Greedy Best-First Search

AI+X
INSTITUTE FOR ARTIFICIAL INTELLIGENCE

Informed Search

- Goal
  - Find solutions more efficiently than uninformed search
- How?
  - Use domain-specific hints about the location of goals
  - The hints come in the form of a **heuristic function**
  - Lower heuristic values indicate nodes closer to the goal
    - Assume these nodes are likely to lead to a solution quickly

Knowledge Check 1

You will travel by car from Tampa to Miami, and you would like to reach the destination as fast as possible. Which city should you go next?

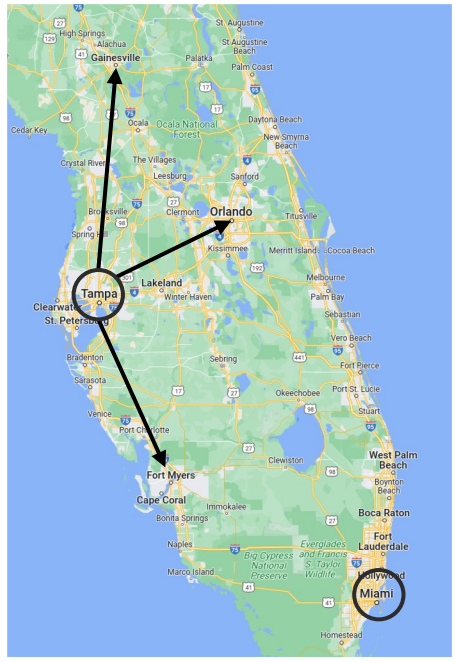


Figure 1. Map of Central & South Florida

A Gainesville

B Orlando

C Fort Myers

Traveling from Arad to Bucharest

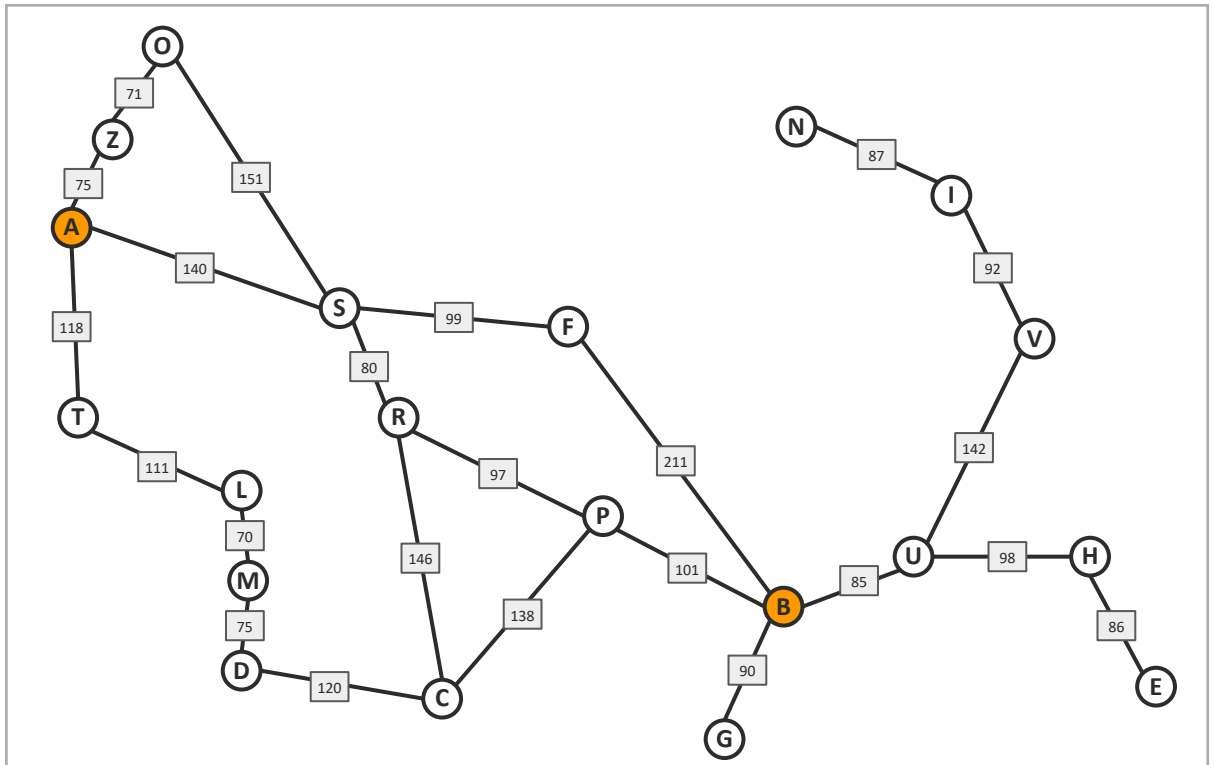


Figure 2. A simplified road map of part of Romania, with road distances in miles.

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

Figure 3. Heuristic function: straight-line distances to Bucharest

Greedy Best-First Search

- Strategy:
  - Expand the node with the lowest heuristic value
- Implementation
  - Fringe (set of leaf nodes) is a priority queue
    - Priority is given to the nodes with lowest heuristic values

