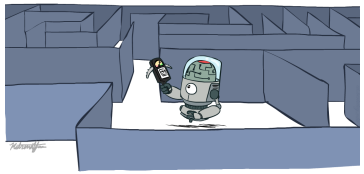
Announcements

- Homework 1: Search
  - On edX - online, instant grading, submit as often as you like.
- Project 1: Search
  - Start early and ask questions. It's longer than most!

1

CS 188: Artificial Intelligence

Informed Search



Instructor: Marco Alvarez

University of Rhode Island

[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley (ai.berkeley.edu)]

2

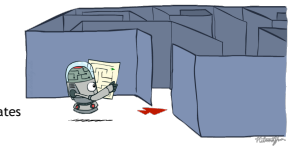
Today

- Informed Search
  - Heuristics
  - Greedy Search
  - A* Search

3

Recap: Search

- Search problem:
  - States (configurations of the world)
  - Actions and costs
  - Successor function (world dynamics)
  - Start state and goal test
- Search tree:
  - Nodes: represent plans for reaching states
  - Plans have costs (sum of action costs)
- Search algorithm:
  - Systematically builds a search tree
  - Chooses an ordering of the fringe (unexplored nodes)
  - Optimal: finds least-cost plans



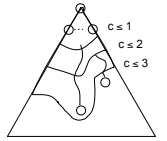
4

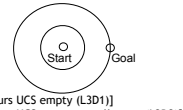
Uninformed Search

5

Uniform Cost Search

- Strategy: expand lowest path cost
- The good: UCS is complete and optimal!
- The bad:
  - Explores options in every "direction"
  - No information about goal location

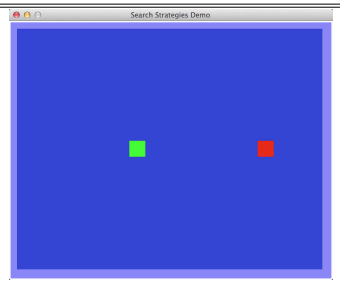




[Demo: contours UCS empty (L3D1)]
[Demo: contours UCS pacman small maze (L3D3)]

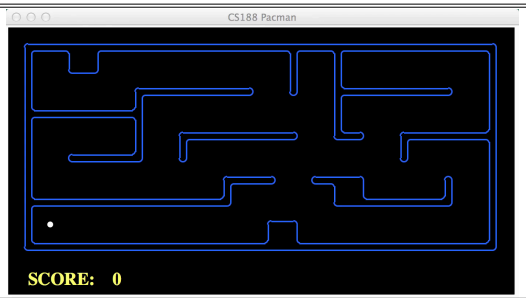
6

Video of Demo Contours UCS Empty



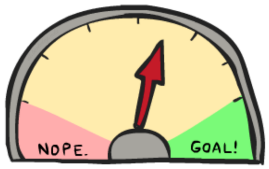
7

Video of Demo Contours UCS Pacman Small Maze



8

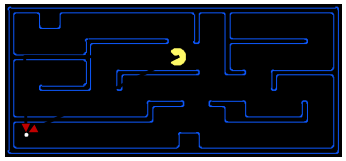
Informed Search

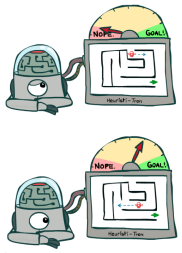


9

Search Heuristics

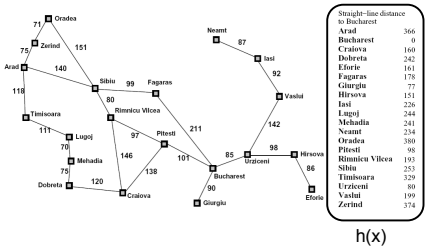
- A heuristic is:
  - A function that *estimates* how close a state is to a goal
  - Designed for a particular search problem
  - Examples: Manhattan distance, Euclidean distance for pathing





10

Example: Heuristic Function



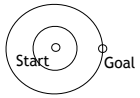
h(x)

11

Effect of heuristics

Guide search ***towards the goal*** instead of ***all over the place***

Informed



Uninformed

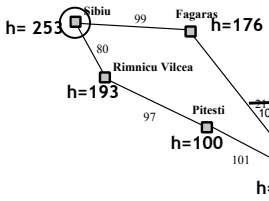
12

Greedy Search

13

Greedy Search

- Expand the node that seems closest…(order frontier by h)
- What can possibly go wrong?

