gilt, ist die Aktion a anwendbar in s.

Zustandsräume: Begriffe

Wir verwenden übliche Begriffe aus der Graphentheorie.

Definition (Pfad)

Sei $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ ein Zustandsraum.

Seien $s^{(0)}, \ldots, s^{(n)} \in S$ Zustände und $\pi_1, \ldots, \pi_n \in A$ Aktionen, so dass $s^{(0)} \xrightarrow{\pi_1} s^{(1)}, \ldots, s^{(n-1)} \xrightarrow{\pi_n} s^{(n)}$.

- $\pi = \langle \pi_1, \ldots, \pi_n \rangle$ ist Pfad von $s^{(0)}$ nach $s^{(n)}$
- Länge des Pfads: $|\pi| = n$
- Kosten des Pfads: $cost(\pi) = \sum_{i=1}^{n} cost(\pi_i)$

Anmerkungen:

- Pfade der Länge 0 sind erlaubt
- manchmal nennt man auch die Zustandsfolge $\langle s^{(0)}, \ldots, s^{(n)} \rangle$ oder die Folge $\langle s^{(0)}, \pi_1, s^{(1)}, \ldots, s^{(n-1)}, \pi_n, s^{(n)} \rangle$ Pfad

Zustandsräume: Begriffe

Weitere Begriffe:

Definition (Lösung, optimal, lösbar, erreichbar, Sackgasse)

Sei $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ ein Zustandsraum.

- Ein Pfad von einem Zustand $s \in S$ zu einem Zustand $s_\star \in S_\star$ ist Lösung für/von s.
- Eine Lösung für s_0 ist Lösung für/von $\mathcal{S}$.
- Optimale Lösungen (für s) haben minimale Kosten unter allen Lösungen (für s).
- Zustandsraum $\mathcal{S}$ ist lösbar, wenn eine Lösung für $\mathcal{S}$ existiert.
- Zustand s ist erreichbar, wenn Pfad von s_0 zu s existiert.
- Zustand s ist Sackgasse, wenn keine Lösung für s existiert.

Repräsentation von Zustandsräumen

Repräsentation von Zustandsräumen

Wie kommt der Zustandsraum in den Computer?

1. als expliziter Graph:
   Knoten (Zustände) und Kanten (Transitionen) explizit repräsentiert,
   z. B. über Adjazenzlisten oder als Adjazenzmatrix

   - für grosse Probleme unmöglich! (zu viel Platzbedarf)
   - Problem klein genug für explizite Darstellung
     $\rightsquigarrow$ einfach zu lösen: Dijkstra $O(|S| \log |S| + |T|)$
   - interessant bei zeitkritischen All-pairs-shortest-path-Anfragen
     (Beispiele: Routenplanung, Pfadplanung in Computerspielen)

Repräsentation von Zustandsräumen

Wie kommt der Zustandsraum in den Computer?

2. als deklarative Beschreibung:
   - kompakte Beschreibung des Zustandsraums als Input
     $\rightsquigarrow$ Zustandsraum exponentiell grösser als Input in Bytes
   - Lösungsverfahren arbeiten direkt mit kompakter Beschreibung
     (können z. B. Problem automatisch umformulieren/vereinfachen)

Repräsentation von Zustandsräumen

Wie kommt der Zustandsraum in den Computer?

3. als Black Box: abstraktes Interface für Zustandsräume

Abstraktes Interface für Zustandsräume

Zustandsraum $\mathcal{S} = \langle S, A, cost, T, s_0, S_\star \rangle$ als Black Box:

- init(): erzeugt den Anfangszustand
  Rückgabewert: der Zustand s_0
- is-goal(s): testet, ob Zustand s ein Zielzustand ist
  Rückgabewert: **true**, falls $s \in S_\star$; sonst **false**
- succ(s): liefert anwendbare Aktionen und Nachfolger von s
  Rückgabewert: Liste der Paare $\langle a, s' \rangle$ mit $s \xrightarrow{a} s'$
- cost(a): liefert die Aktionskosten für Aktion a
  Rückgabewert: die nicht-negative Zahl $cost(a)$

(wird in folgenden Kapiteln leicht, aber entscheidend erweitert)

Black Box

- Wir verwenden im folgenden das Black-Box-Modell.
  - in der Praxis oft etwas aufgeweicht:
    parametrisierte Black-Box-Modelle
  - Beispiel: Anfangszustand nicht fest einprogrammiert,
    sondern Teil des Inputs (z. B. 15-Puzzle)
  - Beispiel: Art und Effekt der Aktionen fest vorgegeben,
    aber Kosten variieren je nach Input (z. B. TSP)
  - ändert nichts daran, dass die Suchverfahren
    den Zustandsraum als Black Box betrachten
    $\rightsquigarrow$ Zustände sind opak
- explizite Graphen verwenden wir nur für illustrierende Beispiele
- spätere Kapitel: deklarative Zustandsräume
  (Handlungsplanung)