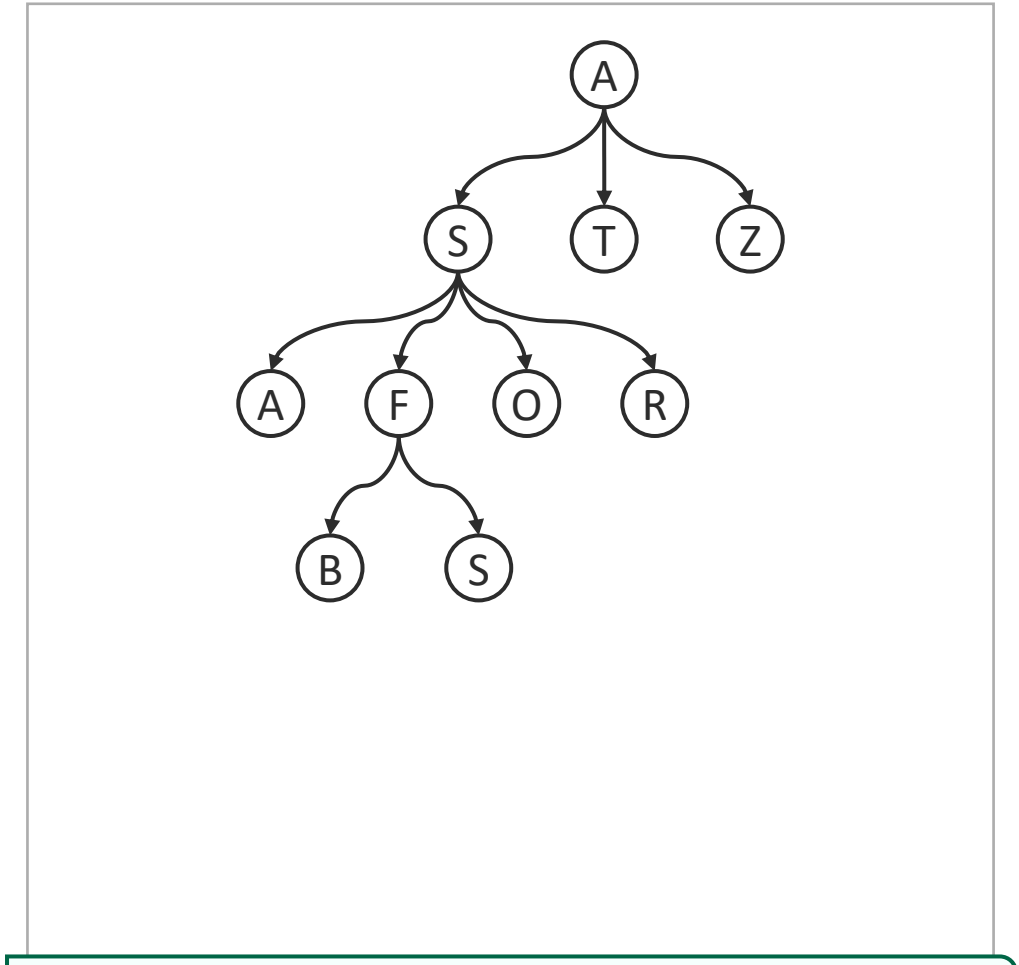


Figure 4. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

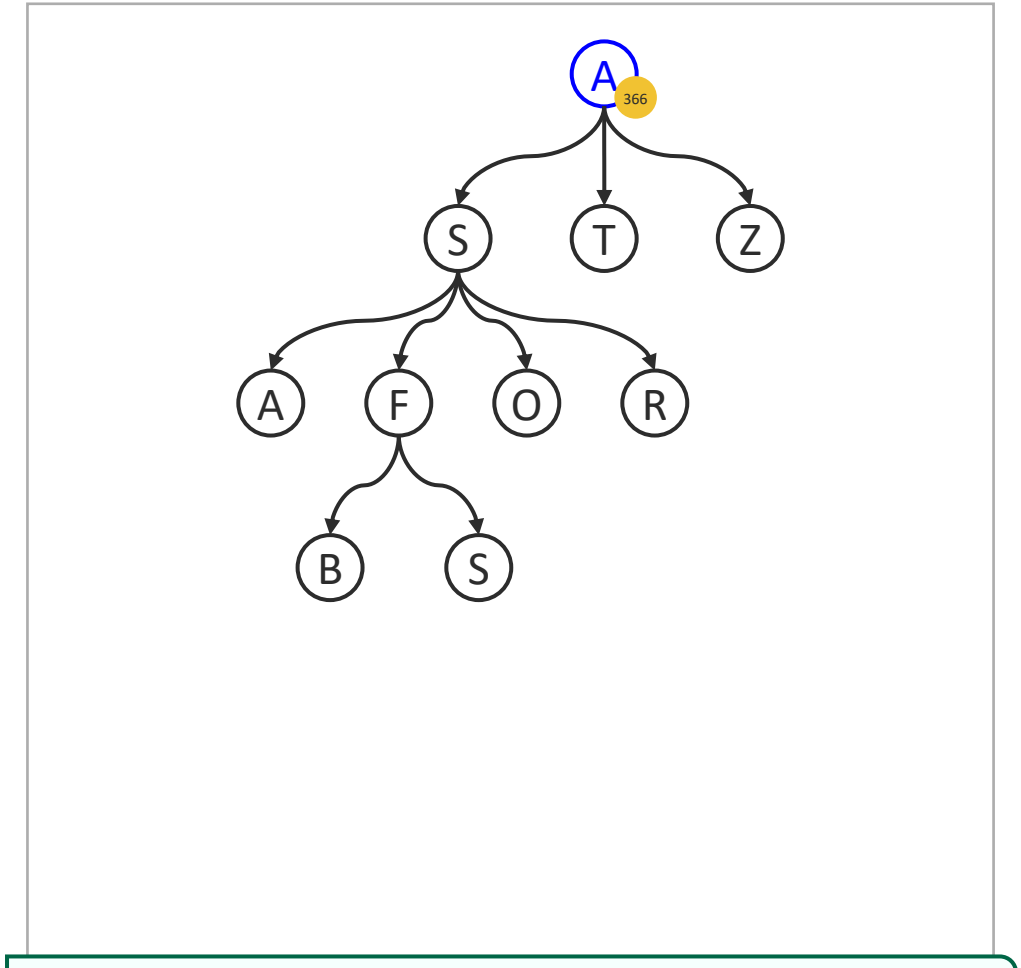


Figure 5. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

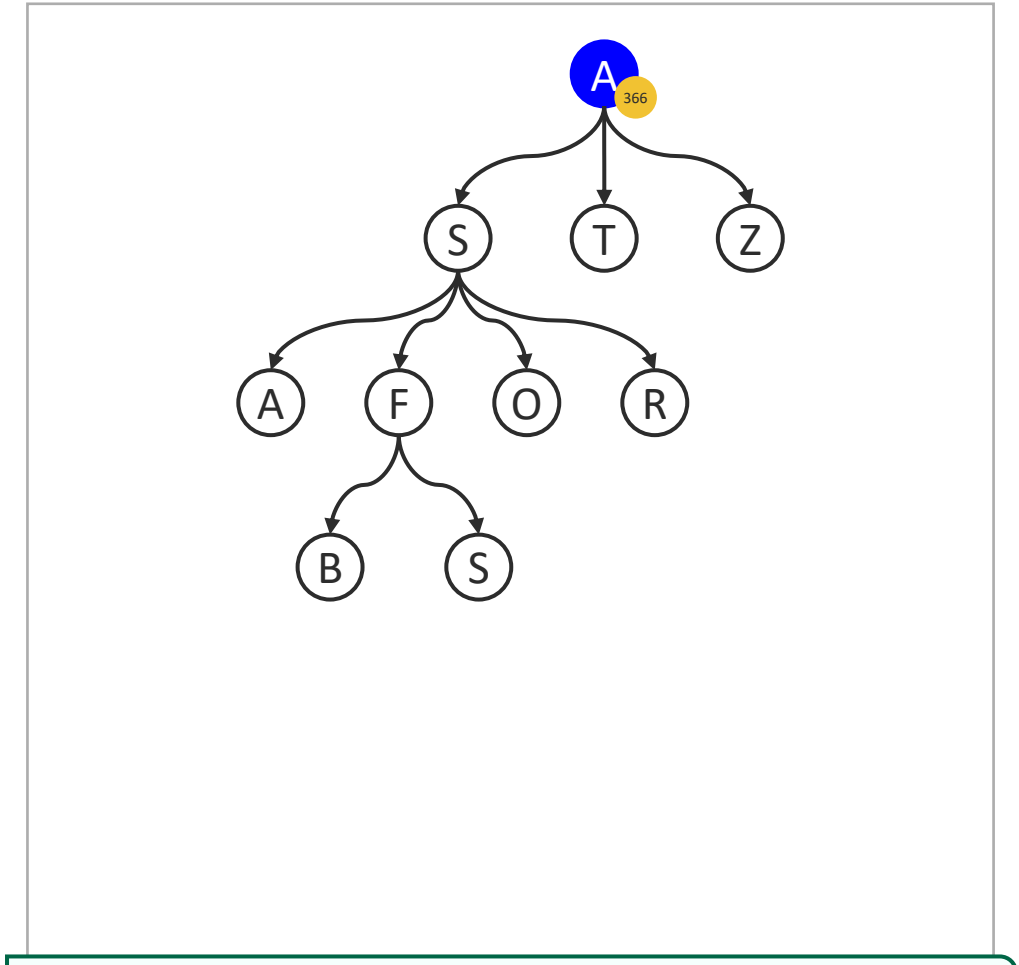


