CSC421 Intro to Artificial Intelligence

Chapter 3
Informed Searching

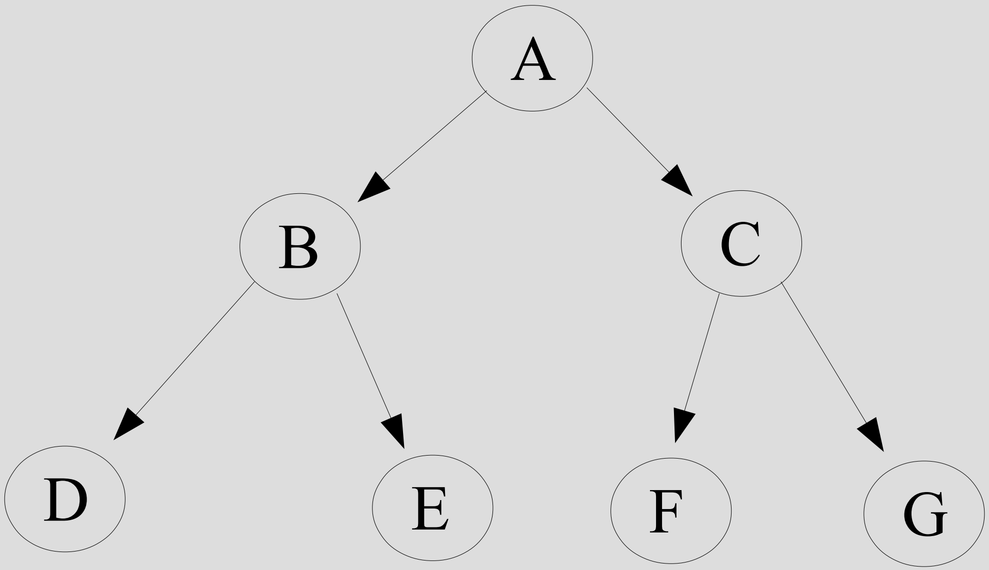
Review

Review

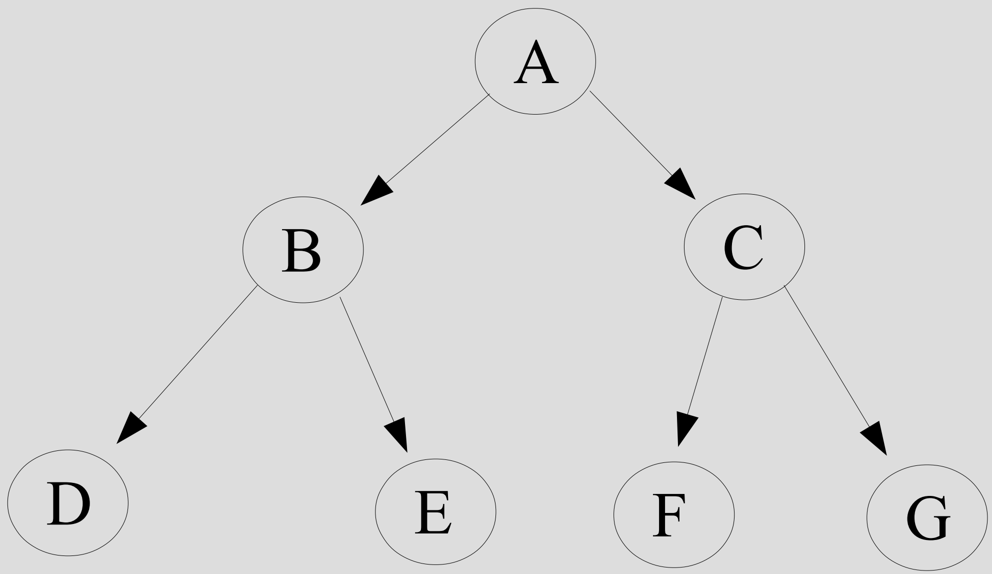
```
function TREE-SEARCH( problem, strategy) returns a solution, or failure
   initialize the search tree using the initial state of problem
   loop do
      if there are no candidates for expansion then return failure
      choose a leaf node for expansion according to strategy
      if the node contains a goal state then return the corresponding solution
      else expand the node and add the resulting nodes to the search tree
```

