CSE 417: Artificial Intelligence

Chapter 4: Informed search algorithms

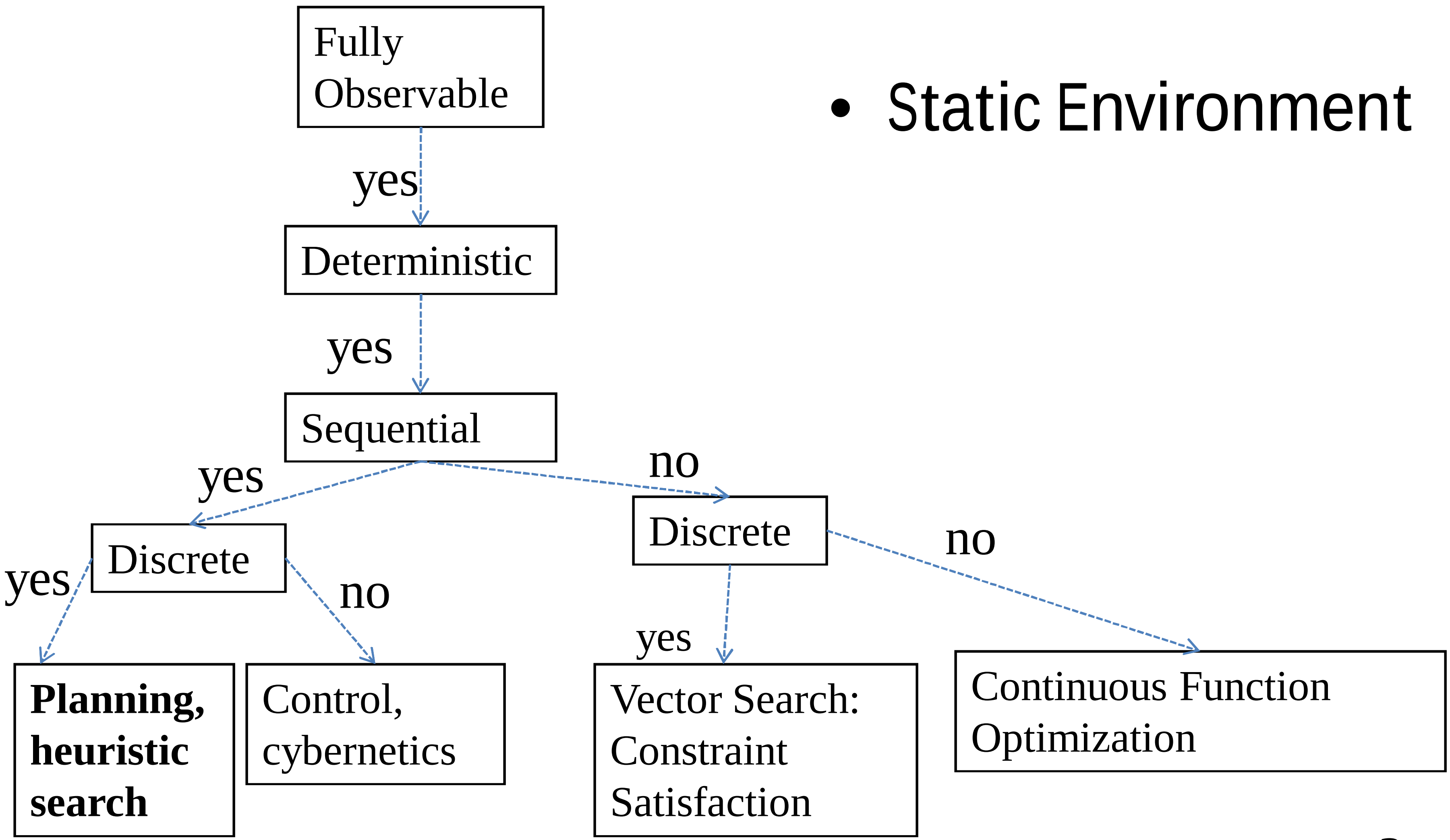
Spring 2015

Department of Computer Science and Engineering (CSE)

Outline

- Best-first search
- A^* search
- Heuristics
- Local search algorithms
- Hill-climbing search
- Simulated annealing search
- Local beam search

Environment Type Discussed In this Lecture



- Static Environment

Review: Tree search

```
function TREE-SEARCH( problem, fringe) returns a solution, or failure
    fringe ← INSERT(MAKE-NODE(INITIAL-STATE[problem]), fringe)
    loop do
        if fringe is empty then return failure
        node ← REMOVE-FRONT(fringe)
        if GOAL-TEST[problem] applied to STATE(node) succeeds return node
        fringe ← INSERTALL(EXPAND(node, problem), fringe)
```

- A search strategy is defined by picking the order of node expansion
- Which nodes to check first?

