Informed Search

10S3001 – Artificial Intelligence

Samuel I. G. Situmeang

Faculty of Informatics and Electrical Engineering

Objectives

Students are able:

- to contrast greedy best-first search and A* search, and
- to apply greedy best-first search and A* search to solve a simple problem.

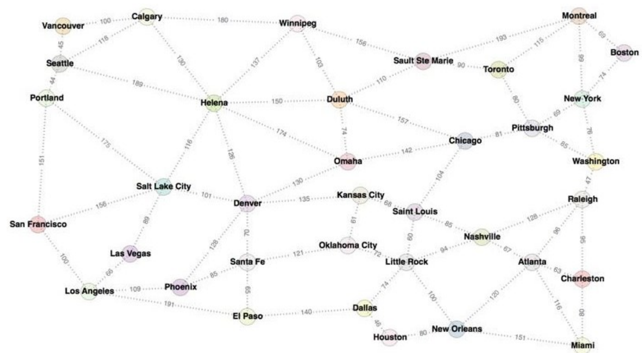
Siswa mampu:

- untuk membedakan *greedy best-first search* dan *A* search*, serta
- untuk menerapkan *greedy best-first search* dan *A* search* untuk menyelesaikan persoalan sederhana.

Informed Search

- **Use domain knowledge!**
  - Are we getting close to the goal?
  - Use a heuristic function that estimates how close a state is to the goal
  - A heuristic does NOT have to be perfect!
  - Example of strategies:
    1. Greedy Best-First Search
    2. A* Search
    3. IDA*

Informed Search



Atlanta	272
Boston	240
Calgary	334
Charleston	322
Chicago	107
Dallas	303
Denver	270
Duluth	110
El Paso	370
Helena	254
Houston	332
Kansas City	176
Las Vegas	418
Little Rock	240
Los Angeles	484
Miami	389
Montreal	193
Nashville	221
New Orleans	322
New York	195
Oklahoma City	237
Omaha	150
Phoenix	396
Pittsburgh	152
Portland	452
Raleigh	251
Saint Louis	180
Salt Lake City	344
San Francisco	499
Santa Fe	318
Sault Ste Marie	0
Seattle	434
Toronto	90
Vancouver	432
Washington	238
Winnipeg	156

The distance is the straight line distance. The goal is to get to Sault Ste. Marie, so all the distances are from each city to Sault Ste. Marie.

Heuristic!

Greedy Best-First Search

- Evaluation function $h(n)$ (heuristic)
- $h(n)$ estimates the cost from n to the closest goal
- Example: $h_{SLD}(n)$ = straight-line distance from n to Sault Ste. Marie
- Greedy search expands the node that **appears** to be closest to goal

Greedy Best-First Search

```
function GREEDY-BEST-FIRST-SEARCH(initialState, goalTest)
    returns SUCCESS or FAILURE :  /* Cost f(n) = h(n) */

    frontier = Heap.new(initialState)
    explored = Set.new()

    while not frontier.isEmpty():
        state = frontier.deleteMin()
        explored.add(state)

        if goalTest(state):
            return SUCCESS(state)

        for neighbor in state.neighbors():
            if neighbor not in frontier ∪ explored:
                frontier.insert(neighbor)
            else if neighbor in frontier:
                frontier.decreaseKey(neighbor)

    return FAILURE
```

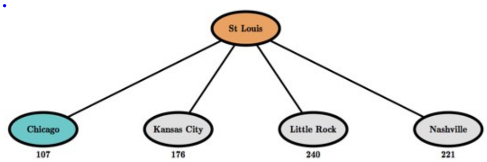
Greedy Best-First Search Example

The initial state:

Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

Greedy Best-First Search Example

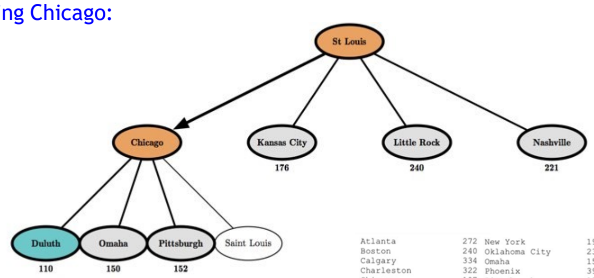
After expanding St Louis:



Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

Greedy Best-First Search Example

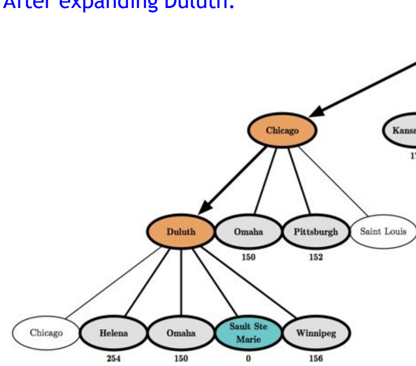
After expanding Chicago:



Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

Greedy Best-First Search Example

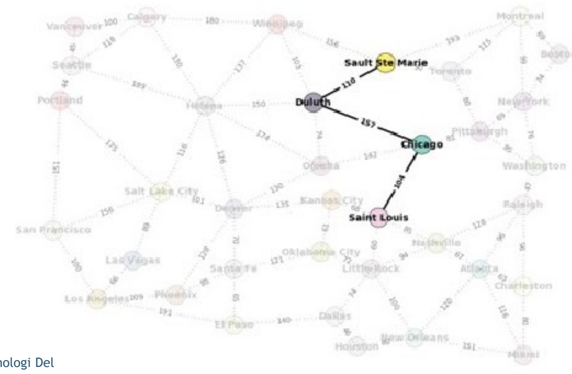
After expanding Duluth:



Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

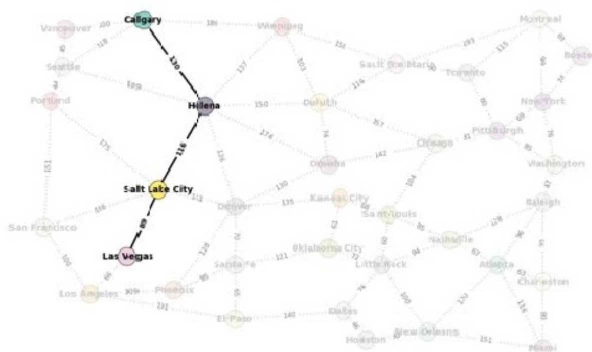
Greedy Best-First Search Examples Using The Map

Start: Saint Louis
Goal: Sault Ste Marie



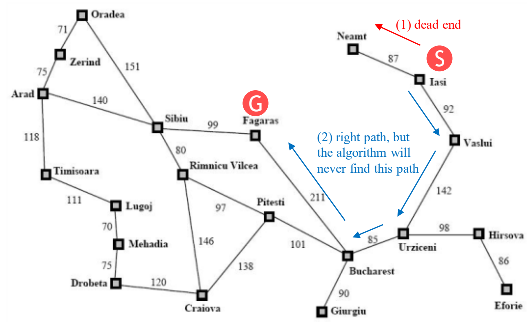
Greedy Best-First Search Examples Using The Map

Start: Las Vegas
Goal: Calgary



Greedy Best-First Search Criteria

- **Complete**: No
- **Time**: $O(b^m)$
- **Space**: $O(b^m)$
- **Optimal**: No



A* Search

- Minimize the total estimated solution cost
- Combines:
  - $g(n)$: cost to reach node n
  - $h(n)$: cost to get from n to the goal
  - $f(n) = g(n) + h(n)$

