Artificial Intelligence

Solving Problem by Searching

Today’s Topic

- Informed Search Strategies: one that uses problem specific knowledge
  - Best First Search & its variant
  - Heuristic Functions
  - Local Search and Optimization

Best-first search

- Idea: use an evaluation function *f(n)* for each node
  - ❑ estimate of "desirability“
  - ❑ Expand most desirable unexpanded node
  - ❑ Evaluation function estimates distance to the goal

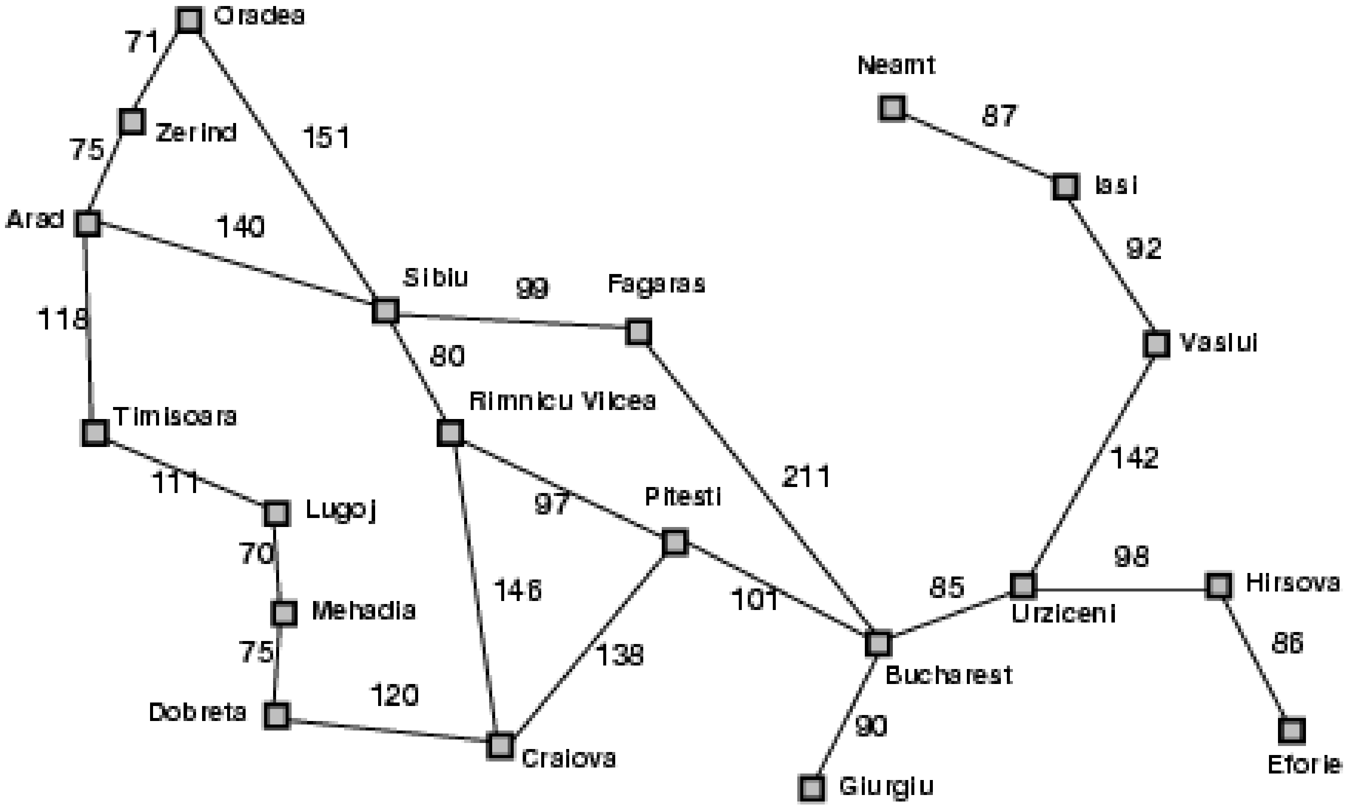
- <u>Implementation</u>:
  Fringe is a priority queue sorted in ascending order of f-values

- Special cases:
  – greedy best-first search
  – A^* search

Heuristic Function

- It maps each state to a numerical value depicts goodness of a node.
  - h(n)= value, where h() is heuristic function and n is current state
- It is estimated cost of cheapest path from initial node to goal
- Example: (In Romania)
  - Straight line distance from Arad to Bucharest
  - Heuristic functions are most common way in which additional knowledge of the problem is imparted to search algorithms

Romania with step costs in km



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Greedy best-first search

- Evaluation function $f(n) = h(n)$ (heuristic)

