Princess Nora University
Faculty of Computer & Information Systems

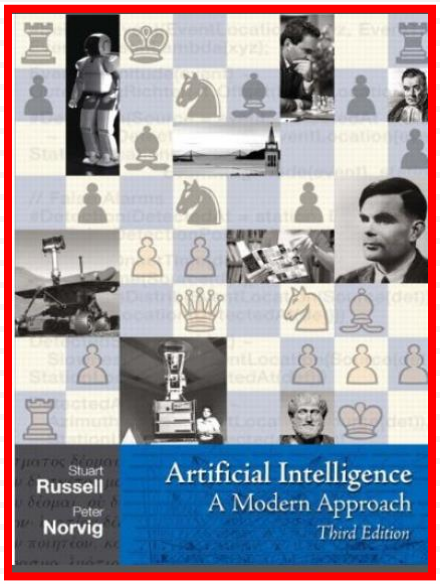
جامعة الأميرة نورة بنت عبد الرحمن
Princess Nora Bint Abdul Rahman University

ARTIFICIAL INTELLIGENCE

(CS 370D)

Computer Science
Department

جامعة الأميرة نورة بنت عبد الرحمن
Princess Nora Bint Abdul Rahman University



College of Computer and Information Sciences

(CHAPTER-3-PART3)

PROBLEM SOLVING AND SEARCH

Searching algorithm

Uninformed Search Algorithms(Blind Search)

3.1 Breadth first Search
3.2 Depth First Search
3.3 Depth limited Search
3.4 Iterative Deeping Search
3. 5 Bidirectional Search

Informed Search (Heuristic Search)

Best First Search
Greedy Search
Perfect Information Search
A* Search
Iterative Deepening A* Search
A* with PathMax

Dr. Abeer Mahmoud
(course coordinator)

informed Search Algorithms (Heuristic Search)

1. Best First Search
2. Greedy Search
3. Perfect Information Search
4. A* Search
5. Iterative Deepening A* Search
6. A* with PathMax

Dr. Abeer Mahmoud
(course coordinator)

Uninformed versus Informed

Uninformed search	Informed search
➢ does not have any additional information on the **quality** of states. ➢ So, it is impossible to determine **which state is the better than others**. As a result, search efficiency depends only on the structure of a state space	➢ heuristically informed search uses a certain kind of information about states in order to guide search along **promising** branches within a state space. ➢ Using problem specific knowledge as hints to guide the search.

Dr. Abeer Mahmoud
(course coordinator)

Uninformed versus Informed (cont)

Uninformed search	Informed search
➢ look for solutions by **systematically** generating new states and checking each of them against the goal. 1. It is very **inefficient** in most cases. 2. Most successor states are "obviously" a bad choice. 3. Such strategies do not use problem-specific knowledge	1. They are almost always **more efficient** than uninformed strategies. 2. May reduce time and space complexities. 3. Evaluation function f(n) measures distance to the goal. 4. Order nodes in **Frontier** according to f(n) and decide which node to expand next.

Informed search & Exploration

- *Modified version from blind search algorithm*

    1. *Greedy best first search*
    2. *A* and its relatives*

- *The family of local search includes methods*

    1. inspired by statistical physics [simulated annealing]
    2. evolutionary biology [genetic algorithms]
    3. online search [in which agent is faced with state space that is completely unknown]

Dr. Abeer Mahmoud
(course coordinator)

Informed search & Exploration
Best first search

Main idea: use an **evaluation function f(n)** for each node

Implementation:

- Order the nodes in **Frontier** in decreasing order of desirability (from low f(n) which means high desirability to high f(n) which means low desirability.)

- There is a whole family of best-first search strategies, each with a different evaluation function.

Special cases:

- Greedy best-first search.
- A* search.

Dr. Abeer Mahmoud
(course coordinator)

Best-first search Algorithm

1-greedy best first search

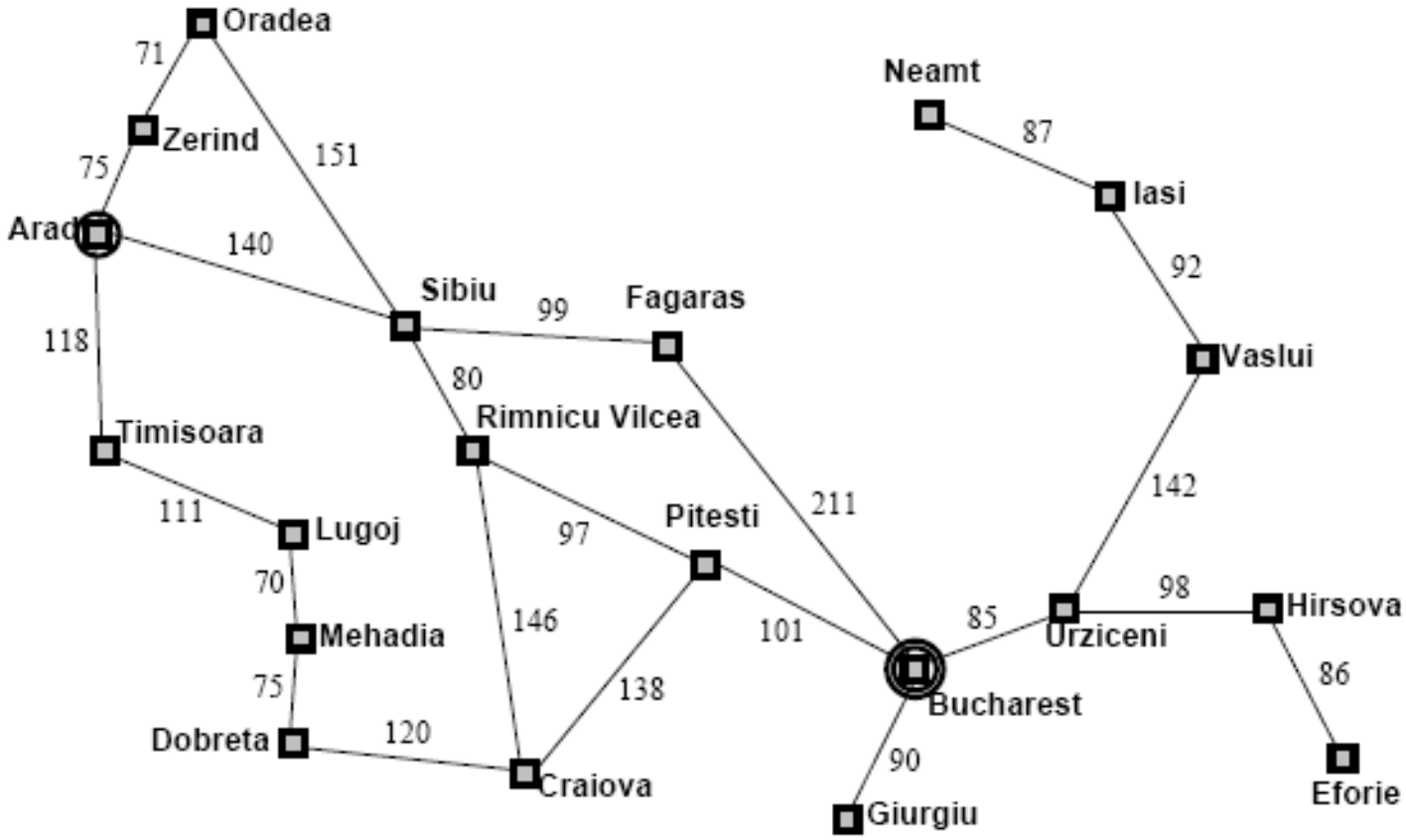
- Tries to expand the node that is closest to the goal
  - Use straight line distance ex : $h_{SLD}(IN(Arad))=366$

