Informed Search strategies

AIMA sections 3.5, 3.6

Summary

- Greedy Best-First search
- A^* search
- Heuristics

Review: Tree search


```
function Tree-Search( problem, frontier) returns a solution, or failure
  frontier ← Insert(Make-Node(problem.Initial-State))
  loop do
    if frontier is empty then return failure
    node ← Pop(frontier)
    if problem.Goal-Test(node.State) then return node
    frontier ← InsertAll(Expand(node, problem))
  end loop
```

A strategy is defined by picking the **order of node expansion**

Best-First search

Idea: use an evaluation function for each node

– estimate of "desirability"

⇒ Expand most desirable unexpanded node

Implementation:

frontier is a queue sorted in decreasing order of desirability

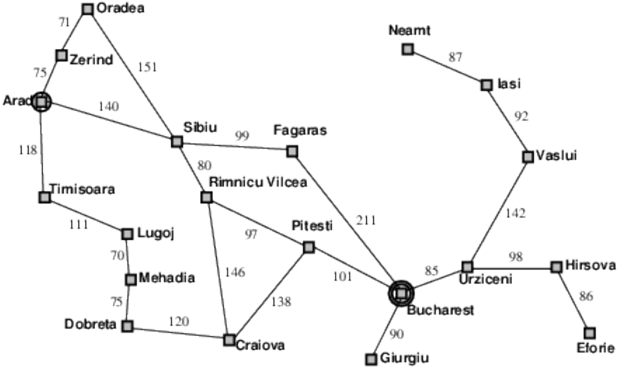
Special cases:

greedy best-first search

A^* search

Romania with straight-line distances to Bucharest

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



Greedy search

Evaluation function $h(n)$ (**h**euristic)

= estimate of cost from n to the closest goal

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

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

Greedy search example

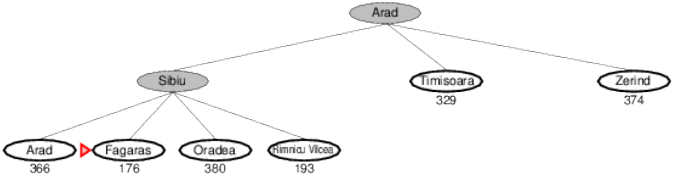
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Dobreta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

Greedy search example

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

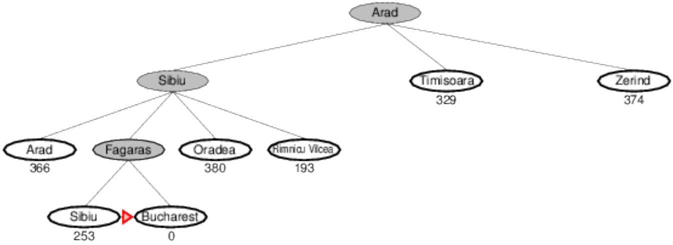
Greedy search example

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



Greedy search example

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



Properties of greedy search

Complete??

Properties of greedy search

Complete?? No–can get stuck in loops, e.g.,
Start: Iasi, Goal: Fagaras
Iasi $\rightarrow$ Neamt $\rightarrow$ Iasi $\rightarrow$ Neamt $\rightarrow \cdots$
Complete in finite space with repeated-state checking
Time??

Properties of greedy search

Complete?? No–can get stuck in loops, e.g.,
Start: Iasi, Goal: Fagaras
Iasi $\rightarrow$ Neamt $\rightarrow$ Iasi $\rightarrow$ Neamt $\rightarrow \cdots$
Complete in finite space with repeated-state checking
Time?? $O(b^m)$, but a good heuristic can give dramatic improvement
Space??

Properties of greedy search

Complete?? No–can get stuck in loops, e.g.,

Start: Iasi, Goal: Fagaras

Iasi $\rightarrow$ Neamt $\rightarrow$ Iasi $\rightarrow$ Neamt $\rightarrow \cdots$

Complete in finite space with repeated-state checking

Time?? $O(b^m)$, but a good heuristic can give dramatic improvement

Space?? $O(b^m)$—keeps all nodes in memory

Optimal??