Knowledge and Heuristics

- Simon and Newell, *Human Problem Solving*, 1972.
- Thinking out loud: experts have strong opinions like “this looks promising”, “no way this is going to work”.
- S&N: intelligence comes from **heuristics** that help find promising states fast.

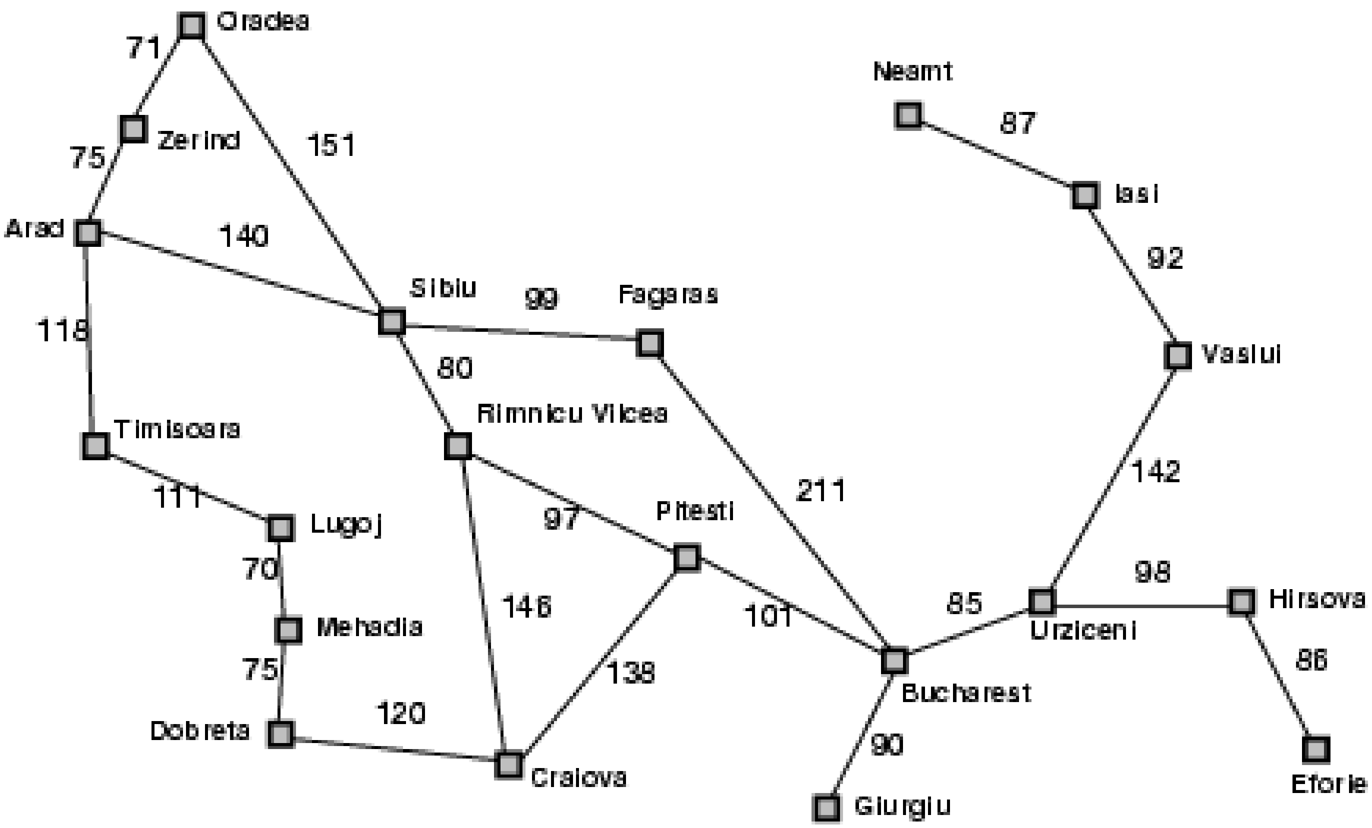
Best-first search

- Idea: use an evaluation function *f(n)* for each node
  - estimate of "desirability"
  - → Expand most desirable unexpanded node
  - →

- Implementation:
  Order the nodes in frontier in decreasing order of desirability