[note that the values of **fn h_{SLD}** cannot be computed from the problem description itself]

Dr. Abeer Mahmoud
(course coordinator)

Greedy Search Example 1

Straight line distances between cities which are additionally provided

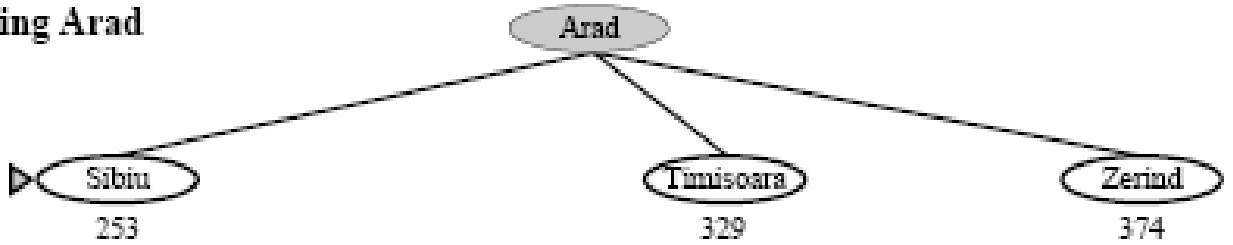


Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

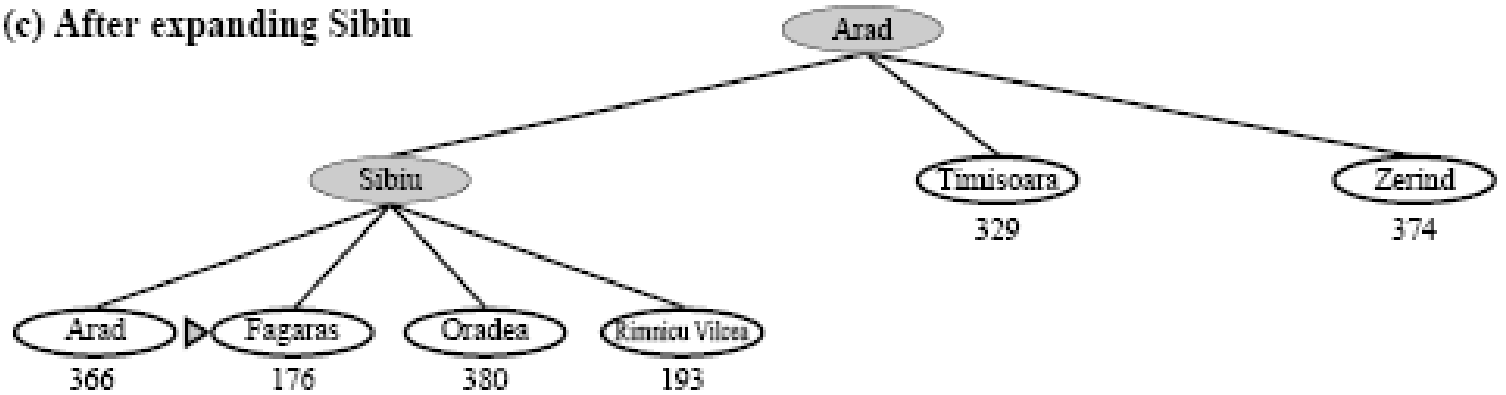
Dr. Abeer Mahmoud
(course coordinator)

