Heuristic Search

Dr. Demetrios Glinos
University of Central Florida

CAP4630 – Artificial Intelligence

Today

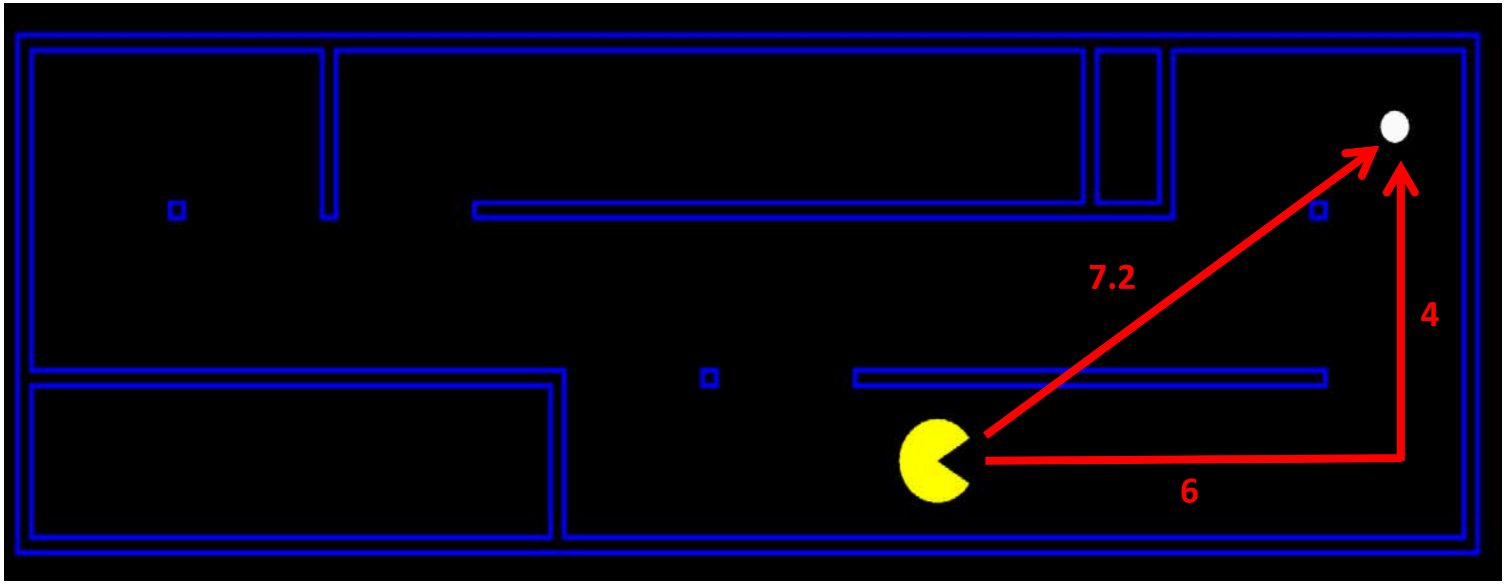
- Informed Search
- Search Heuristics
- Greedy Search
- A* Search
- Admissible Heuristics
- Semi-Lattice of Heuristics
- Consistency of Heuristics
- Effect of Heuristics on Performance
- Optimality