Strategy = picking the order of node expansion

Mini-test

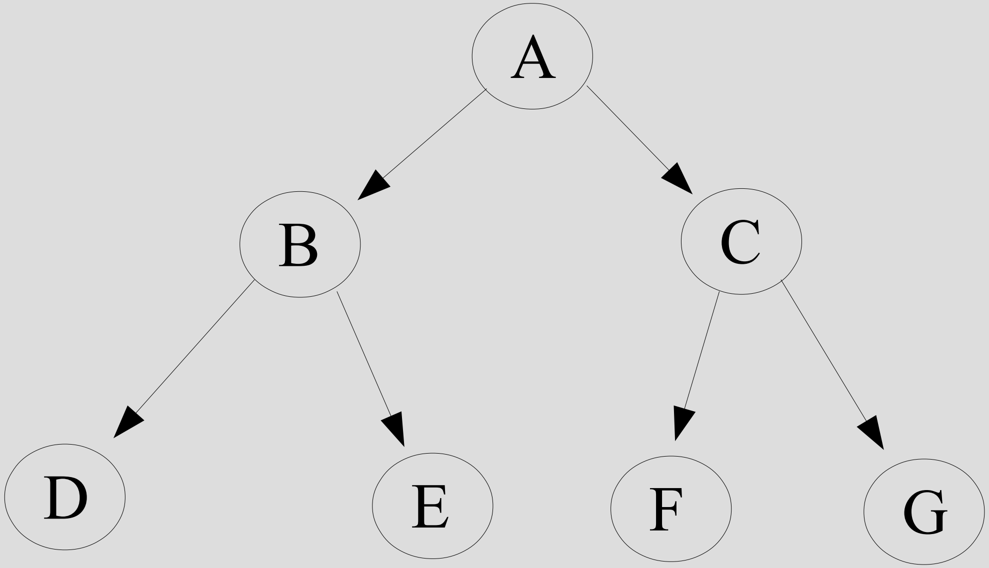


Mini-test



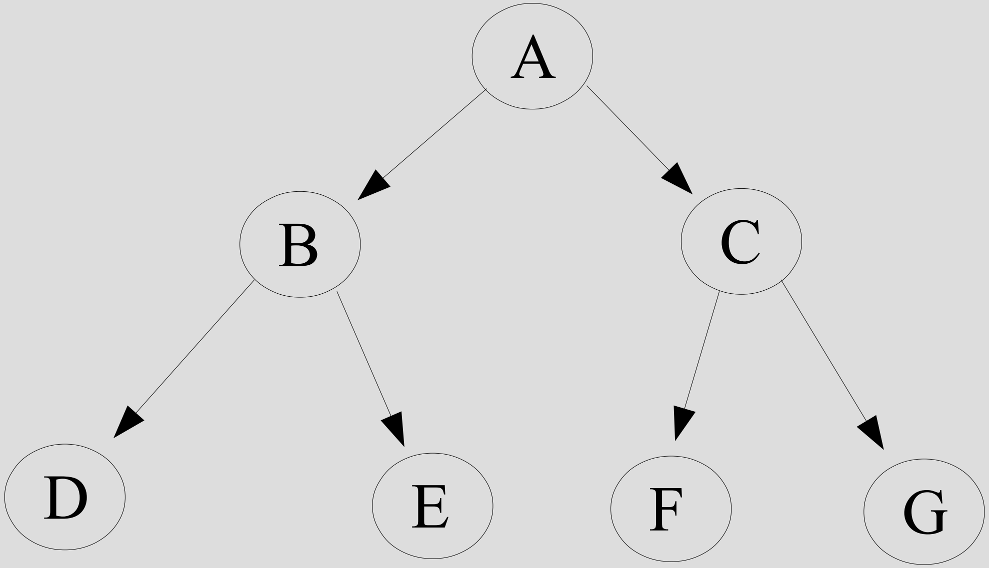
BFS: FIFO fringe =
[A]->[B,C]->[C,D,E]->[D,E,F,G]->...
Order of nodes visited: ABCDEFG

Mini-test



DFS: LIFO fringe =
[A]->[B,C]->[D,E,C]->[E,C]->[C]->[F,G]
Order of nodes visited: ABDECFG

Mini-test



A
ABC
ABDECFG

IDS: Multiple DFS up to depth
Order of nodes visited:
AABCABDECFG

Best-first search

Idea: use an evaluation function for each node – estimate of "desirability"