$f(n)$ is the estimated cost of the cheapest solution through n

A* Search

```
function A-STAR-SEARCH(initialState, goalTest)
    returns SUCCESS or FAILURE : /* Cost f(n) = g(n) + h(n) */

    frontier = Heap.new(initialState)
    explored = Set.new()

    while not frontier.isEmpty():
        state = frontier.deleteMin()
        explored.add(state)

        if goalTest(state):
            return SUCCESS(state)

        for neighbor in state.neighbors():
            if neighbor not in frontier ∪ explored:
                frontier.insert(neighbor)
            else if neighbor in frontier:
                frontier.decreaseKey(neighbor)

    return FAILURE
```

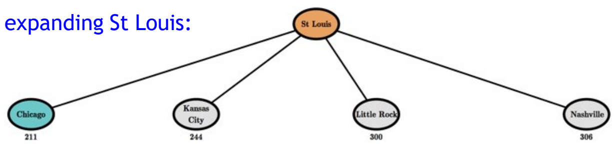
A* Search Example

The initial state:

Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

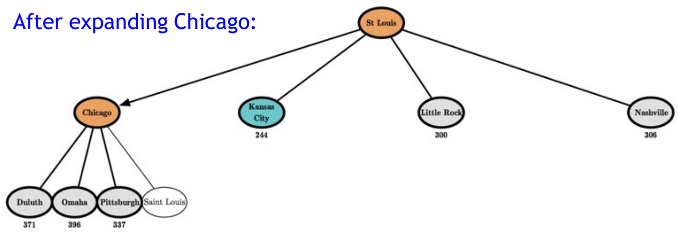
After expanding St Louis:



Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

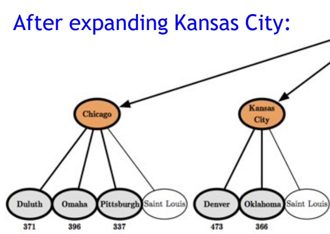
After expanding Chicago:

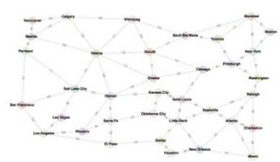


Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

After expanding Kansas City:

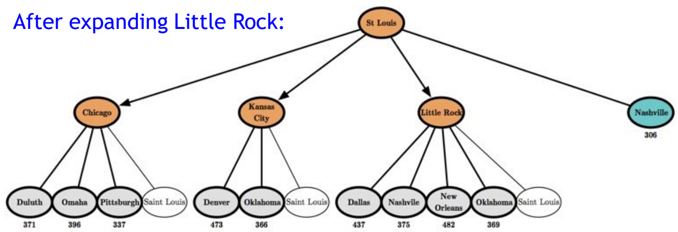


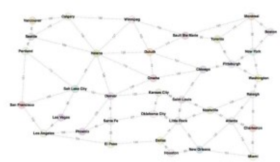


Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

After expanding Little Rock:

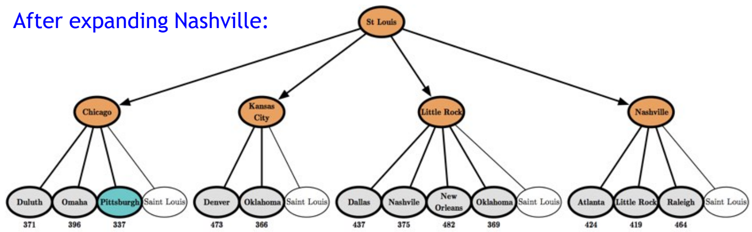


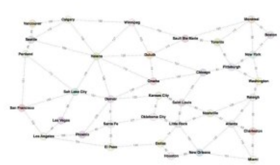


Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

After expanding Nashville:

