Informed Search Algorithms

Adriana Caetano

Uninformed vs Informed

Uninformed	Informed
No additional information	Some information
Explores entire search space: lengthy search	Use additional information: efficient search
Always complete (optimal solution)	Heuristic Function
	May or may not be complete

Heuristic Function : h(n)

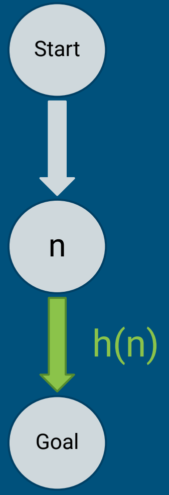
- **Heuristic Function** is an estimate of the cost of the path from the current or initial state to the goal state
- In an informed search algorithm, the heuristic function is used to help choosing a branch from the ones that are available
- The estimated cost should always be lower than or equal to the actual cost of reaching the goal state (admissible heuristic)

$$\mathbf{h(n) <= C(n)}$$

Greedy Best-First Search

Nodes are selected for expansion based on an evaluation function

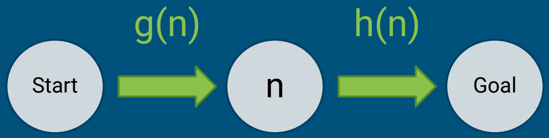
- f(n) = Heuristic Function h(n) = estimated cost from the current state n to the goal state
- Expand the node with the lowest estimated cost



Complete ? No, it can get stuck. Complete only in finite state spaces

Optimal? No, but it’s often efficient

A* Search


$$\begin{tikzcd} \text{Start} \arrow[r, "g(n)"] & n \arrow[r, "h(n)"] & \text{Goal} \end{tikzcd}$$

The most common informed search

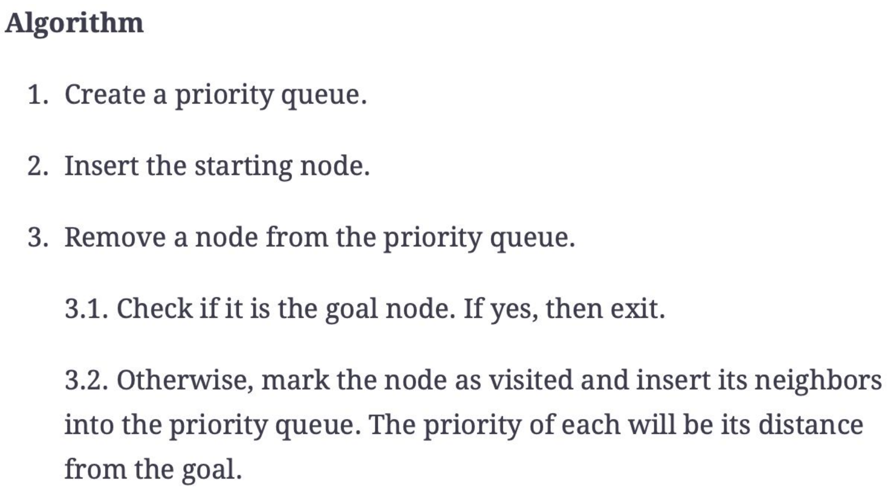
Nodes are selected for expansion based on an evaluation function of the cost from the source to n, g(n), combined with the estimated cost from n to reach the goal, h(n)

- $f(n) = g(n) + h(n)$
- Expand the node with the lowest estimated cost, but it keeps track of last iteration costs. Expands from the lowest cost, and compares costs again, choosing the lowest cost to continue expanding