Expand most desirable unexpanded node

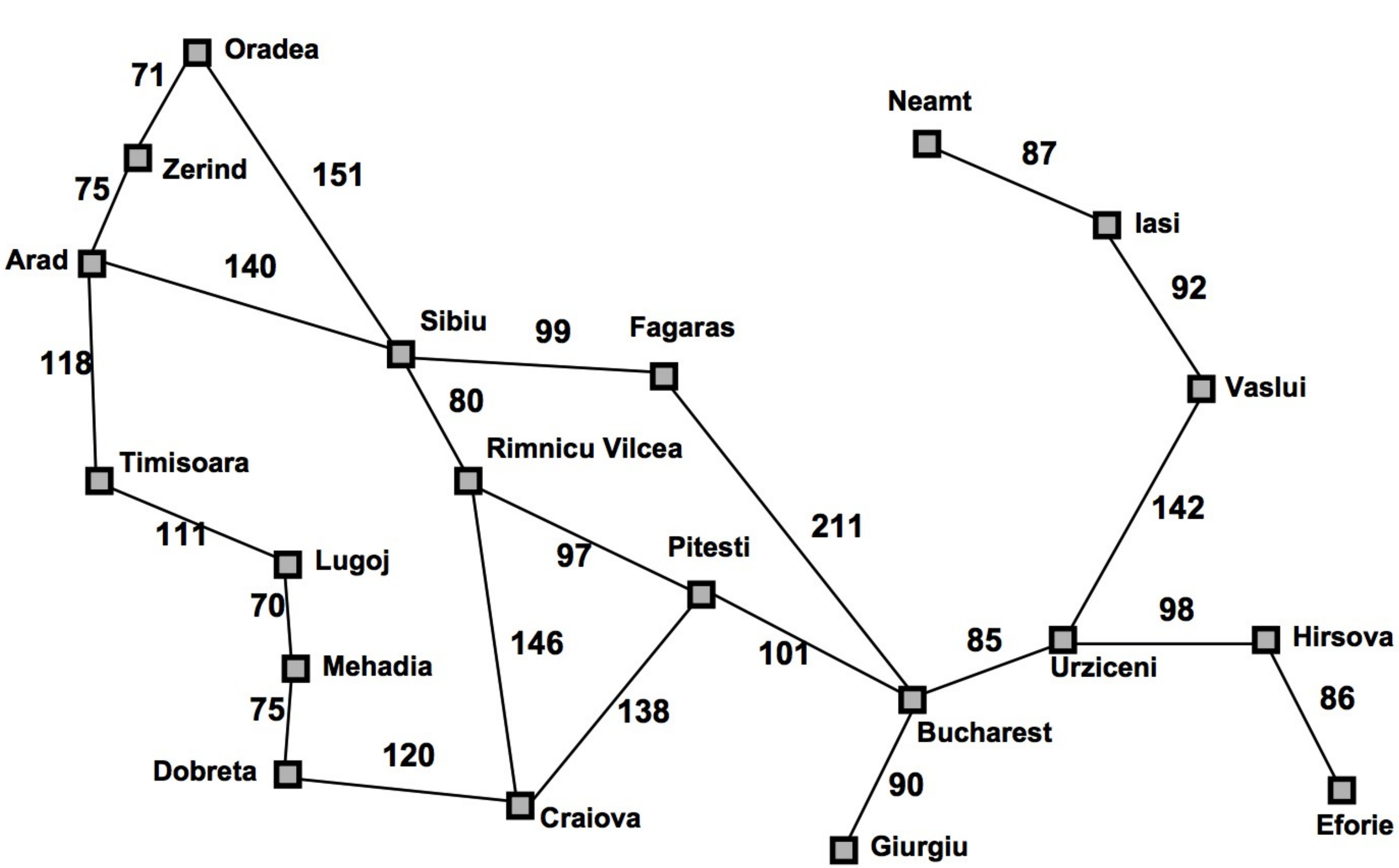
Implementation:

- Fringe is a queue sorted in decreasing order of disirability

Special cases:

- Greedy search
- A*-search

Map with step costs and straight-line distances to goal



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

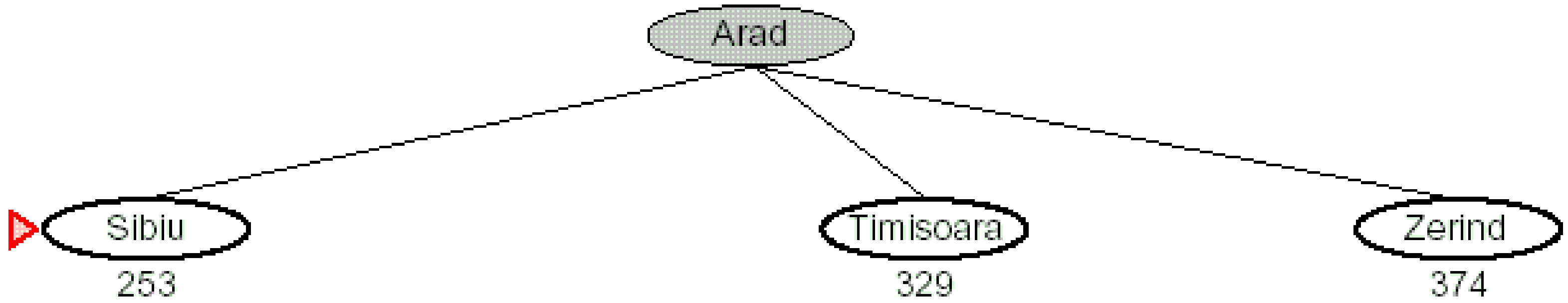
Greedy Search

Evaluation function h(n) (heuristic) = estimate of cost from n to goal

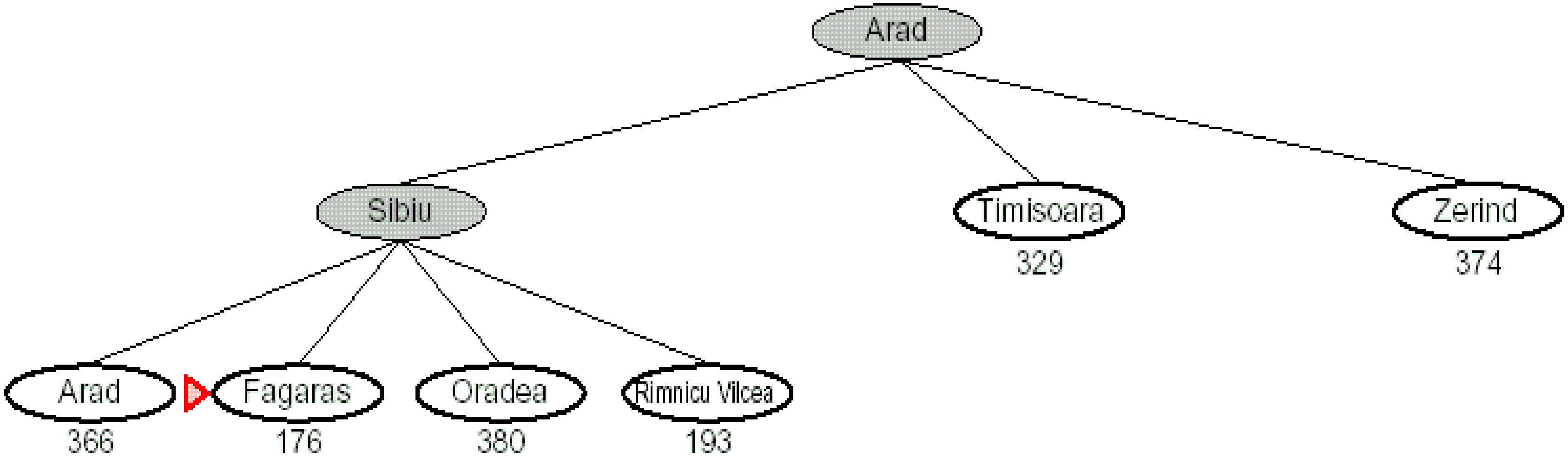
E.g., $h_{SLD}(n)$ = straight-line distance from n to Bucarest

