A* search

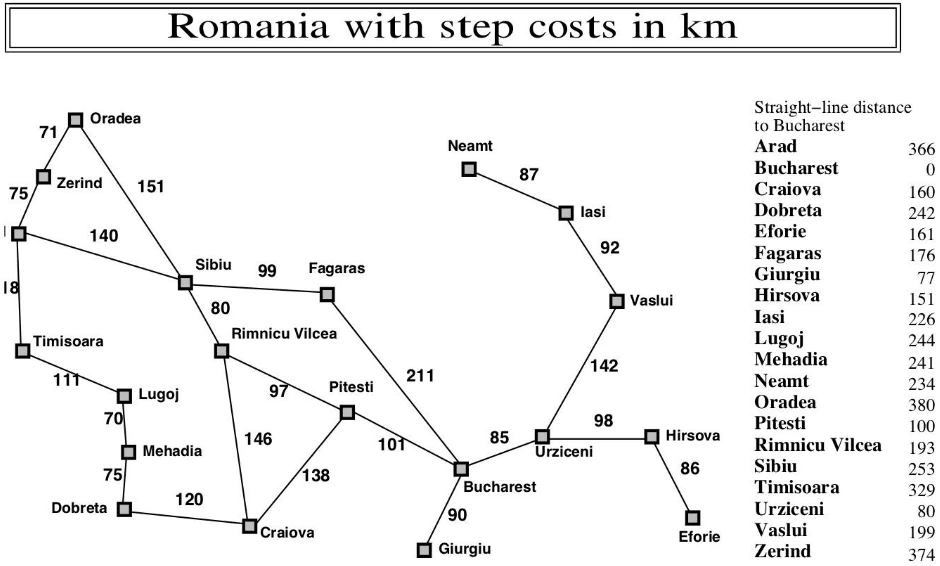
CS 3600

Intro to Artificial Intelligence

Incorporating Domain Knowledge

We need a formal way to introduce our knowledge about the problem to our agent

- "Distance by car is **at least** as much as the straight line distance"
- "A completed Sudoku puzzle has to fill in all the empty squares"
- "A TSP solution has to visit each node"



We can encode each of these as a function that maps states to **approximately remaining path cost** to the goal.

Heuristic

h(s) := approximate cost to goal

Attempt 1: Greedy Best-first Search

UCS, except use **h** instead of **g** as the priority.

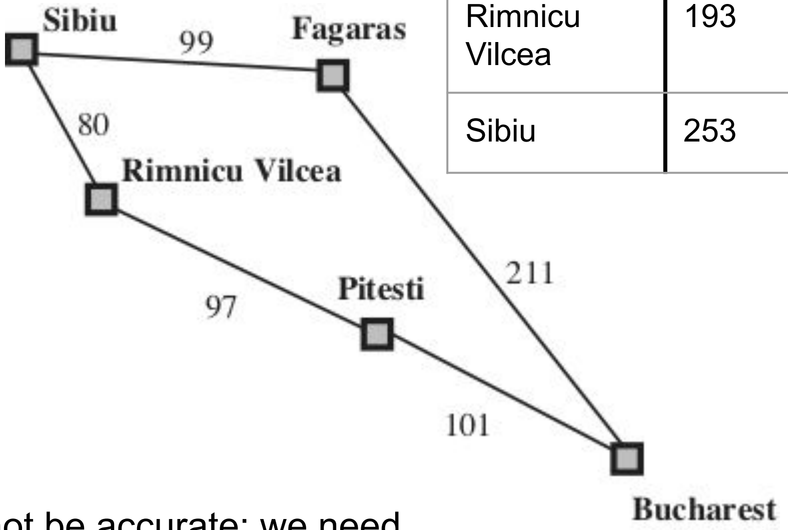
[F(176), R(193)]
[S]

[B(0), R(193)]
[S,F]

Returned path:
S->F->B (cost: 310)

Optimal path:
S->R->P->B (cost 278)

State	h(State)
Bucharest	0
Fagaras	176
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253



Our heuristic might not be accurate; we need to keep track of the **actual cost** as we go.

A* Search

Define a new function for priority

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

The function **f** is an estimate of the “cheapest” solution that passes through n.

A* search is exactly the same as UCS, except using **f**(n) instead of **g**(n) for the priority.

Result: as long as **h** obeys some simple properties, we can **prove** that A* search returns the optimal result!