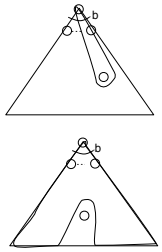


Sibiu-Fagaras-Bucharest =
99+211 = **310**
Sibiu-Rimnicu Vilcea-Pitesti-Bucharest
80+97+101 = **278**

14

Greedy Search

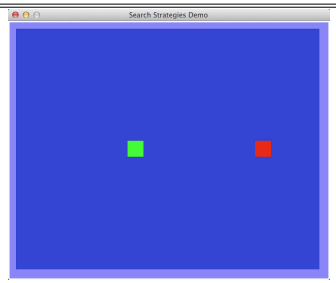
- Strategy: expand a node that **seems** closest to a goal state
  - Heuristic: estimate of distance to nearest goal for each state
- A common case:
  - Best-first takes you straight to the (wrong) goal
- Worst-case: like a badly-guided DFS



[Demo: contours greedy empty (L3D1)]
[Demo: contours greedy pacman small maze (L3D4)]

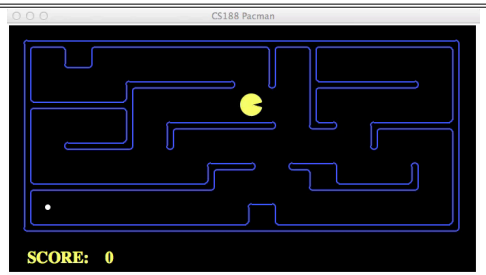
15

Video of Demo Contours Greedy (Empty)



16

Video of Demo Contours Greedy (Pacman Small Maze)

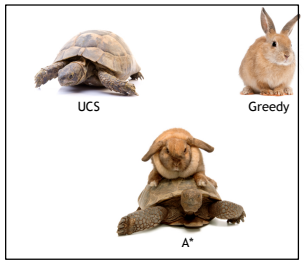


17

A* Search

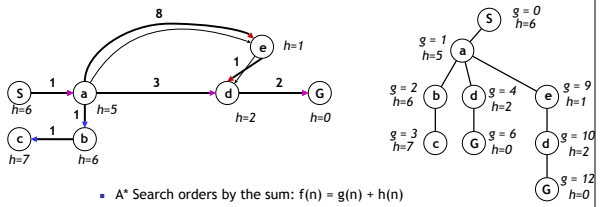
18

A* Search



Combining UCS and Greedy

- Uniform-cost orders by path cost, or *backward cost* g(n)
- Greedy orders by goal proximity, or *forward cost* h(n)

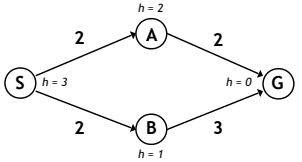


- A* Search orders by the sum: f(n) = g(n) + h(n)

Example: Teg Grenager

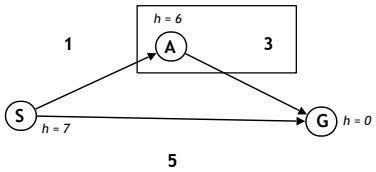
When should A* terminate?

- Should we stop when we enqueue a goal?



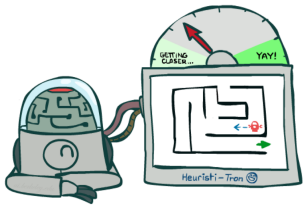
- No: only stop when we dequeue a goal

Is A* Optimal?

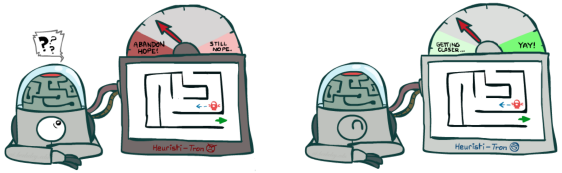


- What went wrong?
- Actual bad goal cost < estimated good goal cost
- We need estimates to be less than actual costs!

Admissible Heuristics



Idea: Admissibility



Inadmissible (pessimistic) heuristics break optimality by trapping good plans on the fringe

Admissible (optimistic) heuristics slow down bad plans but never outweigh true costs

Admissible Heuristics

- A heuristic h is *admissible* (optimistic) if:

$$0 \leq h(n) \leq h^*(n)$$

where $h^*(n)$ is the true cost to a nearest goal

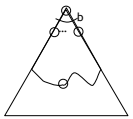
- Example:

- Coming up with admissible heuristics is most of what's involved in using A* in practice.

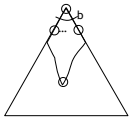
Properties of A*

Properties of A*

Uniform-Cost

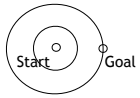


A*



UCS vs A* Contours

- Uniform-cost expands equally in all "directions"

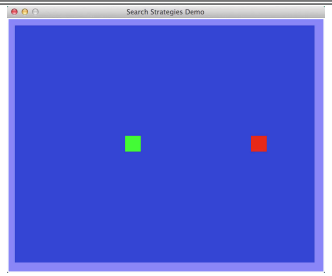


- A* expands mainly toward the goal, but does hedge its bets to ensure optimality

[Demo: contours UCS / greedy / A* empty (L3D1)]
[Demo: contours A* pacman small maze (L3D5)]