Properties of greedy search

Complete?? No–can get stuck in loops, e.g.,
Start: Iasi, Goal: Fagaras
Iasi $\rightarrow$ Neamt $\rightarrow$ Iasi $\rightarrow$ Neamt $\rightarrow \cdots$
Complete in finite space with repeated-state checking
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 $f(n) = g(n) + h(n)$

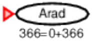
- $g(n)$ = cost so far to reach n
- $h(n)$ = estimated cost to goal from n
- $f(n)$ = estimated total cost of path through n to goal

◇ A* search uses an admissible heuristic
i.e., $h(n) \leq h^*(n)$ where $h^*(n)$ is the **true** cost from n.
(Also require $h(n) \geq 0$, so $h(G) = 0$ for any goal G.)

◇ E.g., $h_{\mathrm{SLD}}(n)$ never overestimates the actual road distance

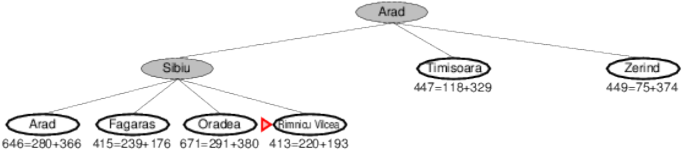
◇ Theorem: A* search is optimal

A* search example

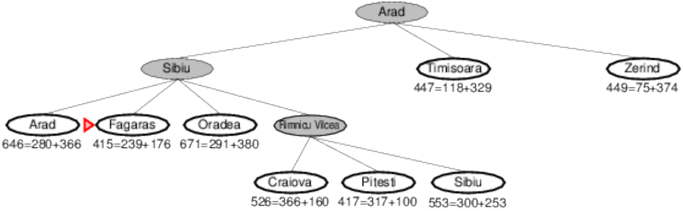


A^* search example

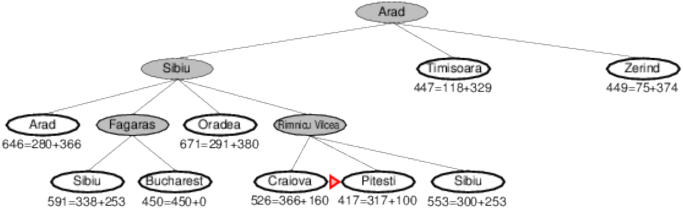
A* search example



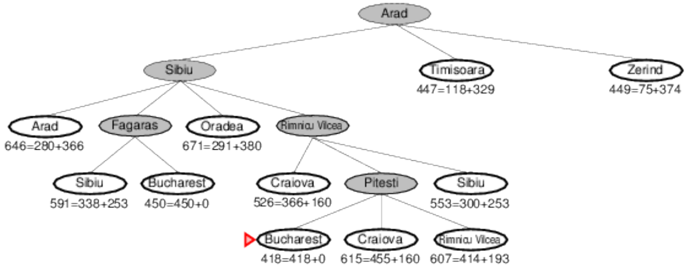
A* search example



A* search example

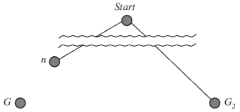


A* search example



Optimality of A* (standard proof)[1]

Suppose some suboptimal goal G_2 has been generated and is in the queue. Let n be an unexpanded node on a shortest path to an optimal goal G_1.



$$\begin{aligned} f(G_2) &= g(G_2) && \text{since } h(G_2) = 0 \\ &> g(G_1) && \text{since } G_2 \text{ is suboptimal} \\ &\geq f(n) && \text{since } h \text{ is admissible} \end{aligned}$$

