

Grundlagen der Künstlichen Intelligenz

13. Klassische Suche: Gierige Bestensuche, A*, Weighted A*

Malte Helmert

Universität Basel

31. März 2014

Grundlagen der Künstlichen Intelligenz

31. März 2014 — 13. Klassische Suche: Gierige Bestensuche, A*, Weighted A*

13.1 Einführung

13.2 Gierige Bestensuche

13.3 A*

13.4 Weighted A*

13.5 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

13.1 Einführung

Worum geht's?

In diesem Kapitel schauen wir uns die Algorithmen aus dem letzten Kapitel genauer an:

- ▶ Gierige Bestensuche
- ▶ A*
- ▶ Weighted A*

13.2 Gierige Bestensuche

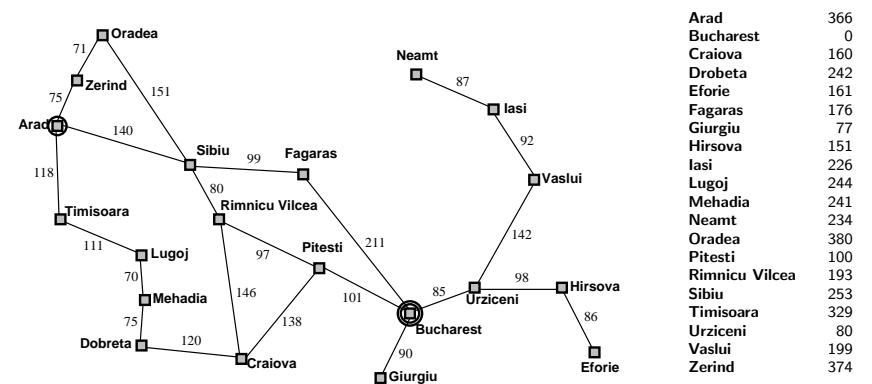
Gierige Bestensuche

Gierige Bestensuche

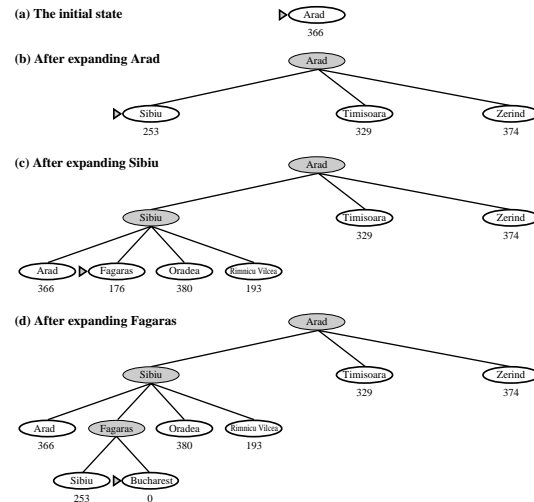
Berücksichtige nur die Heuristik: $f(n) = h(n.state)$

Anmerkung: normalerweise **ohne Reopening** (aus Effizienzgründen)

Beispiel: Gierige Bestensuche für Routenplanung



Beispiel: Gierige Bestensuche für Routenplanung



Gierige Bestensuche: Eigenschaften

- ▶ **vollständig** für **sichere** Heuristiken
(wie alle Varianten der Bestensuche als Graphensuche)
- ▶ **suboptimal**: Lösungen können **beliebig schlecht** sein
- ▶ oft sehr schnell:
in der Praxis einer der schnellsten Suchalgorithmen
- ▶ monotone Transformation von h (z. B. Skalierung, additive Konstante) beeinflusst nicht das Verhalten
(**Warum ist das interessant?**)

13.3 A*

A*

A*

Verbinde gierige Bestensuche mit uniformer Kostensuche:

$$f(n) = g(n) + h(n.state)$$

- ▶ **Abwägen** zwischen Pfadkosten und Nähe zum Ziel
- ▶ $f(n)$ schätzt Gesamtkosten der billigsten Lösung vom Anfangszustand über n zum Ziel

A*: Zitate

Google scholar My Citations Search

Scholar Articles and patents anytime Include citations Create email alert

A formal basis for the heuristic determination of minimum cost paths
 ... NJ Nilsson, B Raphael - Systems Science and ... 1968 - IEEE explore. IEEE.org
 Abstract Although the problem of determining the minimum cost path through a graph arises naturally in a number of interesting applications, there has been no underlying theory to guide the development of efficient search procedures. Moreover, there is no adequate ...
 Cited by 2931 - Related articles - All 13 versions

Correction to a formal basis for the heuristic determination of minimum cost paths
 ... NJ Nilsson, B Raphael - ACM SIGART Bulletin, 1972 - dl.acm.org
 Abstract Our paper on the use of heuristic information in graph searching defined a path-finding algorithm, A*, and proved that it had two important properties. In the notation of the paper, we proved that if the heuristic function $h(n)$ is a lower bound on the true minimal ...
 Cited by 195 - Related articles - All 6 versions