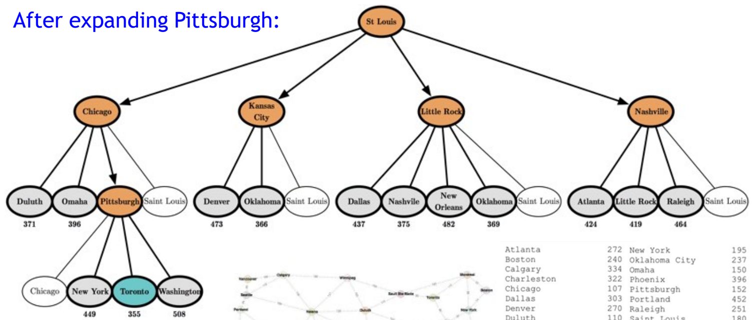




Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

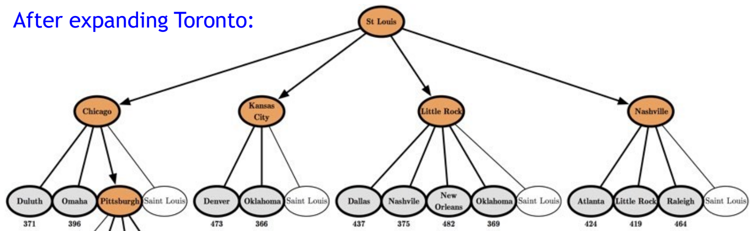
After expanding Pittsburgh:

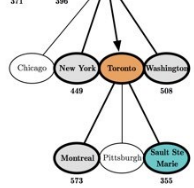


Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

A* Search Example

After expanding Toronto:

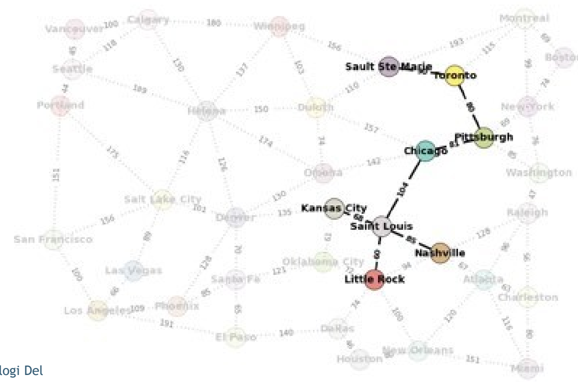




Atlanta	272	New York	195
Boston	240	Oklahoma City	237
Calgary	334	Omaha	150
Charleston	322	Phoenix	396
Chicago	107	Pittsburgh	152
Dallas	303	Portland	452
Denver	270	Raleigh	251
Duluth	110	Saint Louis	180
El Paso	370	Salt Lake City	344
Helena	254	San Francisco	499
Houston	332	Santa Fe	318
Kansas City	176	Sault Ste Marie	0
Las Vegas	418	Seattle	434
Little Rock	240	Toronto	90
Los Angeles	484	Vancouver	432
Miami	389	Washington	238
Montreal	193	Winnipeg	156
Nashville	221		
New Orleans	322		

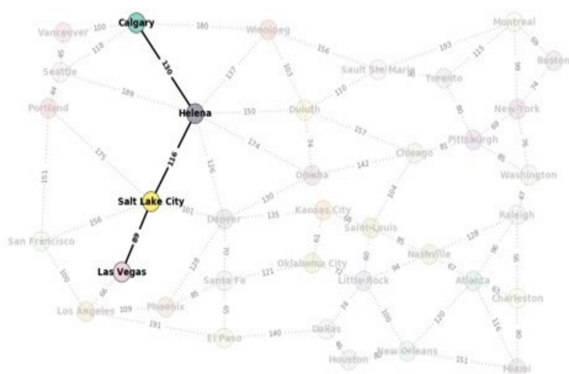
A* Search Examples Using The Map

Start: Saint Louis
Goal: Sault Ste Marie



A* Search Examples Using The Map

Start: Las Vegas
Goal: Calgary



Admissible Heuristics

A good heuristic can be powerful. Only if it is of a “good quality”.

A good heuristic must be admissible.

A* always find the optimal solution. Greedy Search sometimes hits the best solution but sometimes does not. So, A* does a pretty good job at finding the optimal solution. However, it must rely on a good heuristic because a good heuristic can turn out to be powerful one that can lead us to the optimal solution.

So, a good heuristic is known to be powerful if it has the property of being what we call admissible. What is it?