Greedy search expands the node that appears to be closest to goal

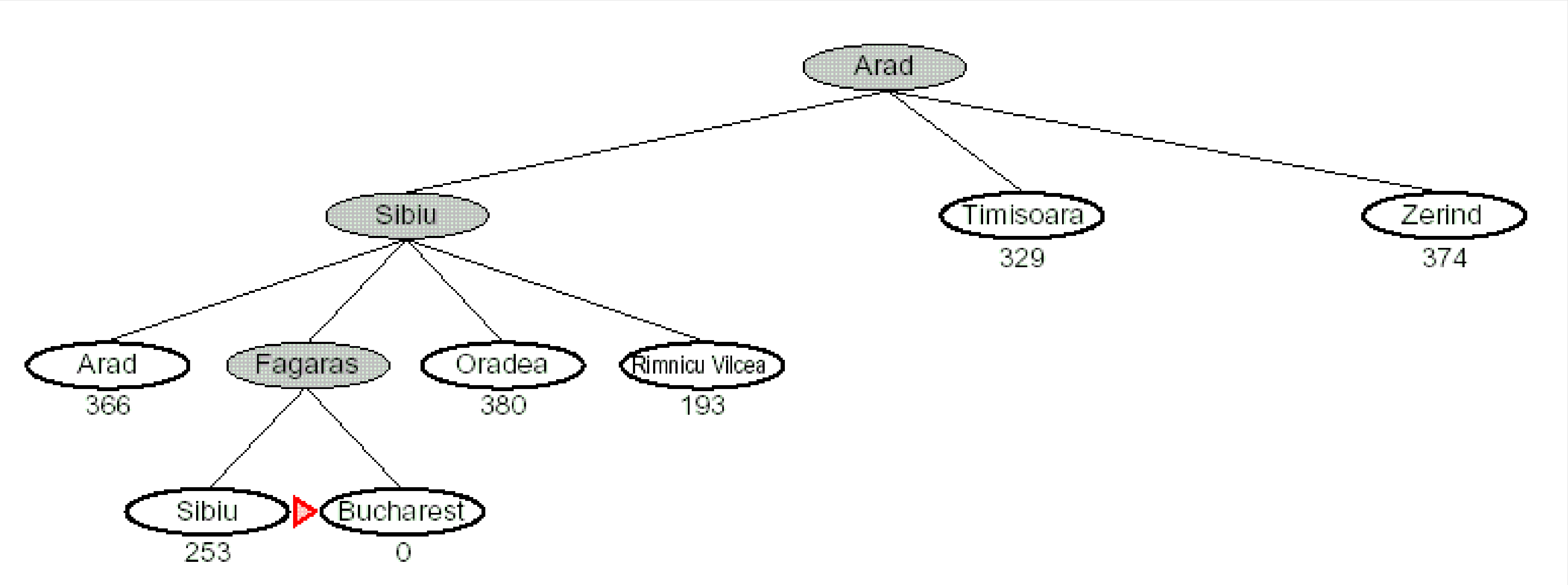
Greedy Search Example



Greedy Search Example



Greedy Search Example



Properties of Greedy Search

Complete: no it can get stuck in loops – however complete with repeated state checking

Time: $O(b^m)$ but good heuristic can give dramatic improvements in many cases

Space: $O(b^m)$

Optimal: No, why ?

A-* search

Idea: avoid expanding paths that are already expensive

Evaluation function $f(n) = g(n) + h(n)$

- $g(n)$: cost so far to reach n
- $h(n)$: estimated cost to goal from n
- $f(n)$: estimated total cost of path through n to goal

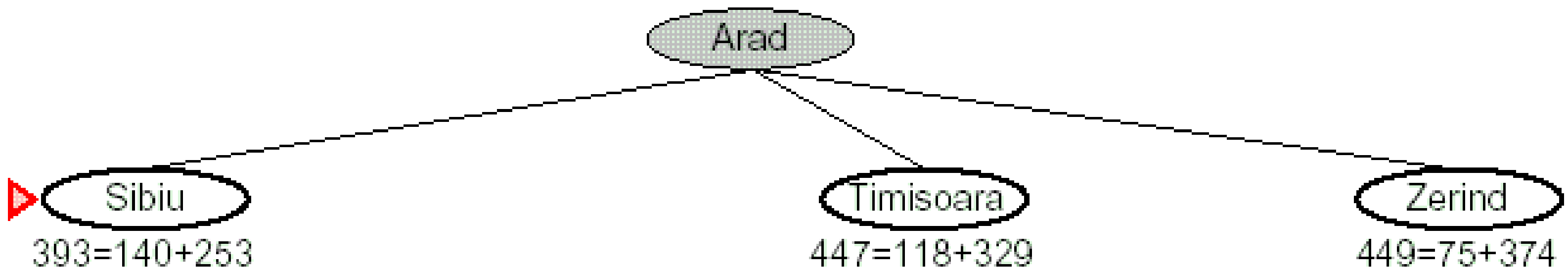
A-* search needs to use an admissable heuristic i.e always <= true cost