(a) The initial state

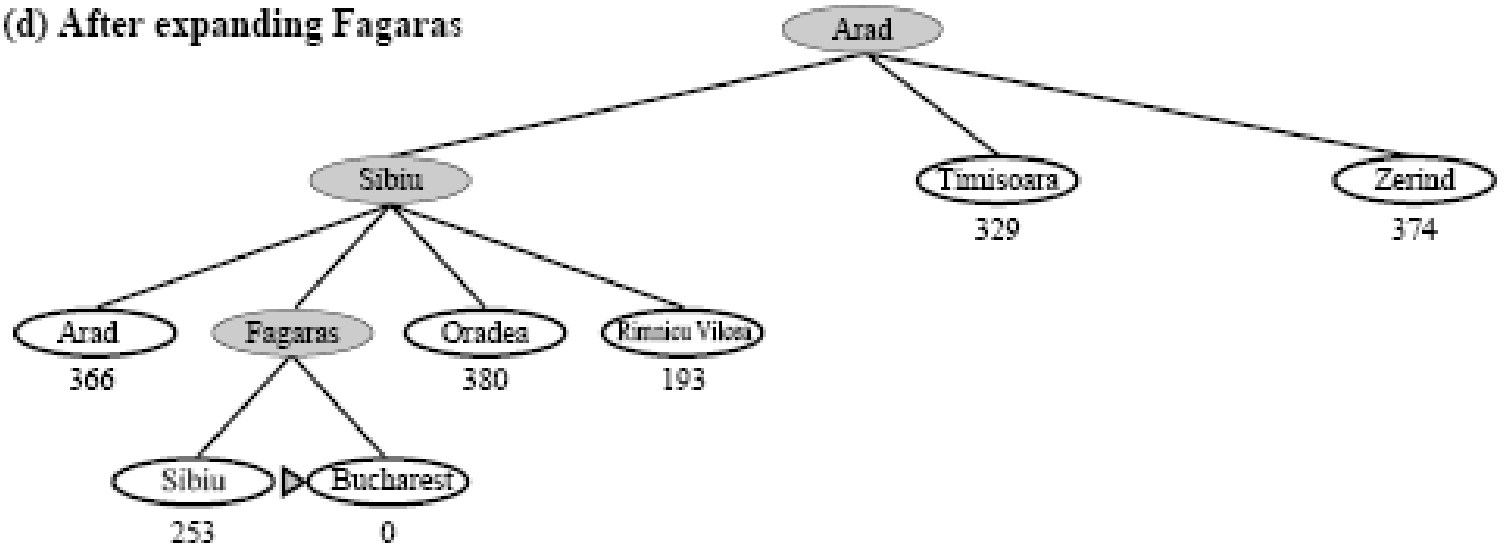
(b) After expanding Arad



(c) After expanding Sibiu



(d) After expanding Fagaras



moud
ıator)

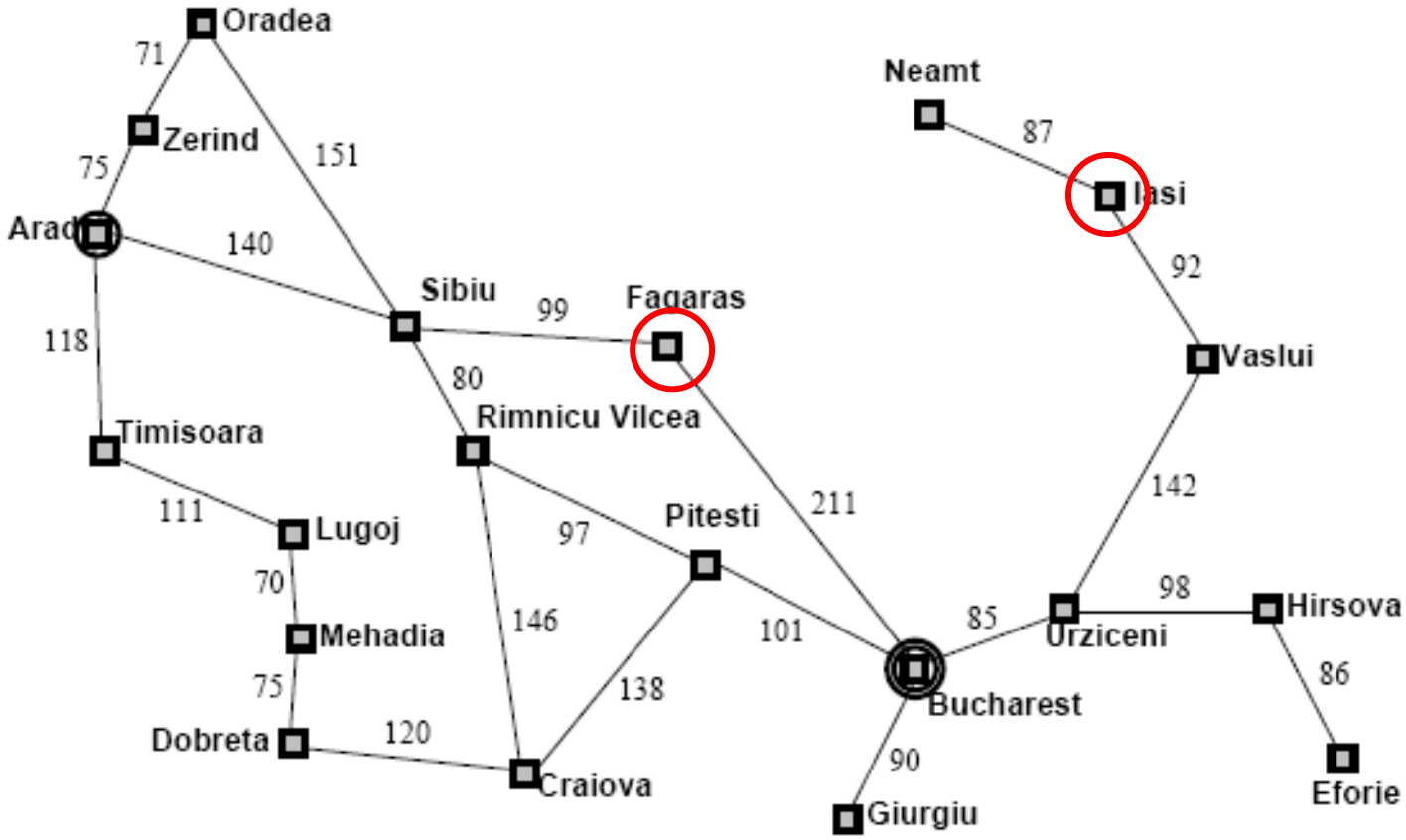
The greedy best first search using h_{SLD} finds a solution without ever expanding a node that is not on solution path, hence its cost is minimal

This show why the algorithm is called greedy [at each step it tries to get as close to goal as it can]

Dr. Abeer Mahmoud
(course coordinator)

Greedy Search Example 2

Straight line distances between cities which are additionally provided



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

Dr. Abeer Mahmoud
(course coordinator)