Informed Search

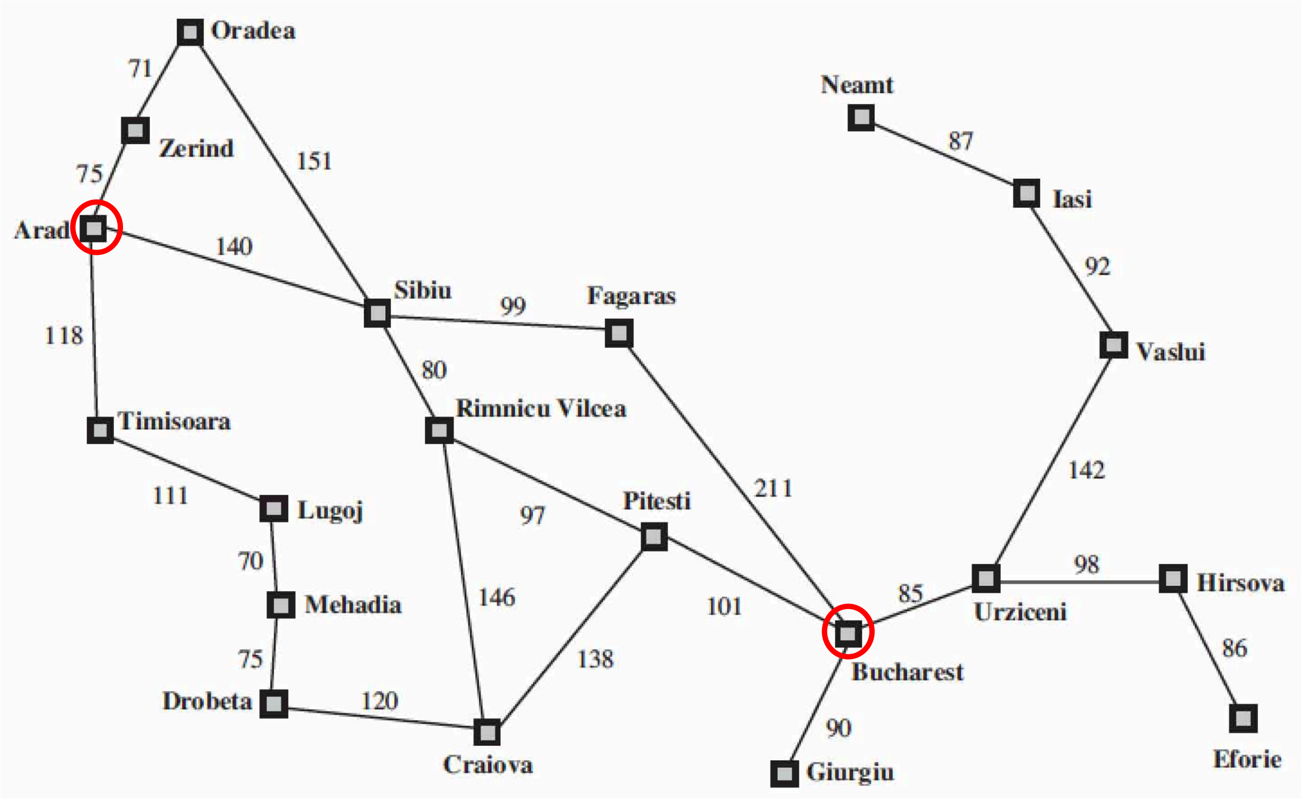
- **Basic idea:** Use some information to know if we are making progress towards goal
  - Are we getting "hotter"
  - Are we getting "colder"

Search Heuristics

- Heuristic defined:
  - A *function* `f:{States}` → `R`
  - that *estimates* "how close" a state is to a goal
  - heuristics are *problem-specific*
    - Examples for Pac-Man: Manhattan distance, Euclidean distance



Example: Heuristic Function



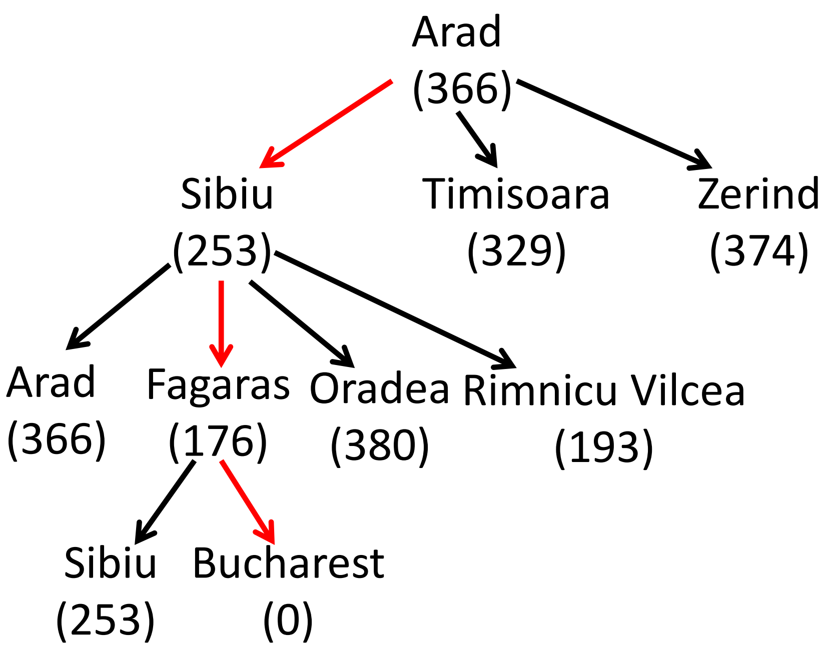
Heuristic function h(x)