Complete? Yes

Optimal? Yes, provided a good heuristic function

Greedy Best-First Search



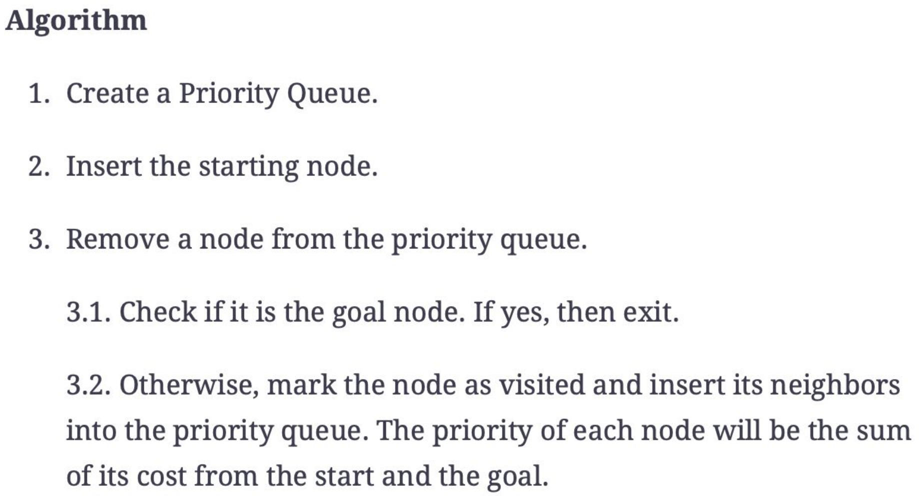
Algorithm

1. Create a priority queue.
2. Insert the starting node.
3. Remove a node from the priority queue.

   3.1. Check if it is the goal node. If yes, then exit.

   3.2. Otherwise, mark the node as visited and insert its neighbors into the priority queue. The priority of each will be its distance from the goal.

A* Search



Algorithm

1. Create a Priority Queue.
2. Insert the starting node.
3. Remove a node from the priority queue.

   3.1. Check if it is the goal node. If yes, then exit.

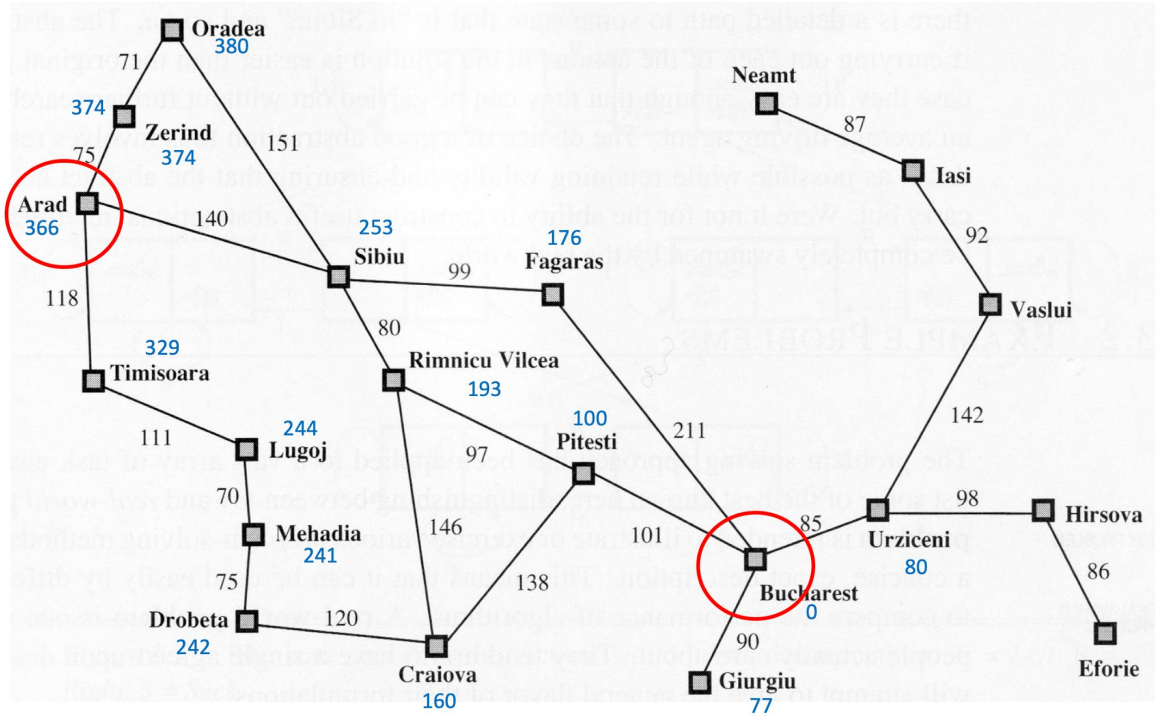
   3.2. Otherwise, mark the node as visited and insert its neighbors into the priority queue. The priority of each node will be the sum of its cost from the start and the goal.

