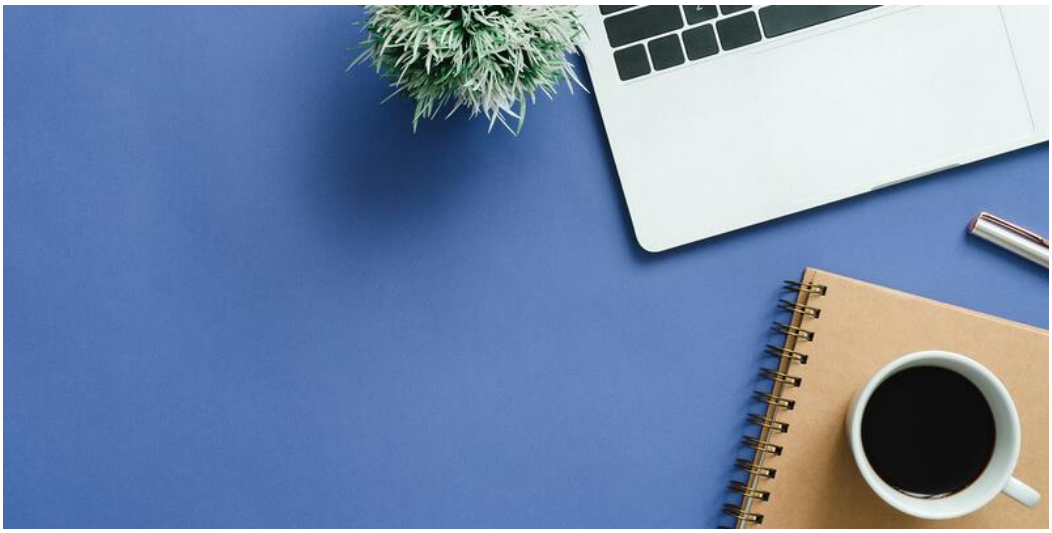
Artificial and Computational Intelligence

Assignment 9

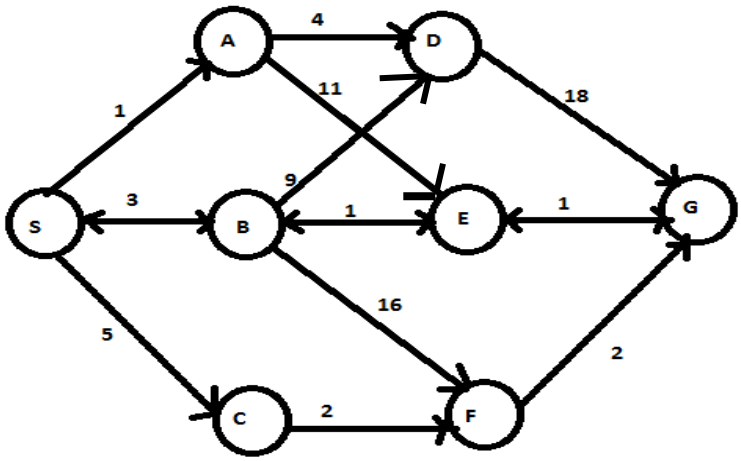
Group 112

CONTENTS



PROBLEM STATEMENT

Monk visits the land of Islands. There are a total of **8** islands numbered from S to **G**. Some pairs of islands are connected to each other by **Bidirectional** bridges running over water. Monk hates to cross these bridges as they require a lot of efforts. He is standing at Island #S and wants to reach the Island #G. Find the minimum the number of bridges that he shall have to cross, if he takes the optimal route.