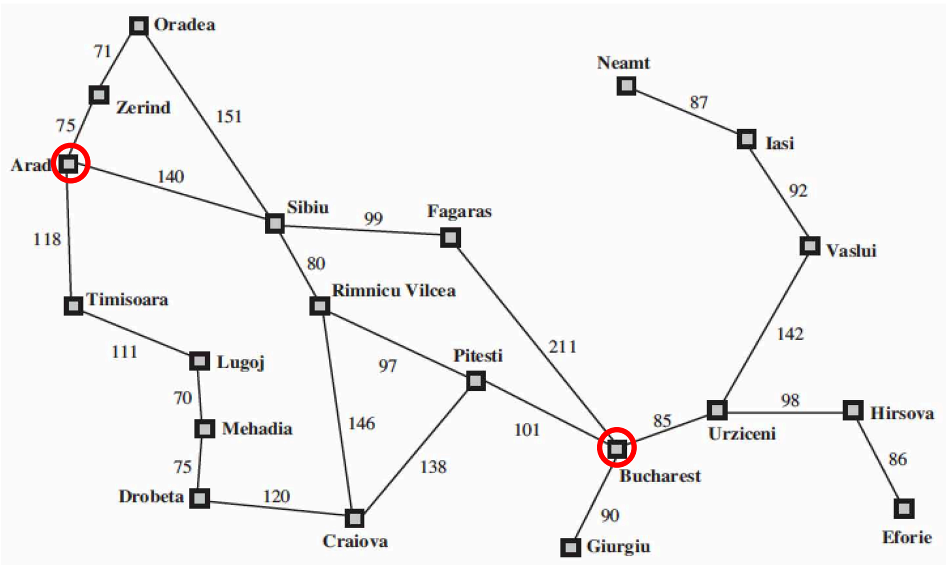
Straight-line distance to Bucharest

Arad	366
Bucharest	0
Craiova	160
Fagaras	176
Sibiu	253
. . .	
Zerind	374

Greedy Search

Strategy: Expand the node that is closest to goal according to the heuristic function