Figure 6. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

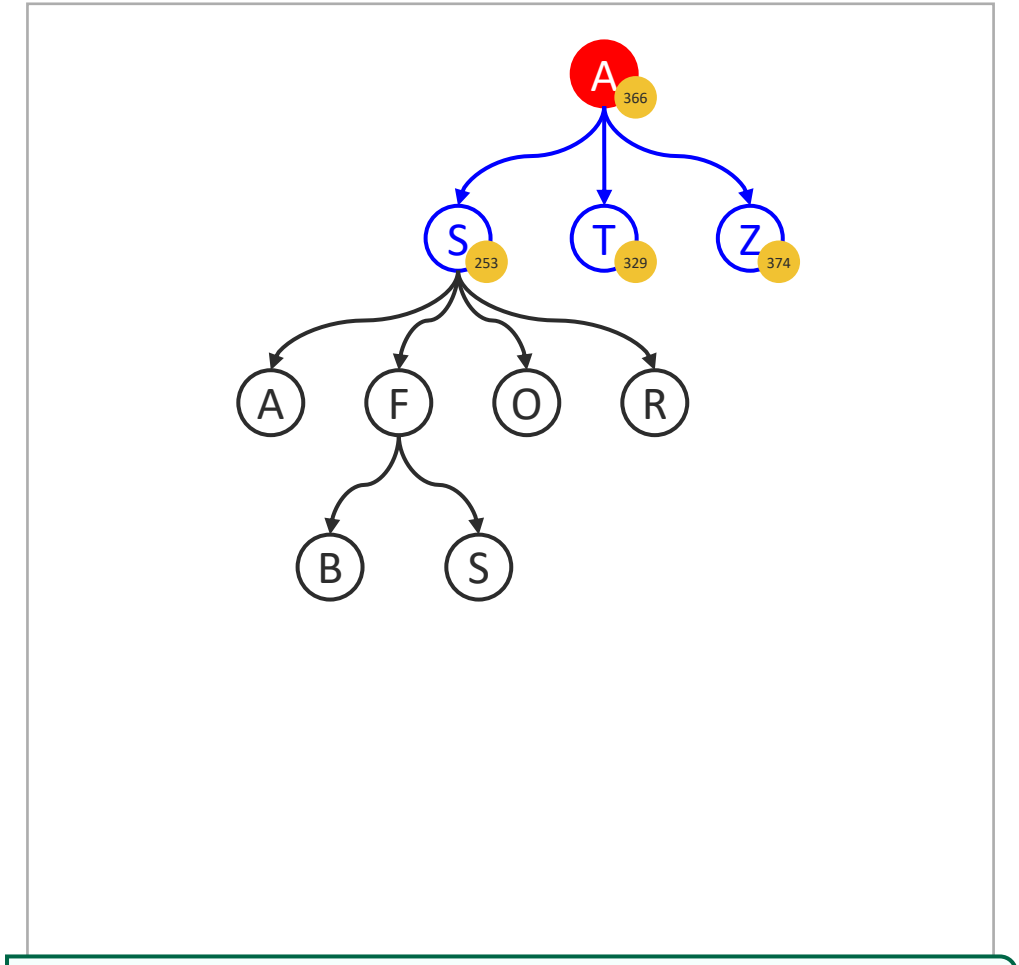


Figure 7. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

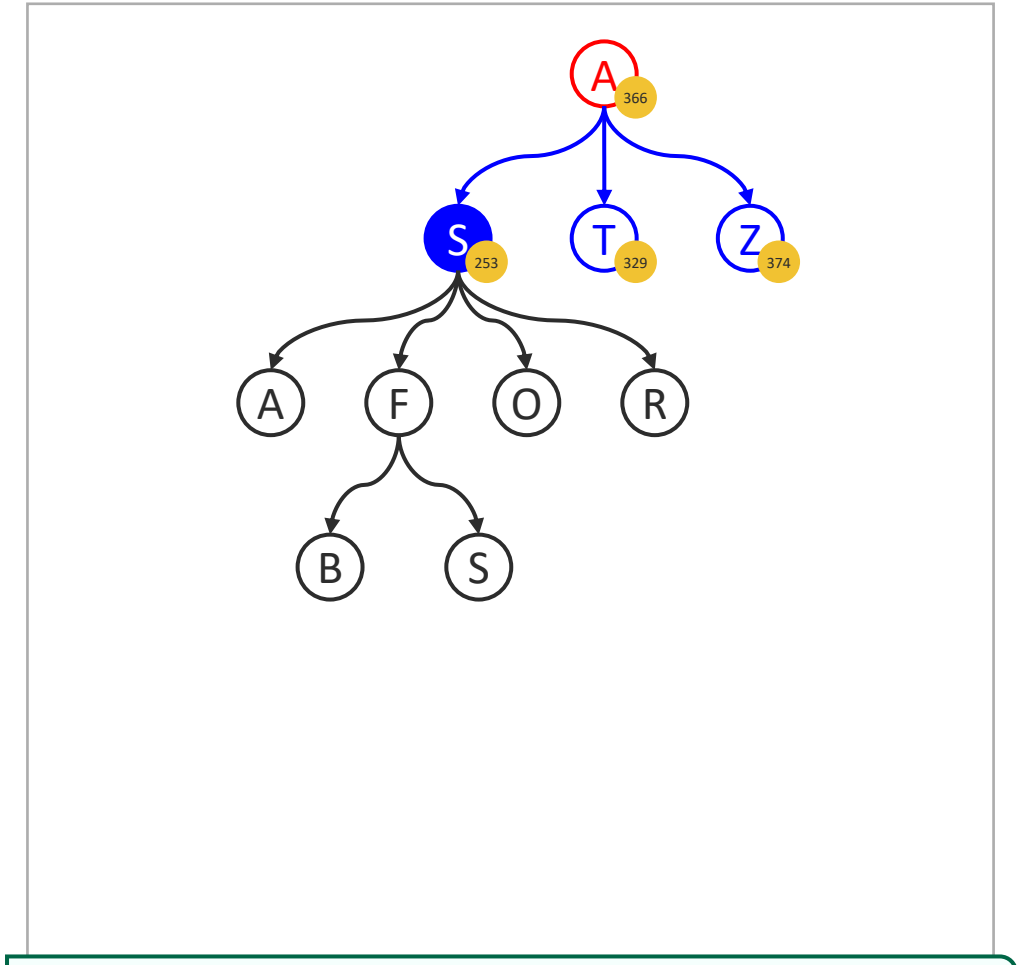


Figure 8. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