Note: As per instructor clarification in discussion forum, we have updated the graph. "*Kindly consider A-E and B-D as uni-directional edge. A---->E, B------>D*"

ALGORITHM SELECTION

For given graph, we selected Weighted A* Algorithm. Reason

1. It is better fit "informed search" for graph traversal and path search problems.
2. It is complete (for finite nodes), optimal (depending on heuristics) with exponential time complexity. However not efficient in space parameters. As we have finite nodes and smaller graph, we thought of using A* Algorithm for given problem statement.
3. We initially implemented A-star algorithm, but this was not fulfilling the additional requirement of monk disliking of bidirectional bridges. For this reason, we implemented weighted A-star algorithm to add weights so that the algorithm does not take bi-direction bridges due to additional weights for bi-directional bridges.
4. The value for weights was adjusted after some trial and error.

DATA STRUCTURE

For graph representation, we have used Adjacency List with Edges and cost

For Heuristics & Weights, we have used HashMap with node and corresponding values.

For Open & Close list we have used set, this can be optimized further using priority queue.

For storing the g(n) we are using map

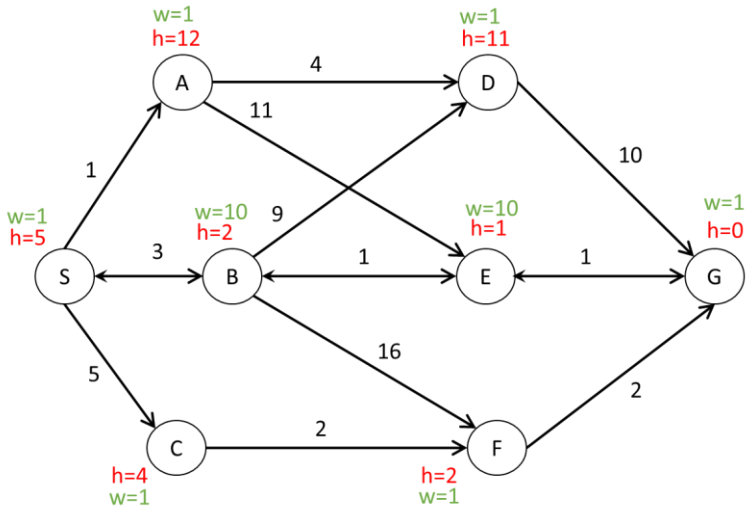
COST FUNCTION

Cost function for Weighted A* Algorithm f(n) = g(n) + w(n) * h(n), where

- n is next node to be explored,
- g(n) is the cost of the path from start node to n,
- w(n) is the weight associated with node,

- h(n) is the heuristic function that estimates of the cheapest path from n to the goal.

Heuristic Function must be **Admissible**, h(n) ≤ h*(n), h*(n) is optimal cost to reach goal from node n, h(n) is the heuristic / indicated cost to reach goal from node 'n'. It should never overestimate the actual cost to get to the goal, this will guarantee that A* will always return optimal path from start to goal.



Admissible Heuristic h(n) < h*(n),

For the given graph, we set the h(n) = h*(n) i.e., optimal value to reach goal state "G" from node 'n'.

To avoid the bi-directional bridges, we set the w(n) = 10 and for unidirectional bridge we keep weight as 1. The value for weights was adjusted after few trial and error. We started with weights = 2 for bi-directional bridges node, and finally for weight=10 we got the algorithm taking alternate route (avoid bi-directional bridges / nodes)

PYTHON IMPLEMENTATION

group112-aci-assignment-9-astar-weig

group112-aci-assignment-9-astar-weig

Output Path

```
Path: ['S', 'C', 'F', 'G']
```

Output Cost

```
Path: Length=3 , Cost=9
```

CONCLUSION

We first attempted to implement the problem solution using A-star (without weights) algorithm, we got following path which is shortest, but monk had to cross bi-directional bridge.

```
Path: ['S', 'B', 'E', 'G']
Path: Length=3 , Cost=5
```

By improving algorithm - adding + adjusting weights, we get the path which avoids the bi-directional bridges / path (fulfils the requirement), resulting in following path.

```
Path: ['S', 'C', 'F', 'G']
Path: Length=3 , Cost=9
```