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

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

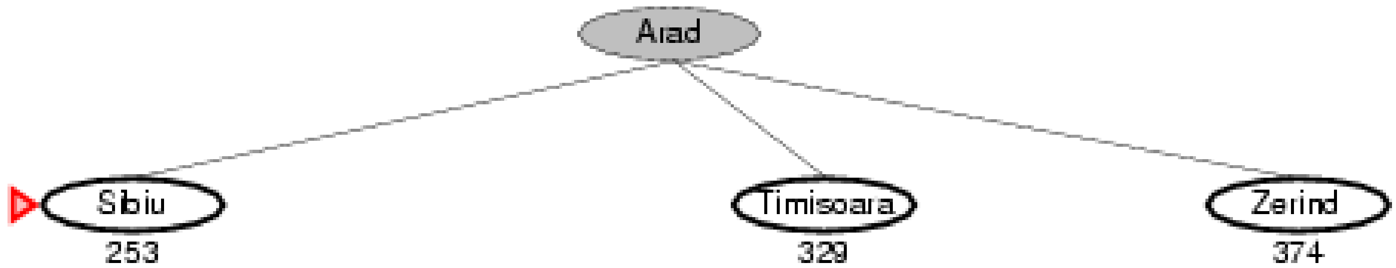
Greedy best-first search

- Evaluation function
  - $f(n) = h(n)$ (heuristic)
  - = estimate of cost from n to *goal*

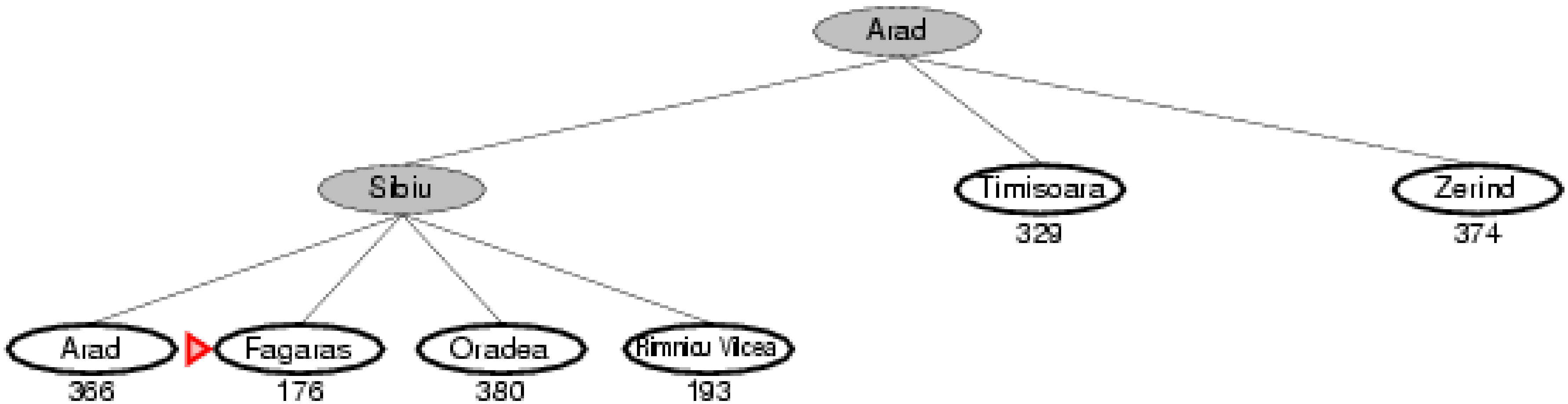
- e.g., $h_{SLD}(n)$ = straight-line distance from n to Bucharest
- Greedy best-first search expands the node that appears to be closest to goal
-