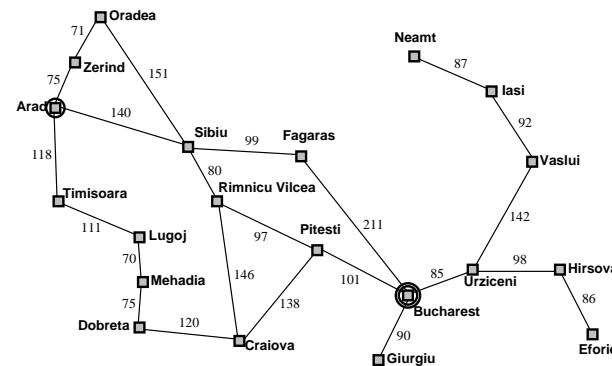
[citation] Bi-directional search
 I Pohl - 1970 - IBM TJ Watson Research Center
 Cited by 337 - Related articles - Find in IDS Basel/Bern - All 2 versions

[citation] and B. Raphael. A formal basis for heuristic determination of minimum path cost
 ... NJ Nilsson - IEEE Transaction on SSC, 1968
 Cited by 28 - Related articles

[citation] Research and applications—artificial intelligence
 B Raphael, R Duda, RE Fikes, PE Hart, N Nilsson... - Final Report, SRI Project, 1971
 Cited by 13 - Related articles

A mobile automaton: An application of artificial intelligence techniques
 NJ Nilsson - 1969 - DTIC Document
 A MOBILE ALGORITHM: AN APPLICATION OF ARTIFICIAL INTELLIGENCE TECHNIQUES

Beispiel: A* für Routenplanung



Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Beispiel: A* für Routenplanung

(a) The initial state



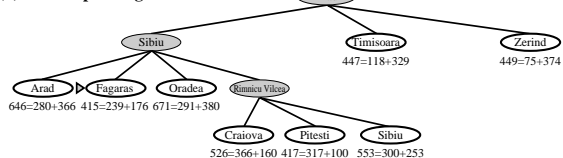
(b) After expanding Arad



(c) After expanding Sibiu

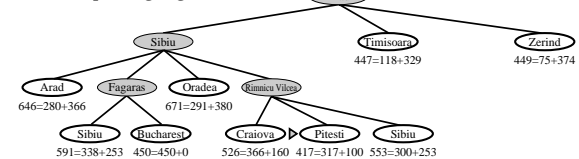


(d) After expanding Rimnicu Vilcea

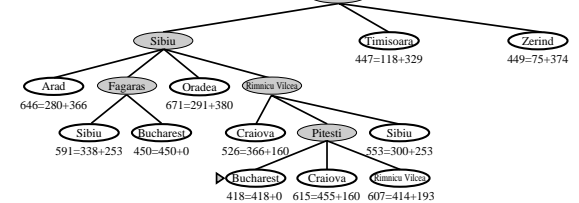


Beispiel: A* für Routenplanung

(e) After expanding Fagaras



(f) After expanding Pitesti



A*: Eigenschaften

- ▶ **vollständig** für **sichere** Heuristiken
(wie alle Varianten der Bestensuche als Graphensuche)
- ▶ **mit Reopening**:
optimal, wenn Heuristik **zulässig**
- ▶ **ohne Reopening**:
optimal, wenn Heuristik **konsistent** und **zulässig**

↔ Beweise: Kapitel 15 und 16

A*: Implementierungsaspekte

Einige praktische Anmerkungen für die Implementierung von A*:

- ▶ **häufiger Bug**: kein Reopening implementiert, obwohl die Heuristik nicht konsistent ist
- ▶ **häufiger Bug**: Duplikatstest „zu früh“
(beim Erzeugen der Suchknoten)
- ▶ **häufiger Bug**: Zieltest „zu früh“
(beim Erzeugen der Suchknoten)
- ▶ all diese Bugs führen zum Verlust der Optimalität und können lange unentdeckt bleiben

13.4 Weighted A*

Weighted A*

Weighted A*

A* mit stärker gewichteter Heuristik: $f(n) = g(n) + w \cdot h(n.state)$,
wobei **Gewicht** $w \in \mathbb{R}_0^+$ mit $w \geq 1$ ein frei wählbarer Parameter ist

Anmerkung: $w < 1$ ist denkbar, aber meist keine gute Idee
(Warum nicht?)

Weighted A*: Eigenschaften

Gewichtparameter kontrolliert "Gierigkeit" der Suche:

- ▶ $w = 0$: wie uniforme Kostensuche
- ▶ $w = 1$: wie A*
- ▶ $w \rightarrow \infty$: wie gierige Bestensuche

Mit $w \geq 1$ analoge Eigenschaften zu A*:

- ▶ h zulässig:
gefundene Lösung garantiert höchstens Faktor w so teuer wie Optimum, wenn Reopening verwendet wird
- ▶ h konsistent und zulässig:
gefundene Lösung garantiert höchstens Faktor w so teuer wie Optimum; kein Reopening nötig

(ohne Beweis)

13.5 Zusammenfassung

Zusammenfassung

Bestensuche als Graphensuche mit Bewertungsfunktion f :

- ▶ $f = h$: **gierige Bestensuche**
suboptimal, oft sehr schnell
- ▶ $f = g + h$: **A***
optimal, wenn h zulässig und konsistent
oder wenn h zulässig und **Reopening** verwendet wird
- ▶ $f = g + w \cdot h$: **Weighted A***
für $w \geq 1$ Suboptimalitätsfaktor höchstens w
unter den gleichen Bedingungen wie für Optimalität von A*