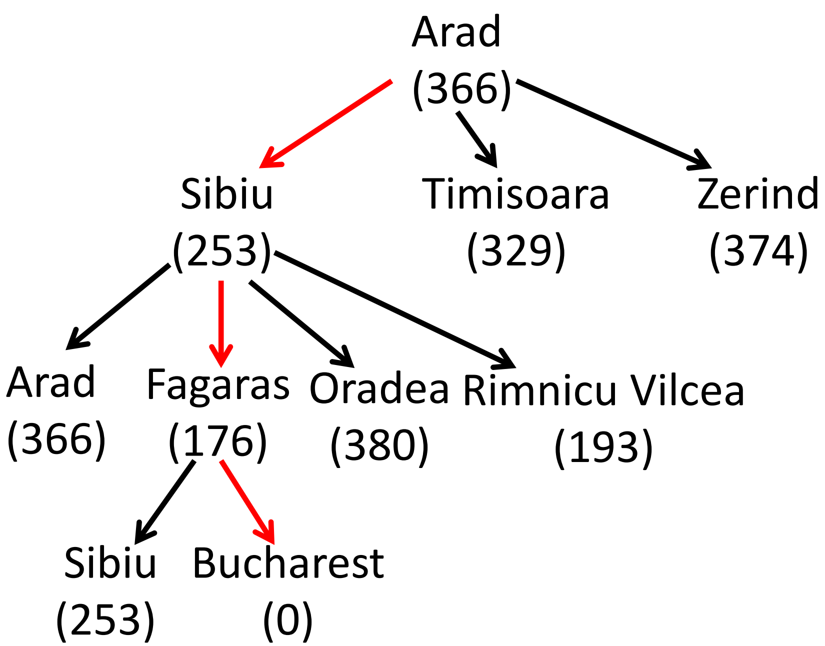


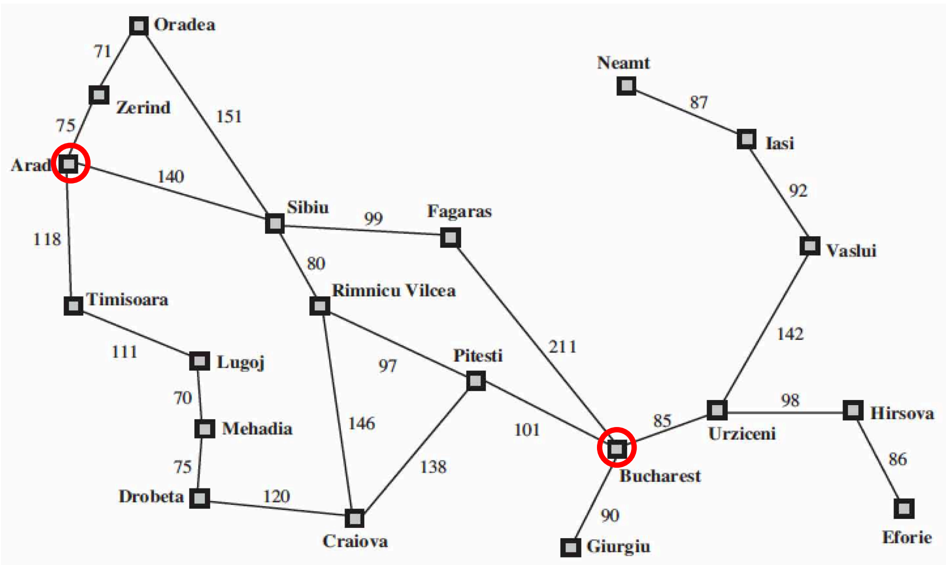


Total distance traveled: 450

Greedy Search

Strategy: Expand the node that is closest to goal according to the heuristic function





Total distance traveled: 450

Shortest path distance: 418

Q: What went wrong, here?

A* Search

- Pronounced: "A-star" search
- Basic Idea: Combine UCS and Greedy
  - UCS path cost is "***backward cost***" ***g(n)***
  - Greedy value is "***forward cost***" ***h(n)***
- A* Search orders by the minimum value of the sum of these two numbers:

$$\boxed{f(n) = g(n) + h(n)}$$

Q: If h(n) = 0, what kind of search is this?

Q: If g(n) = 0, what kind of search is this?

A* Search

- Pronounced: "A-star" search

- Basic Idea: Combine UCS and Greedy

  - UCS path cost is "***backward cost***" $g(n)$
  - Greedy value is "***forward cost***" $h(n)$

- A* Search orders by the minimum value of the sum of these two numbers:

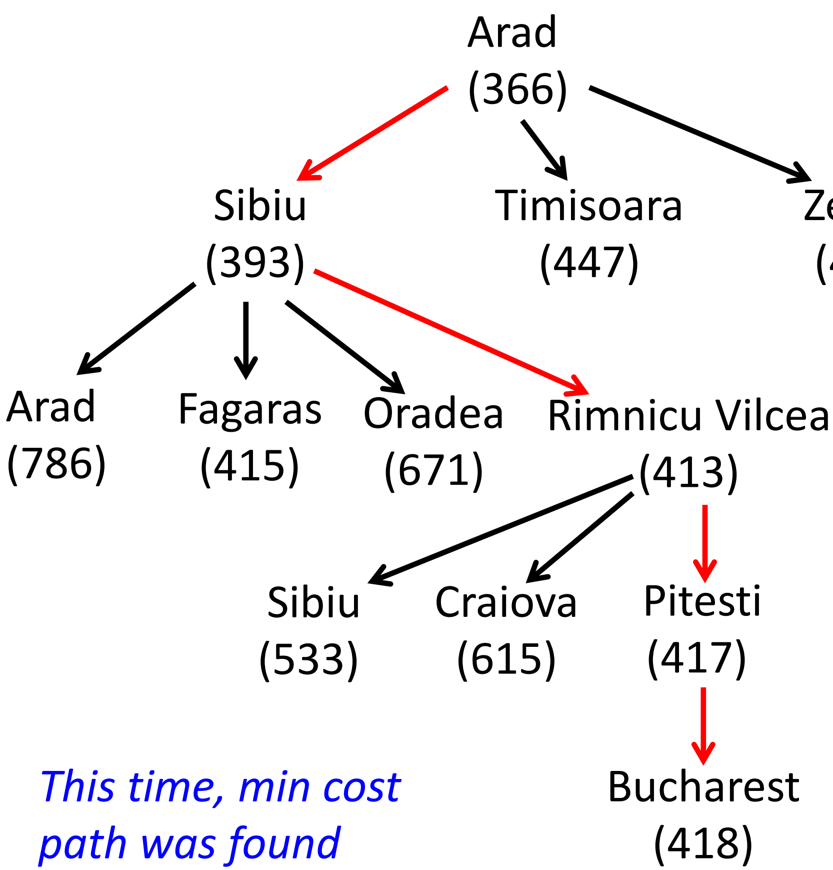
$$\boxed{f(n) = g(n) + h(n)}$$

Q: If h(n) = 0, what kind of search is this? UCS

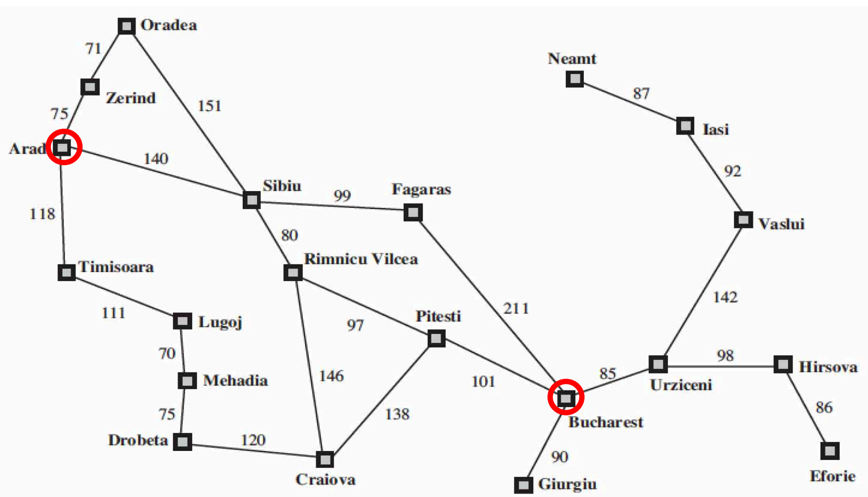
Q: If g(n) = 0, what kind of search is this? Greedy

A* Search Example

Strategy: Order by minimum value of ***f(n) = g(n) + h(n)***



Note:
g(n) is cumulative cost so far
h(n) is NOT cumulative

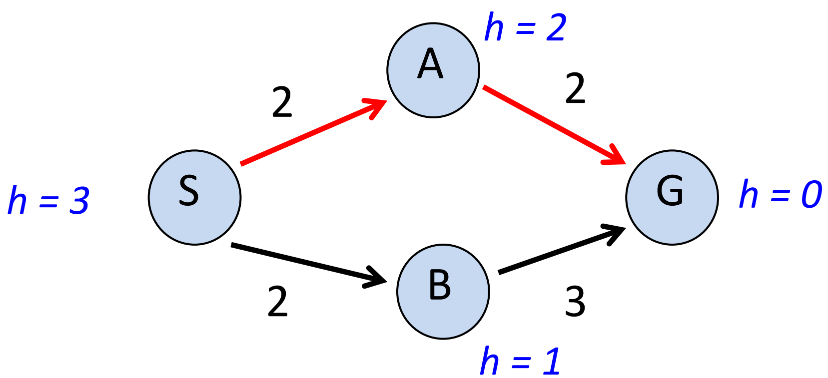


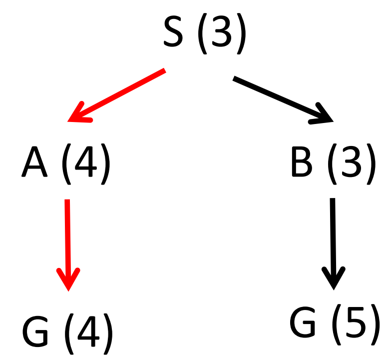
This time, min cost path was found

demo: astar

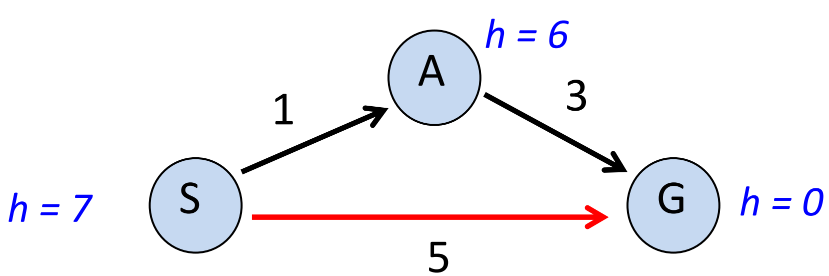
A* Implementation Notes

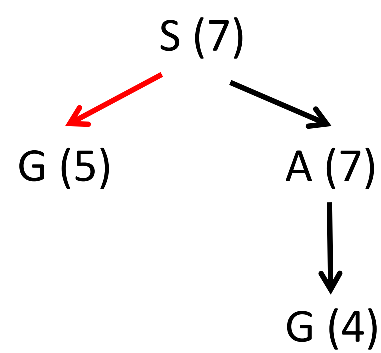
- Stop when dequeue a goal, not when enqueue it





- A* can fail if the heuristic is "bad"