Greedy best-first search example

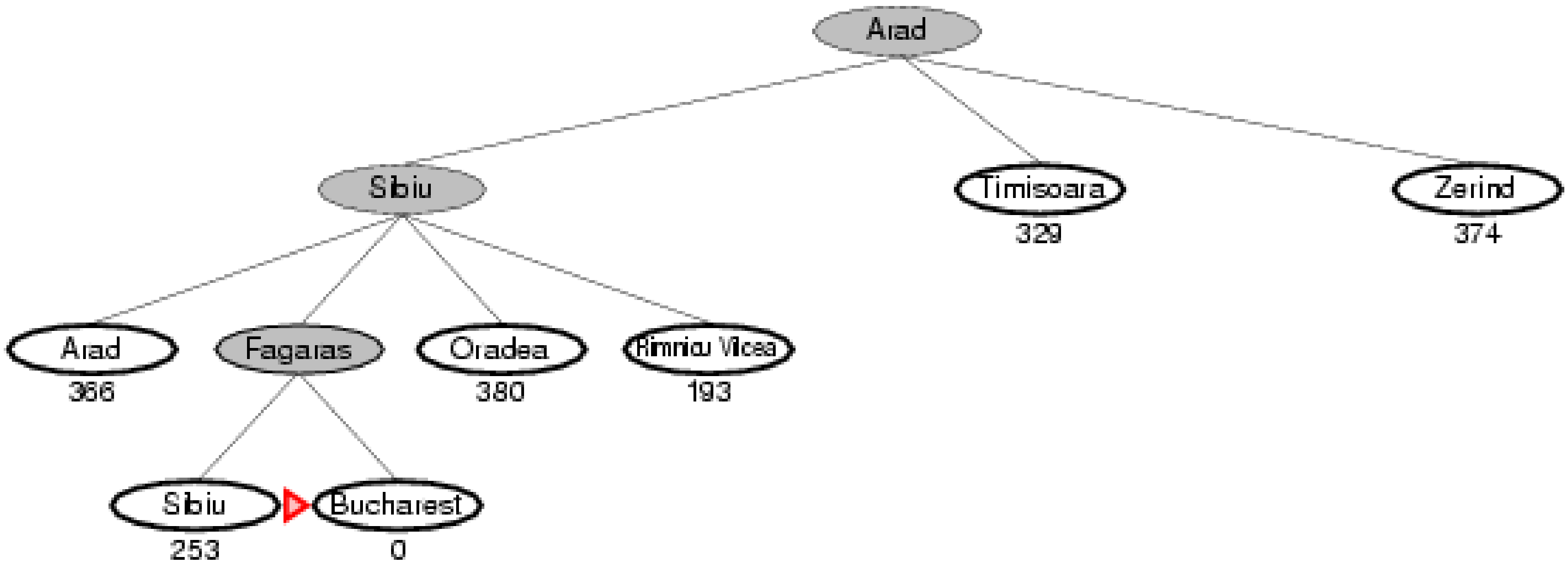
Greedy best-first search example



Greedy best-first search example



Greedy best-first search example



http://aispace.org/search/

Properties of greedy best-first search

- Complete? No – can get stuck in loops,
  - e.g. as Oradea as goal
    - 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
-

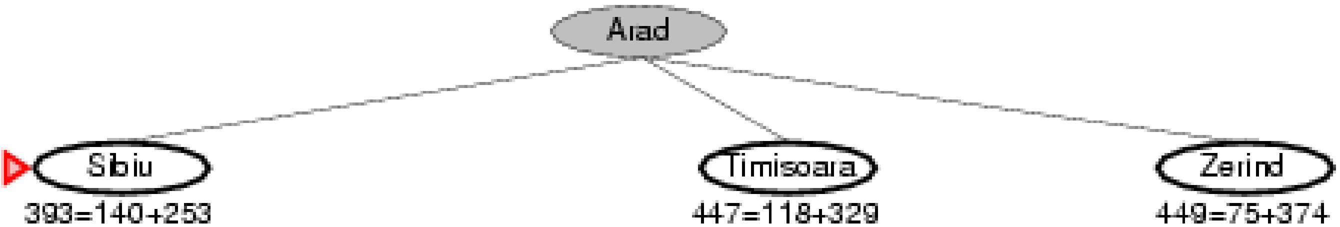
A* search

- Idea: avoid expanding paths that are already expensive.
- Very important!

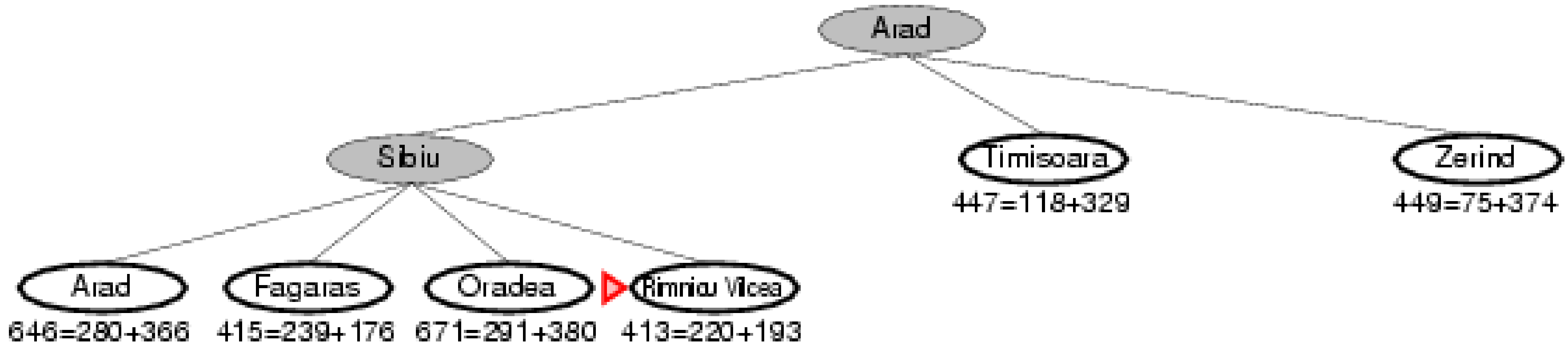
- Evaluation function $f(n) = g(n) + h(n)$
- $g(n)$ = cost so far to reach n
- $h(n)$ = estimated cost from n to goal
- **$f(n)$ = estimated total cost of path through n to goal**

