# Day 24

GREEDY TECHNIQUE

## Greedy Technique

Constructs solution to an *optimization problem* through a sequence of choices that are:

- *feasible* (valid for the problem)
- *locally optimal* (most attractive choice among all currently feasible)
- *irrevocable* (cannot be undone later)

- Success relies on the problem having
  - *Principle of optimal substructure*: optimal solution is composed of optimal solutions to subinstances
  - *Greedy property*: most attractive local choice is always a part of an optimal solution

# Applications of Greedy Strategy

For some problems, yields an optimal solution for every instance:

- change making for "normal" coin denominations
- minimum spanning tree (MST)
- single-source shortest paths
- simple scheduling problems
- Huffman codes

For most, does not, but can be useful for fast approximations:

- traveling salesperson problem (TSP)
- knapsack problem
- other combinatorial optimization problems

# Change-Making Problem

Given unlimited amounts of coins of denominations

$$d_1 > d_2 > \cdots > d_m,$$

give change for amount $n$ with the minimum possible number of coins

Examples. Does the greedy strategy work?

1. Denominations $25¢$, $10¢$, $5¢$, $1¢$. Give change for $n = 48¢$
2. Denominations $25¢$, $10¢$, $1¢$. Give change for $n = 30¢$

Change-making problem has the greedy property for some ("normal"?) sets of denominations, but not all.

# Best-First Search in Detail

Consider the following map:

It displays distances between cities in Romania.

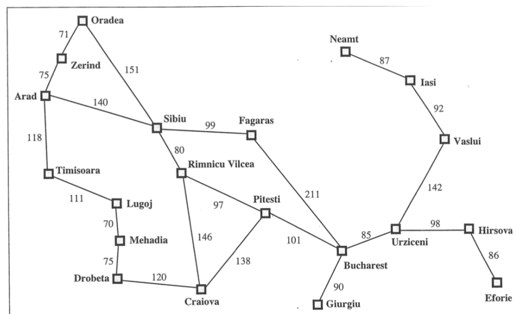



**Figure 3.2** A simplified road map of part of Romania.

Source of figure: Russel and Norvig, AIMA, 3rd edition, p 68

# Best-First Search in Detail

Here is a table with the straight-line distances between the cities on the map and Bucharest.

Suppose we wish to travel from Arad to Bucharest.

In best-first search, we will only use the information on the right.

| | | | |
|---|---|---|---|
| **Arad** | 366 | **Mehadia** | 241 |
| **Bucharest** | 0 | **Neamt** | 234 |
| **Craiova** | 160 | **Oradea** | 380 |
| **Drobeta** | 242 | **Pitesti** | 100 |
| **Eforie** | 161 | **Rimnicu Vilcea** | 193 |
| **Fagaras** | 176 | **Sibiu** | 253 |
| **Giurgiu** | 77 | **Timisoara** | 329 |
| **Hirsova** | 151 | **Urziceni** | 80 |
| **Iasi** | 226 | **Vaslui** | 199 |
| **Lugoj** | 244 | **Zerind** | 374 |

**Figure 3.22** Values of $h_{SLD}$—straight-line distances to Bucharest.

Source of figure: Russel and Norvig, AIMA, 3rd edition, p 93

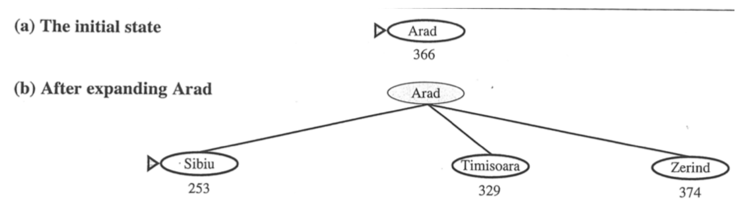
# Best-First Search in Detail

We first look at Arad.