For example h_{SDL} is always less than the true distance (at least in Euclidean geometry)

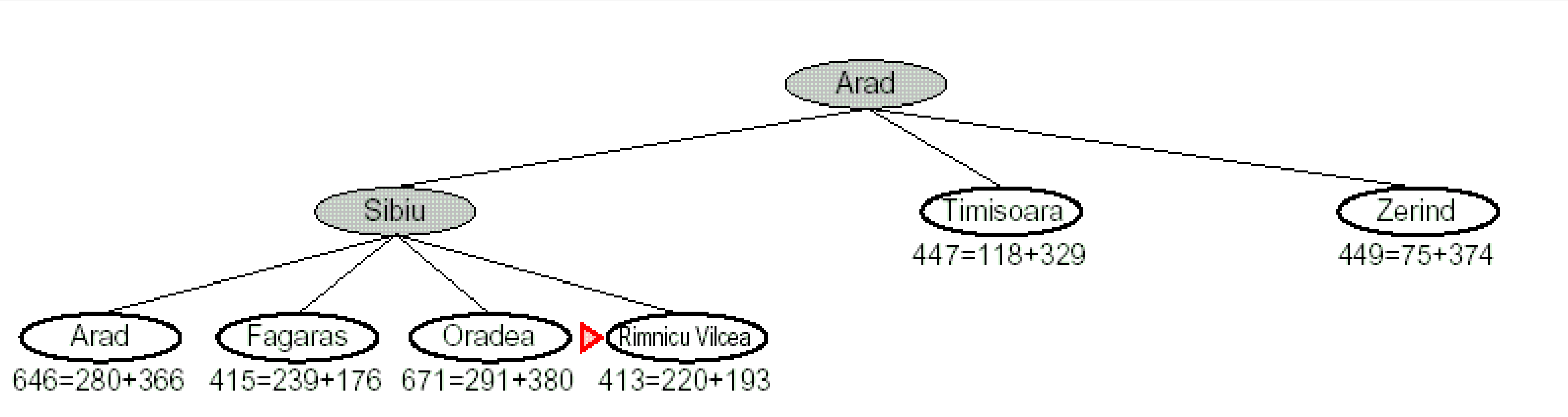
Theorem: A* is optimal

A* example

A* example


Arad
Sibiu
393=140+253
Timisoara
447=118+329
Zerind
449=75+374

A* example



Gradually adds *f-contours* of nodes
Nice easy optimality proof read the book

A* properties

Complete: Yes, unless there are infinitely many nodes with f <= f(G)

Time: exponential in [relative error in * length of solution]

Memory: Keeps all nodes in memory

Optimality: yes

Problems

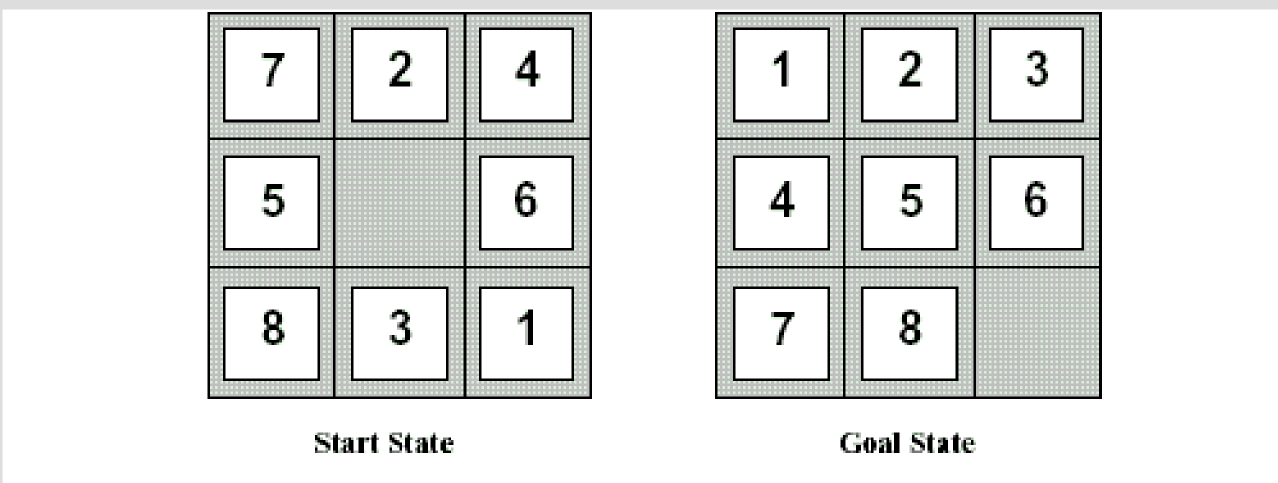
- Exponential growth for most optimal solution
- Sometimes good-enough ok (suboptimal)
- Memory-intensive (read book for some approaches to reducing memory load)