- Consider the problem of getting from Iasi to Fagras
- The heuristic suggests that Neamt be expanded first because it is closest to Fagaras but it is like dead end
- The solution is to go first to Vaslui a step that is actually farther from the goal according to the heuristic & then continue to Urzicent, Bucharest and Fagaras.
- In this case , then heuristic causes unnecessary needs to be expanded

Greedy best first search

- *Resembles depth first search in the way it prefers to follow a single path all the way to goal but it will back up when it hits a dead end*
- *It is not optimal (greedy) and incomplete (because of backtracking)*

Best-first search Algorithm

2-A* best first search

- ***Main idea***: *avoid expanding paths that are already expensive.*
- *Minimizing the total estimated solution cost*
- *It evaluate a node by*

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

Cost so far to reach n

Estimated Cost to get from n to goal

- Path cost is g and heuristic function is h
  - f(state) = g(state) + h(state)
  - Choose smallest overall path cost (known + estimate)

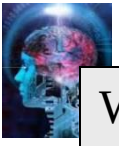
Dr. Abeer Mahmoud
(course coordinator)

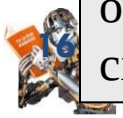
A* Search Example

Distances to Bucharest

Town	SLD
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244

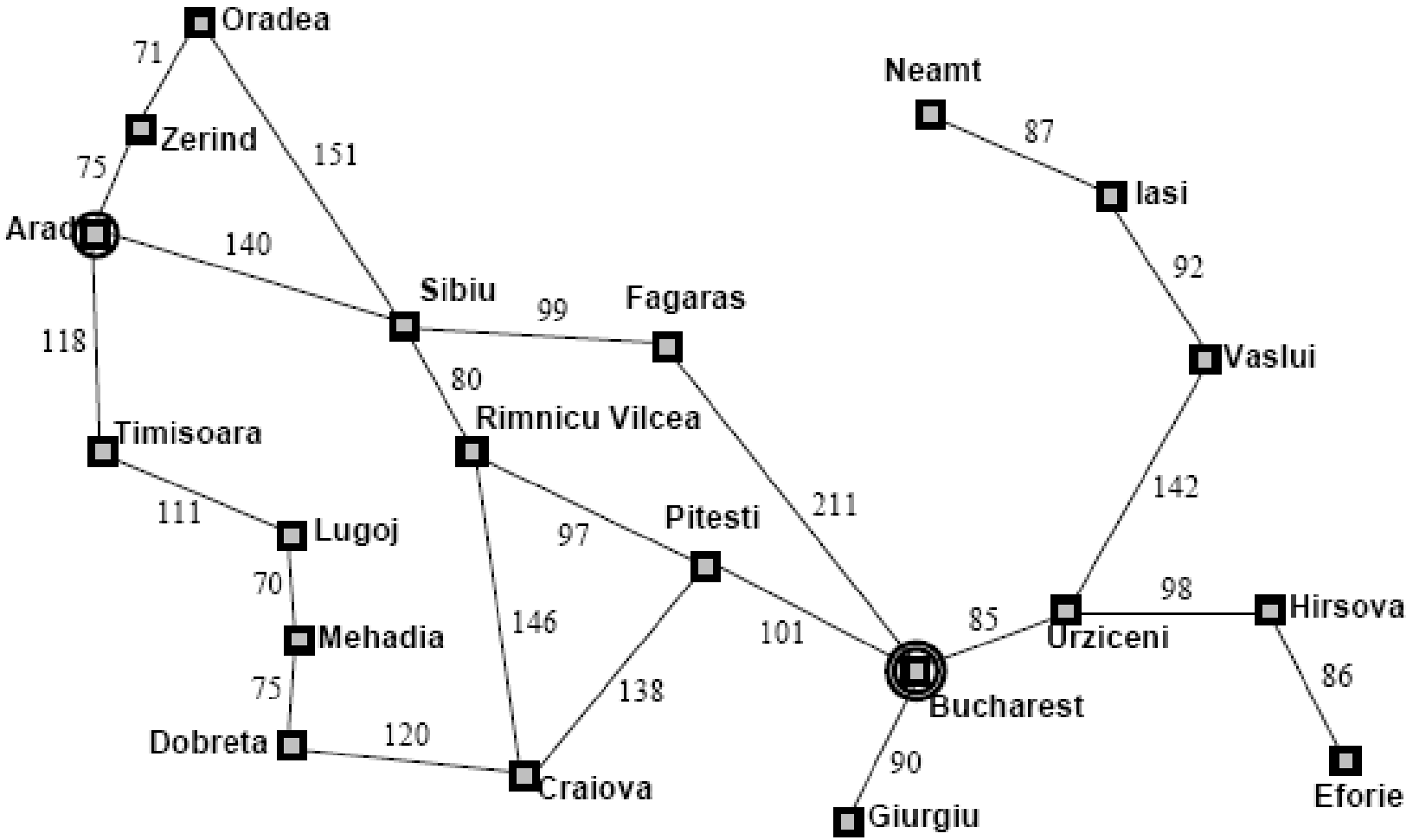
Town	SLD
Mehadai	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374





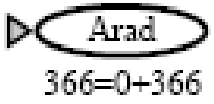
We can use straight line distances as an admissible heuristic as they will never overestimate the cost to the goal. This is because there is no shorter distance between two cities than the straight line distance. Press space to continue with the slideshow.

