Uniform cost search vs. A* graph search

- Essentially, we have replaced the path function $g(n)$ in Uniform cost search with the evaluation function $f(n)$.

Uniform cost search vs. A* graph search

- Essentially, we have replaced the path function $g(n)$ in Uniform cost search with the evaluation function $f(n)$.
- Can we be sure that when the goal state is popped out of the priority queue in A* graph search the function $f(n)$ will be minimized?

Conditions for non-decreasing $f(n)$ in A* search

1. Consistent heuristic:

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

Conditions for non-decreasing $f(n)$ in A* search

1. Consistent heuristic:

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

2. Consistent heuristic $\Rightarrow$ non-decreasing $f(n)$:

Let n' be a successor state of node n along a path being explored.

$$f(n') = g(n') + h(n')$$

Conditions for non-decreasing $f(n)$ in A* search

1. Consistent heuristic:

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

2. Consistent heuristic $\Rightarrow$ non-decreasing $f(n)$:

Let n' be a successor state of node n along a path being explored.

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

Conditions for non-decreasing $f(n)$ in A* search

1. Consistent heuristic:

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

2. Consistent heuristic $\Rightarrow$ non-decreasing $f(n)$:

Let n' be a successor state of node n along a path being explored.

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

Conditions for non-decreasing $f(n)$ in A* search

1. Consistent heuristic:

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

2. Consistent heuristic $\Rightarrow$ non-decreasing $f(n)$:

Let n' be a successor state of node n along a path being explored.

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

A* graph search optimality

- *For any node n, when n is popped out of the priority queue, we have found minimum value for $f(n)$.*
  Is the above sentence correct?

A* graph search optimality

- *For any node n, when n is popped out of the priority queue, we have found minimum value for $f(n)$.*
  Is the above sentence correct?
- *For any node n, when n is popped out of the priority queue, we have found minimum value for $g(n)$.*
  Is the above sentence correct?

A* graph search optimality

- *For any node n, when n is popped out of the priority queue, we have found minimum value for $f(n)$.*
  Is the above sentence correct?
- *For any node n, when n is popped out of the priority queue, we have found minimum value for $g(n)$.*
  Is the above sentence correct?
- Will A* graph search give minimum path length to goal state?

Properties of heuristic function

1. Consistent heuristic:

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

Properties of heuristic function

1. Consistent heuristic:

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

2. Admissibility
   - Consistency $\Rightarrow$ Admissibility

Is A* search better than Uniform cost search?

Is A* search better than Uniform cost search?