Simple A* example

[**R**(273), **F**(275)]
[S]

[F(275), **P(277)**]
[S,R]

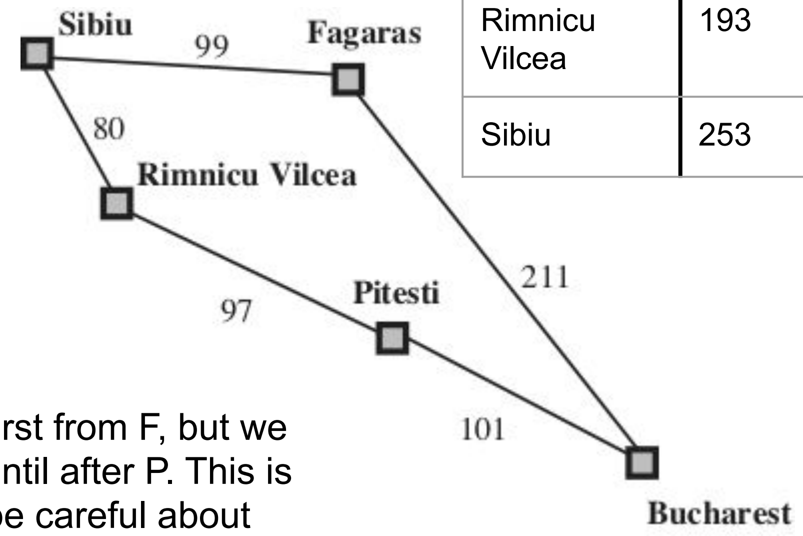
[P(277), **B(310)**]
[S,R,F]

[**B(278)**, B(310)]
[S,R,F,P]

Returned Solution:
[S->R, R->P, P->B]

Note: we **add** B first from F, but we don't **expand** B until after P. This is why we need to be careful about when we stop searching

State	h(State)
Bucharest	0
Fagaras	176
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253



Admissibility - Don’t overestimate

Let’s define the “true cost to go” as h*

$$
\begin{aligned}
G_n &:= \text{ the closest goal to } n \\
h^*(n) &:= \text{ true cost from } n \text{ to } G_n
\end{aligned}
$$

We say a heuristic is **admissible** if it **never overestimates**

$$h(n) \leq h^*(n), \quad \forall n$$

Notice: UCS is A* with h(n) = 0 (null heuristic)

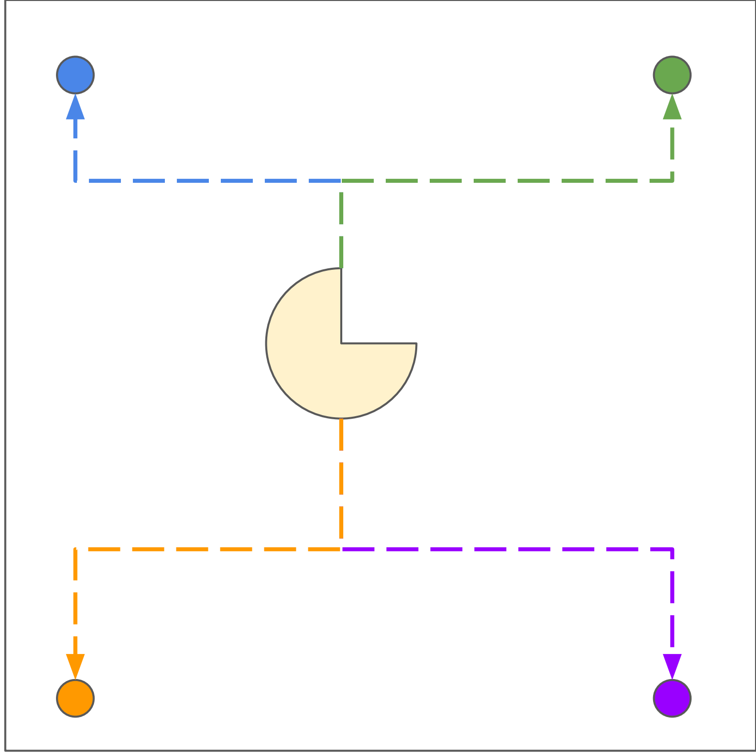
Admissibility - examples

Which of these are admissible heuristics?