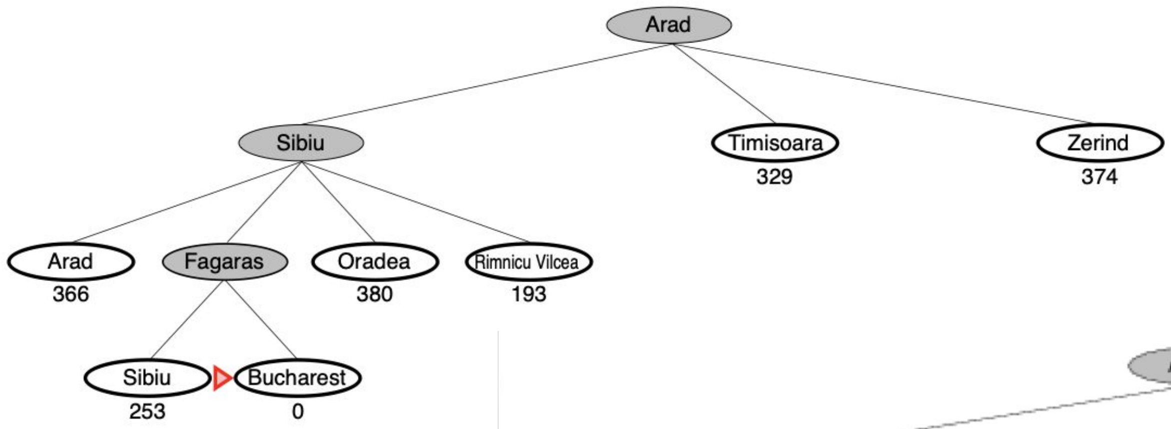
Greedy Best-First Search vs A* Search

Example:

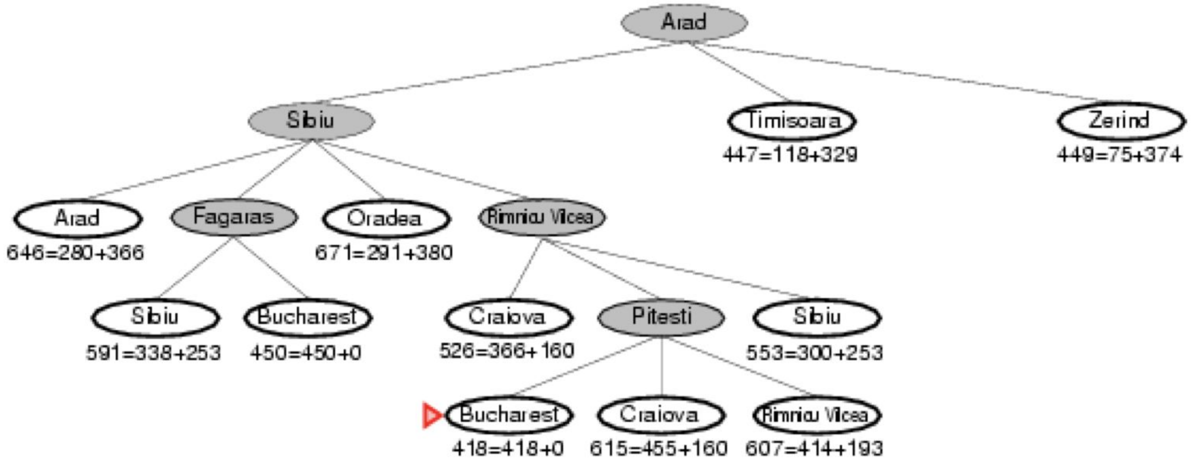
Route-finding problem

Heuristic Function = straight-line distances from the current node to the goal node (in blue)