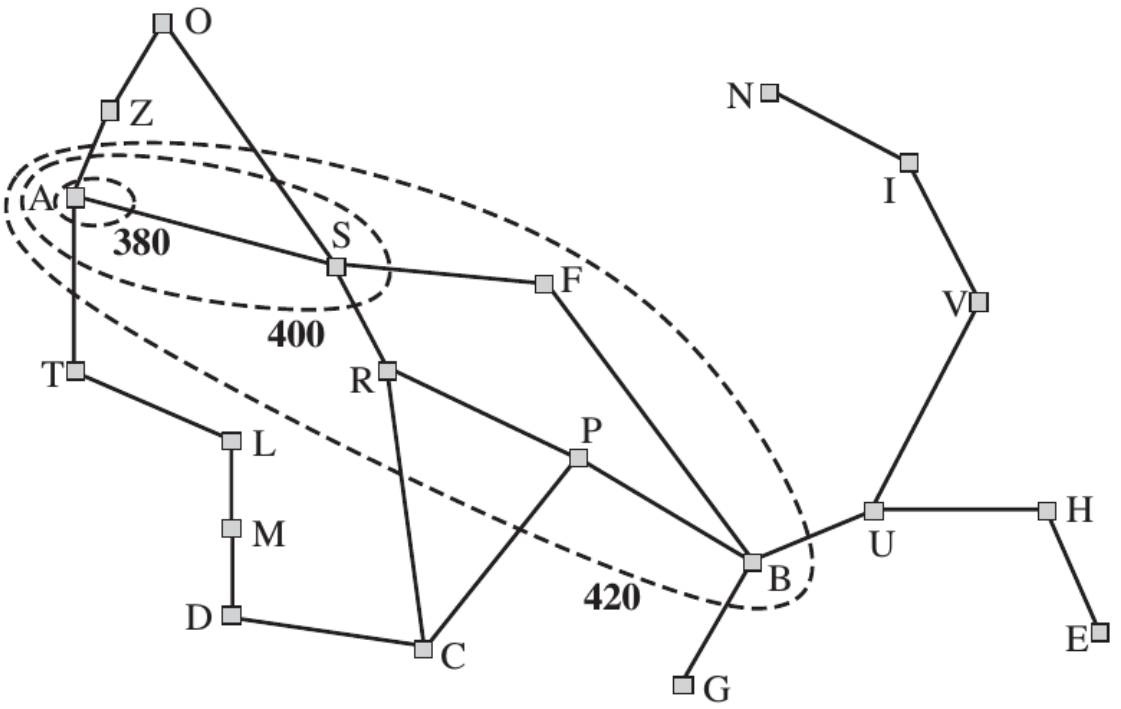


Figure 3.25 Map of Romania showing contours at $f = 380$, $f = 400$, and $f = 420$, with Arad as the start state. Nodes inside a given contour have f-costs less than or equal to the contour value.

A* graph search

- A* search is complete.

A* graph search

- A* search is complete.
- A* search is optimally efficient.

A* graph search

- A* search is complete.
- A* search is optimally efficient.
- In the worst case, A* search will expand all nodes with $f(n) \leq C^*$.
  (Time complexity analysis is difficult.)

A* graph search

- A* search is complete.
- A* search is optimally efficient.
- In the worst case, A* search will expand all nodes with $f(n) \leq C^*$.
  (Time complexity analysis is difficult.)
- Number of nodes generated is still exponential in d.

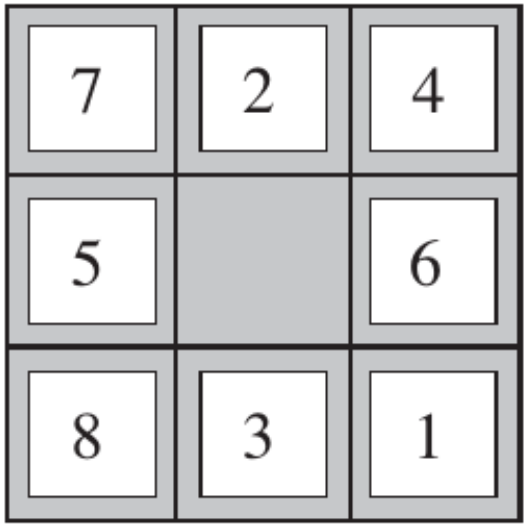
Further Improvements

- Iterative deepening A* (IDA*)

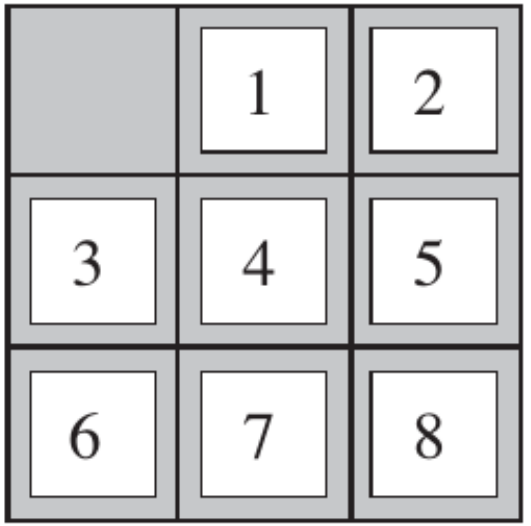
Further Improvements

- Iterative deepening A* (IDA*)
- Recursive best-first search (RBFS)
- Memory-bounded A* (MA*)