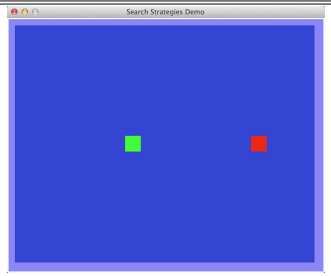
28

Video of Demo Contours (Empty) -- UCS



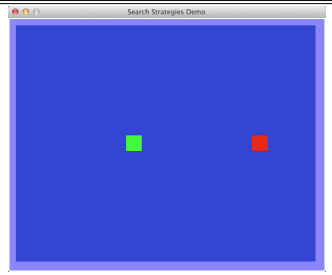
29

Video of Demo Contours (Empty) -- Greedy



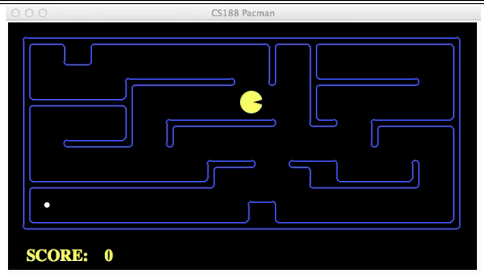
30

Video of Demo Contours (Empty) – A*



31

Video of Demo Contours (Pacman Small Maze) – A*



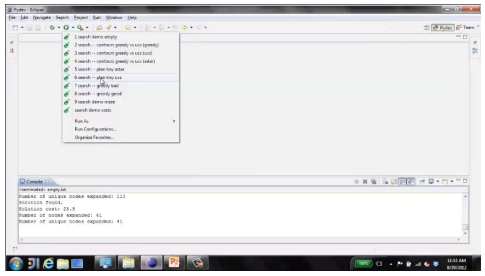
32

Comparison

Greedy Uniform Cost A*

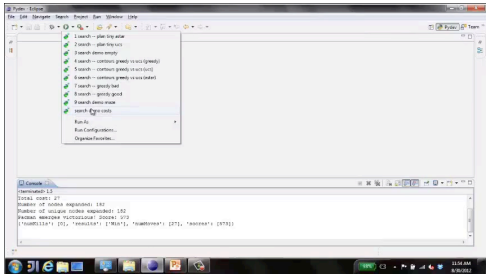
33

Video of Demo Pacman (Tiny Maze) – UCS / A*



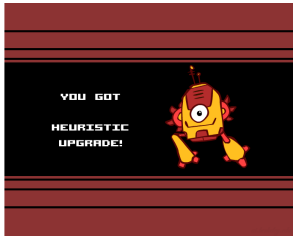
34

Video of Demo Empty Water Shallow/Deep – Guess Algorithm



35

Creating Heuristics



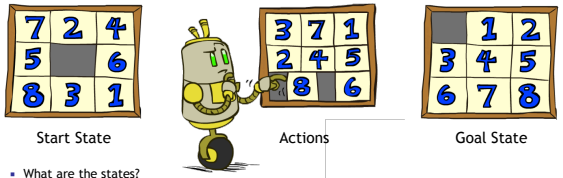
36

Creating Admissible Heuristics

- Most of the work in solving hard search problems optimally is in coming up with admissible heuristics
- Often, admissible heuristics are solutions to *relaxed problems*, where new actions are available

- Inadmissible heuristics are often useful too

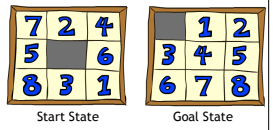
Example: 8 Puzzle

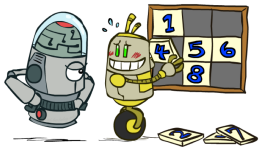


- What are the states?
- How many states?
- What are the actions?
- How many successors from the start state?
- What should the costs be?

8 Puzzle I

- Heuristic: Number of tiles misplaced
- Why is it admissible?
- h(start) = 8
- This is a *relaxed-problem* heuristic



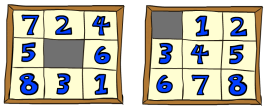


	Average nodes expanded when the optimal path has...		
	...4 steps	...8 steps	...12 steps
UCS	112	6,300	3.6×10^6
TILES	13	39	227

Statistics from Andrew Moore

8 Puzzle II

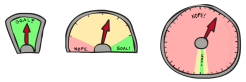
- What if we had an easier 8-puzzle where any tile could slide any direction at any time, ignoring other tiles?
- Total *Manhattan* distance
- Why is it admissible?
- h(start) = 3 + 1 + 2 + … = 18



	Average nodes expanded when the optimal path has...		
	...4 steps	...8 steps	...12 steps
TILES	13	39	227
MANHATTAN	12	25	73

8 Puzzle III

- How about using the *actual cost* as a heuristic?
  - Would it be admissible?
  - Would we save on nodes expanded?
  - What's wrong with it?



- With A*: a trade-off between quality of estimate and work per node
  - As heuristics get closer to the true cost, you will expand fewer nodes but usually do more work per node to compute the heuristic itself