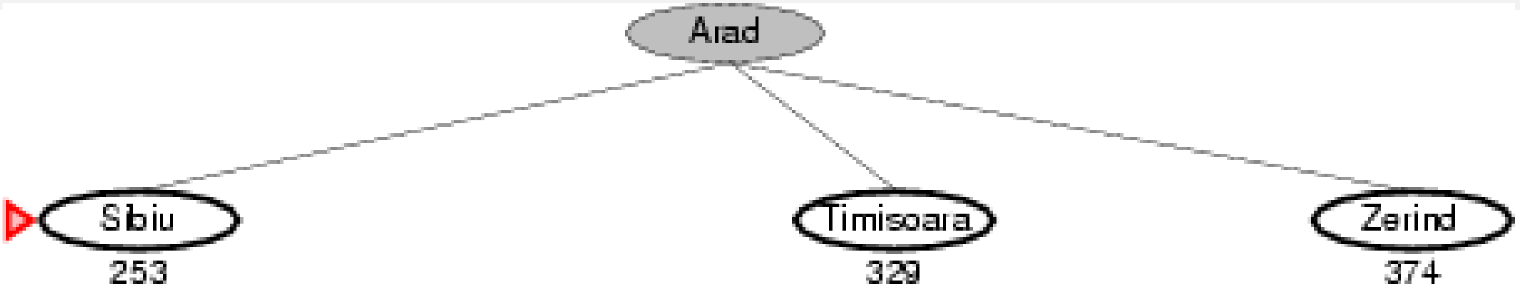
  = estimate of cost from n to *goal*

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

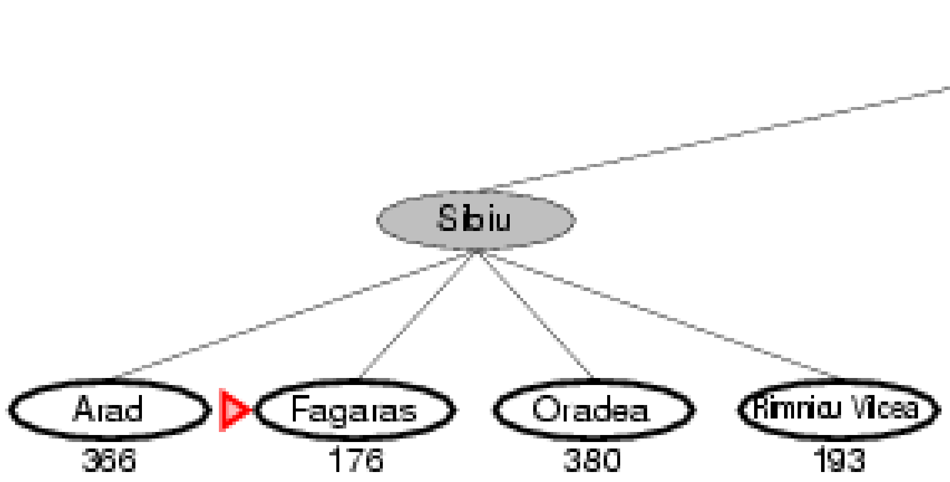
- Greedy best-first search expands the node that appears to be closest to goal
  - f(n)=h(n)

