Artificial Intelligence

For HEDSPI Project

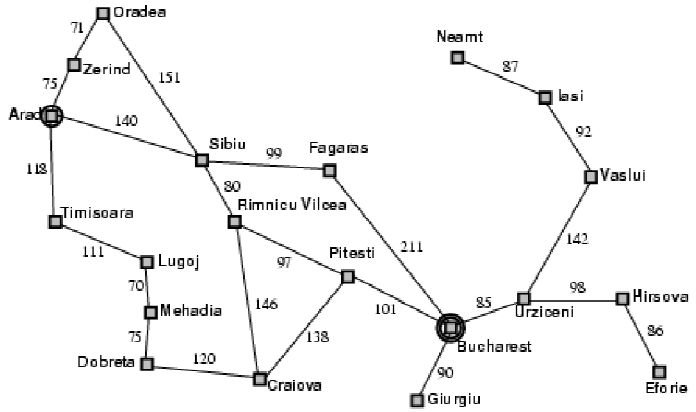
Lecturer 4 - Search

Lecturers :
Dr. Le Thanh Huong
Dr. Tran Duc Khanh
Dr. Hai V. Pham
School of ICT, HUST

Outline

- Graph search
- Best-first search
- A* search

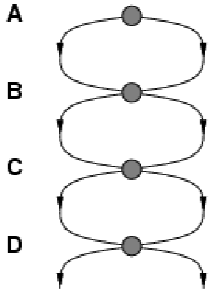
Graph search

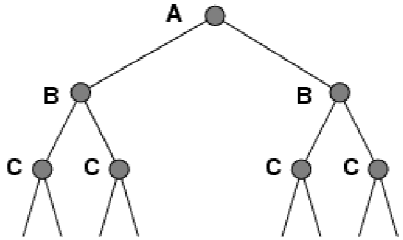


Get from Arad to Bucharest as quickly as possible

Graph search

- Failure to detect repeated states can turn a linear problem into an exponential one!





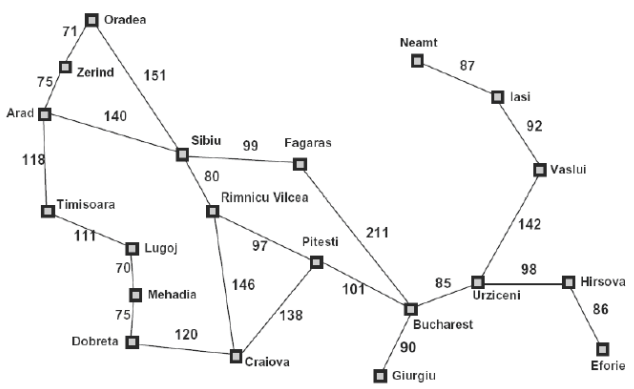
- Very simple fix: never expand a node twice

Graph search

```
function Graph-Search(problem, fringe) returns a solution, or failure
fringe ← Insert(Make-Node(Initial-State(problem)), fringe);
closed ← an empty set
while (fringe not empty)
   node ← RemoveFirst(fringe);
   if (Goal-Test(problem, State(node))) then return Solution(node);
   if (State(node) is not in closed then
      add State(node) to closed
      fringe ← InsertAll(Expand(node, problem), fringe);
   end if
end
return failure;
```

- Never expand a node twice!

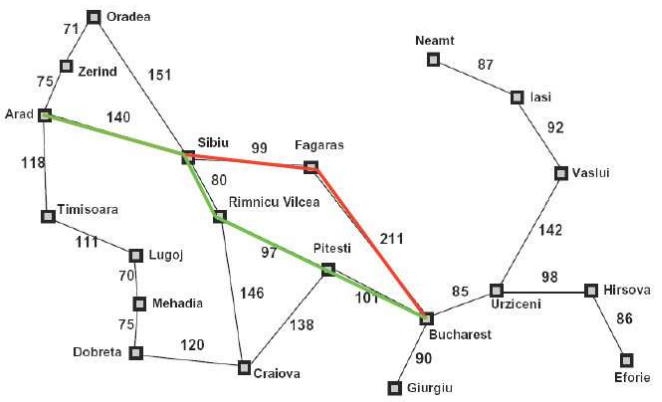
Straight Line Distances



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Best-first search

- Idea: use an evaluation function *f(n)* for each node
  - estimate of "desirability"
  - → Expand most desirable unexpanded node
- Order the nodes in fringe in decreasing order of desirability
- Special cases:
  - greedy best-first search
  - A* search



Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

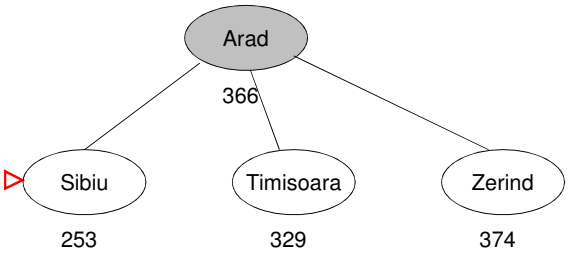
Greedy Best-First Search

- Evaluation function $f(n) = h(n)$ (heuristic)
  = estimate of cost from n to *goal*
- e.g., $h_{SLD}(n)$ = straight-line distance from n to Bucharest
- Greedy best-first search expands the node that appears to be closest to goal

Greedy best-first search example

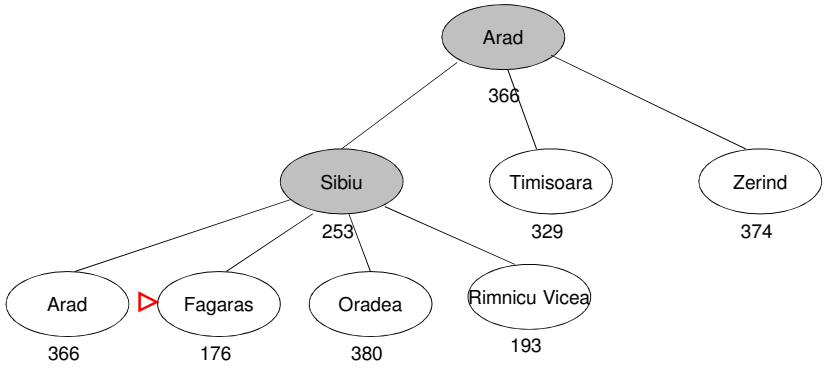
366

Greedy best-first search example



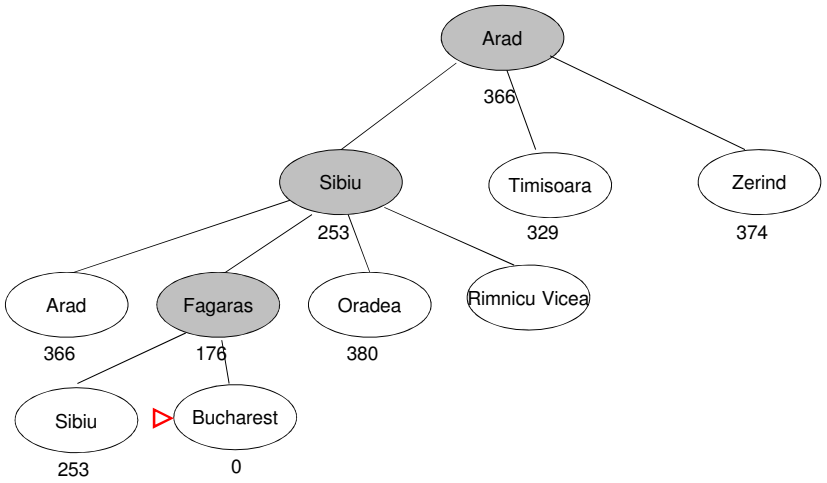
11

Greedy best-first search example



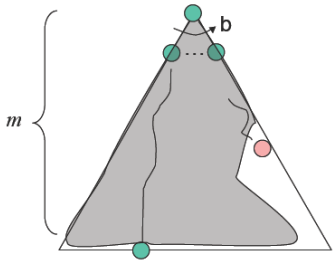
12

Greedy best-first search example



Greedy Best-First Search

- Complete? No – can get stuck in loops, e.g., Iasi → Neamt → Iasi → Neamt → …
- Time? $O(b^m)$, but a good heuristic can give dramatic improvement
- Space? $O(b^m)$ -- keeps all nodes in memory
- Optimal? No



- What do we need to do to make it complete?