A* Search Example

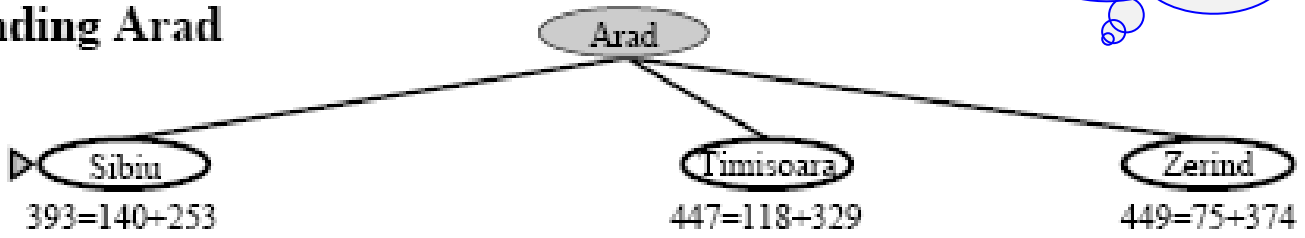


A* Search Example

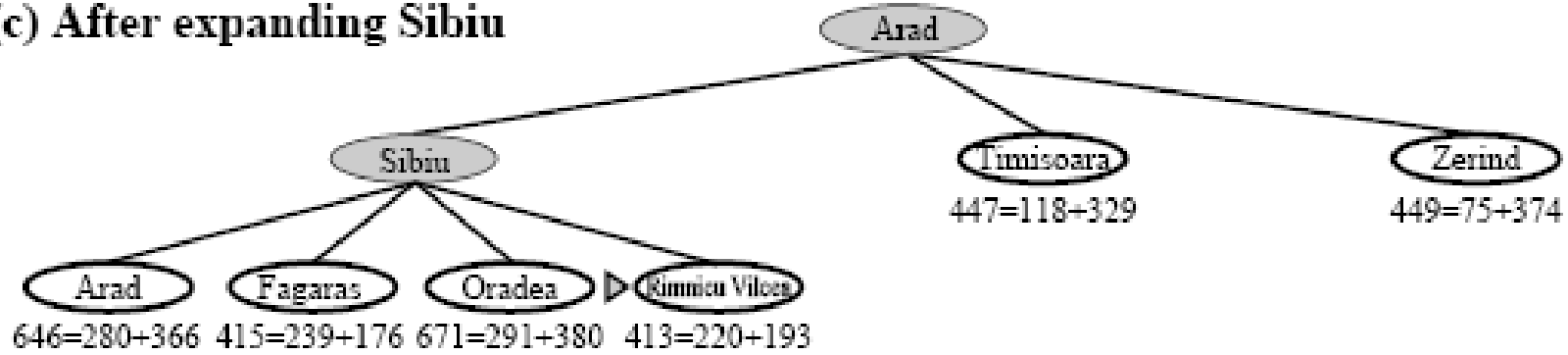
(a) The initial state



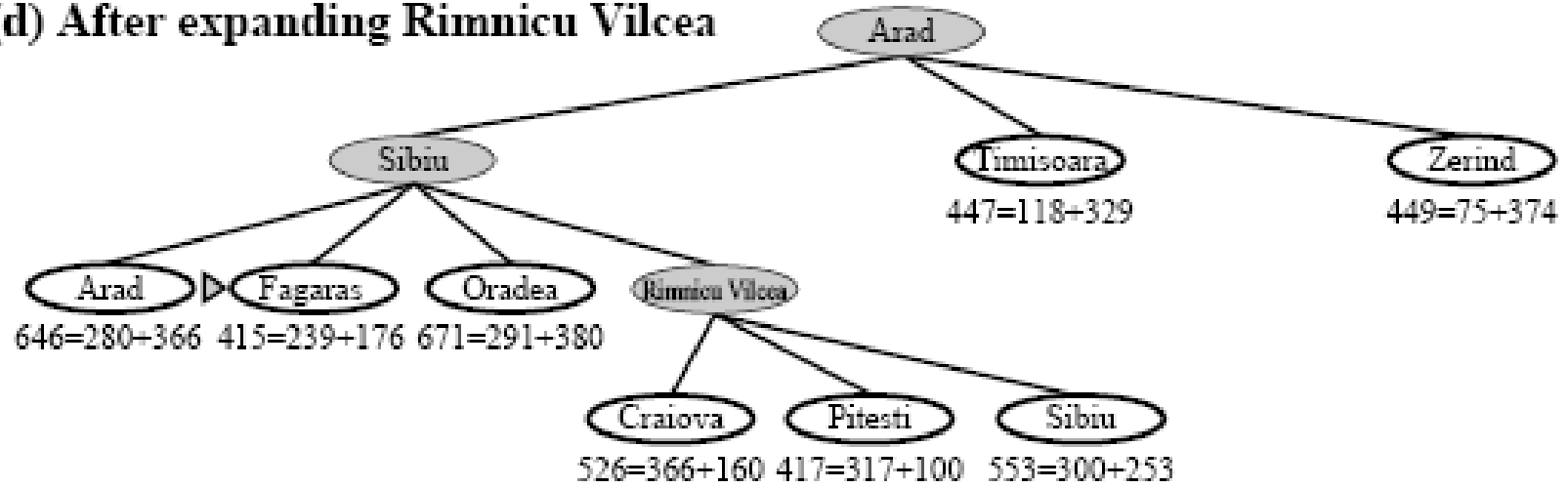
(b) After expanding Arad



(c) After expanding Sibiu

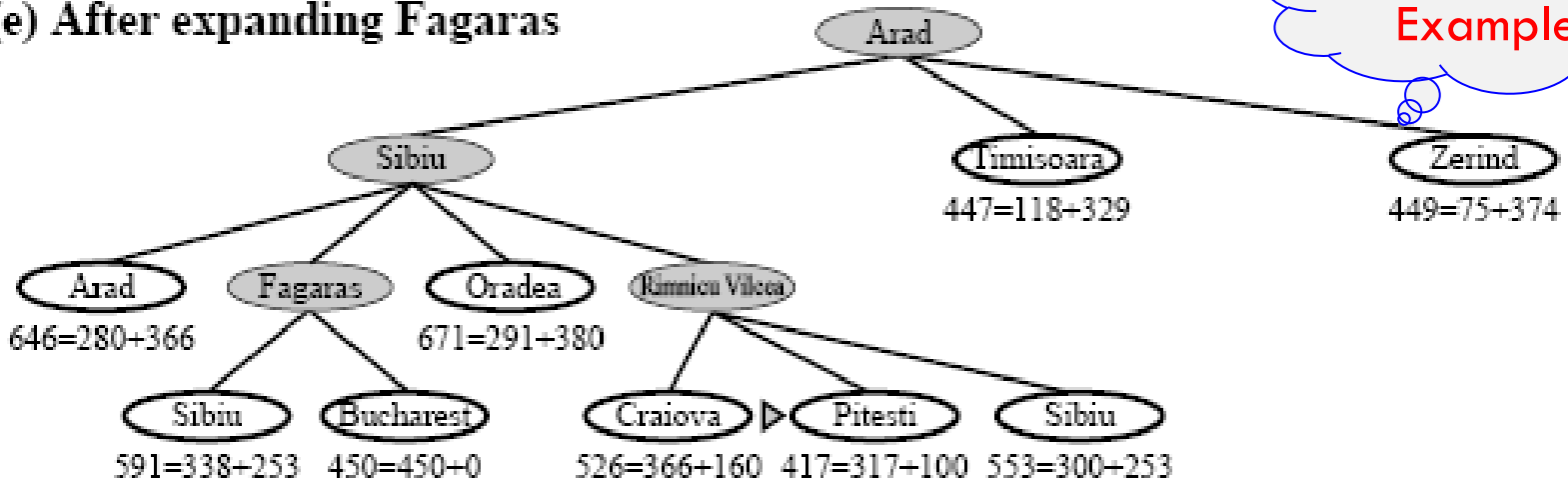


(d) After expanding Rimnicu Vilcea