Greedy best-first search example

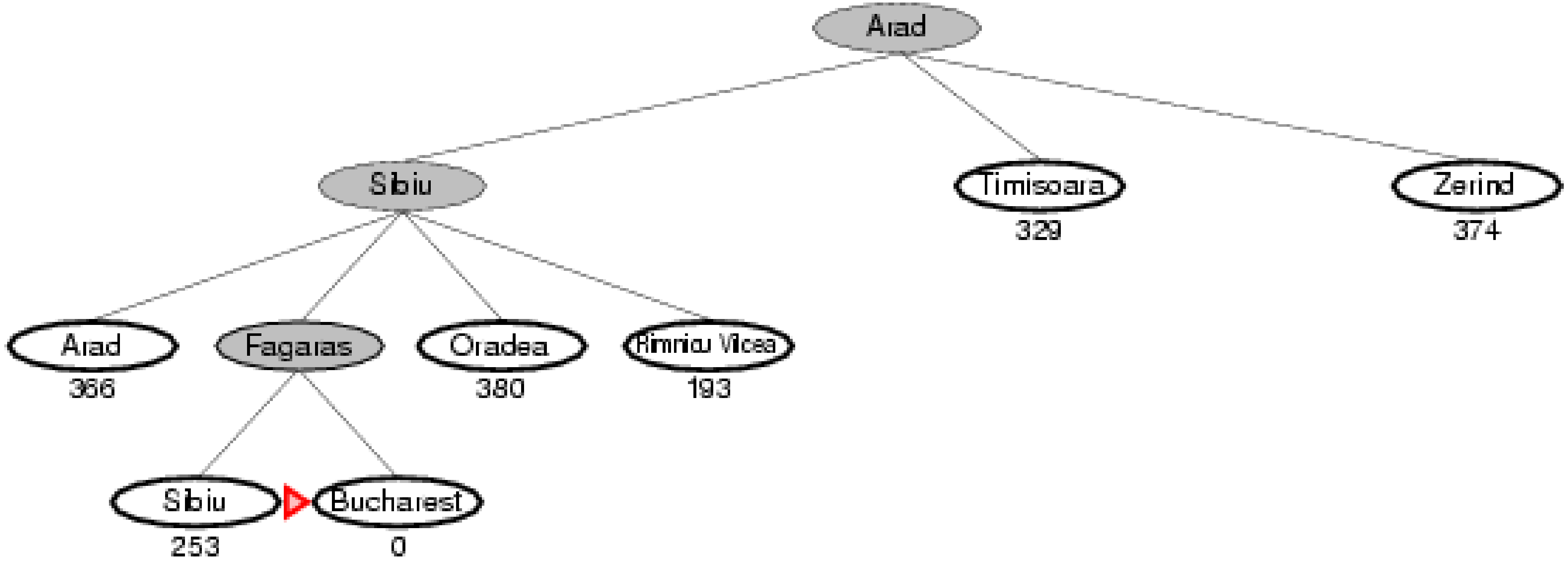
Greedy best-first search example



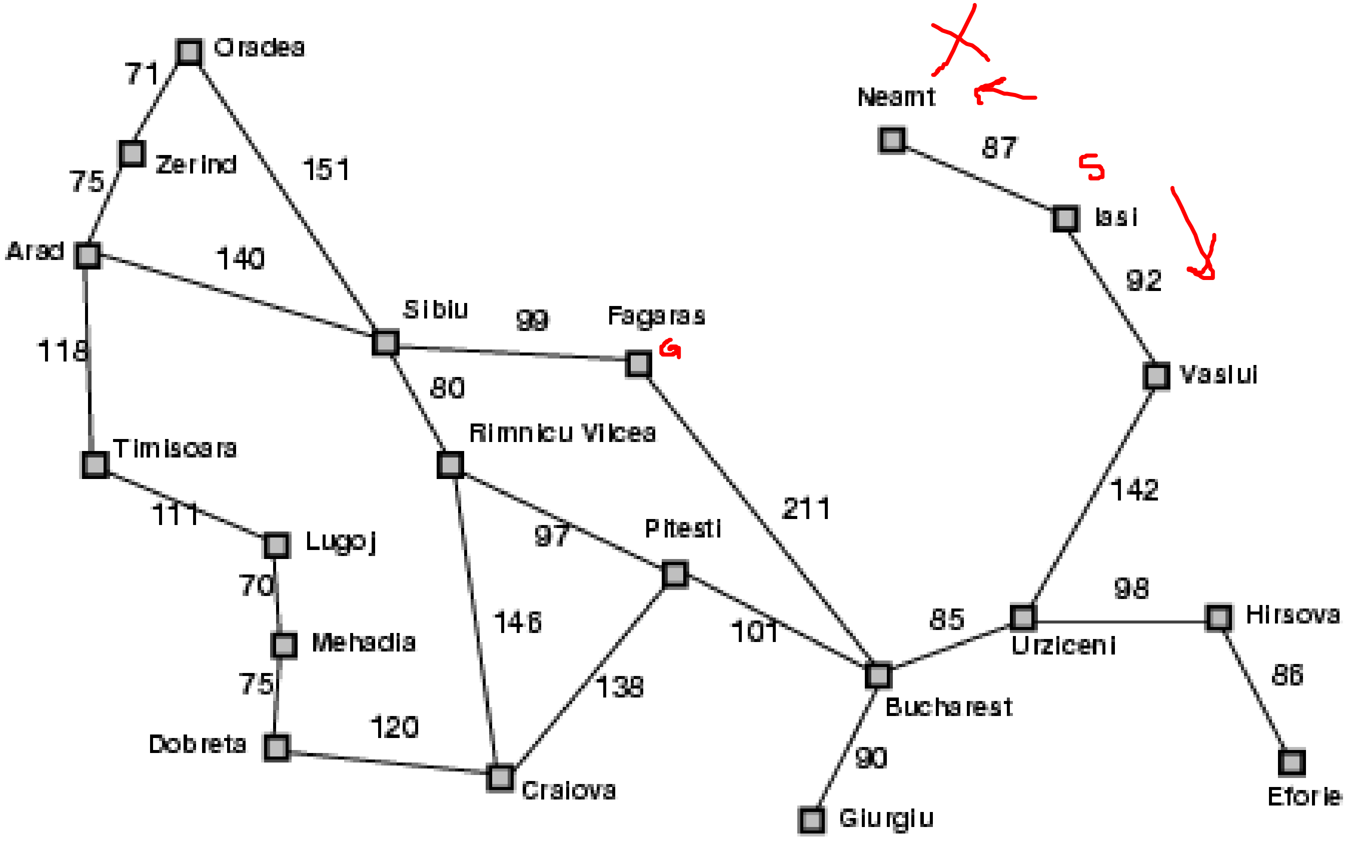
Greedy best-first search example



Greedy best-first search example



Greedy Best First Search



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Properties of greedy best-first search