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

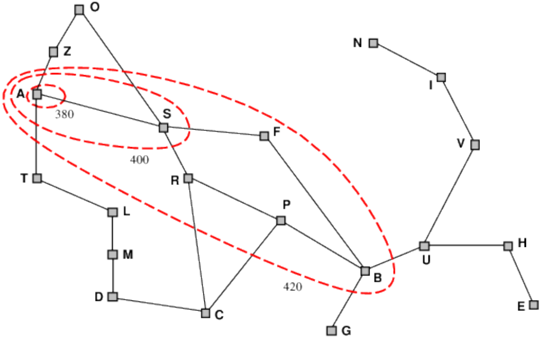
[1] Tree-Search + Admissible Heuristic

Optimality of A* (more useful)

Lemma: A* expands nodes in order of increasing f value[2]

Gradually adds "f-contours" of nodes (cf. breadth-first adds layers)

Contour i has all nodes with $f = f_i$, where $f_i < f_{i+1}$



[2] if heuristic is **consistent**

Properties of A*

Complete??

Properties of A*

Complete?? Yes, unless there are infinitely many nodes with $f \leq f(G)$

Time??

Properties of A*

Complete?? Yes, unless there are infinitely many nodes with $f \leq f(G)$

Time?? Exponential in [relative error in h $\times$ length of soln.]

Space??

Properties of A*

Complete?? Yes, unless there are infinitely many nodes with $f \leq f(G)$

Time?? Exponential in [relative error in h $\times$ length of soln.]

Space?? Keeps all nodes in memory

Optimal??

Properties of A*

Complete?? Yes, unless there are infinitely many nodes with $f \leq f(G)$
Time?? Exponential in [relative error in h $\times$ length of soln.]
Space?? Keeps all nodes in memory
Optimal?? Yes—cannot expand f_{i+1} until f_i is finished
A* expands all nodes with $f(n) < C^*$
A* expands some nodes with $f(n) = C^*$
A* expands no nodes with $f(n) > C^*$ $\rightarrow$ A* is **optimally efficient** (for a given heuristic)

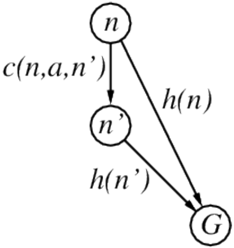
Proof of lemma: Consistency

A heuristic is consistent if

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

If h is consistent, we have

$$\begin{aligned} f(n') &= g(n') + h(n') \\ &= g(n) + c(n, a, n') + h(n') \\ &\geq g(n) + h(n) \\ &= f(n) \end{aligned}$$



I.e., $f(n)$ is nondecreasing along any path.

Admissible vs Consistent Heuristic

consistency $\rightarrow$ admissible
Can be proved by induction on the path to goal
admissible $\not\rightarrow$ consistency
Find a counter example...

Tree-Search + admissible Heuristic $\rightarrow$ optimality of A^*

Graph-Search + admissible Heuristic $\not\rightarrow$ optimality of A^*

- Can discard the optimal path to a repeated node

Graph-Search + **consistent** Heuristic $\rightarrow$ optimality of A^*

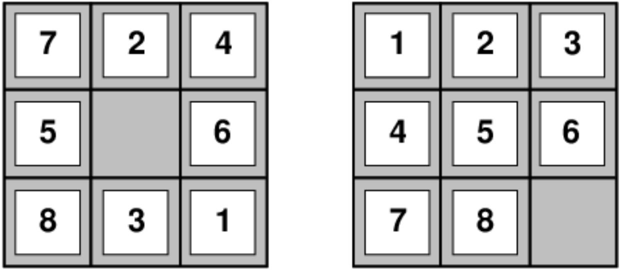
Admissible heuristics

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)