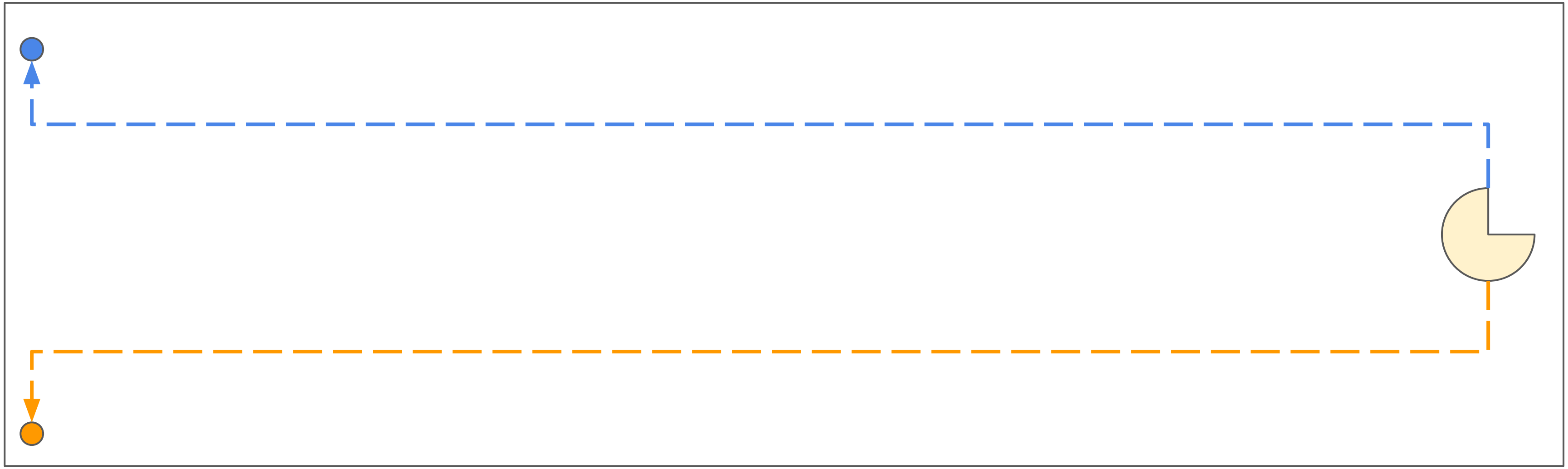
Vacuum world:
h(n) = “number of cells that are not marked clean”

Romania:
h(n) = “Straight Line Distance to Bucharest”

Pac-Man (eating all the pellets):
h(n) = “Sum of the Manhattan distance to remaining pellets”



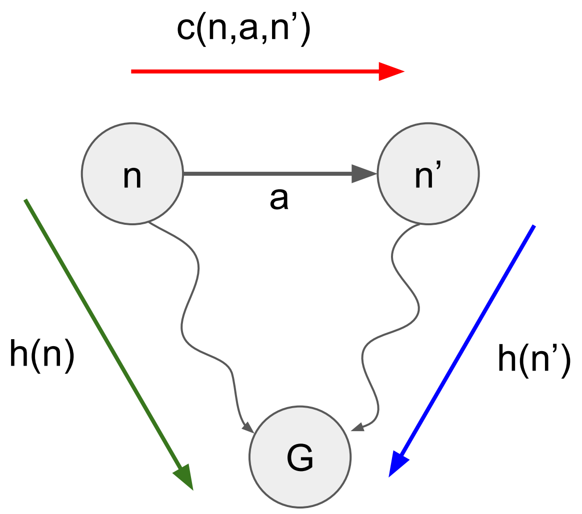
Admissibility - Pac-Man



Moving towards one pellet may put you closer to another pellet!

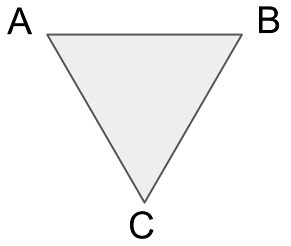
Consistency - the triangle inequality

$$\underbrace{h(n)} \leq \underbrace{c(n,a,n')} + \underbrace{h(n')}$$
$$\forall n, \forall a, \forall n'$$



Any **consistent** heuristic is also **admissible**, but not all admissible heuristics are consistent.

Consistency is important for ensuring that we don't have to backtrack to handle shortcuts as we will soon see



AC <= AB+BC

A* optimality - proof sketch (1)

First, note that the sequence of $f(n_i)$ values along **any path** $(S, n_1, n_2, ..., n_k)$ is non-decreasing

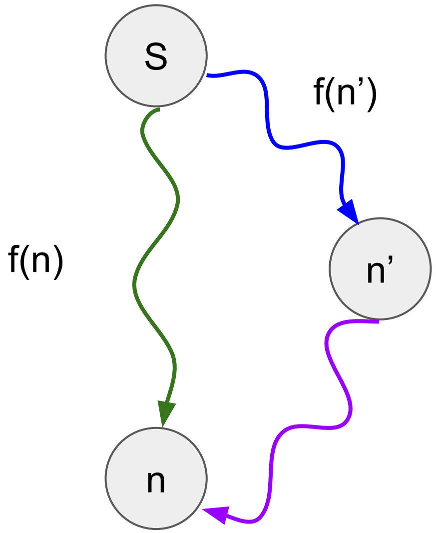
$$\begin{aligned} f(n_i) &= g(n_i) + h(n_i) && \text{(definition)} \\ &\leq g(n_i) + c(n_i, a_i, n_{i+1}) + h(n_{i+1}) && \text{(consistency)} \\ &= g(n_{i+1}) + h(n_{i+1}) = f(n_{i+1}) && \text{(definition)} \end{aligned}$$

A* optimality - proof sketch (2)

Next, whenever a node is expanded, the optimal path to that node has been found.

