UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Workbook:

Iterative Deepening A* (IDA*)

Albert Sanchis
Jorge Civera

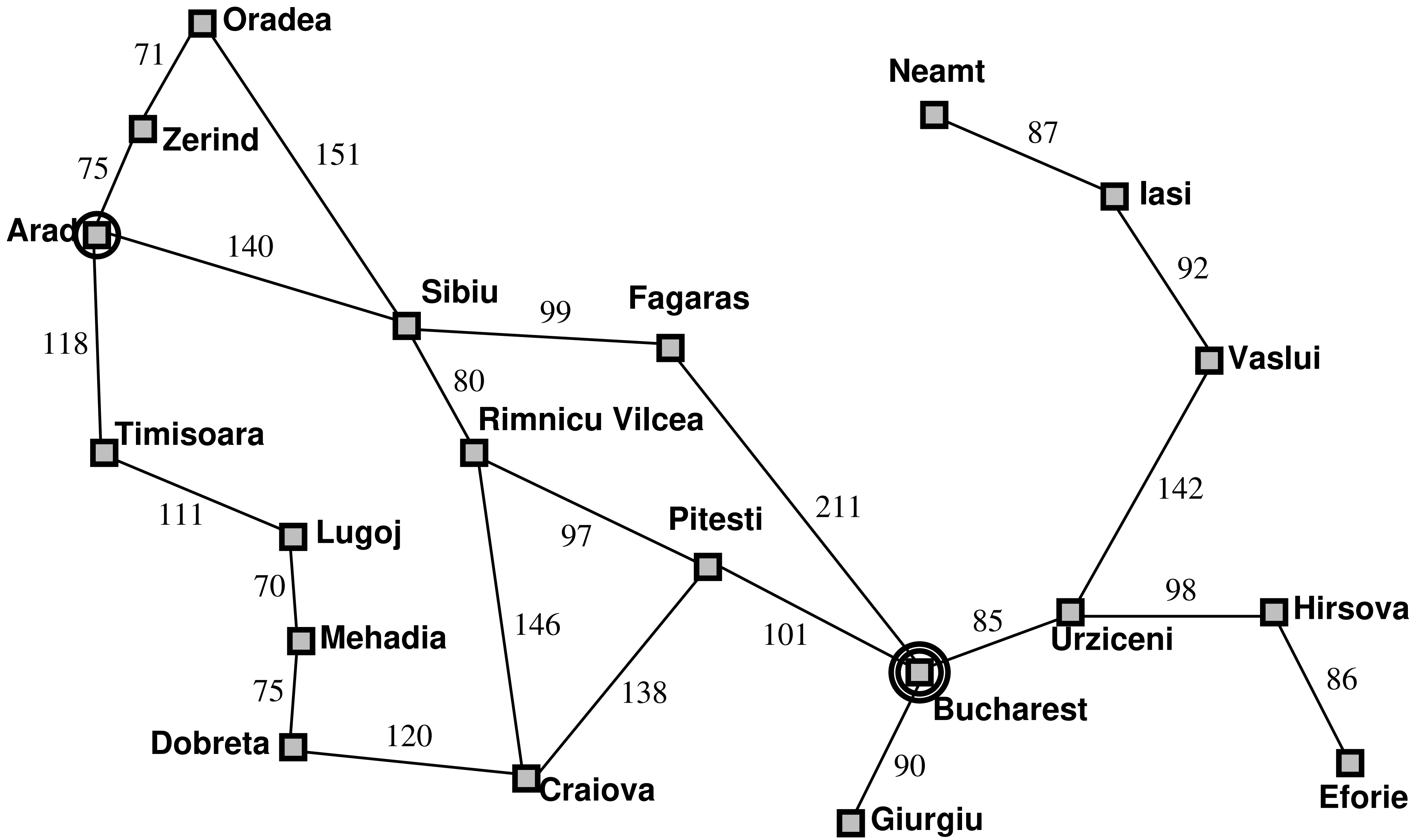
Departament de Sistemes Informàtics i Computació

Learning objectives

- To describe the *Iterative Deepening A** (IDA*) algorithm.
- To draw the tree of IDA* search.
- To apply IDA* search to a well-known problem.
- To analyze the quality of IDA* search.

Problem: Shortest path between two points

Shortest path from Arad to Bucarest [1]:



Actions(Arad) = {Move(Sibiu), Move(Timisoara), Move(Zerind)}.

Problem: Shortest path between two points

Straight-line distances to Bucharest:

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

1 The IDA* algorithm (main) [2]

IDA(G, s', h**)** // G weighed graph, s' start, h ***heuristic***

$P = InitStack(s')$ // Init ***Path*** with source node

$b = h(s')$ // Init bound with $f_{s'} = h(s')$

while True:

$(nextb, r) =$ **BT**(G, P, h, b)//BT returns next bound and goal state

if $r \neq$ NULL: **return** P // if solution, return ***Path*** to goal

if $nextb = \infty$: **return** NULL // no children to compute next bound

$b = nextb$ // bound updated for next iteration

The IDA* algorithm (backtracking) [2]

```
BT(G, P, h, b)                        // G weighed graph, Path P, h, bound b
  s = Top(P)                          // Path: extract node from stack
  f_s = g_s + h(s)                    // f value of node being explored
  if f_s > b: return (f_s,NULL)       // b exceeded return to compute nextb
  if Goal(s): return (f_s, s)         // solution found!
  min = ∞                             // children's minimum f value
  n = FirstAdjacent(G, s)             // generation: n first child of s
  while n ≠ NULL:                     // while there are children left to explore
    if n ∉ P:                         // n not in Path to avoid loops
      Push(P, n)                      // add child to the Path being explored
      (nextb, r) = BT(G, P, h, b)     // child returns min f and goal state
      if r ≠ NULL: return (nextb, r)  // if r solution, get out recursion
      if nextb < min: min = nextb     // update minimum f value
      Pop(P)                          // Discard last child from Path
    n = NextAdjacent(G, s, n)         // generation: n next child of s
  return (min, NULL)                  // sol. not found, return minimum f
```

- ***Question 1***: Draw the search tree as a result of applying the IDA* algorithm to the problem of finding the shortest path from Arad to Bucarest.
- ***Question 2***: Does the IDA* algorithm find a solution?
- ***Question 3***: If the answer is “Yes”:
  - What is the solution found?
  - What is the cost of this solution?
  - Is this the solution of minimum cost?

References

[1] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, third edition, 2010.

[2] R. E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27:97–109, 1985.