Heuristic Functions for 8-puzzle problem

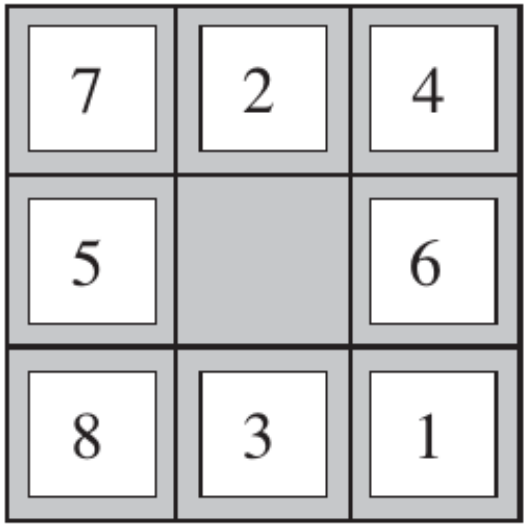


Start State

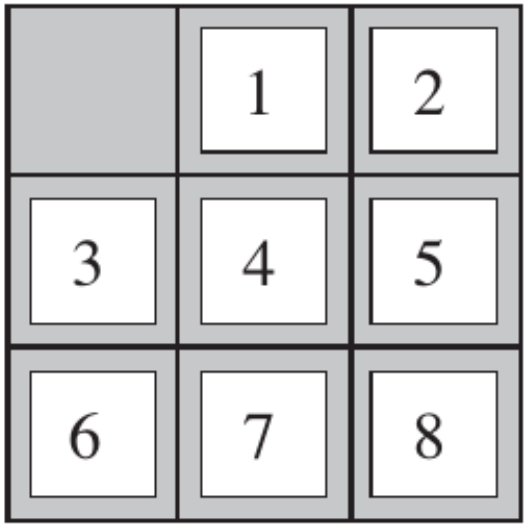


Goal State

Heuristic Functions for 8-puzzle problem



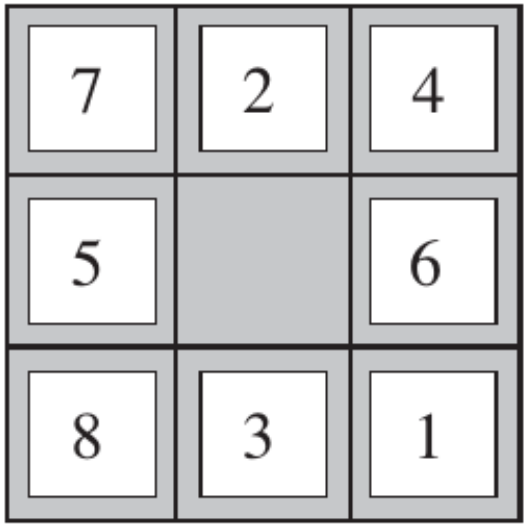
Start State



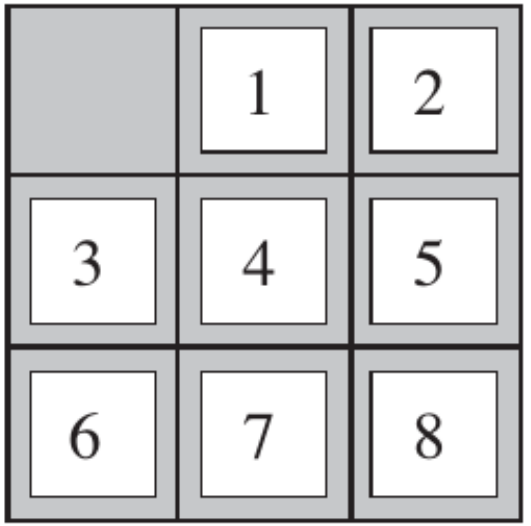
Goal State

1. Number of misplaced tiles (h_1)

Heuristic Functions for 8-puzzle problem



Start State



Goal State

1. Number of misplaced tiles (h_1)
2. Manhattan distance (h_2)