Beispiele

Beispiel 1: Blocks world

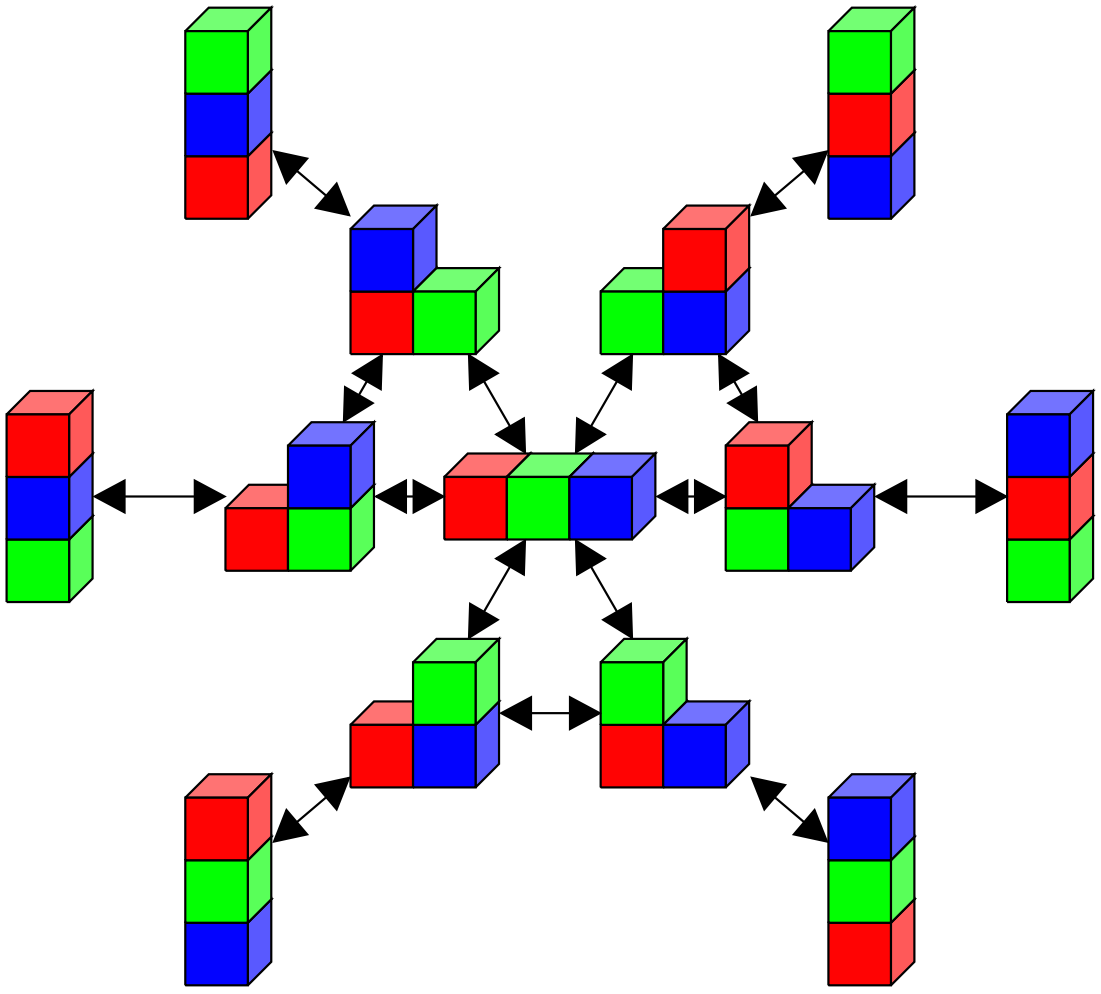
- Die Blocks world (Klötzchenwelt) ist ein traditionelles Beispielproblem in der KI.
- Wir werden sie häufiger als Beispiel verwenden.

Aufgabe: Blocks world

- Farbige Blöcke liegen auf einem Tisch.
- Sie können zu Türmen gestapelt werden, wobei immer nur ein Block auf einmal bewegt werden kann.
- Unsere Aufgabe ist es, eine gegebenen Zielkonfiguration zu erreichen.