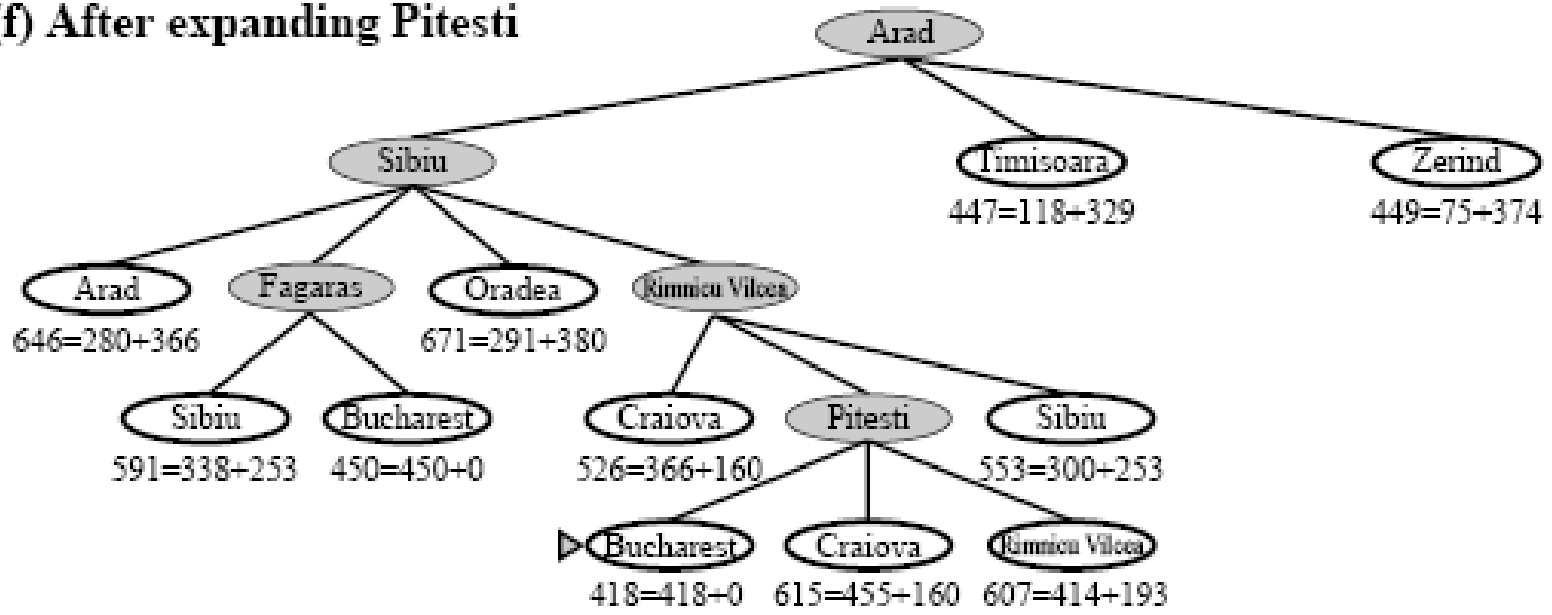


A* Search Example

(e) After expanding Fagaras



(f) After expanding Pitesti



Id
(course coordinator)

Properties of A*

- Complete? Yes
- Time? Exponential
- Space? Keeps all nodes in memory
- Optimal? Yes

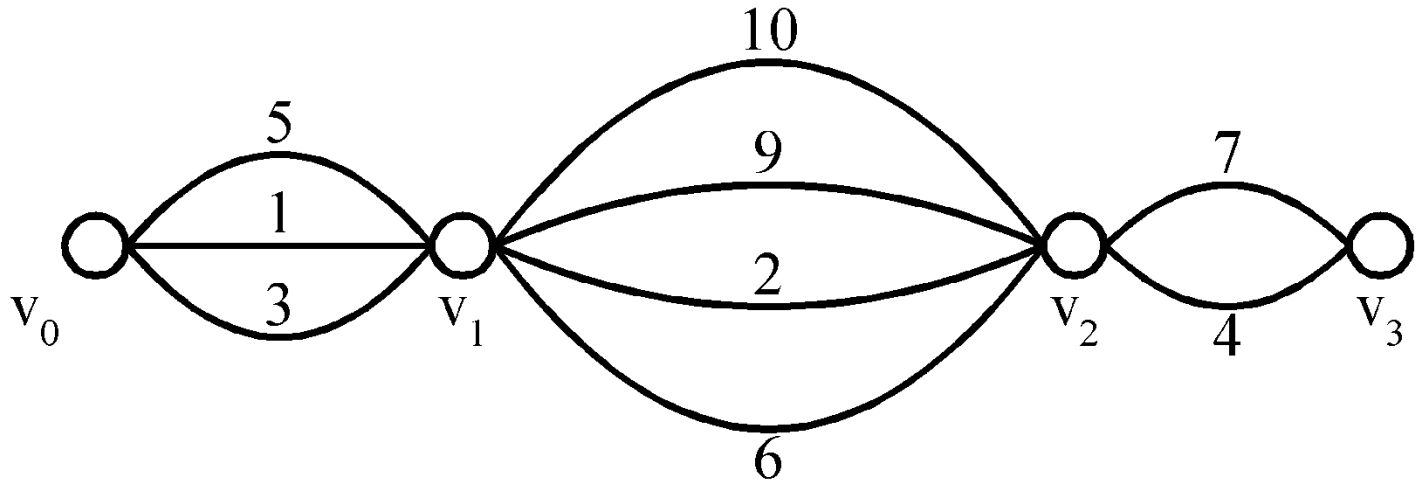
Dr. Abeer Mahmoud
(course coordinator)