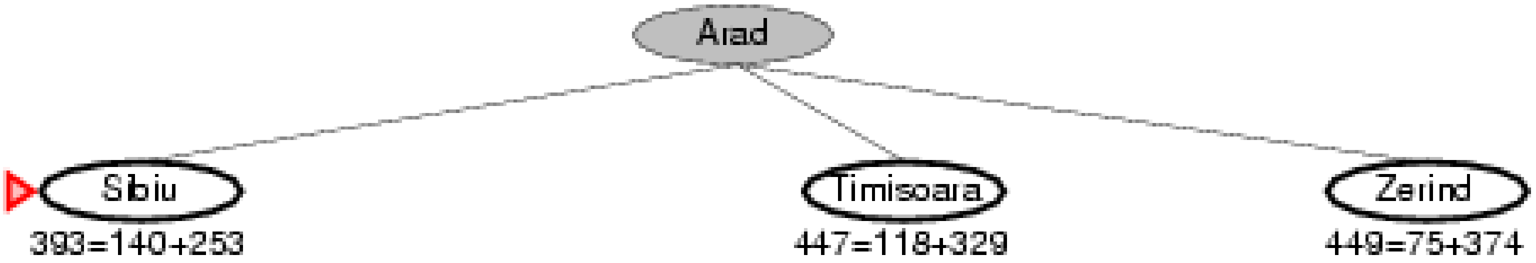
- Complete? No – can get stuck in loops, e.g., Iasi → Neamt → Iasi → Neamt →
- Time? $O(b^m)$, but a good heuristic can give dramatic improvement
- Space? $O(b^m)$ -- keeps all nodes in memory
- Optimal? No

A^* search

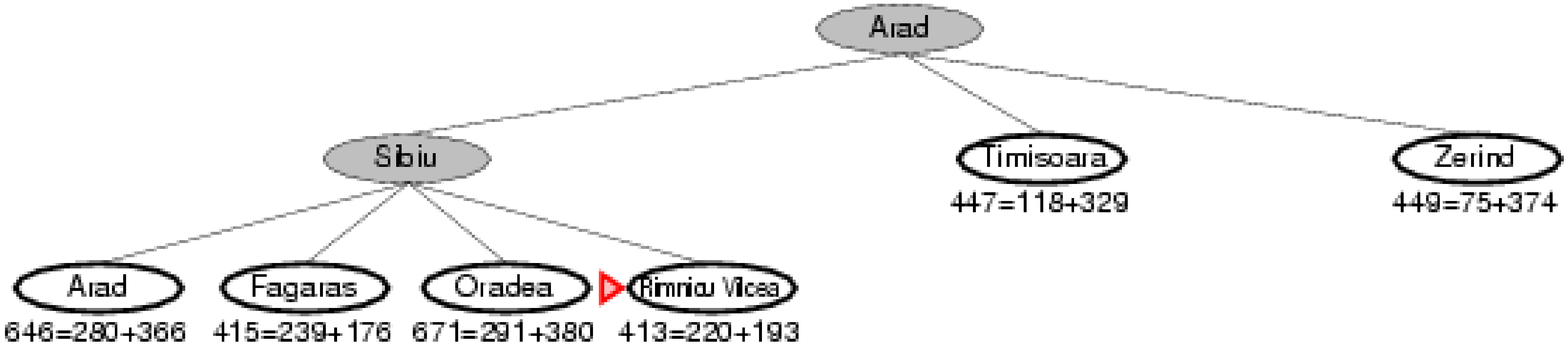
- Idea: avoid expanding paths that are already expensive
- Evaluation function $f(n) = g(n) + h(n)$
  - – $g(n)$ = cost from start/node to n
  - – $h(n)$ = estimated cost from n to goal node
  - – $f(n)$ = estimated total cost of path through n to goal

