

Grundlagen der Künstlichen Intelligenz

10. Klassische Suche: Heuristiken

Malte Helmert

Universität Basel

24. März 2014

Grundlagen der Künstlichen Intelligenz

24. März 2014 — 10. Klassische Suche: Heuristiken

10.1 Einführung

10.2 Heuristiken

10.3 Beispiele

10.4 Zusammenfassung

Klassische Suche: Überblick

Kapitelüberblick klassische Suche:

- ▶ 3.–5. Einführung
- ▶ 6.–9. Basisalgorithmen
- ▶ 10.–17. heuristische Algorithmen
 - ▶ 10. Heuristiken
 - ▶ 11. Analyse von Heuristiken
 - ▶ 12. Bestensuche als Graphensuche
 - ▶ 13. Gierige Bestensuche, A*, Weighted A*
 - ▶ 14. IDA*
 - ▶ 15. A*: Optimalität, Teil I
 - ▶ 16. A*: Optimalität, Teil II
 - ▶ 17. A*: Vollständigkeit und Komplexität

10.1 Einführung

Informierte Suchalgorithmen

- ▶ bisher betrachtete Suchalgorithmen: **blind**, da sie keine Problemaspekte ausser der formalen Definition (Zustandsraum) berücksichtigen
 - ▶ **Problem**: Skalierbarkeit \rightsquigarrow Zeit- und Speicherlimits bereits für scheinbar **einfache** Probleme erreicht
 - ▶ **Idee**: versuche (problemspezifische) Kriterien zu finden, um **gute** von **schlechten** Zuständen zu unterscheiden \rightsquigarrow **bevorzuge gute Zustände**
- \rightsquigarrow **informierte** („heuristische“) Suchalgorithmen

10.2 Heuristiken

Heuristiken

Definition (Heuristik)

Sei S ein Zustandsraum mit Zuständen s .
Eine **Heuristikfunktion** oder **Heuristik** für S ist eine Funktion

$$h : S \rightarrow \mathbb{R}_0^+ \cup \{\infty\},$$

die jeden Zustand auf eine nicht-negative Zahl (oder ∞) abbildet.

Heuristiken: Intuition

Idee: $h(s)$ schätzt Abstand (= Kosten des billigsten Pfads) von s zum nächsten Zielzustand

- ▶ Heuristiken können **beliebige** Funktionen sein
- ▶ **Intuition**: je näher h am wirklichen Zielabstand, desto effizienter ist die Suche, die sie benutzt

Heuristiken werden manchmal über **Suchknoten** statt Zuständen definiert, aber diese Allgemeinheit ist selten nützlich. (**Warum?**)

Warum „Heuristik“?

Was bedeutet „Heuristik“?

- ▶ Heuristik: von Altgriechisch $\epsilon\upsilon\rho\iota\sigma\kappa\omega$ (= Ich finde)
 \rightsquigarrow vergleiche: $\epsilon\upsilon\eta\eta\kappa\alpha!$
- ▶ popularisiert durch George Pólya: How to Solve It (1945)
- ▶ in der Informatik oft verwendet für:
 Daumenregel, inexakter Algorithmus
- ▶ in der KI-Suche **Fachterminus** für **Zielabstandsschätzer**

Repräsentation von Heuristiken

In unserem Black-Box-Modell sind Heuristiken ein zusätzliches Element des Zustandsraum-Interfaces:

Zustandsräume als Black Box (erweitert)

- ▶ `init()`
- ▶ `is_goal(s)`
- ▶ `succ(s)`
- ▶ `cost(a)`
- ▶ **$h(s)$** : Heuristikwert für Zustand s
 Ergebnis: nicht-negative ganze Zahl oder ∞

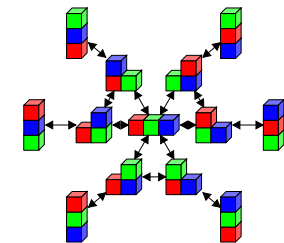
10.3 Beispiele

Beispiel: Blocks world

mögliche Heuristik:

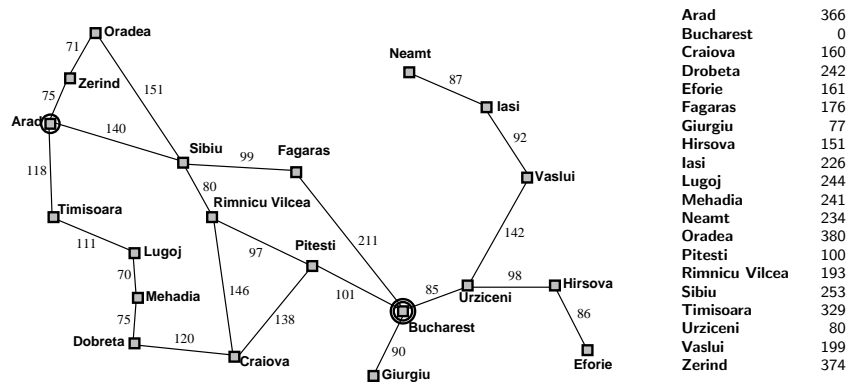
Zähle die Blöcke x , die auf y liegen und im Ziel auf $z \neq y$ liegen müssen (inkl. Fall, wo y oder z der Tisch ist)

Wie genau ist diese Heuristik?



Beispiel: Routenplanung in Rumänien

mögliche Heuristik: Luftliniendistanz nach Bukarest



Beispiel: 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

mögliche Heuristik: Anzahl Personen am falschen Flussufer

- ↪ mit unserer Formalisierung von Zuständen als Tripel $\langle m, c, b \rangle$:
- $$h(\langle m, c, b \rangle) = m + c$$

10.4 Zusammenfassung

- ▶ **Heuristiken** schätzen Abstand eines Zustands vom Ziel
- ▶ Heuristiken können verwendet werden, um die Suche auf **viel versprechende Zustände** zu **fokussieren**
- ↪ **später**: Algorithmen, die Heuristiken ausnutzen

Zusammenfassung