A* search example

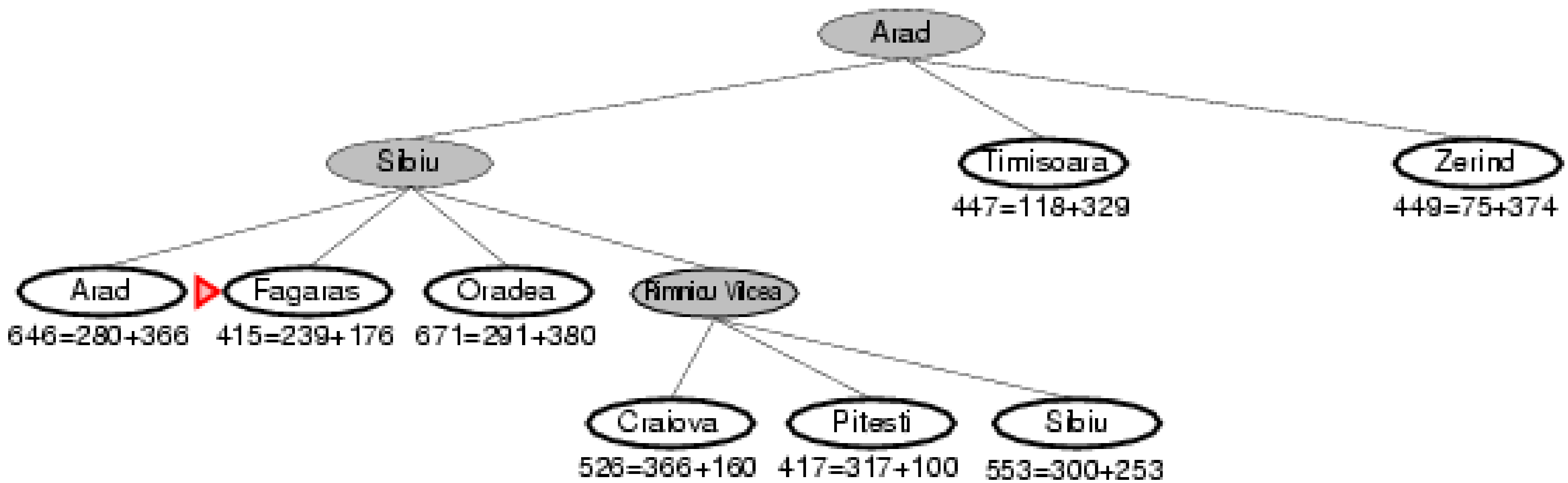
A* search example



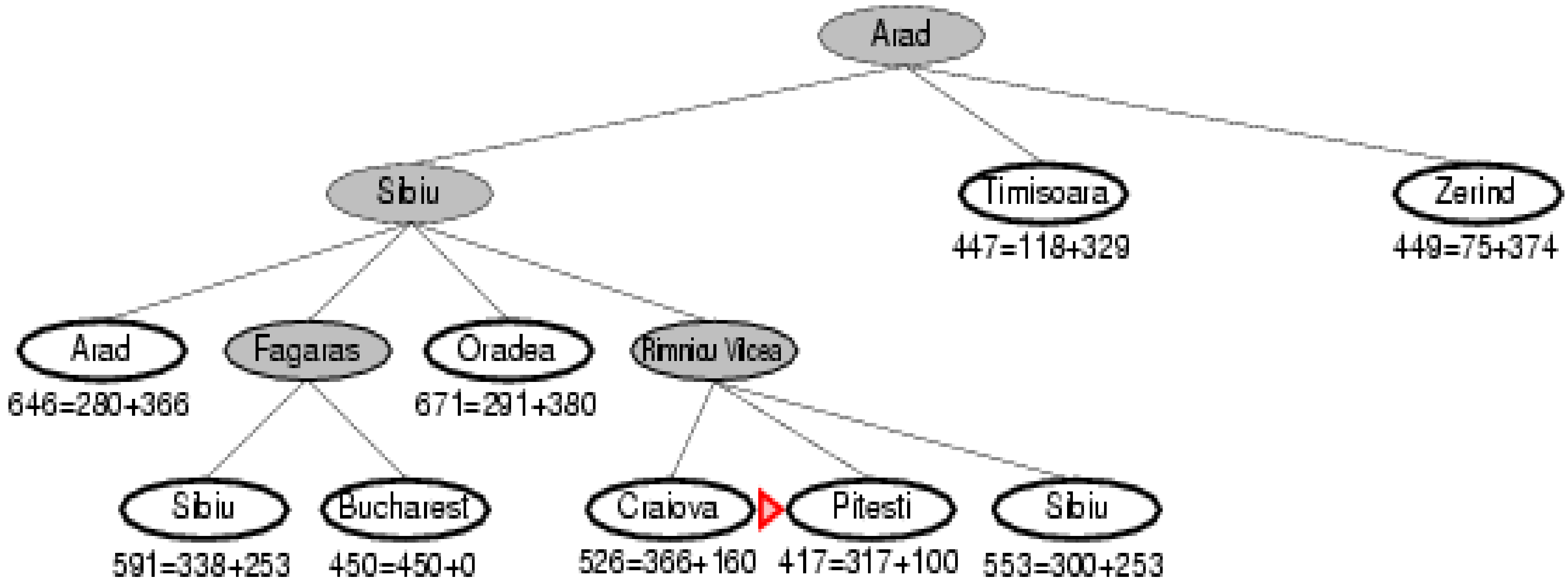
A* search example



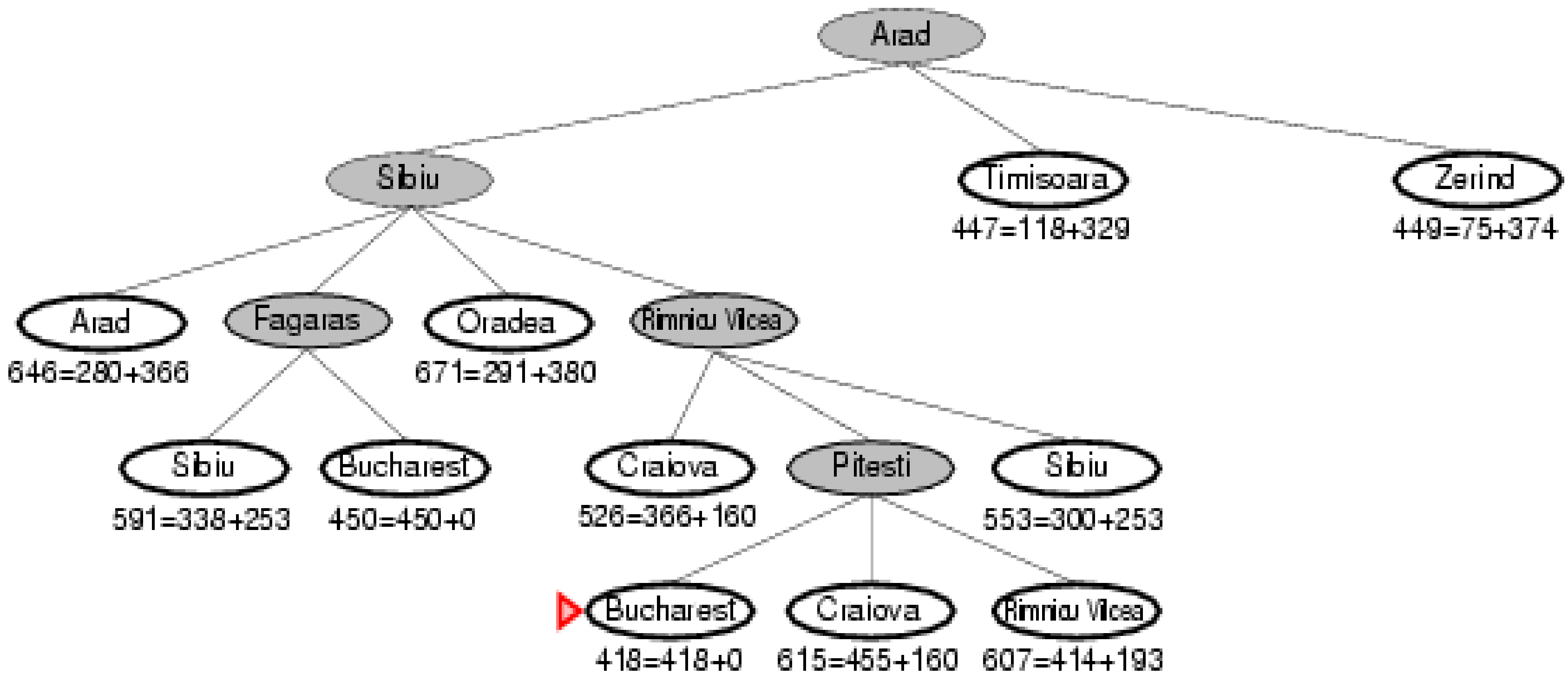
A* search example



A* search example



A* search example



http://aispace.org/search/

- We stop when the node with the lowest f-value is a goal state.
- Is this guaranteed to find the shortest path?

Admissible heuristics

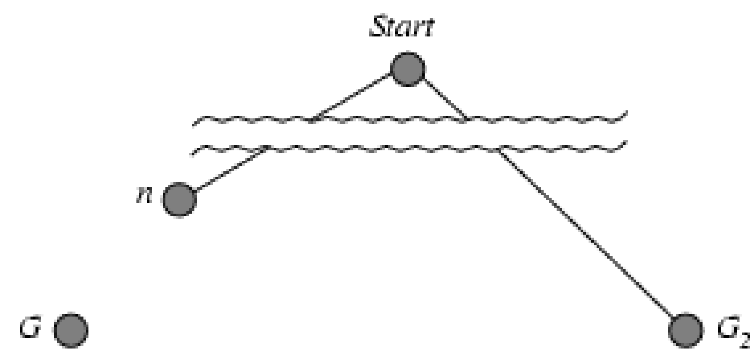
- A heuristic $h(n)$ is admissible if for every node n, $h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost to reach the goal state from n.