Examples

Dr. Abeer Mahmoud
(course coordinator)

Shortest paths

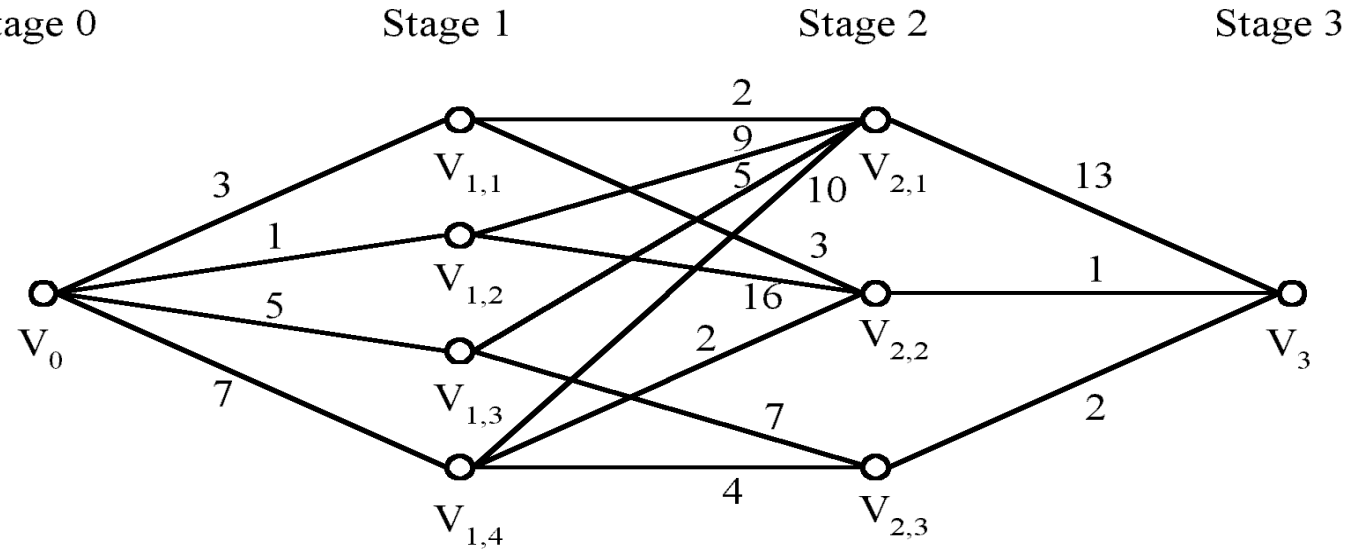
- Find a shortest path from v_0 to v_3???
- Can the **greedy** method solve this problem???
- The shortest path: 1 + 2 + 4 = 7.



Dr. Abeer Mahmoud
(course coordinator)

Shortest paths on a multi-stage graph

- Find a shortest path from v_0 to v_3 in the multi-stage graph.



- Greedy method: $v_0v_{1,2}v_{2,1}v_3 = 23$
- Optimal: $v_0v_{1,1}v_{2,2}v_3 = 7$
- The greedy method does not work.

Dr. Abeer Mahmoud
(course coordinator)

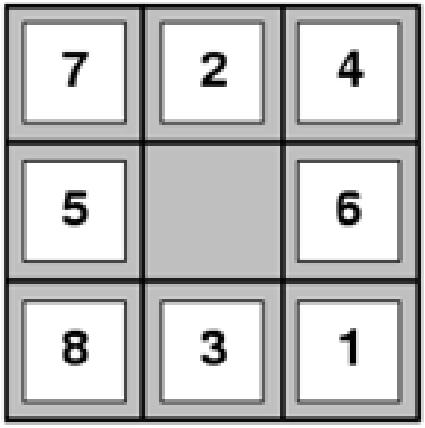
Admissible heuristics (accepted evaluation function)

E.g., for the 8-puzzle:

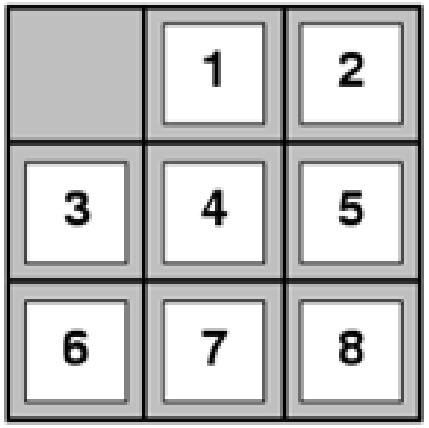
$h_1(n)$ = number of misplaced tiles[tiles in wrong places)

- $h_2(n)$ = total Manhattan distance [how many moves to reach right place]

(i.e., no. of squares from desired location of each tile)



Start state



Goal state

- $h_1(S) = ?$
- $h_2(S) = ?$

Dr. Abeer Mahmoud
(course coordinator)