Proof by contradiction:
Assume n’ is on the optimal path to n. Then f(n’) <= f(n) (by non-decreasing f just shown). But since A* expands nodes in priority order by f, n’ must have already been expanded.

Finally, apply the same logic to any of the goal nodes to see that the first goal node expanded by A* must also be the goal node with smallest f



A* properties

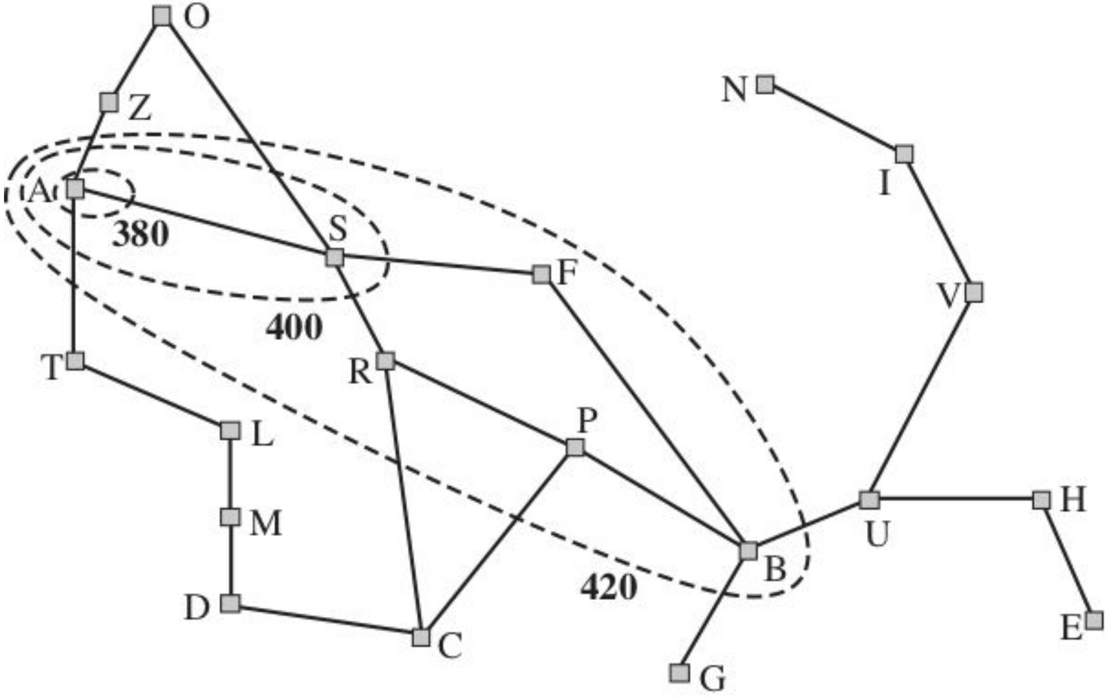
With a consistent heuristic, the following is true about A* (Where C* is the path cost of the optimal solution)

- A* expands **all nodes** with $f(n)<C^*$
- A* may expand **some nodes** with $f(n) = C^*$
- A* will **not** expand **any** nodes with $f(n) > C^*$
- A* is complete if all step costs are positive and the branching factor is finite
- A* is **optimally efficient** among algorithms that expand all nodes with $f(n)<C^*$

So, is A* with consistent heuristics the answer? Why not always use $h(n)=0$?

Effect of the heuristic on explored states

h(n) “shapes” the search towards the best solution.



Heuristic design - preview

Sometimes, admissible/consistent heuristics just expand too many states

We can still perform A* with an inadmissible heuristic, we just can't guarantee optimality anymore

Not all admissible/consistent heuristics are created equal: Making a good choice for the heuristic has strong implications for the running time and memory usage, because it impacts the **effective branching factor**.

Summary and preview

Wrapping up

- A* is like UCS, except using $f(n) = g(n)+h(n)$ for the priority
- In order for A* to be optimal, we need $h(n)$ to have certain properties:
- **Admissibility**: never over estimates true cost to goal
- **Consistency**: $h(n) \le c(n,a,n')+h(n')$ (triangle inequality)
- The choice of the heuristic has a big impact on performance

Next time

- Full Romania example
- Dealing with inconsistent heuristics
- Heuristic design techniques