- An admissible heuristic never overestimates the cost to reach the goal, i.e., it is optimistic.

- Example: $h_{SLD}(n)$ (never overestimates the actual road distance)
- Negative Example: Fly heuristic: if wall is dark, then distance from exit is large.

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

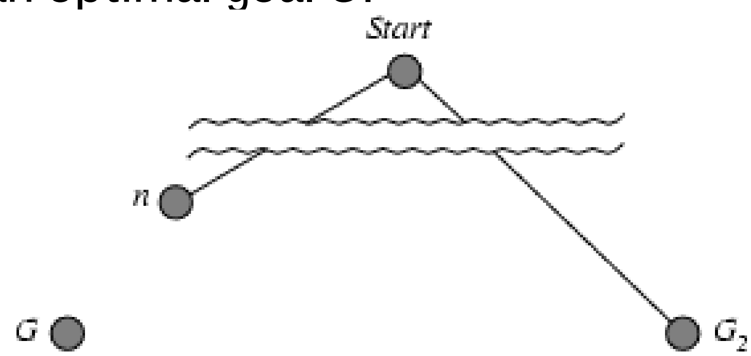
Optimality of A^* (proof)

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



- $f(G_2) = g(G_2)$ since $h(G_2) = 0$ because h is admissible
- $g(G_2) > g(G)$ since G_2 is suboptimal, cost of reaching G is less.
- $f(G) = g(G)$ since $h(G) = 0$
- $f(G_2) > f(G)$ from above

Optimality of A* (proof)

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

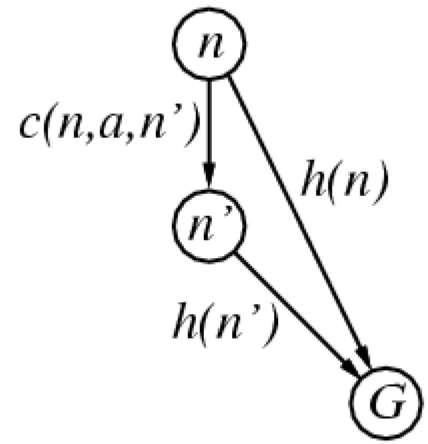
- f(G_2) > f(G) from above
- h(n) ≤ h*(n) since h is admissible, h* is minimal distance.
- g(n) + h(n) ≤ g(n) + h^*(n)
- f(n) ≤ f(G)

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*,
-

  $h(n) \le c(n,a,n') + h(n')$



- Intuition: can't do worse than going through n'.