We expand the Arad node by adding its neighbors as children.

| | | | |
|---|---|---|---|
| **Arad** | 366 | **Mehadia** | 241 |
| **Bucharest** | 0 | **Neamt** | 234 |
| **Craiova** | 160 | **Oradea** | 380 |
| **Drobeta** | 242 | **Pitesti** | 100 |
| **Eforie** | 161 | **Rimnicu Vilcea** | 193 |
| **Fagaras** | 176 | **Sibiu** | 253 |
| **Giurgiu** | 77 | **Timisoara** | 329 |
| **Hirsova** | 151 | **Urziceni** | 80 |
| **Iasi** | 226 | **Vaslui** | 199 |
| **Lugoj** | 244 | **Zerind** | 374 |

**Figure 3.22** Values of $h_{SLD}$—straight-line distances to Bucharest.




Source of figure: Russel and Norvig, AIMA, 3rd edition, p 94

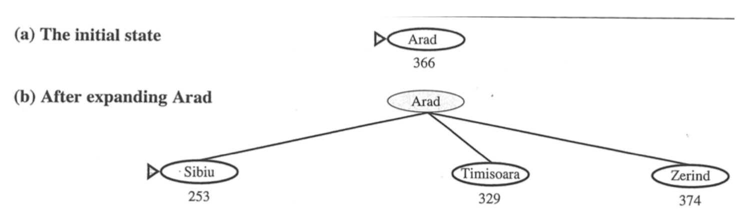
# Best-First Search in Detail

We now pick the bode with the least straight-line distance.

This is Sibiu

| | | | |
|---|---|---|---|
| **Arad** | 366 | **Mehadia** | 241 |
| **Bucharest** | 0 | **Neamt** | 234 |
| **Craiova** | 160 | **Oradea** | 380 |
| **Drobeta** | 242 | **Pitesti** | 100 |
| **Eforie** | 161 | **Rimnicu Vilcea** | 193 |
| **Fagaras** | 176 | **Sibiu** | 253 |
| **Giurgiu** | 77 | **Timisoara** | 329 |
| **Hirsova** | 151 | **Urziceni** | 80 |
| **Iasi** | 226 | **Vaslui** | 199 |
| **Lugoj** | 244 | **Zerind** | 374 |

**Figure 3.22** Values of $h_{SLD}$—straight-line distances to Bucharest.




Source of figure: Russel and Norvig, AIMA, 3rd edition, p 94

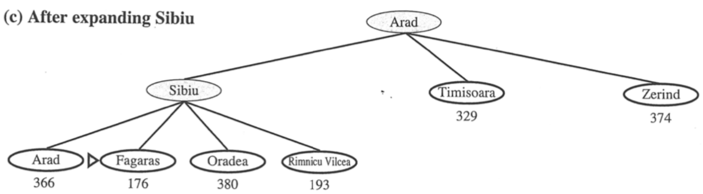
# Best-First Search in Detail

After expanding Sibiu, we will pick Fagaras next, because it has the lowest value.

| | | | |
|---|---|---|---|
| **Arad** | 366 | **Mehadia** | 241 |
| **Bucharest** | 0 | **Neamt** | 234 |
| **Craiova** | 160 | **Oradea** | 380 |
| **Drobeta** | 242 | **Pitesti** | 100 |
| **Eforie** | 161 | **Rimnicu Vilcea** | 193 |
| **Fagaras** | 176 | **Sibiu** | 253 |
| **Giurgiu** | 77 | **Timisoara** | 329 |
| **Hirsova** | 151 | **Urziceni** | 80 |
| **Iasi** | 226 | **Vaslui** | 199 |
| **Lugoj** | 244 | **Zerind** | 374 |

**Figure 3.22** Values of $h_{SLD}$—straight-line distances to Bucharest.

**(c) After expanding Sibiu**




Source of figure: Russel and Norvig, AIMA, 3rd edition, p 94

# Best-First Search in Detail

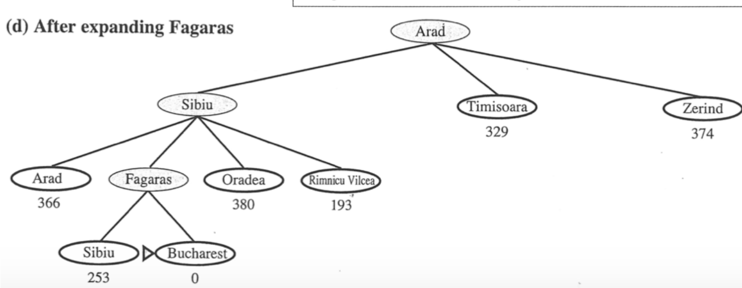
We now see Bucharest as one of the children.