Admissable Heuristic

e.g., for the 8-puzzle

$h_1(n)$ = number of misplaced tiles

$h_2(n)$ = total Manhattan distance (i.e #squares from desired location of each tile)



$h_1(S) = ?$

$h_2(S) = ?$

Dominance

If $h_2(n) >= h_1(n)$ for all n (both admissable) then h_2 dominates h_1 is better for search

Typical search costs:

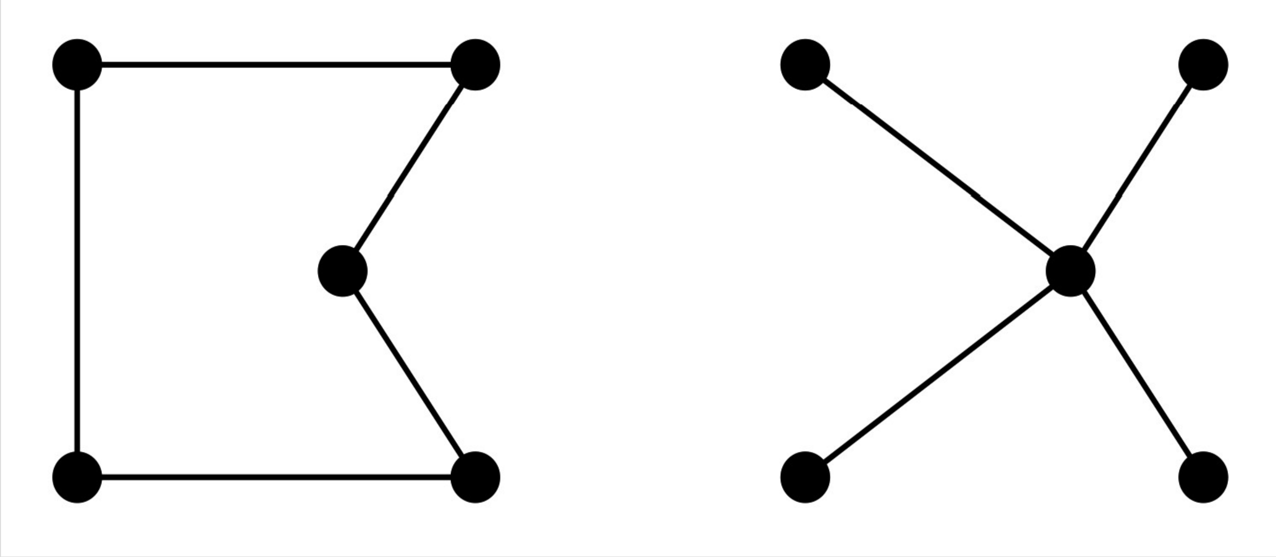
- D=14
  - IDS 3,473,941 nodes
  - A*(h1) = 539 nodes
  - A*(h2) = 113 nodes

Relaxed problems

- Admissable heuristics can be derived from the exact solution cost of a relaxed version of the problem
- If the rules of the 8-puzzle are relaxed so that a tile can move anywhere then h1 gives the shortest solution
- What about h2 ?
- Key point: the optimal solution cost of a relaxed problem is no greater than the optimal solution of the real problem

More relaxed problems

- Well known example: traveling salesman problem
- Find the shortest tour visiting all cities exactly once



Minimum spanning tree can be computed in $O(n^2)$ and is a lower bound on the shortest (open) tour

Summary

Heuristic functions estimate costs of shortest paths
Good heuristics can dramatically reduce search cost
Greedy best-first search expands lowest h
- Incomplete, not always optimal