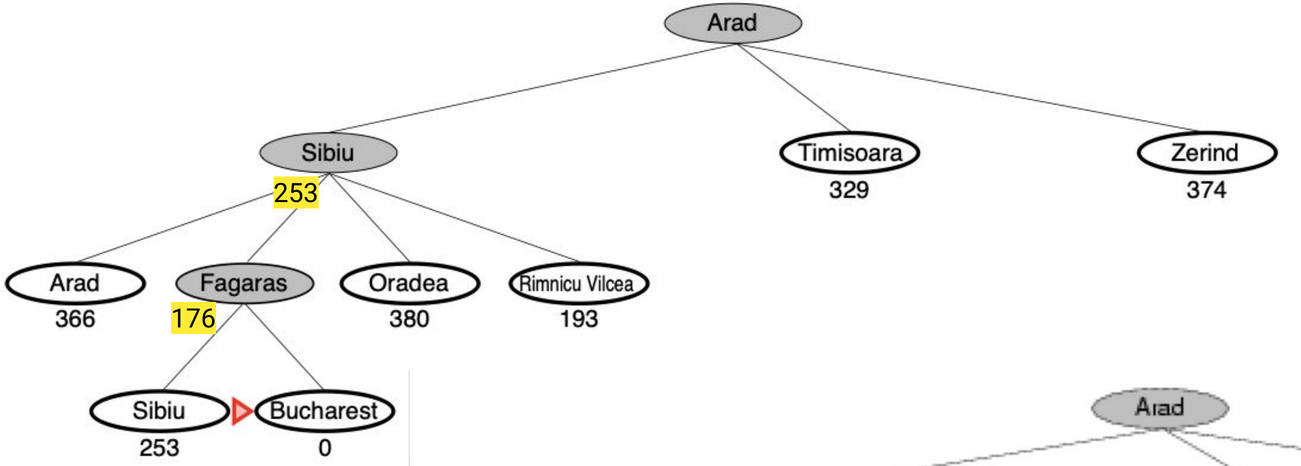
Greedy Best-First Search



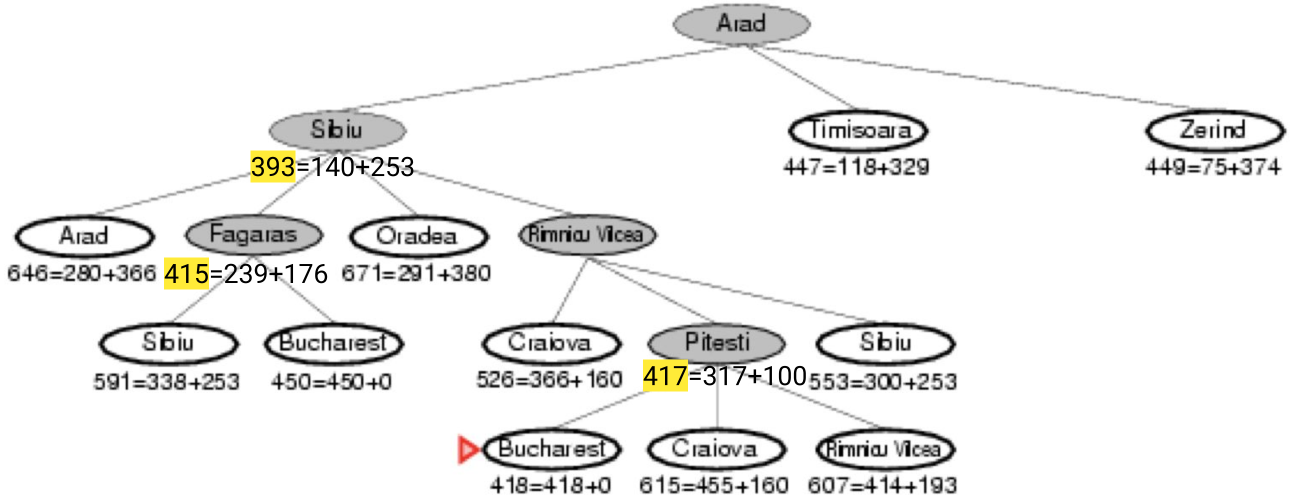
A* Search

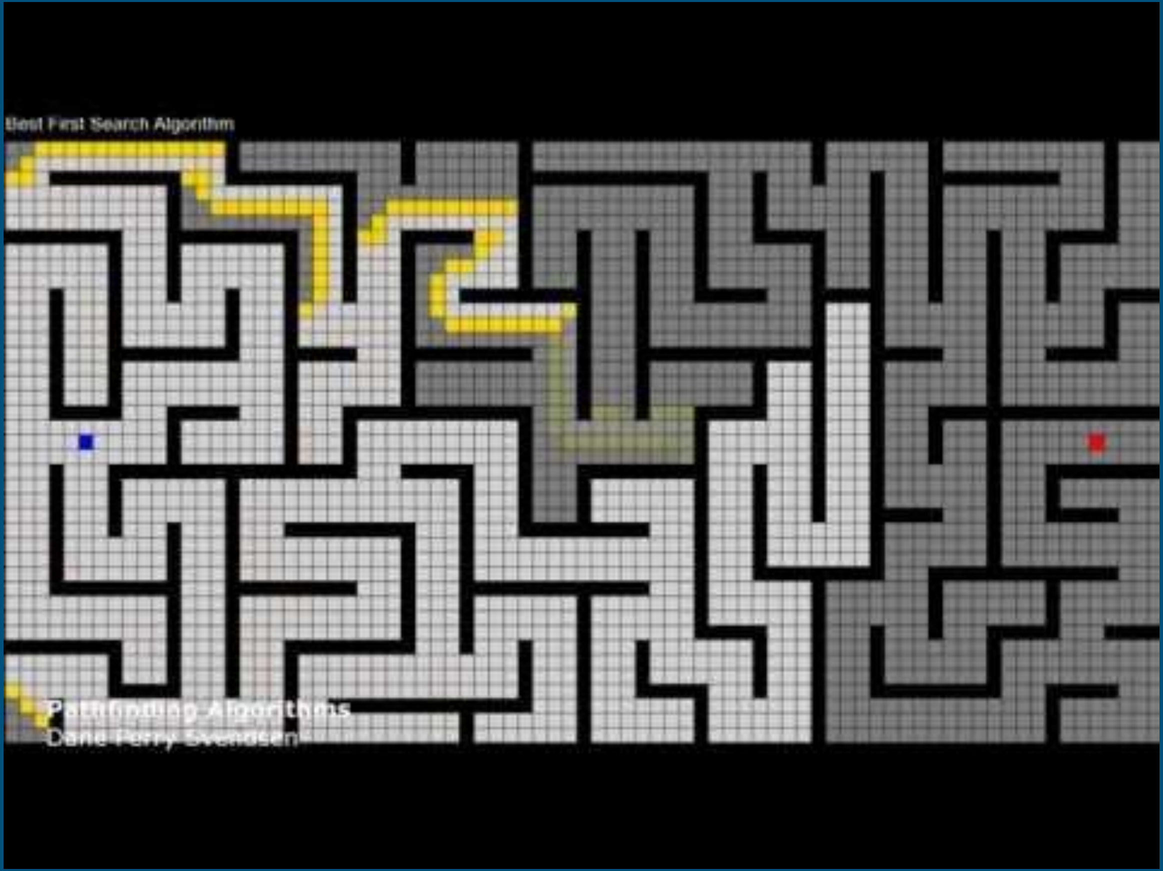
Greedy Best-First Search vs A* Search

y Best-First Search



A* Search




Best First Search Algorithm
Pathfinding Algorithms
Dane Perry Svendsen