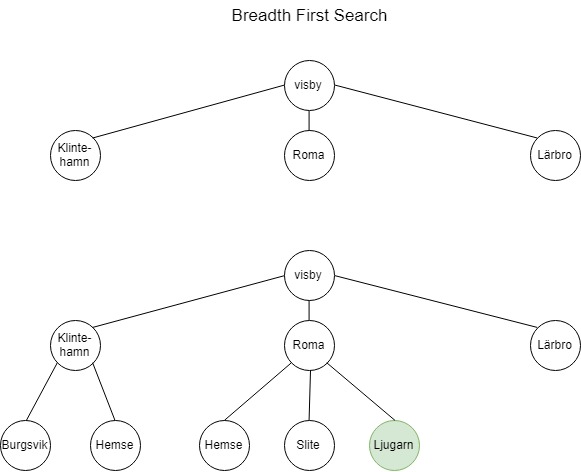
**Examining Three Search Algorithms**

We have a road map of Gotland where we want to find the shortest path from Visby to Ljugarn. The roads distance is represented with the numbers next to the roads. We chose to examine Breadth First Search, Uniform-cost Search and A\* Search.

**Breadth First Search**

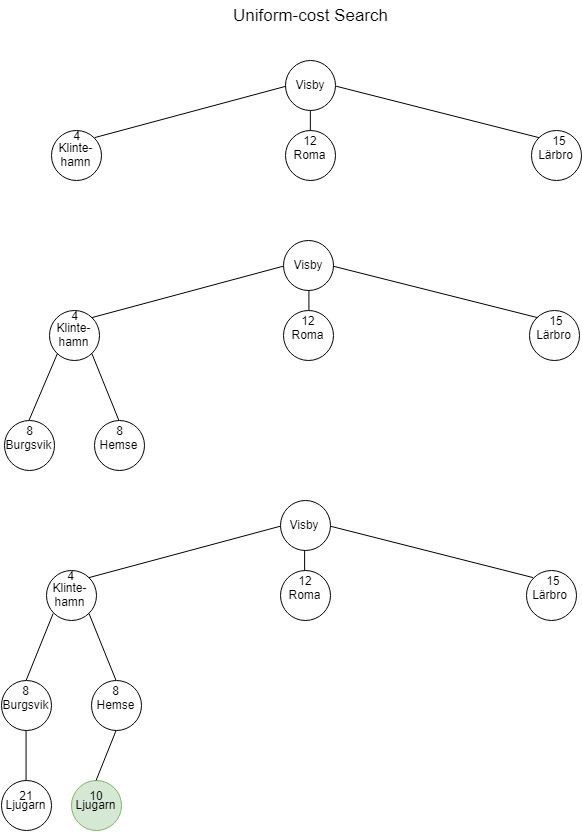
Breadth first search will return the path with the least amount of towns passed. BFS will return the “shallowest” nod with the goal state and does not take the distance from one town to another into account. The cost of the actions in our world are not the same for all, this means that BFS could chose the path with the least amount of nodes in them but with the longest distance and will not be optimal.



**Memory for Breadth First Search**

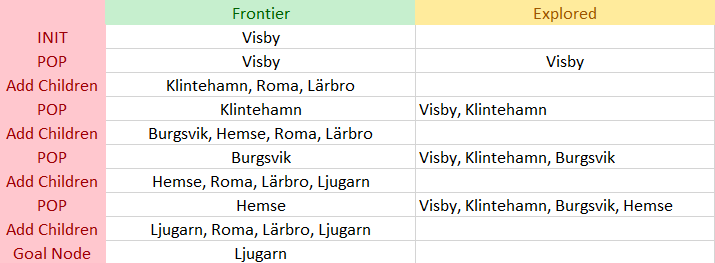


The path will be: Visby -> Roma -> Ljugarn with a total cost of 15.

**Uniform-cost Search**

Uniform-cost Search takes the distance into account while choosing what path to expand. It uses a priority based queue that is based on the lowest path cost from the root node. Because the cost of the actions are not the same for all, Uniform-cost Search is a better fit for this kind of problem comparing to BFS.

**Memory for Uniform-cost Search**



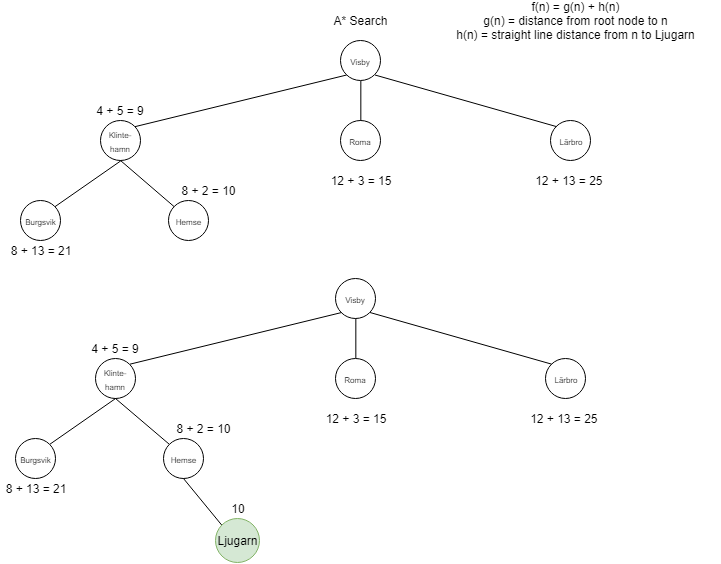
The path will be: Visby -> Klintehamn -> Ljugarn with a total path cost of 10. Compared to BFS, UCS is more effective in this example.

**A\* Search**

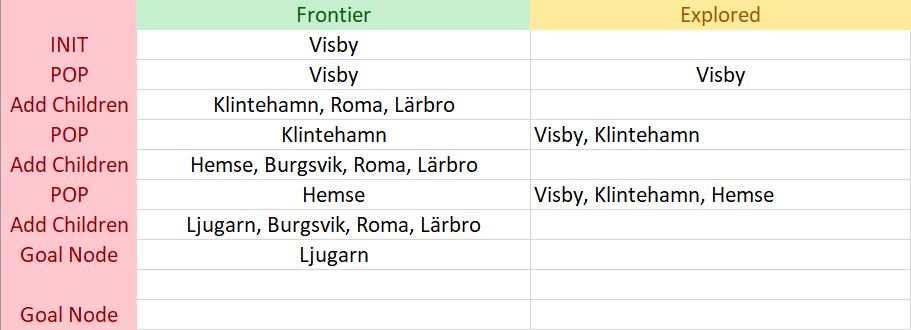
A\* Search takes the distance and an estimated distance to the goal into account while choosing what path to expand. It uses a priority-based queue that is ordered on f(n).

*f(n) = g(n) + h(n)* where *g(n) = the distance from root node to n, h(n) is the straight line distance from node n to goal node.*

The search will continue until a goal node has a lower f(n) than any other node in the priority queue.

We chose to use the straight line distance from node n to goal node because it is admissible and will not overestimate the cost.

**Memory for A\* Search**

****

The path will be: Visby -> Klintehamn -> Ljugarn with a total path cost of 10. A\* chose the same path as Uniform-cost Search but it found the path in fewer iterations. It did not have to expand Burgsvik and therefore it is more efficient.