A* search example

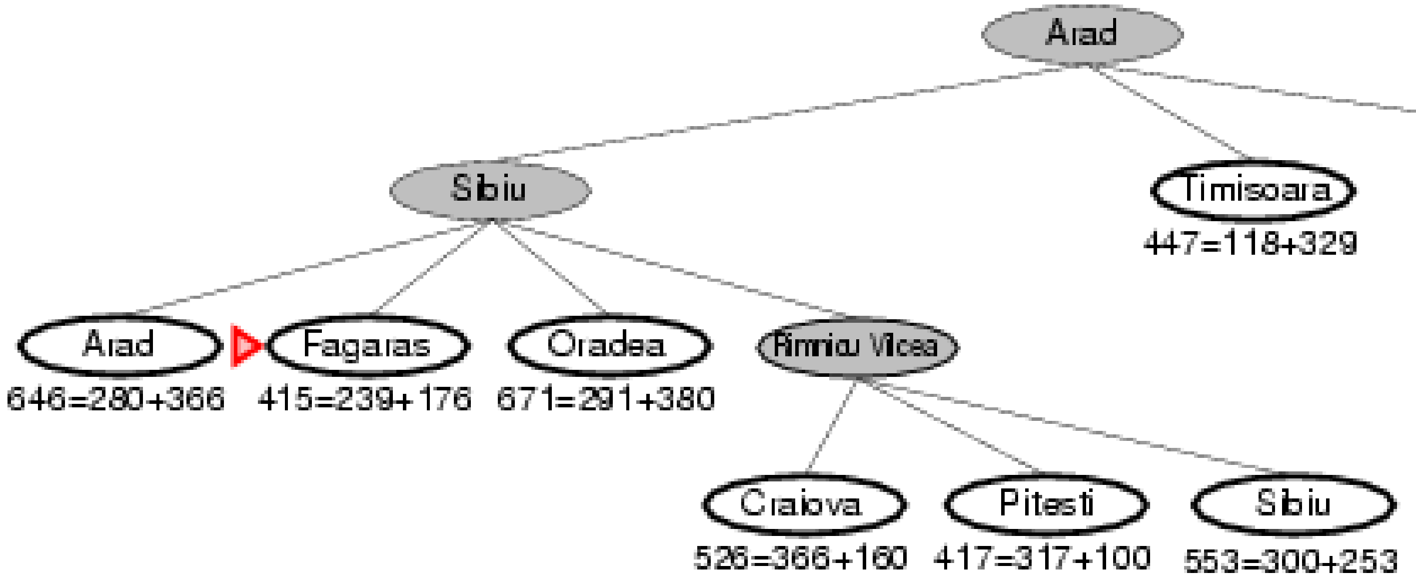
A* search example



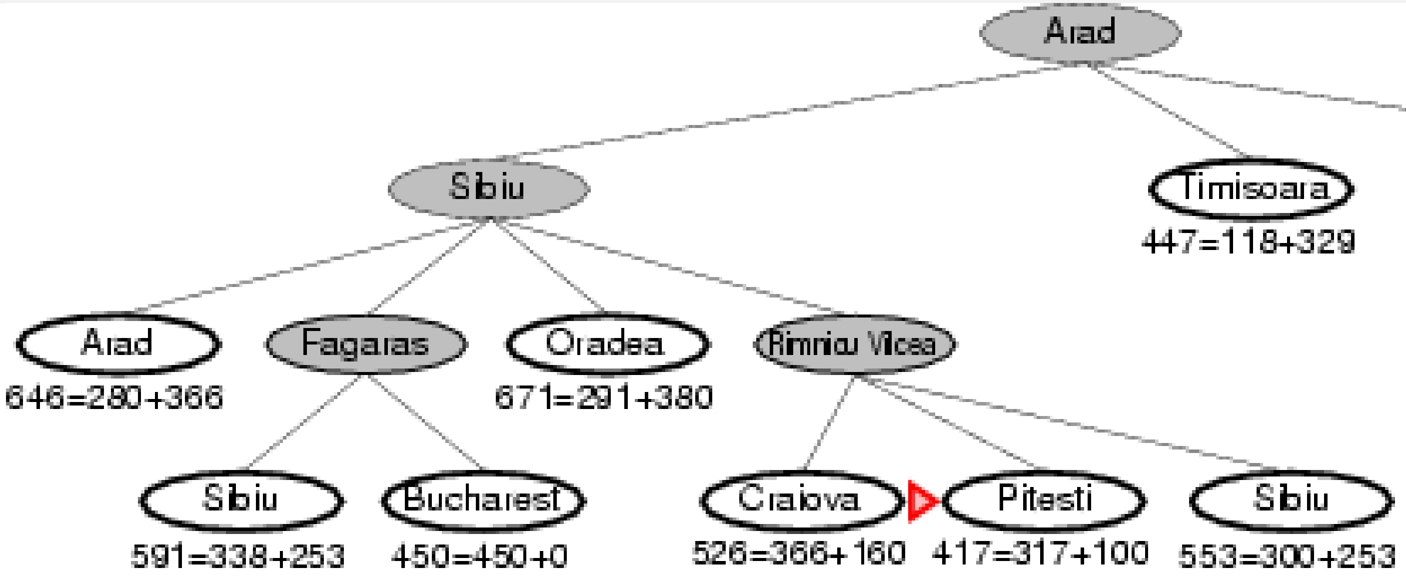
A* search example



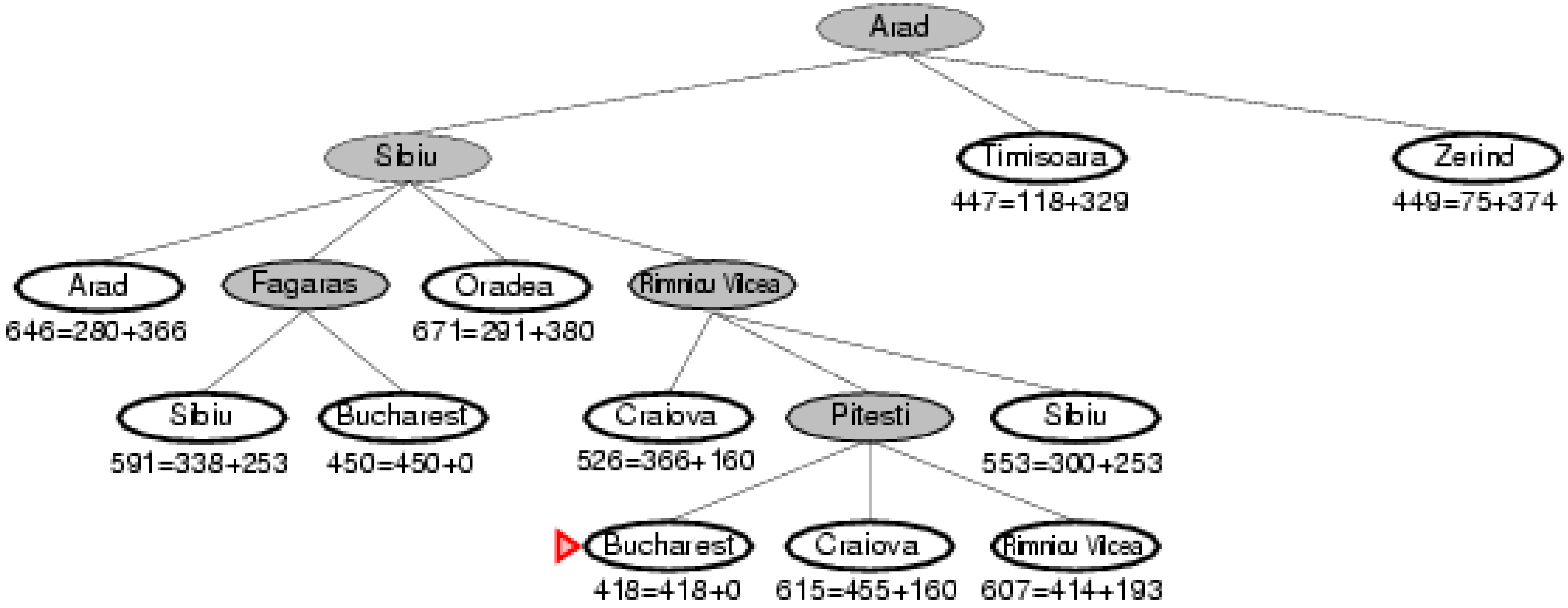
A* search example



A* search example



A* search example



Conditions for Optimality

- Admissible
- Consistency

A^* is Admissible

- A* will never overestimate the cost of reaching the goal.
- The cost to reach goal is guaranteed to be lower or equal to the actual cost.
- This property ensures that **the algorithm will eventually find the optimal solution, if one exists.**

