Informed search algorithms

Chapter 4, Sections 1–2

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

Best-first search

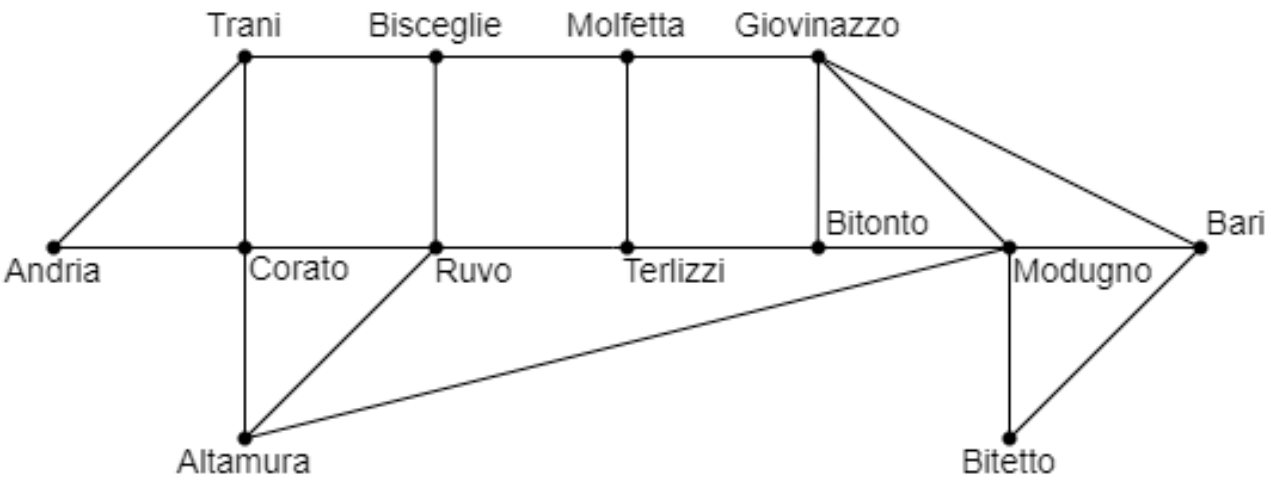
Idea: use an evaluation function for each node
– estimate of "desirability"

$\Rightarrow$ Expand most desirable unexpanded node

Implementation:
fringe is a queue sorted in decreasing order of desirability

Special cases:
greedy search
A* search

Example: Apulia



Cities coordinate

Andria: (41.2316, 16.2917)
Corato: (41.1465, 16.4147)
Altamura: (40.8302, 16.5545)
Ruvo: (41.1146, 16.4886)
Terlizzi: (41.1321, 16.5461)
Bisceglie: (41.243, 16.5052)
Trani: (41.2737, 16.4162)
Molfetta: (41.2012, 16.5983)
Giovinazzo: (41.1874,16.6682)
Bitonto: (41.1118, 16.6902)
Modugno: (41.0984, 16.7788)
Bari: (41.1187, 16.852)
Bitetto: (41.040, 16.748)

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

$\mathbf{A^*}$ 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.)

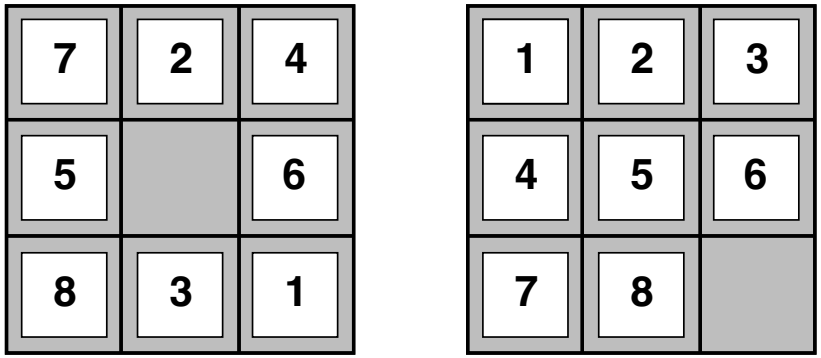
E.g., $h_{\mathrm{SLD}}(n)$ never overestimates the actual road distance

Theorem: A* search is optimal

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

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