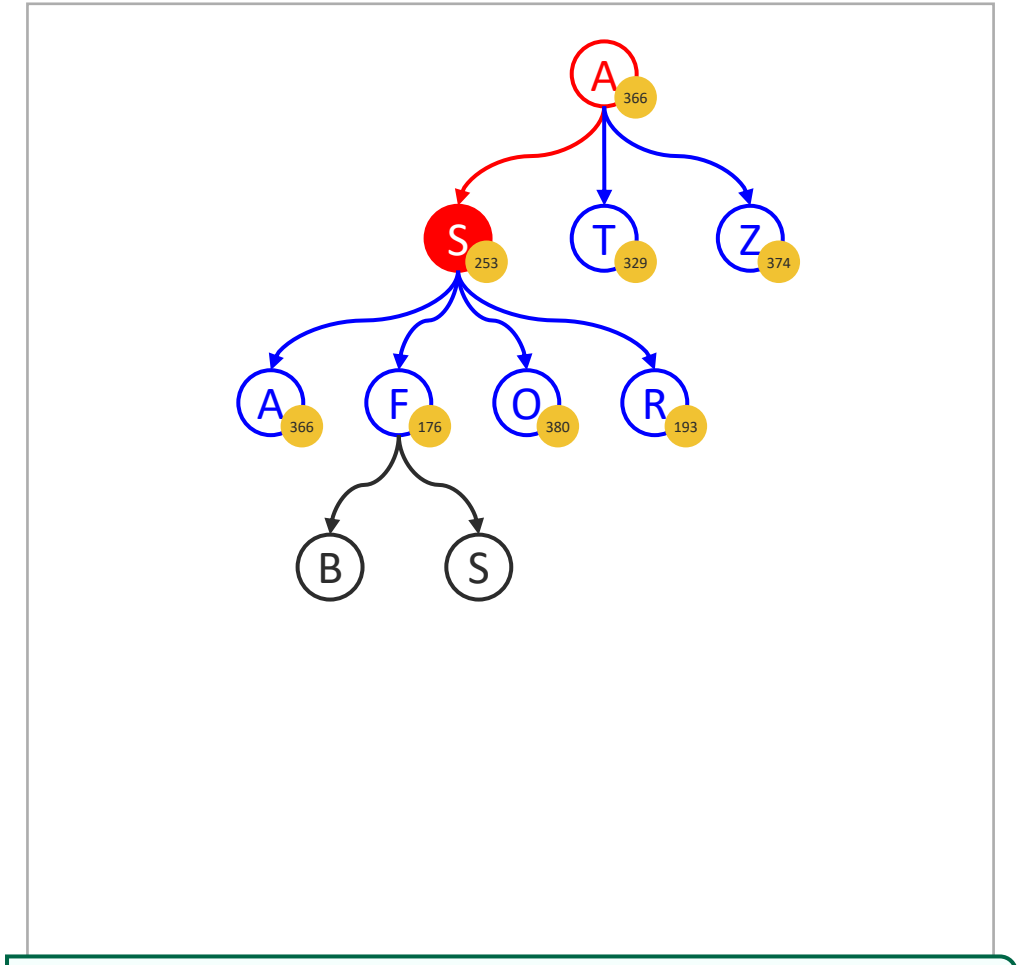


Figure 9. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

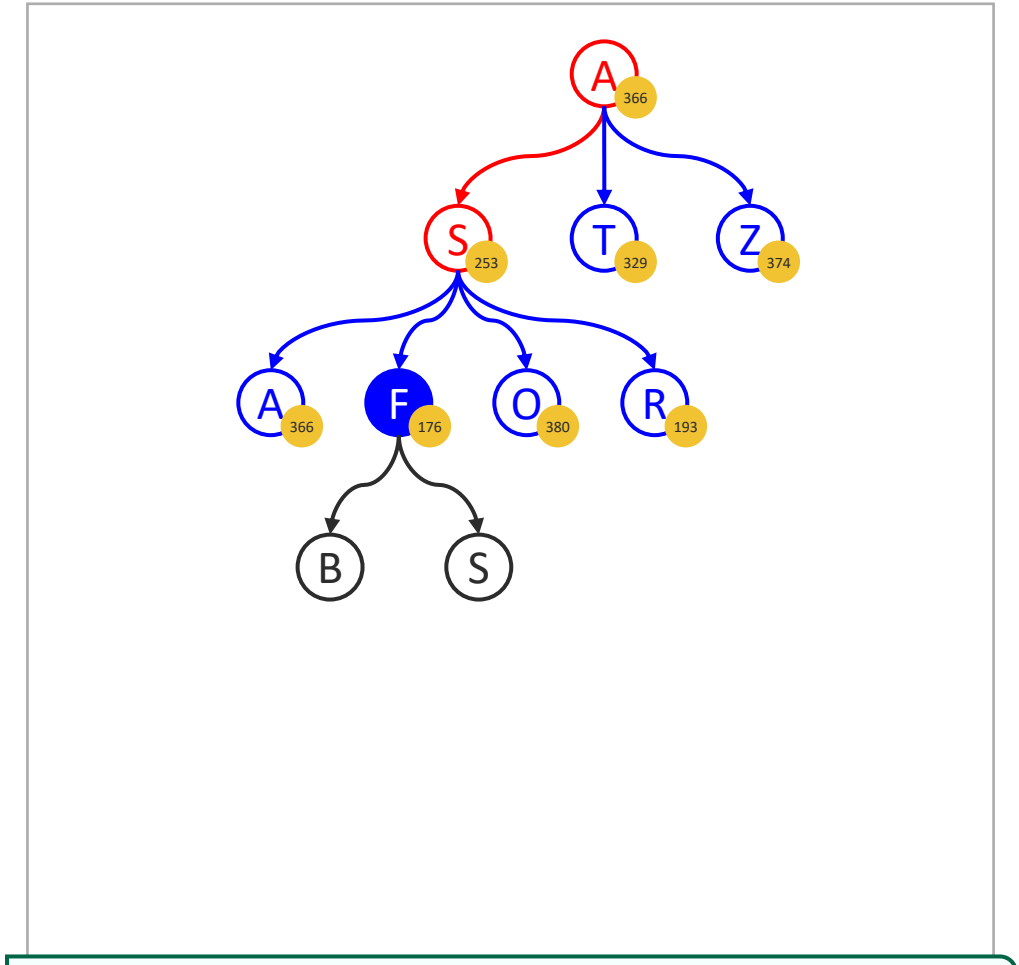


Figure 10. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