Admissible Heuristics

- Basic idea: The heuristic value should never be greater than the actual cost

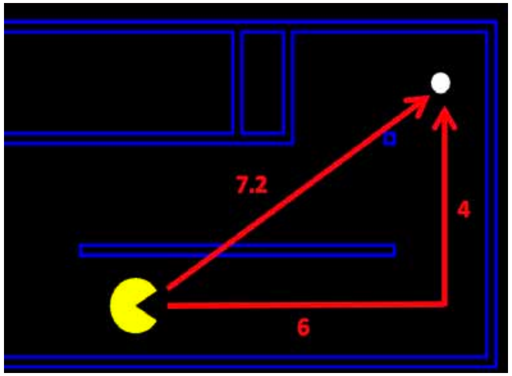
- A heuristic *h* is admissible (optimistic) if:

$$\boxed{0 \leq h(n) \leq h^*(n)}$$

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

- Example:
  - Manhattan and Euclidean distance for Pac-Man

- In practice:
  - Finding an admissible heuristic can be challenging
  - Often done by considering a "relaxed" model



Semi-Lattice of Heuristics

- Dominance: $h_a \geq h_b$ if

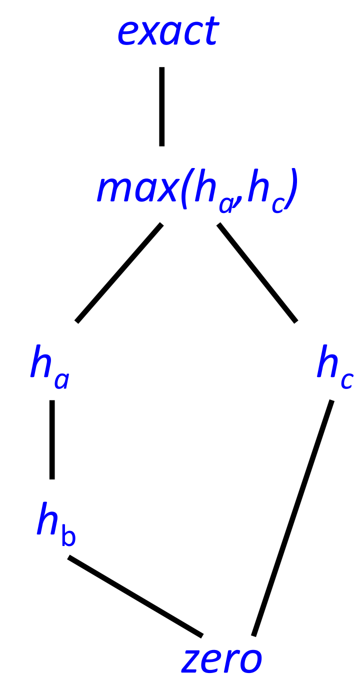
  $$\forall n : h_a(n) \geq h_b(n)$$

- Max of admissible heuristics is admissible

  $$h_{new}(n) = \max[\, h_a(n), h_c(n) \,]$$

  → heuristics form a ***semi-lattice***

- Trivial heuristics
  - Top of lattice is the exact heuristic
  - Bottom of lattice is the zero heuristic (what search method is this?)



(dominance hierarchy)

Consistency of Heuristics

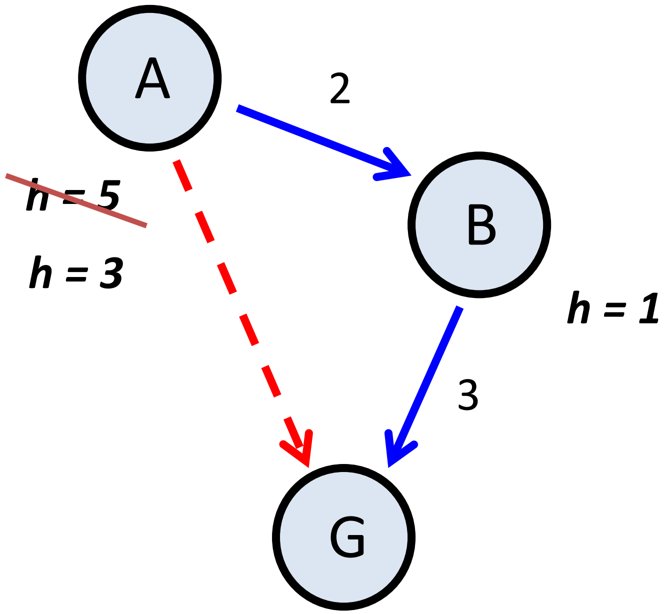
- Consistency is a stronger property than admissibility
- Admissibility:

  h(A) ≤ actual cost from A to G

- Consistency:

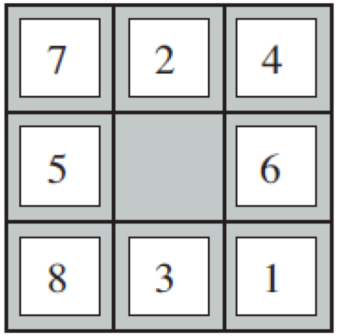
  h(A) – h(B) ≤ actual cost (A, B)

- Impact of heuristic being consistent:
  - The f-value along a path never decreases

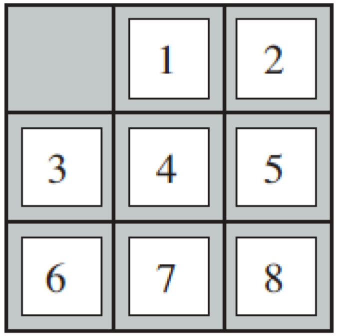


Effect of Heuristic on Performance

- 8-puzzle:
  - h1 = number of misplaced tiles
  - h2 = sum of Manhattan distances of misplaced tiles