⇒ A* search

- Can we make it optimal? → No

A* search

- Idea: Expand unexpanded node with lowest evaluation value
- Evaluation function *f(n) = g(n) + h(n)*
- *g(n)* = cost so far to reach *n*
- *h(n)* = estimated cost from *n* to goal
- *f(n)* = estimated total cost of path through *n* to goal
- Nodes are ordered according to f(n).

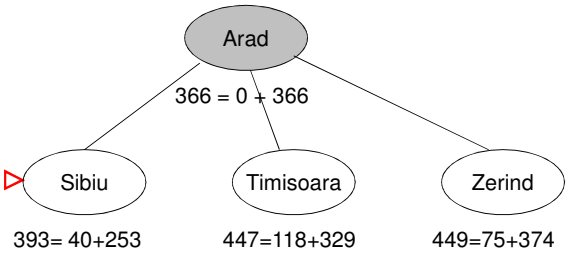
15

A* search example

366 = 0 + 366

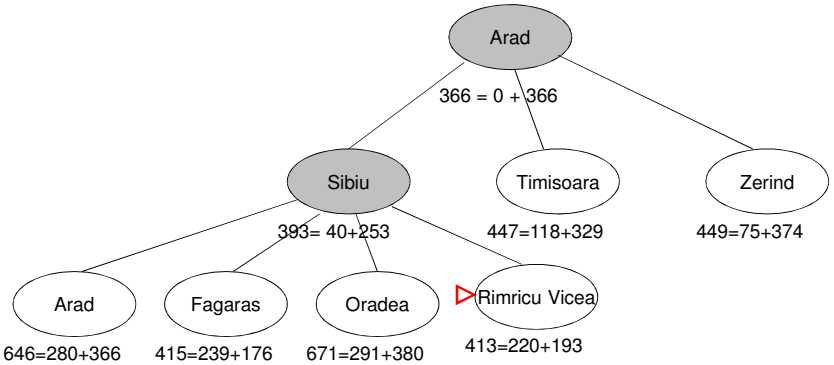
16

A* search example



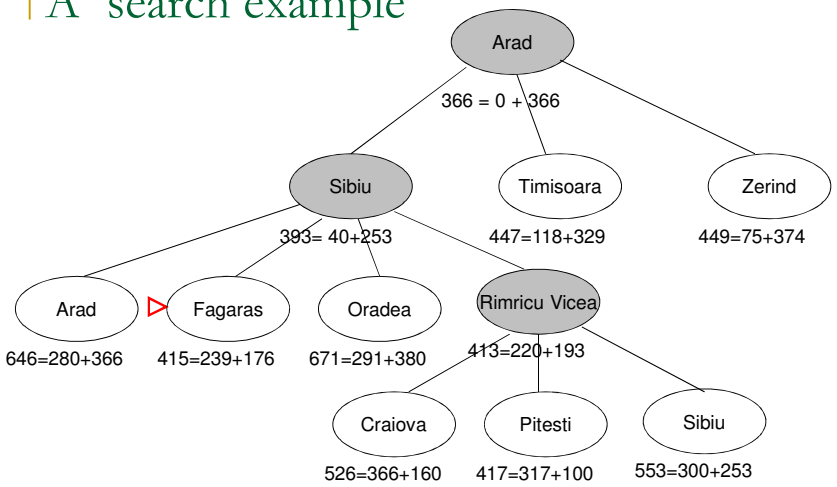
17

A* search example



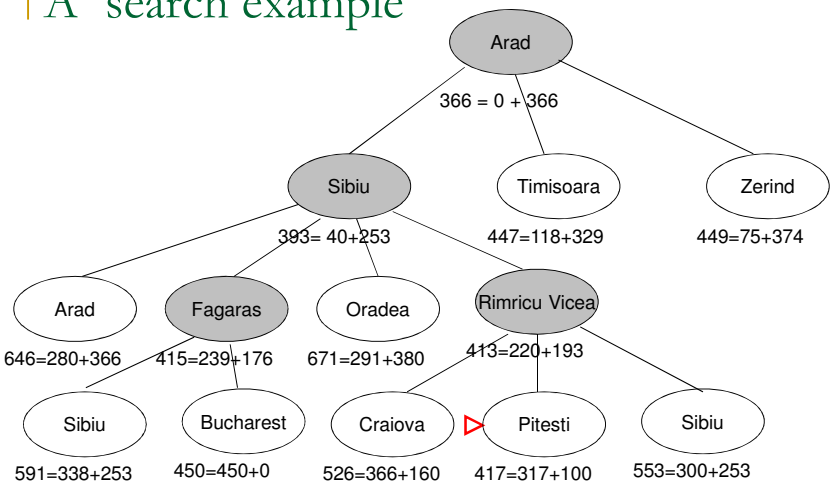
18

A* search example



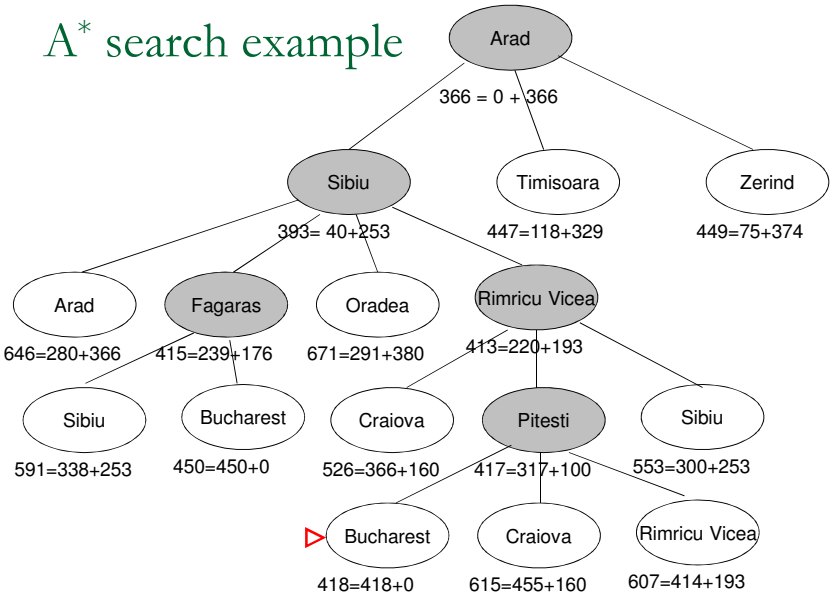
19

A* search example



20

A* search example



Can we Prove Anything?

- If the state space is finite and we avoid repeated states, the search is complete, but in general is not optimal
- If the state space is finite and we do not avoid repeated states, the search is in general not complete
- If the state space is infinite, the search is in general not complete

22

Admissible heuristic

- Let h*(N) be the **true** cost of the optimal path from N to a goal node
- Heuristic h(N) is admissible if:

$$0 \leq h(N) \leq h^*(N)$$

- An admissible heuristic is always optimistic

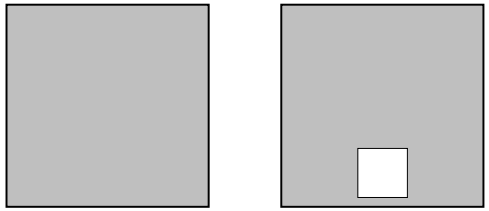
23

Admissible heuristics

The 8-puzzle:

- $h_1(n)$ = number of misplaced tiles
- $h_2(n)$ = total Manhattan distance

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



- $h_1(S)$ = ? 7
- $h_2(S)$ = ? 2+3+3+2+4+2+0+2 = 18

24

```
ERROR: rangecheck
OFFENDING COMMAND: get

STACK:

1
[[-168 -341 1093 960 ]]
0
/descender
[(Start State)]
-savelevel-
-savelevel-
```