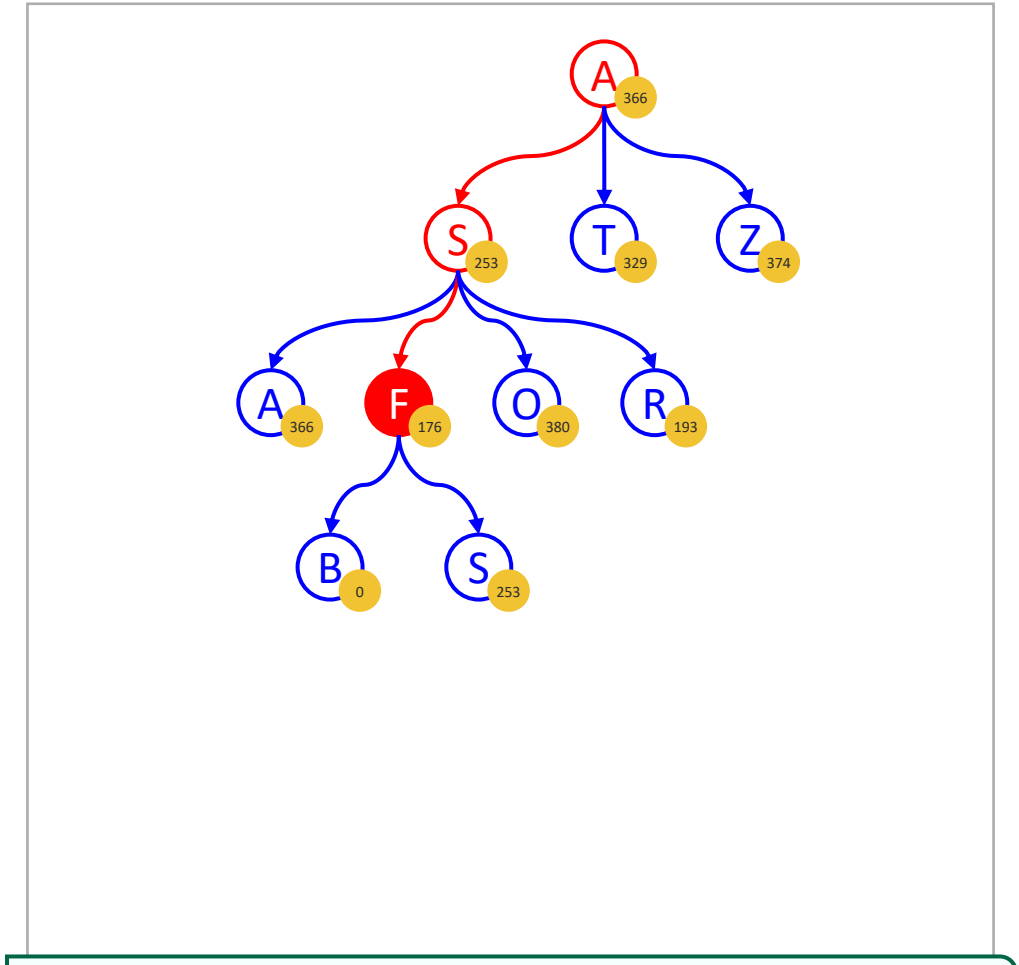


Figure 11. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root
while True:
  choose node with lowest heuristic value for expansion
  if there are no candidates for expansion
    return failure
  if chosen node is a goal state