Admissible heuristics

- A heuristic *h(n)* is admissible if for every node *n*,
  $h(n) \leq h^*(n)$, where $h^*(n)$ is the true/actual cost to reach the goal state from *n and h(n) is the estimated cost to reach the goal*

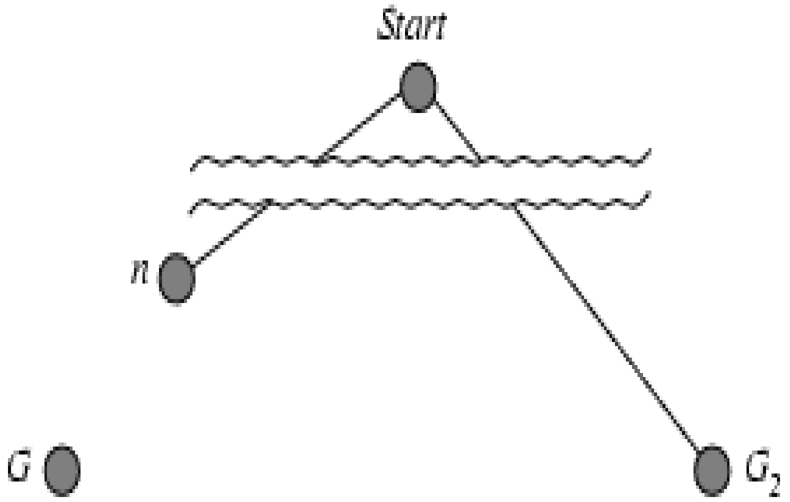
- An admissible heuristic never overestimates the cost to reach the goal, i.e., it is optimistic

- Example: $h_{SLD}(n)$ (never overestimates the actual road distance)

- Theorem: If *h(n)* is admissible, A^* using TREE−SEARCH is optimal

Optimality of A* (proof)