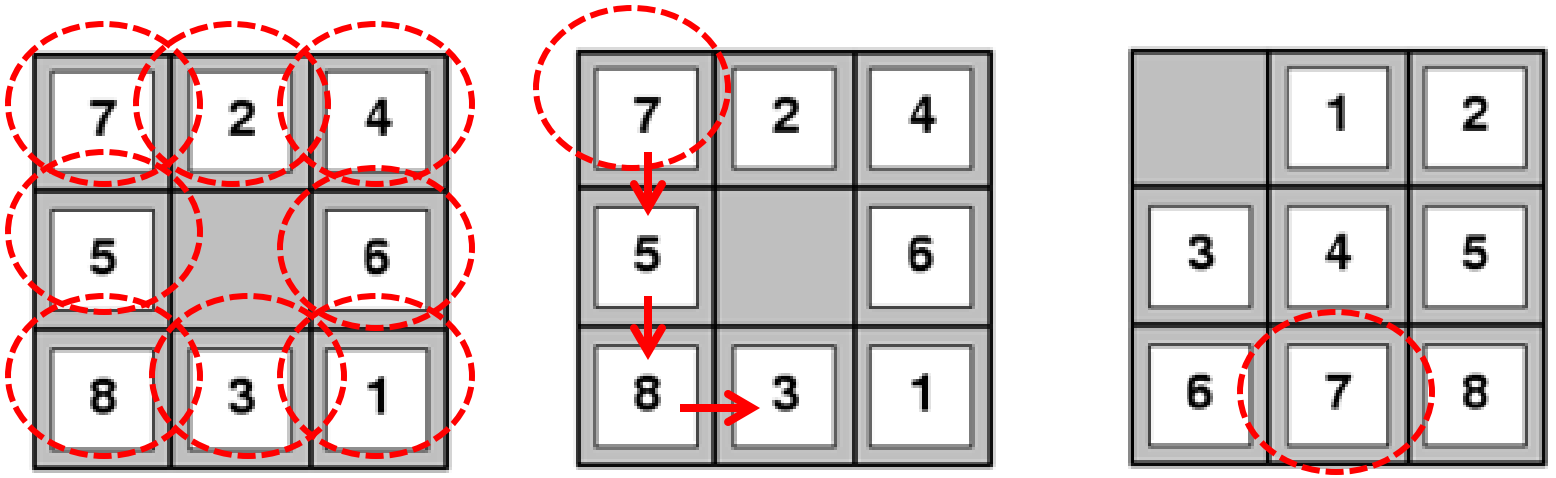
Admissible heuristics (accepted evaluation function)

E.g., for the 8-puzzle:

$h_1(n)$ = number of misplaced tiles[tiles in wrong places)

- $h_2(n)$ = total Manhattan distance [how many moves to reach right place]

(i.e., no. of squares from desired location of each tile)



- **$h_1(S) = 8$**
- $h_2(S) = ?$

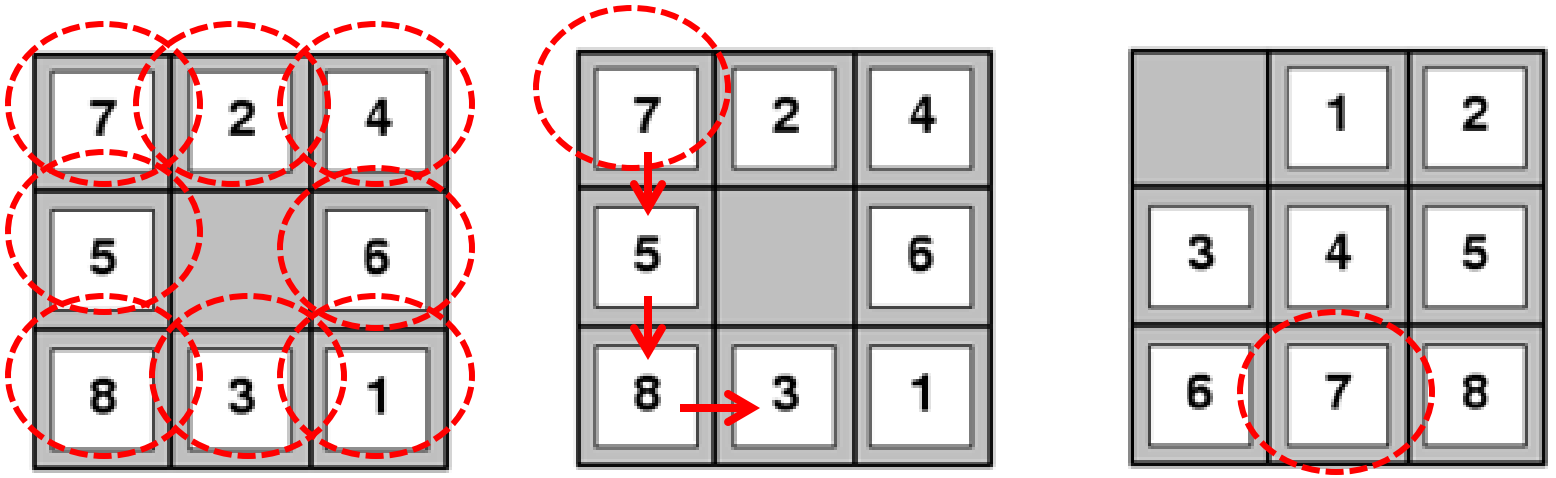
Dr. Abeer Mahmoud
(course coordinator)

Admissible heuristics (accepted evaluation function)

E.g., for the 8-puzzle:

- $h_1(n)$ = number of misplaced tiles[tiles in wrong places)
- $h_2(n)$ = total Manhattan distance [how many moves to reach right place]

(i.e., no. of squares from desired location of each tile)



- $h_1(S) = 8$
- $h_2(S) = 3+1+2+2+2+3+3+2 = 18$

Dr. Abeer Mahmoud
(course coordinator)

Dominance (which is better)

- If $h_2(n) \geq h_1(n)$ for all n (both admissible)
- then h_2 dominates h_1
- h_2 is better for search

- **Typical search costs (average number of nodes expanded):**

- *d=12* IDS = 3,644,035 nodes
  $A^*(h_1)$ = 227 nodes
  $A^*(h_2)$ = 73 nodes
- *d=24* IDS = too many nodes
  $A^*(h_1)$ = 39,135 nodes
  $A^*(h_2)$ = 1,641 nodes

Dr. Abeer Mahmoud
(course coordinator)

Thank you

End of Chapter 3-Part3

Dr. Abeer Mahmoud
(course coordinator)