    return path from root to node
  else
    expand node and add neighbors to the tree
```

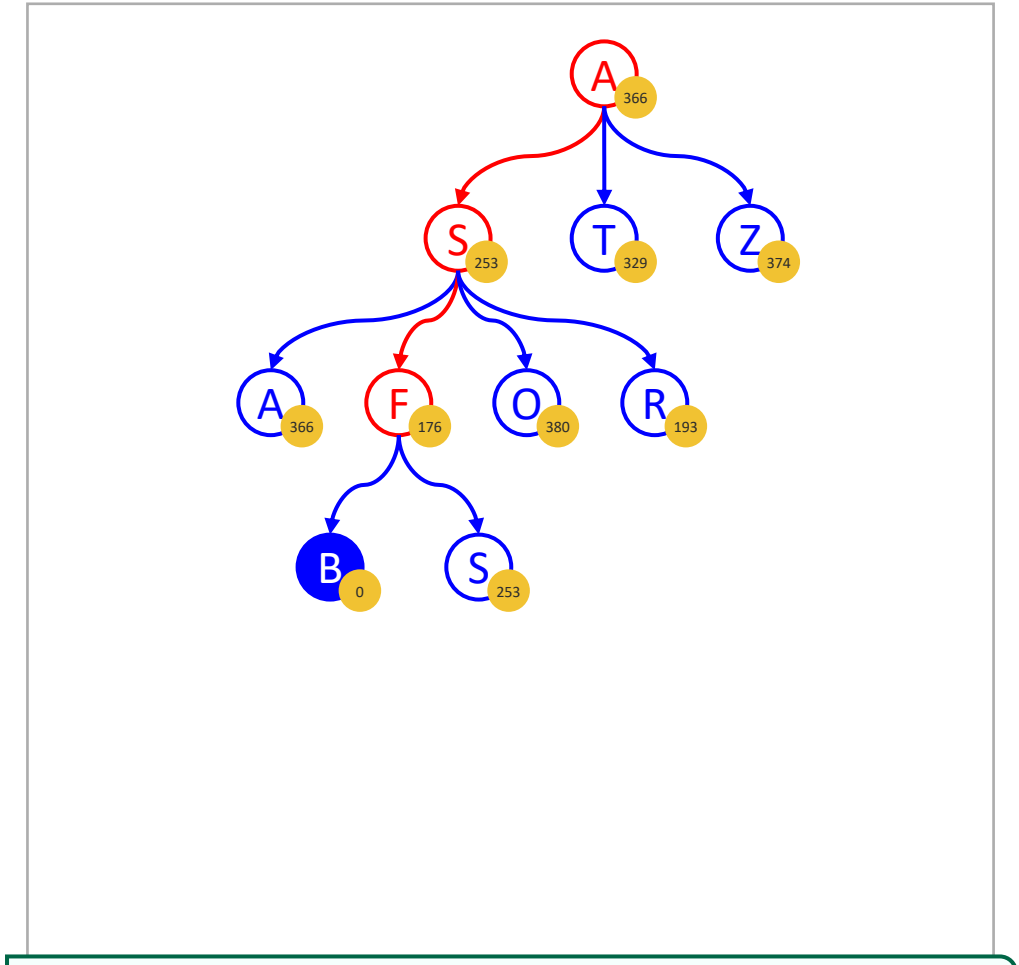


Figure 12. Search tree

Greedy Best-First Search

```
set initial state of problem as the tree root

while True:

  choose node with lowest heuristic value for
expansion

  if there are no candidates for expansion

    return failure

  if chosen node is a goal state

    return path from root to node

  else

    expand node and add neighbors to the tree
```

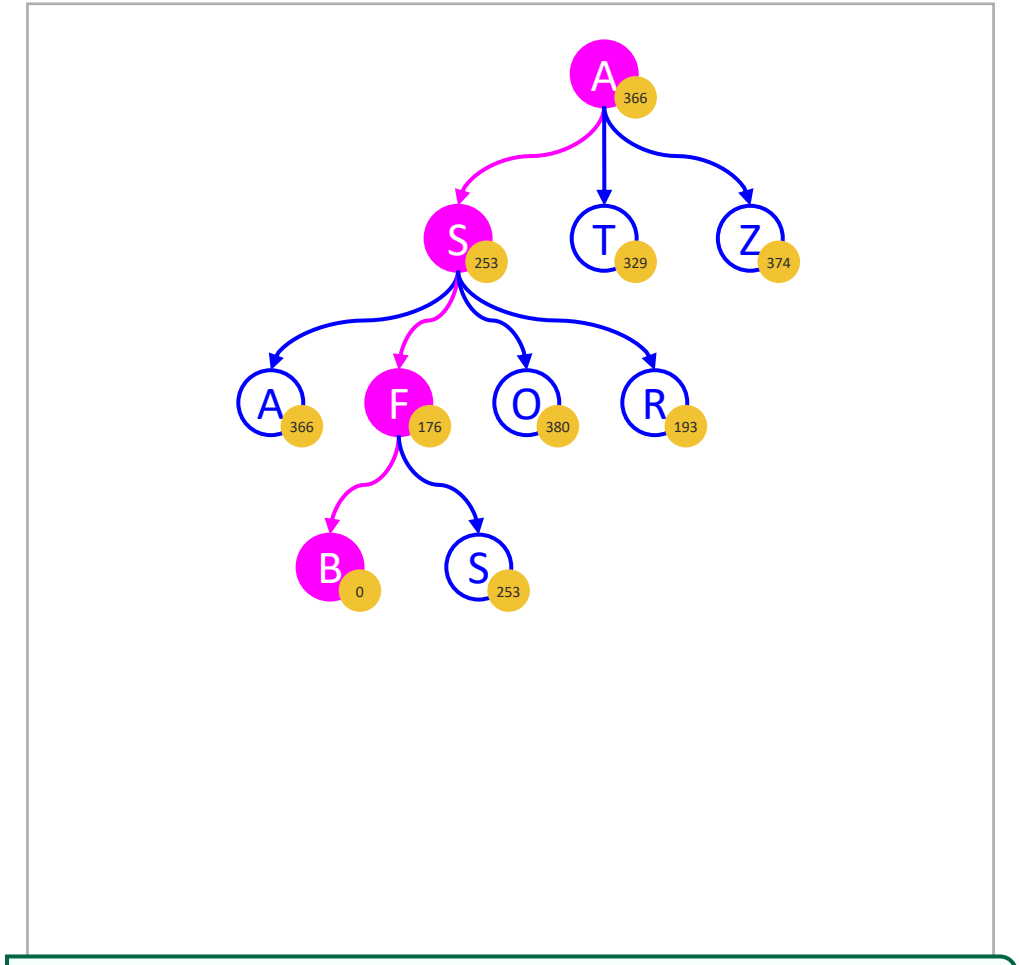


Figure 13. Search tree

Traveling from Arad to Bucharest

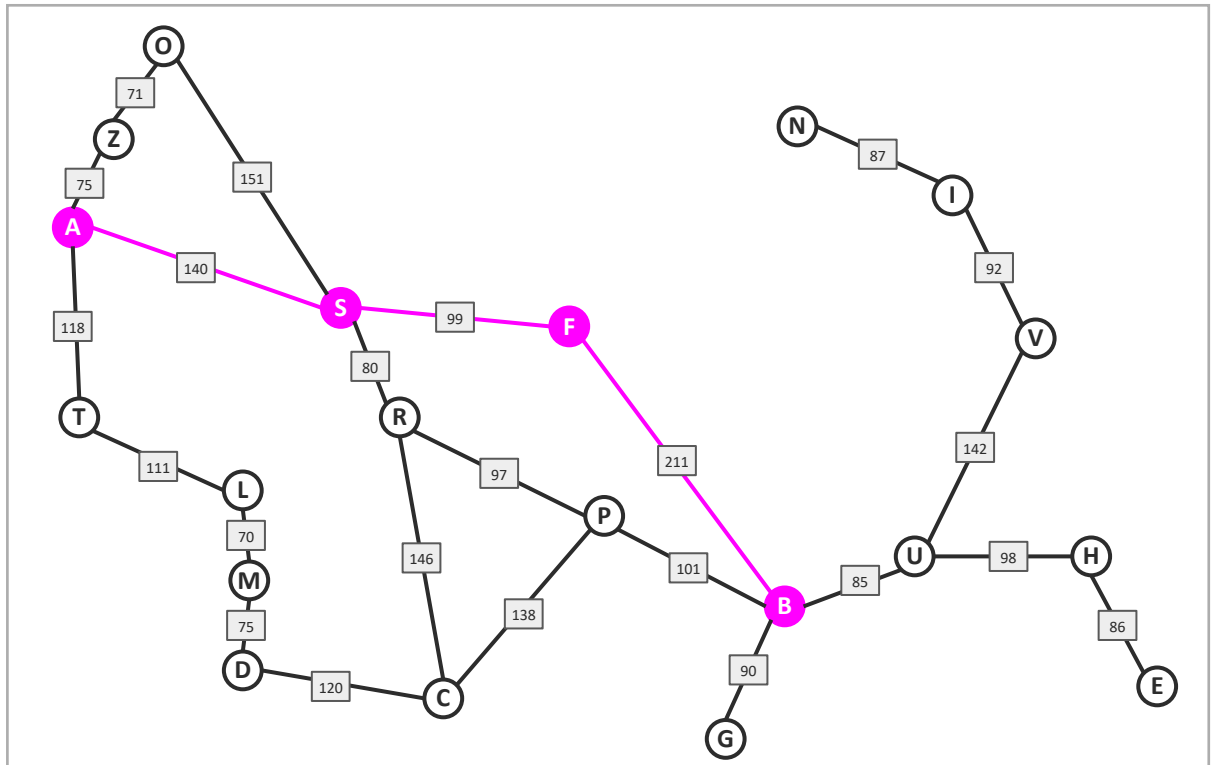


Figure 14. A simplified road map of part of Romania, with road distances in miles.

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

Figure 15. Heuristic function: straight-line distances to Bucharest

Knowledge Check 2

Is Greedy Best-First Search optimal?

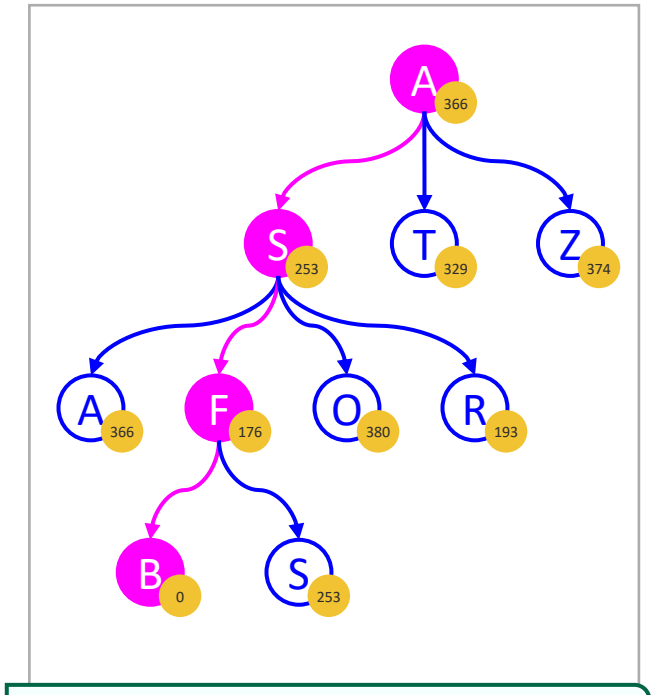
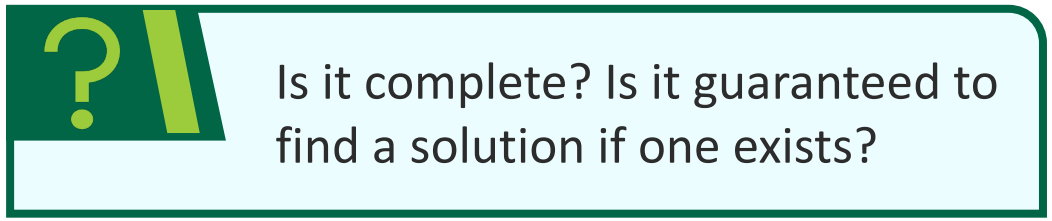


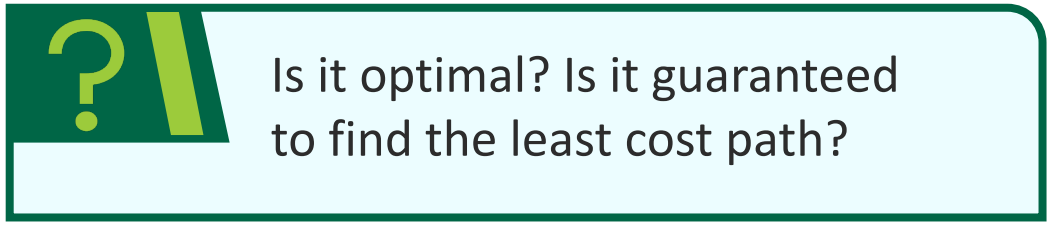
Figure 16. Search tree

A. Yes

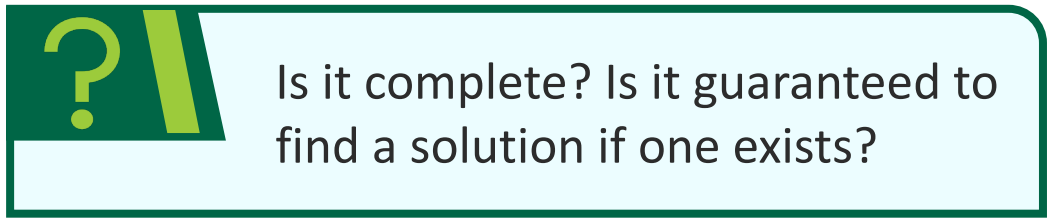
B. No

Greedy Best-First Search Analysis

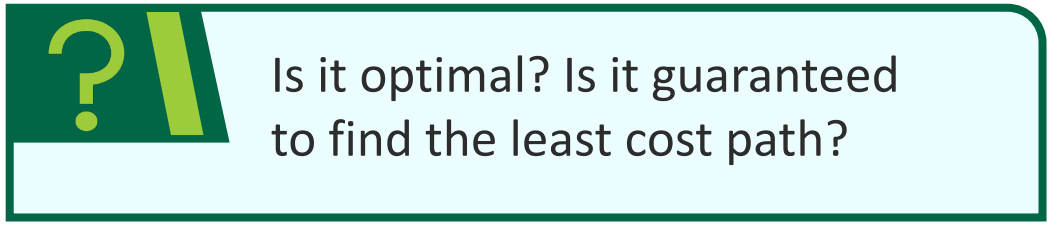




Greedy Best-First Search Analysis



- Complete in finite state spaces, but not infinite ones.



Greedy Best-First Search Analysis

Is it complete? Is it guaranteed to find a solution if one exists?

- Complete in finite state spaces, but not infinite ones.

Is it optimal? Is it guaranteed to find the least cost path?

- No. On each iteration it tries to get as close to a goal as it can, but greediness can lead to worse results than being careful.

What is its time complexity?

What is its space complexity?

Greedy Best-First Search Analysis

Is it complete? Is it guaranteed to find a solution if one exists?

- Complete in finite state spaces, but not infinite ones.