Beispiel: Blocks world mit drei Blöcken

(Aktionsnamen der Übersicht halber weggelassen;
Anfangszustand und Ziel können unterschiedlich gewählt werden)



Blocks world: formale Definition

Zustandsraum $\langle S, A, cost, T, s_0, S_\star \rangle$ Blocks world mit n Blöcken

Zustandsraum Blocks world

Zustände S:

Partitionierungen von $\{1, 2, \ldots, n\}$ in nichtleere geordnete Listen

Beispiel $n = 3$:

- $\{\langle 1,2,3\rangle\}$, $\{\langle 1,3,2\rangle\}$, $\{\langle 2,1,3\rangle\}$, $\{\langle 2,3,1\rangle\}$, $\{\langle 3,1,2\rangle\}$, $\{\langle 3,2,1\rangle\}$
- $\{\langle 1,2\rangle, \langle 3\rangle\}$, $\{\langle 2,1\rangle, \langle 3\rangle\}$, $\{\langle 1,3\rangle, \langle 2\rangle\}$, $\{\langle 3,1\rangle, \langle 2\rangle\}$, $\{\langle 2,3\rangle, \langle 1\rangle\}$, $\{\langle 3,2\rangle, \langle 1\rangle\}$
- $\{\langle 1\rangle, \langle 2\rangle, \langle 3\rangle\}$

Blocks world: formale Definition

Zustandsraum $\langle S, A, cost, T, s_0, S_\star \rangle$ Blocks world mit n Blöcken

Zustandsraum Blocks world

Aktionen A:

- $\{move_{b,b'} \mid b, b' \in \{1, \ldots, n\}$ mit $b \neq b'\}$
  - Bewege Block b auf Block b'.
  - Beide müssen jeweils oberster Block in einem Turm sein.
- $\{totable_b \mid b \in \{1, \ldots, n\}\}$
  - Bewege Block b auf den Tisch ($\rightsquigarrow$ neuer Turm entsteht).
  - Muss oberster Block in einem Turm sein.

Aktionskosten $cost$:

$cost(a) = 1$ für alle Aktionen a

Blocks world: formale Definition

Zustandsraum $\langle S, A, cost, T, s_0, S_\star \rangle$ Blocks world mit n Blöcken

Zustandsraum Blocks world

Transitionen:

Beispielhaft für $a = move_{2,3}$:

Transition $s \xrightarrow{a} s'$ existiert genau dann, wenn

- $s = \{\langle b_1, \ldots, b_k, 2\rangle, \langle c_1, \ldots, c_m, 3\rangle\} \cup X$ und
- falls $k > 0$: $s' = \{\langle b_1, \ldots, b_k\rangle, \langle c_1, \ldots, c_m, 3, 2\rangle\} \cup X$
- falls $k = 0$: $s' = \{\langle c_1, \ldots, c_m, 3, 2\rangle\} \cup X$

Blocks world: formale Definition

Zustandsraum $\langle S, A, cost, T, s_0, S_\star \rangle$ Blocks world mit n Blöcken

Zustandsraum Blocks world

Anfangszustand s_0 und Zielzustände $S_\star$:

Eine mögliche Definition für $n = 3$:

- $s_0 = \{\langle 1, 3\rangle, \langle 2\rangle\}$
- $S_\star = \{\{\langle 3, 2, 1\rangle\}\}$

(im allgemeinen frei wählbar)

