Foundations of Artificial Intelligence

B9. State-Space Search: Heuristics

Malte Helmert

University of Basel

March 16, 2026

State-Space Search: Overview

Chapter overview: state-space search

- B1–B3. Foundations
- B4–B8. Basic Algorithms
- B9–B15. Heuristic Algorithms
    - B9. Heuristics
    - B10. Analysis of Heuristics
    - B11. Best-first Graph Search
    - B12. Greedy Best-first Search, A^*, Weighted A^*
    - B13. IDA*
    - B14. Properties of A^*, Part I
    - B15. Properties of A^*, Part II

Introduction

Informed Search Algorithms

search algorithms considered so far:

- uninformed ("blind"): use no information besides formal definition to solve a problem
- scale poorly: prohibitive time (and space) requirements for seemingly simple problems (time complexity usually $O(b^d)$)

Informed Search Algorithms

search algorithms considered so far:

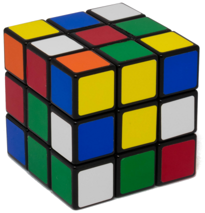
- uninformed ("blind"): use no information besides formal definition to solve a problem
- scale poorly: prohibitive time (and space) requirements for seemingly simple problems (time complexity usually $O(b^d)$)

example: $b = 13$; 10^5 nodes/second

d	nodes	time
4	30 940	0.3 s
6	$5.2 \cdot 10^6$	52 s
8	$8.8 \cdot 10^8$	147 min
10	10^{11}	17 days
12	10^{13}	8 years
14	10^{15}	1 352 years
16	10^{17}	$2.2 \cdot 10^5$ years
18	10^{20}	$38 \cdot 10^6$ years

Informed Search Algorithms

Rubik's cube:



- branching factor: ≈ 13
- typical solution length: 18

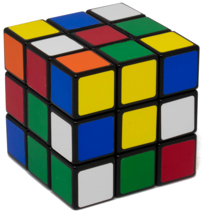
example: $b = 13$; 10^5 nodes/second

d	nodes	time
4	30 940	0.3 s
6	$5.2 \cdot 10^6$	52 s
8	$8.8 \cdot 10^8$	147 min
10	10^{11}	17 days
12	10^{13}	8 years
14	10^{15}	1 352 years
16	10^{17}	$2.2 \cdot 10^5$ years
18	10^{20}	$38 \cdot 10^6$ years

Richard Korf, Finding Optimal Solutions to Rubik's Cube Using Pattern Databases (AAAI, 1997)

Informed Search Algorithms

Rubik's cube:



- branching factor: ≈ 13
- typical solution length: 18

search algorithms considered now:

- idea: try to find (problem-specific) criteria to distinguish good and bad states
- heuristic ("informed") search algorithms prefer good states

Richard Korf, Finding Optimal Solutions to Rubik's Cube Using Pattern Databases (AAAI, 1997)

Heuristics

Heuristics

Definition (heuristic)

Let $\mathcal{S}$ be a state space with states S.
A heuristic function or heuristic for $\mathcal{S}$ is a function

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

mapping each state to a nonnegative number (or ∞).

Heuristics: Intuition

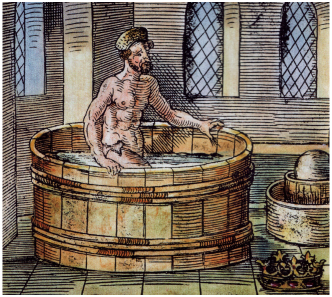
idea: $h(s)$ estimates distance ($=$ cost of cheapest path) from s to closest goal state

- heuristics can be arbitrary functions
- intuition:
  1. the closer h is to true goal distance, the more efficient the search using h
  2. the better h separates states that are close to the goal from states that are far, the more efficient the search using h

Why "Heuristic"?

What does "heuristic" mean?

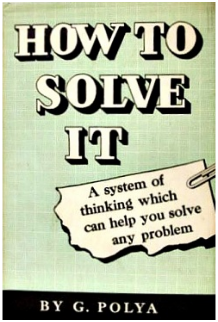
- from ancient Greek ἑυρισκω (= I find)
- same origin as ἑυρηκα!



Why "Heuristic"?

What does "heuristic" mean?

- from ancient Greek ἑυρισκω (= I find)
- same origin as ἑυρηκα!
- popularized by George Pólya: How to Solve It (1945)
- in computer science often used for: rule of thumb, inexact algorithm
- in state-space search technical term for goal distance estimator



Representation of Heuristics

In our black box model, heuristics are an additional element of the state space interface:

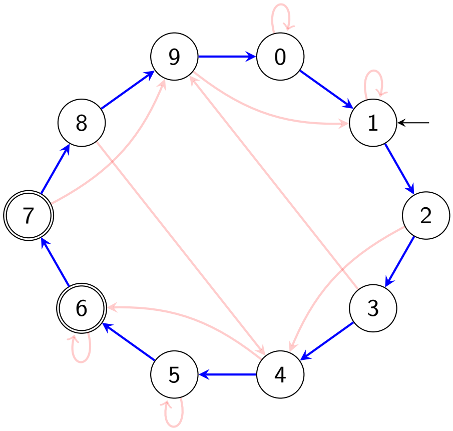
State Spaces as Black Boxes (Extended)

