

Grundlagen der Künstlichen Intelligenz

5. Klassische Suche: Zustandsräume

Malte Helmert

Universität Basel

6. März 2015

Grundlagen der Künstlichen Intelligenz

6. März 2015 — 5. Klassische Suche: Zustandsräume

5.1 Klassische Suchprobleme

5.2 Formalisierung

5.3 Zustandsraumsuche

5.4 Zusammenfassung

5.1 Klassische Suchprobleme

Klassische Suchprobleme informell

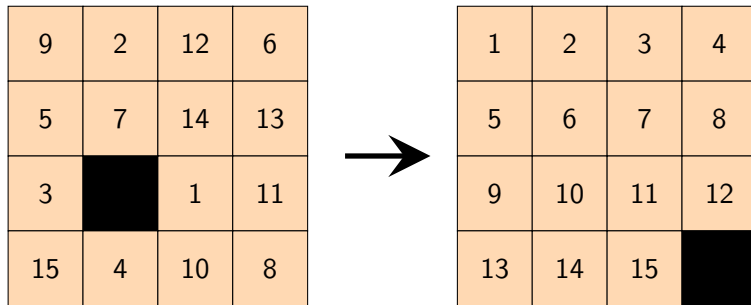
Eine der „einfachsten“ und **wichtigsten** Klassen von KI-Problemen sind **(klassische) Suchprobleme**.

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

⇒ können alle verallgemeinert werden
(aber nicht in diesem Vorlesungsteil)

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!

Klassische Suche: Überblick

Kapitelüberblick klassische Suche:

- ▶ 5.–7. Grundlagen
 - ▶ 5. Zustandsräume
 - ▶ 6. Repräsentation von Zustandsräumen
 - ▶ 7. Beispiele von Zustandsräumen
- ▶ 8.–12. Basisalgorithmen
- ▶ 13.–20. heuristische Algorithmen

5.2 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**
- ▶ $cost : A \rightarrow \mathbb{R}_0^+$ **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 $\mathcal{S} = \langle S, A, cost, T, s_0, S_* \rangle$ ein Zustandsraum.

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

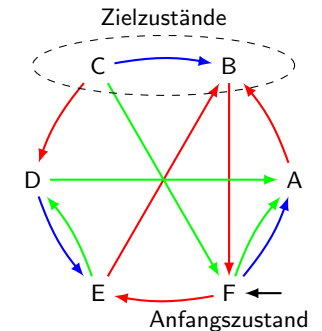
Wir sagen $\mathcal{S}$ **hat die Transition** $\langle s, a, s' \rangle$, falls $\langle s, a, s' \rangle \in T$ und schreiben dafür $s \xrightarrow{a} s'$ sowie $s \rightarrow s'$, wenn a nicht interessiert.

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

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

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

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

- $\pi = \langle \pi_1, \dots, \pi_n \rangle$ 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 $\langle s^{(0)}, \dots, s^{(n)} \rangle$ oder die Folge $\langle s^{(0)}, \pi_1, s^{(1)}, \dots, s^{(n-1)}, \pi_n, s^{(n)} \rangle$ **Pfad**

Zustandsräume: Begriffe

Weitere Begriffe:

Definition (erreichbar, Lösung, optimal)

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

- ▶ Zustand s ist **erreichbar**, wenn Pfad von s_0 zu s existiert.
- ▶ 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).

5.3 Zustandsraumsuche

Zustandsraumsuche

Zustandsraumsuche

Zustandsraumsuche ist das algorithmische Problem, Lösung in Zustandsräumen zu finden oder zu beweisen, dass keine Lösung existiert.

Bei der **optimalen Zustandsraumsuche** wird zusätzlich gefordert, dass nur optimale Lösungen gefunden werden.

Lernziele Zustandsraumsuche

Lernziele zum Thema Zustandsraumsuche

- ▶ **Zustandsraumsuche verstehen:**
Worin besteht das Problem und wie formalisieren wir es?
- ▶ **Suchalgorithmen bewerten:**
Vollständigkeit, Optimalität, Zeit-/Speicheraufwand
- ▶ **Suchalgorithmen kennenlernen:**
uninformiert und informiert, Baum- und Graphensuche
- ▶ **Heuristiken für Suchalgorithmen bewerten:**
Zielerkennung, Sicherheit, Zulässigkeit, Konsistenz
- ▶ **effiziente Implementierung** von Suchalgorithmen
- ▶ **experimentelle Bewertung** von Suchalgorithmen
- ▶ **Entwurf und Vergleich von Heuristiken** für Suchalgorithmen

5.4 Zusammenfassung

Zusammenfassung

- ▶ **klassische Suchprobleme:** finde Aktionsfolge vom Anfangszustand zu einem Zielzustand
- ▶ **Performance-Mass:** Summe von Aktionskosten
- ▶ Formalisierung über **Zustandsräume:**
 - ▶ **Zustände, Aktionen, Aktionskosten, Transitionen, Anfangszustand, Zielzustände**
- ▶ Begriffe zu Transitionen, Pfaden, Lösungen
- ▶ Definition (optimale) Zustandsraumsuche