A* search expands lowest g + h
- Complete and optimal

Admissable heuristics can be derived from the exact solution of relaxed problems

Food for thought

- A* in modern computer games
- Simplifying search space without compromising solution
  - Hierarchical path finding
  - Waypoints
  - Navigation mesh
- Memory issues
  - Node bank
  - IDA*
- Game examples
  - Age of empires
  - Civilization
  - World of Warcraft

Waypoint and Navigation Graph

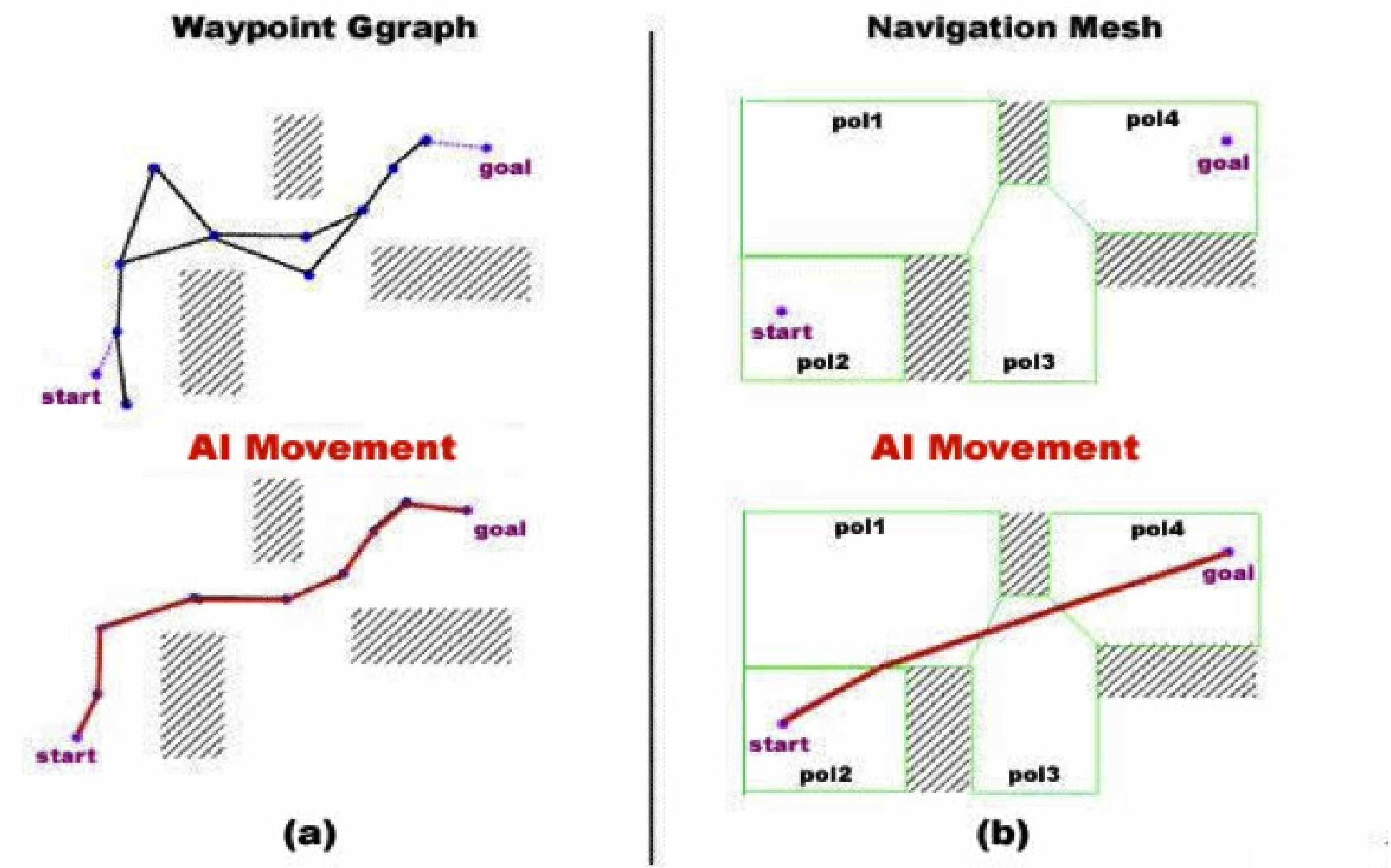


Fig. 3 Different representations of waypoint graph and NavMesh [7]

Age of Empires