It is the goal, so the algorithm terminates.

| | | | |
|---|---|---|---|
| **Arad** | 366 | **Mehadia** | 241 |
| **Bucharest** | 0 | **Neamt** | 234 |
| **Craiova** | 160 | **Oradea** | 380 |
| **Drobeta** | 242 | **Pitesti** | 100 |
| **Eforie** | 161 | **Rimnicu Vilcea** | 193 |
| **Fagaras** | 176 | **Sibiu** | 253 |
| **Giurgiu** | 77 | **Timisoara** | 329 |
| **Hirsova** | 151 | **Urziceni** | 80 |
| **Iasi** | 226 | **Vaslui** | 199 |
| **Lugoj** | 244 | **Zerind** | 374 |

**Figure 3.22** Values of $h_{SLD}$—straight-line distances to Bucharest.

**(d) After expanding Fagaras**




Source of figure: Russel and Norvig, AIMA, 3rd edition, p 94

# Best-First Search in Detail

As you can tell from the map, the path found, i.e. from Arad, through Sibiu, Fagaras to Bucharest is not the shortest path.

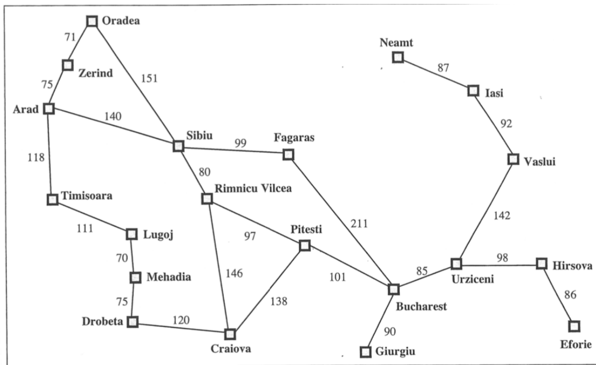



**Figure 3.2** A simplified road map of part of Romania.

Source of figure: Russel and Norvig, AIMA, 3rd edition, p 94

# Greedy Algorithms

As a rule, greedy algorithms are intuitive and simple.

It is usually easy to figure out how to proceed in a greedy manner.

More difficult: proving that a greedy algorithm yields an optimal solution when it does.

Techniques:

1. Use **mathematical induction** to show a partially constructed greedy solution on each iteration can be extended to an optimal solution
2. Argue that **no other algorithm could do better** than greedy algorithm's choice **at every step**
3. Argue that the **result obtained** by the greedy algorithm is **optimal** based on the output
4. Proof by **contradiction**: consider an arbitrary optimal solution, and show that if it is better than the greedy solution, this leads to a contradiction

# Proof by method (2)

Consider a 10 x 10 chess board.

What is the minimum number of moves in which a knight can traverse from one corner to the diagonally opposed corner.

Greedy algorithm?

Jump as close to goal as possible on each move.

How many moves?

6

Solve 1 + 3k = 10, then k*2

Each move decreases the Manhattan distance to goal by 3

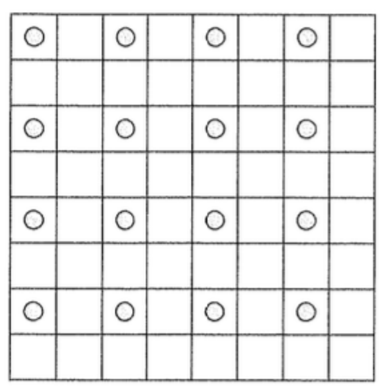
Cannot do better with a knight.

# Proof by method (3)

Consider an 8 x 8 chess board.

Place chips so that no two chips are placed:

- On the same square or
- Adjacent
  - Horizontally
  - Vertically
  - diagonally



# Proof by method (3)

Partition board into 16 4x4 squares

Impossible to place more than one chip into each square while satisfying constraints