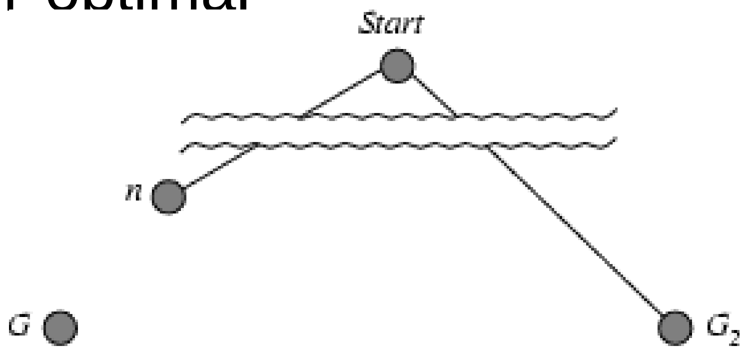
- Suppose some suboptimal goal G_2 has been generated and is in the fringe. Let *n* be an unexpanded node in the fringe such that *n* is on a shortest path to an optimal goal *G*.



- $f(G_2) = g(G_2)$ since $h(G_2) = 0$
- $g(G_2) > g(G)$ since G_2 is suboptimal
- $f(G) = g(G)$ since $h(G) = 0$
- $f(G_2) > f(G)$ from above

Optimality of A* (proof)

- Suppose some suboptimal goal G_2 has been generated and is in the fringe. Let n be an unexpanded node in the fringe such that n is on a shortest path to an optimal goal G.



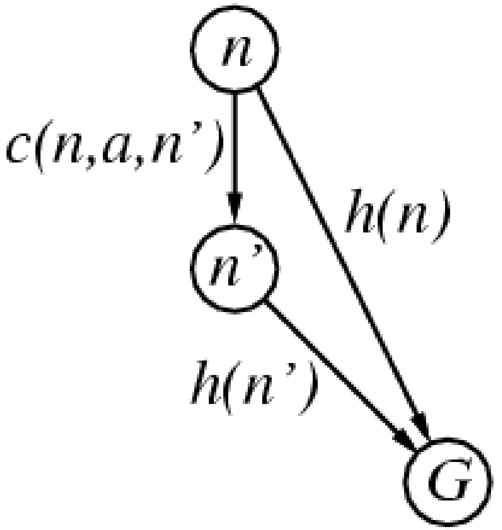
- $f(G_2) > f(G)$ from above
- $h(n) \le h^*(n)$ since h is admissible, h* is minimal distance/actual cost
- $g(n) + h(n) \le g(n) + h^*(n)$
- $f(n) \le f(G)$

Hence $f(G_2) > f(n)$, and A* will never select G_2 for expansion

Consistent heuristics

- A heuristic is consistent if for every node *n*, every successor *n'* of *n* generated by any action *a*, the estimated cost of reaching the goal from node n, *h(n),* is not greater than the step cost of getting to *n'*, (g(n'))+ *estimated cost of reaching the goal from n' (f(n'))*

$h(n) \leq g(n') + h(n')$
$h(n) \leq c(n,a,n') + h(n')$



traingle inequality

- Intuition: can't do worse than going through n'.
- If *h* is consistent, we have

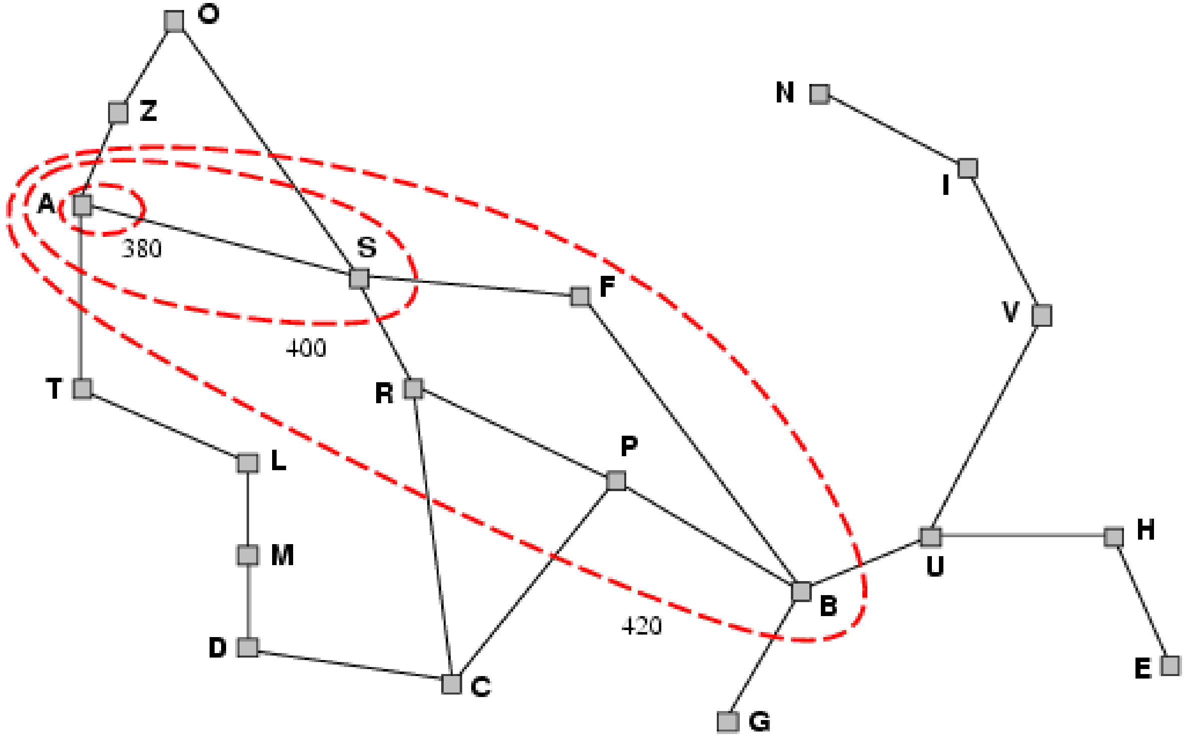
$$f(n') = g(n') + h(n') = g(n) + c(n,a,n') + h(n') \geq g(n) + h(n) = f(n)$$