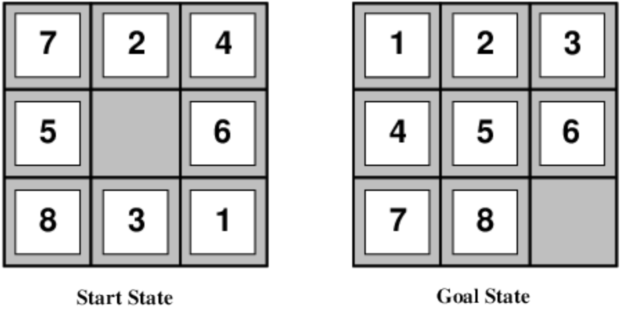
Start State Goal State

$h_1(S) = ??$

$h_2(S) = ??$

Admissible heuristics

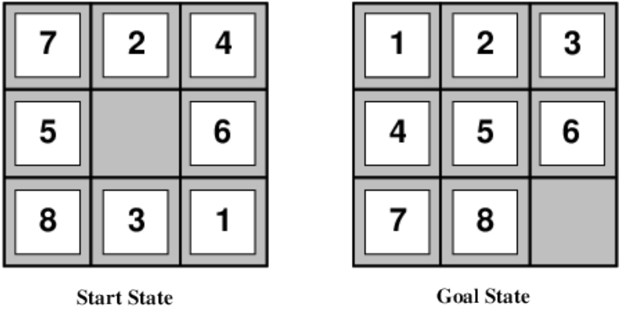
E.g., for the 8-puzzle:
$h_1(n)$ = number of misplaced tiles
$h_2(n)$ = total Manhattan distance
(i.e., no. of squares from desired location of each tile)



$h_1(S) =??$ 6
$h_2(S) =??$

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)



$h_1(S) =??$ 6
$h_2(S) =??$ 4+0+3+3+1+0+2+1 = 14

Dominance

If $h_2(n) \geq h_1(n)$ for all n (both admissible)
then h_2 dominates h_1 and is better for search
Typical search costs:

$d = 14$	IDS = 3,473,941 nodes
	$A^*(h_1)$ = 539 nodes
	$A^*(h_2)$ = 113 nodes
$d = 24$	IDS ≈ 54,000,000,000 nodes
	$A^*(h_1)$ = 39,135 nodes
	$A^*(h_2)$ = 1,641 nodes

Given any admissible heuristics h_a, h_b,

$$h(n) = \max(h_a(n), h_b(n))$$

is also admissible and dominates h_a, h_b

Relaxed problems

Admissible 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 $h_1(n)$ 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

Key point: the optimal solution cost of a relaxed problem is no greater than the optimal solution cost of the real problem

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, for forward search)

Admissible heuristics can be derived from exact solution of relaxed problems

Exercise: Going from Lugoj to Bucharest

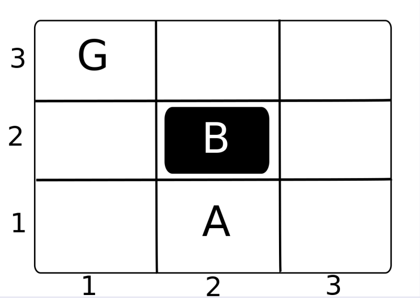
From Lugoj to Bucharest

♢ Trace the operation of A* search applied to the problem of going from Lugoj to Bucharest using the straight-line distance heuristic.

♢ Trace the operation of greedy best-first search applied to the problem of going from Lugoj to Bucharest using the straight-line distance heuristic.

Exercise: Navigation

Navigation with obstacles



The figure shows an artificial environment where an agent A is positioned in the square (1, 2)[a], the goal G is in (3, 1), and there is a block B in (2, 2). The agent can not pass through blocks and can move in the four directions (Up, Down, Left, Right).

[a] where the position is (row,column)

Exercise: Navigation II

Navigation with obstacles II

Formalize the problem of reaching G as a state problem

- Describe the state space, the initial and final state.
- Describe the operators.
- Find an admissible heuristics for A^*.
- Assume the operators have cost 1, draw the tree generated by A^*.

Exercise: confusing problems for greedy best-first search

confusing problems for greedy best-first search

When going from Iasi to Fagaras the straight-line distance heuristic result in poor performance for greedy best-first search. But from Fagaras to Iasi it is perfect. Are there problems for which the heuristic is misleading in both directions ?