- init()
- is_goal(s)
- succ(s)
- cost(a)
- h(s): heuristic value for state s
  result: nonnegative integer or ∞

Examples

Bounded Inc-and-Square

bounded inc-and-square:



possible heuristics:

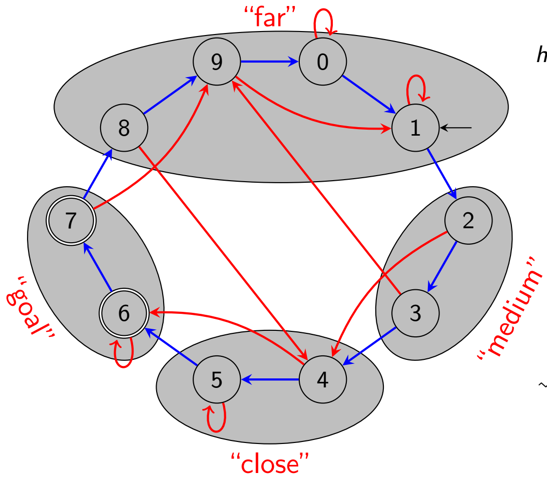
$$h_1(s) = \begin{cases} 0 & \text{if } s = 7 \\ (16 - s) \bmod 10 & \text{otherwise} \end{cases}$$

$\rightsquigarrow$ number of *inc* actions to goal

How accurate is this heuristic?

Bounded Inc-and-Square

bounded inc-and-square:



possible heuristics:

$$h_1(s) = \begin{cases} 0 & \text{if } s = 7 \\ (16 - s) \bmod 10 & \text{otherwise} \end{cases}$$

$\rightsquigarrow$ number of *inc* actions to goal

$$h_2(s) = \begin{cases} 0 & \text{if } s \text{ is a “goal”} \\ 1 & s \text{ is “close”} \\ 2 & s \text{ is “medium”} \\ 3 & s \text{ is “far”} \end{cases}$$

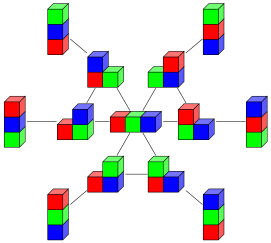
$\rightsquigarrow$ categorize states

How accurate is this heuristic?

Example: Blocks World

possible heuristic:

count blocks x that currently lie on y and must lie on $z \neq y$ in the goal (including case where y or z is the table)

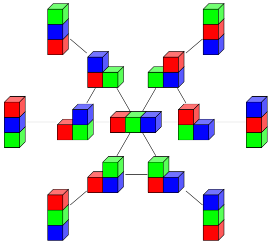


Example: Blocks World

possible heuristic:

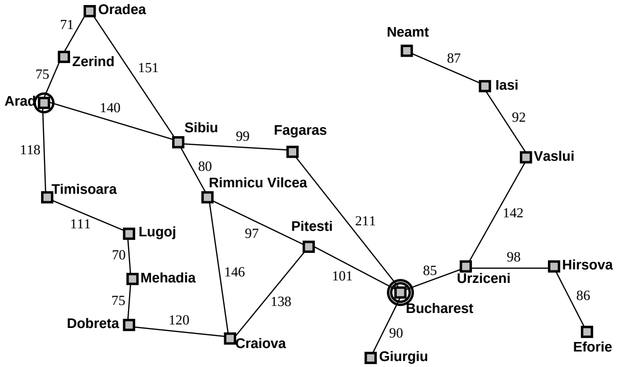
count blocks x that currently lie on y and must lie on $z \neq y$ in the goal (including case where y or z is the table)

How accurate is this heuristic?



Example: Route Planning in Romania

possible heuristic: straight-line distance to Bucharest



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

Example: Missionaries and Cannibals

Setting: Missionaries and Cannibals

- Six people must cross a river.
- Their rowing boat can carry one or two people across the river at a time (it is too small for three).
- Three people are missionaries, three are cannibals.
- Missionaries may never stay with a majority of cannibals.

possible heuristic: number of people on the wrong river bank

Example: Missionaries and Cannibals

Setting: Missionaries and Cannibals

- Six people must cross a river.
- Their rowing boat can carry one or two people across the river at a time (it is too small for three).
- Three people are missionaries, three are cannibals.
- Missionaries may never stay with a majority of cannibals.

possible heuristic: number of people on the wrong river bank

$\rightsquigarrow$ with our formulation of states as triples $\langle m, c, b \rangle$:
$h(\langle m, c, b \rangle) = m + c$

Summary

Summary

- heuristics estimate distance of a state to the goal
- can be used to focus search on promising states

$\leadsto$ soon: search algorithms that use heuristics