Theorem: If *h(n)* is consistent, A* using GRAPH-SEARCH is optimal

Optimality of A*

- A* expands nodes in order of increasing *f* value
  - This means that nodes with lower *f*-values are expanded before nodes with higher *f*-values.

- Gradually adds "f-contours" of nodes
- Contour i has all nodes with f=fi, where fi < fi+1



Properties of A*

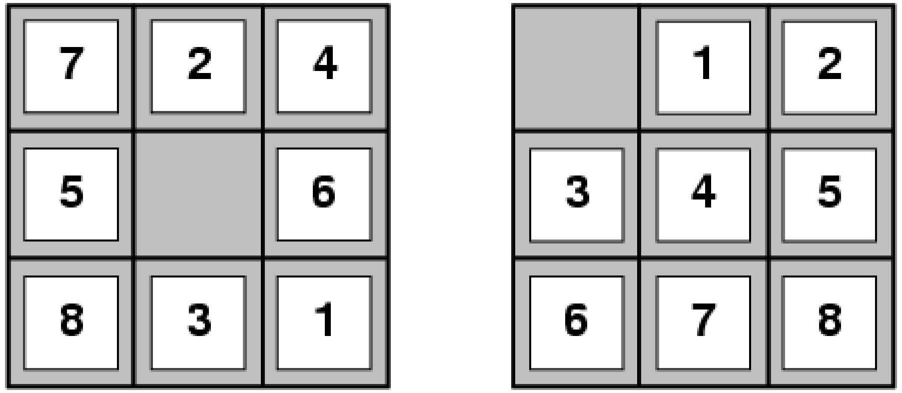
- Complete? Yes (unless there are infinitely many nodes with $f \leq f(G)$)
- Time? Exponential
- Space? Keeps all nodes in memory
- Optimal? Yes

Admissible heuristics

E.g., for 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)



Start State Goal State

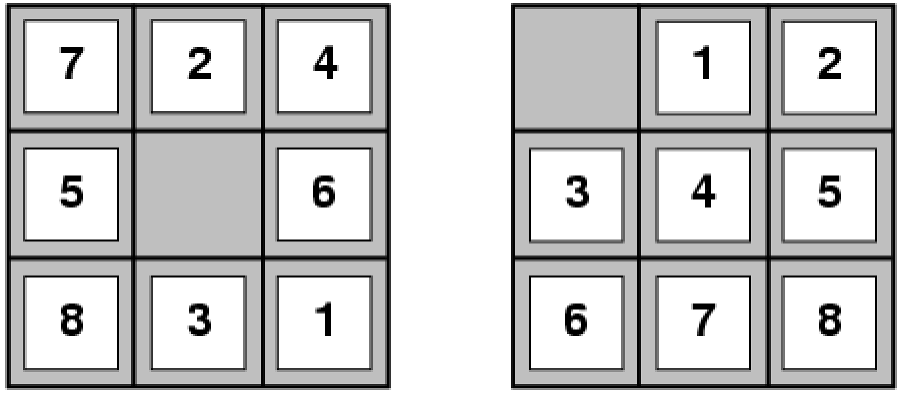
- [$h_1(S)$ = ?]()
- [$h_2(S)$ = ?]()

Admissible heuristics

E.g., for 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)



Start State Goal State

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

Relaxed problems

- A problem with fewer restrictions on the actions is called a relaxed problem
- The cost of an optimal solution to a relaxed problem is an admissible heuristic for the original problem
- If the rules of the 8-puzzle are relaxed so that a tile can move anywhere, then $h_1(n)$ gives the shortest solution
- If the rules are relaxed so that a tile can move to any adjacent square, then $h_2(n)$ gives the shortest solution