Blocks world: Eigenschaften

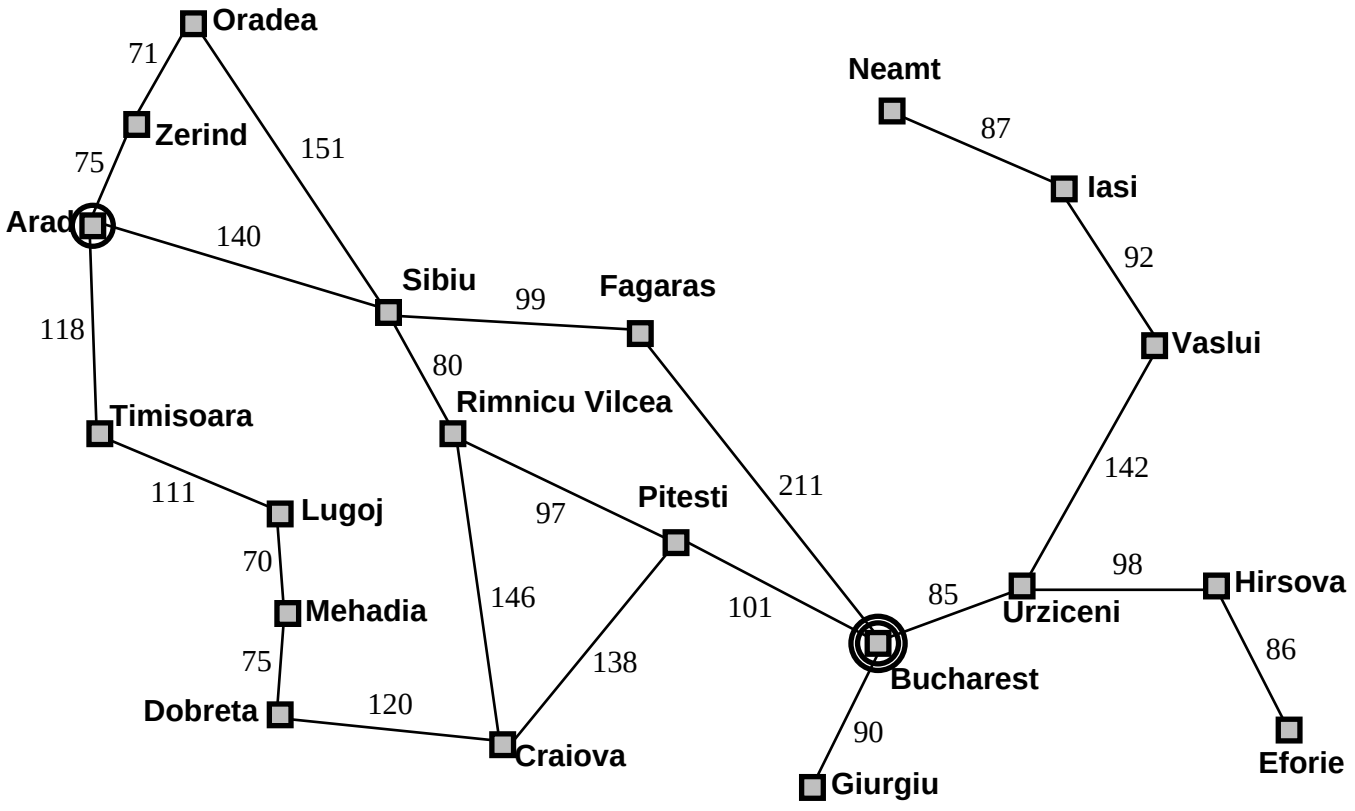
Blöcke	Zustände	Blöcke	Zustände
1	1	10	58941091
2	3	11	824073141
3	13	12	12470162233
4	73	13	202976401213
5	501	14	3535017524403
6	4051	15	65573803186921
7	37633	16	1290434218669921
8	394353	17	26846616451246353
9	4596553	18	588633468315403843

- Für jeden gegebenen Anfangs- und Zielzustand mit n Blöcken finden einfache Algorithmen Lösungen in $O(n)$ Zeit. (Wie?)
- Optimale Lösungen zu finden ist NP-vollständig (für eine kompakte Problembeschreibung).

Beispiel 2: Routenplanung in Rumänien

Aufgabe: Routenplanung in Rumänien

Wir machen Urlaub in Rumänien und sind derzeit in Arad.
Unser Rückflug startet morgen in Bukarest.



Rumänien formal

Zustandsraum Routenplanung in Rumänien

- Zustände S: $\{\text{arad}, \text{bucharest}, \text{craoiva}, \ldots, \text{zerind}\}$
- Aktionen A: $move_{c,c'}$ für je zwei Städte c und c', die durch einzelnen Strassenabschnitt verbunden
- Aktionskosten $cost$: siehe Abbildung, z. B. $cost(move_{\text{iasi,vaslui}}) = 92$
- Transitionen: $s \xrightarrow{a} s'$ genau dann, wenn $a = move_{s,s'}$
- Anfangszustand: $s_0 = \text{arad}$
- Zielzustände: $S_\star = \{\text{bucharest}\}$

Beispiel 3: Missionare und Kannibalen

Aufgabe: Missionare und Kannibalen

- sechs Personen müssen einen Fluss überqueren
- sie besitzen ein Boot, mit dem ein oder zwei Personen über den Fluss rudern können (mehr passen nicht hinein)
- drei der Personen sind Missionare, drei sind Kannibalen
- Missionare dürfen nicht mit einer Mehrheit an Kannibalen allein gelassen werden

Missionare und Kannibalen formal

Zustandsraum Missionare und Kannibalen

Zustände S:

Zahlentripel $\langle m, c, b \rangle \in \{0, 1, 2, 3\} \times \{0, 1, 2, 3\} \times \{0, 1\}$:

- Anzahl Missionare m,
- Kannibalen c und
- Boote b

am linken Flussufer

Anfangszustand: $s_0 = \langle 3, 3, 1 \rangle$

Ziel: $S_\star = \{\langle 0, 0, 0 \rangle, \langle 0, 0, 1 \rangle\}$

Aktionen, Aktionskosten, Transitionen: ?

Zusammenfassung

Zusammenfassung

- klassische Suchprobleme: finde Aktionsfolge vom Anfangszustand zu einem Zielzustand
- Performance-Mass: Summe von Aktionskosten
- Formalisierung über Zustandsräume
- Repräsentation: explizit, deklarativ oder Black-Box