Start State

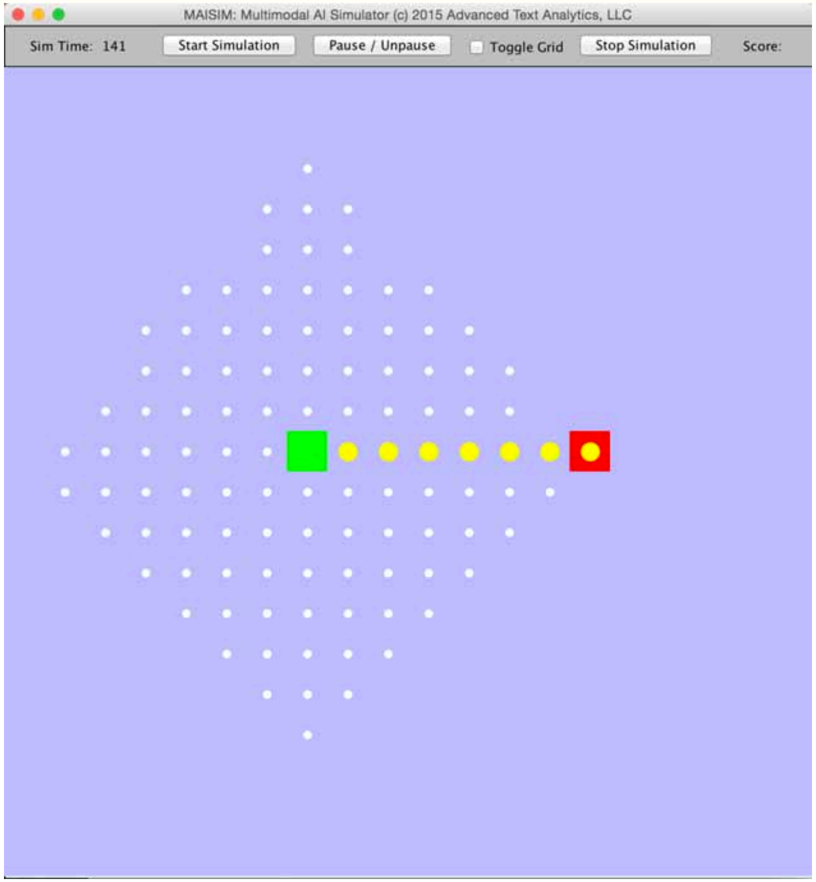


Goal State

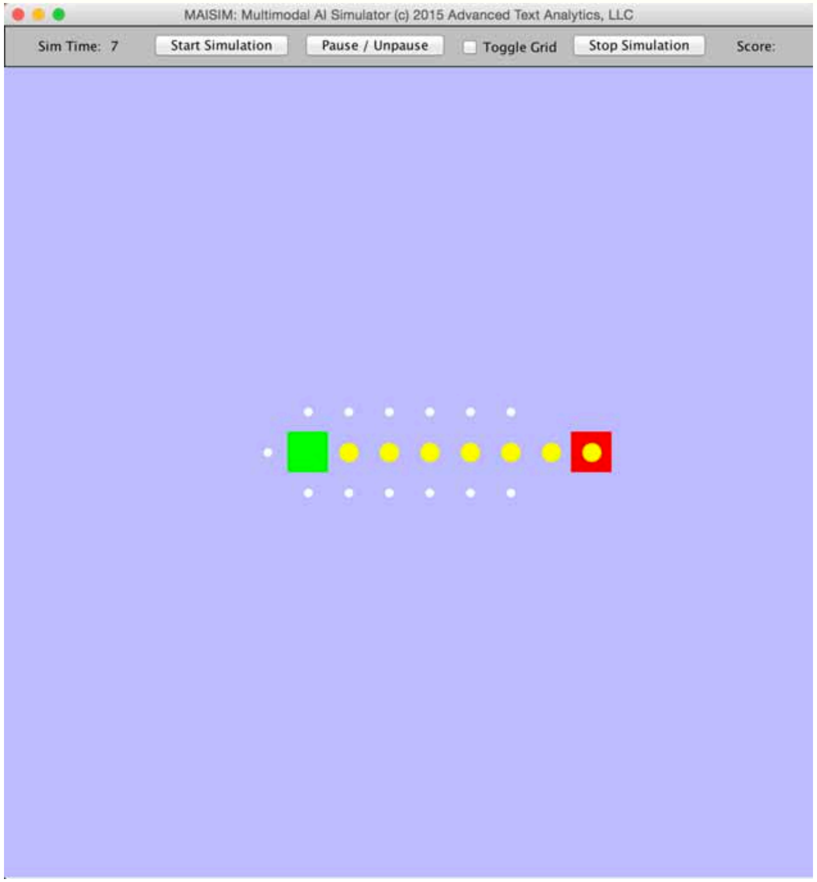
- Number of nodes examined:

# moves in solution	h1	h2
6	20	18
12	277	73
18	3056	363
24	39135	1641

UCS v. A* Contours



UCS expands equally in all directions



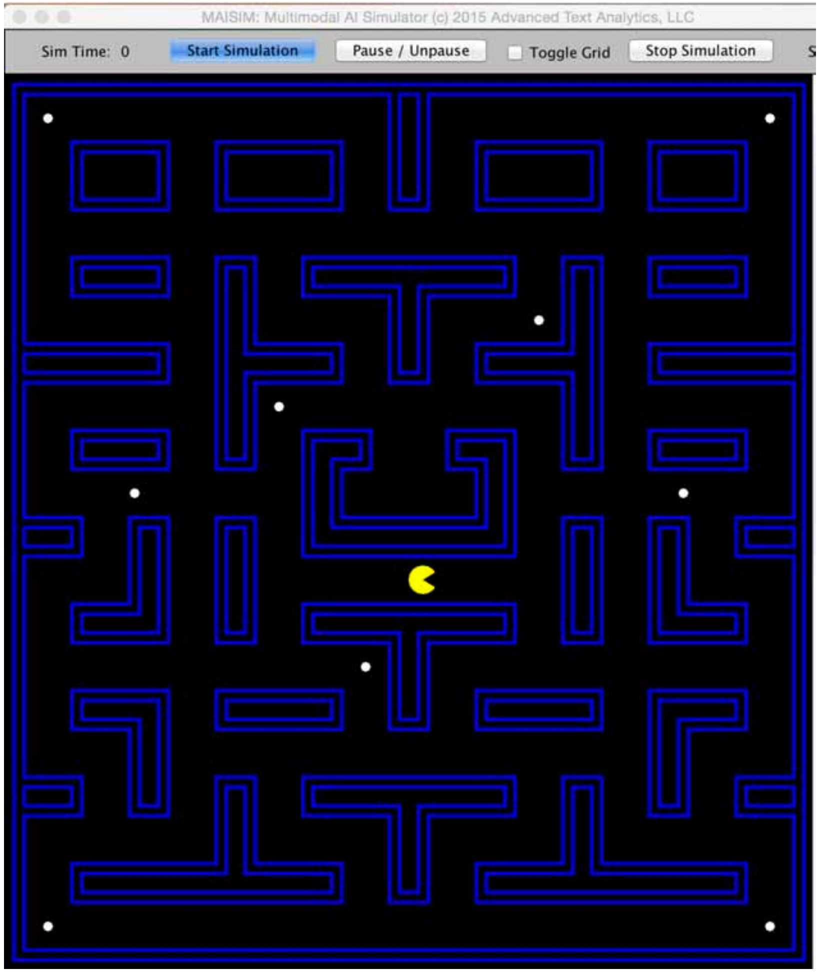
A* expands toward goal, but also some in other directions to ensure optimality

A* Applications

- Pathing/routing applications
- Resource planning/logistics
- Robot motion planning
- Video games/simulations
- Language understanding
- Machine translation
- Speech recognition

- ... and more

- ... including our Pac-Man tour



A* for Pac-Man Tour

- A* heuristic: Manhattan distance to nearest remaining food, plus maximum manhattan distance from any remaining food to any other remaining food

- 9 food pellets:

	UCS	A*
run time	1.230 sec	0.565 sec
nodes examined	162,926	70,160
solution path length	153	153

- 14 food pellets:

	UCS	A*
run time	63.902	28.800 sec
nodes examined	5,303,471	2,683,625
solution path length	185	185

A* and Optimality

- General tree search (states can appear multiple times)
  - A* is optimal if heuristic is admissible
  - UCS is optimal (special case, h=0, so also admissible)
- Graph search (no state appears more than once)
  - A* is optimal if heuristic is consistent
  - UCS is optimal (special case, h=0, so also consistent)
- Notes:
  - Consistency implies admissibility
  - Most admissible heuristics from relaxed problems are also consistent