Is it optimal? Is it guaranteed to find the least cost path?

- No. On each iteration it tries to get as close to a goal as it can, but greediness can lead to worse results than being careful.

What is its time complexity?

- Can go through the entire search tree, so $O(b^m)$

What is its space complexity?

Greedy Best-First Search Analysis

Is it complete? Is it guaranteed to find a solution if one exists?

- Complete in finite state spaces, but not infinite ones.

Is it optimal? Is it guaranteed to find the least cost path?

- No. On each iteration it tries to get as close to a goal as it can, but greediness can lead to worse results than being careful.

What is its time complexity?

- Can go through the entire search tree, so $O(b^m)$

What is its space complexity?

- Equivalent to a BFS, so $O(b^m)$

You have reached the end of the lecture.

Image/Figure References

Figure 1. Map of Central & South Florida. Retrieved from: maps.google.com
Figure 2. A simplified road map of part of Romania, with road distances in miles. Retrieved from: Russel & Norvig, Artificial Intelligence: A Modern Approach, 4th edition, Pearson, 2021.
Figure 3. Heuristic function: straight-line distance to BucharestFigure 4-13. Search tree..
Figure 14. A simplified road map of part of Romania, with road distances in miles. Retrieved from: Russel & Norvig, Artificial Intelligence: A Modern Approach, 4th edition, Pearson, 2021.
Figure 15. Heuristic function: straight-line distance to BucharestFigure 16. Search tree. Other images were purchased from Getty Images and with permission to use.