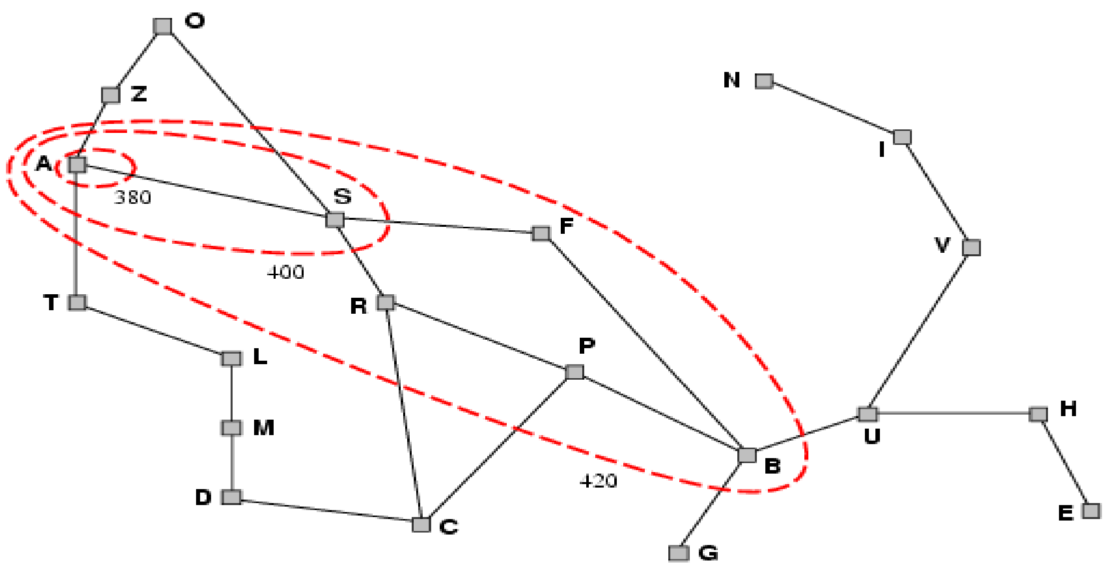
- If *h* is consistent, we have
-

f(n') = g(n') + h(n') = g(n) + c(n,a,n') + h(n')
≥ g(n) + h(n) = f(n)

- i.e.**, *f(n)* is non-decreasing along any path.**
-
- Theorem: If *h(n)* is consistent, A* using GRAPH-SEARCH is optimal
-

Optimality of A^*

- A^* expands nodes in order of increasing *f* value
-
- http://aispace.org/search/
- Gradually adds "*f*-contours" of nodes
- Contour *i* has all nodes with $f=f_i$, where $f_i < f_{i+1}$
-



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
-

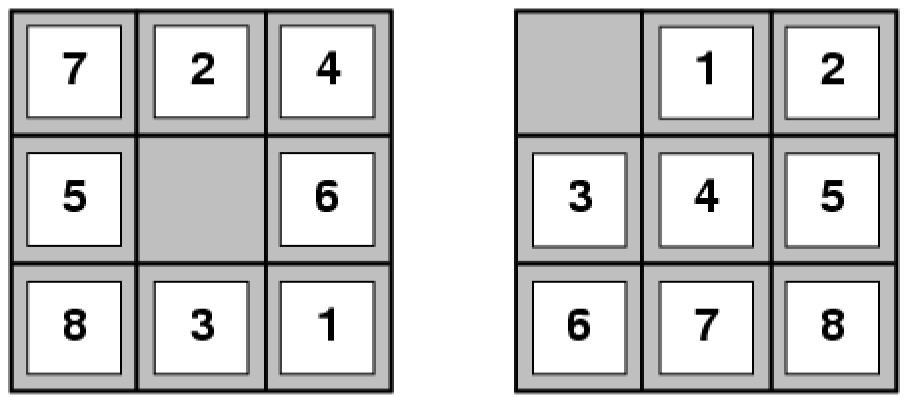
Admissible heuristics

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)

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



Start State Goal State

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)

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

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

Dominance

- 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
-

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
-

Summary

- Heuristic functions estimate costs of shortest paths
- Good heuristics can dramatically reduce search cost
- Greedy best-first search expands lowest h
  - incomplete and not always optimal
- A$*$ search expands lowest g + h
  - complete and optimal
  - also optimally efficient (up to tie-breaks)
- Admissible heuristics can be derived from exact solution of relaxed problems

Missionaries and Cannibals

- Old puzzle: has been around since 700 AD.
  Solved by Computer!
- Try it at home!
- Good for depth-first search: basically, linear solution path.
- Another view of **informed search**: we use so much domain knowledge and constraints that depth-first search suffices.
- The problem graph is larger than the problem statement.

✘ Taking the state graph as input seems problematic.