Admissible Heuristics

- An **admissible** heuristic never overestimates the cost to reach the goal, that is it is **optimistic**.
- A heuristic h is admissible if

$$\forall\, node\ \ n, h(n) \leq h^{*}(n)$$

where h^{*} is true cost to reach the goal from n.

- h_{SLD} (used as a heuristic in the map example) is admissible because it is by definition the shortest distance between two points.

Admissible heuristic is a heuristic that never overestimates the cost to reach the goal. That is it is optimistic, so it always has less than what it needs to reach the goal.

A heuristic h is admissible if for all node n, the value of the heuristic at the node n is actually bounded by the true cost to reach the goal from the node n.

So, in the example of the map, we use the heuristic h_{SLD} that actually is the straightest line between two cities. And this heuristic is admissible because it's always less than the true cost to reach the node.

This can be illustrated by a small example. If we have two cities A and B, the straightest distance between the two cities is actually not a real configuration. It's just an estimation of the distance between A and B. However, if you use a true path between A and B through roads, you would end up with some path that actually links the two cities, but whatever distance you take is going to be bigger than the straight line distance between A and B.

A* Search Criteria

- **Complete**: Yes
- **Time**: exponential
- **Space**: keeps every node in memory, the biggest problem
- **Optimal**: Yes!

Complete because if there is a solution, you will find it.
Unfortunately, the time is still exponential, and the space is still exponential. The main reason is that it keeps all the nodes at memory to find the best possible solution of the optimal solution, which is the biggest problem. And as in the previous example, there actually lots of nodes were grey because they are all in the fringe waiting to be explored. So, there is an important use of memory to solve the problem.
It is optimal. If there is a solution, we find that it finds the best possible solution, which is a solution with the least cost in your search.

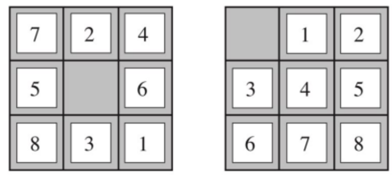
A* Optimality

If $h(n)$ is admissible, A* using tree search is optimal.

- **Rationale**:
  - Suppose G_o is the optimal goal.
  - Suppose G_s is some suboptimal goal.
  - Suppose n is an unexpanded node in the fringe such that n is on the shortest path to G_o.
  - $f(G_s) = g(G_s)$ since $h(G_s) = 0$
    $f(G_o) = g(G_o)$ since $h(G_o) = 0$
    $f(G_s) > g(G_o)$ since G_s is suboptimal
    Then $f(G_s) > f(G_o)$ . . . (1)
  - $h(n) \leq h^*(n)$ since h is admissible
    $g(n) + h(n) \leq g(n) + h^*(n) = g(G_o) = f(G_o)$
    Then, $f(n) \leq f(G_o)$ . . . (2)
    From (1) and (2) $f(G_s) > f(n)$
    so A* will never select G_s during the search and hence A* is optimal.

A* will always select the node that will lead to the optimal goal.

Heuristics



Start State Goal State

- The solution is 26 steps long.
- $h_1(n)$ = number of misplaced tiles
- $h_2(n)$ = total Manhattan distance (sum of the horizontal and vertical distances).
- $h_1(n) = 8$
- Tiles 1 to 8 in the start state gives: $h_2(n) = 3 + 1 + 2 + 2 + 2 + 3 + 3 + 2 = 18$ which does not overestimate the true solution.

Last note about heuristics. We have seen heuristics in the context of maps. Straight-line distance turned out to be a good heuristic when you explore the map. In other contexts, you may need to find other kinds of heuristic.
26 steps is the true cost/optimal cost, to reach the goal states starting from the start state.
Both h_1 and h_2 never overestimate the true cost to find the solution. I prefer h_2 because it's closer as a number to the real total number of steps.

References

- S. J. Russell and P. Borvig, *Artificial Intelligence: A Modern Approach (4th Edition)*, Prentice Hall International, 2020.

eof