Artificial Intelligence Lab
Assignment 3

Tapan Meena
160010039

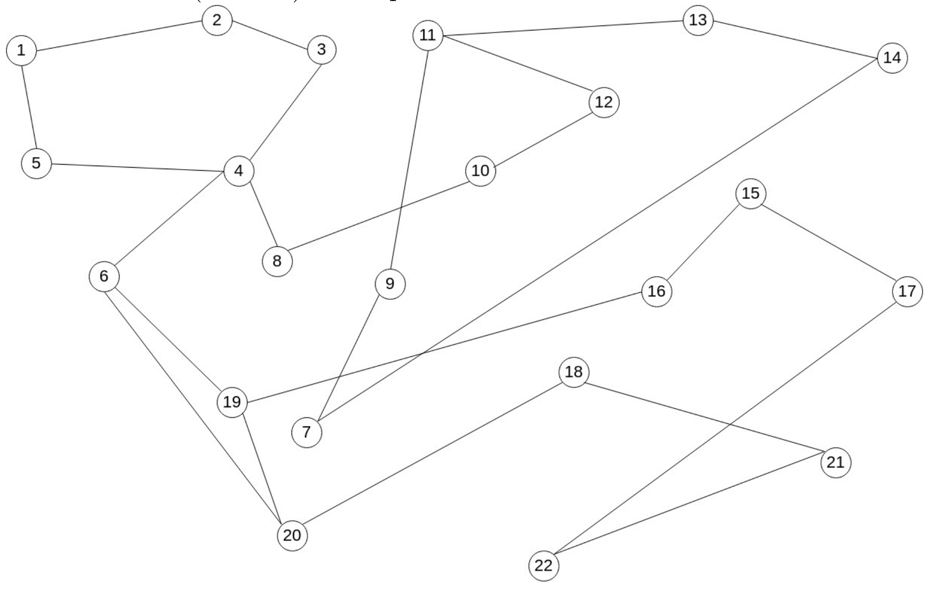
Arpit Shukla
160010012

27 January 2019

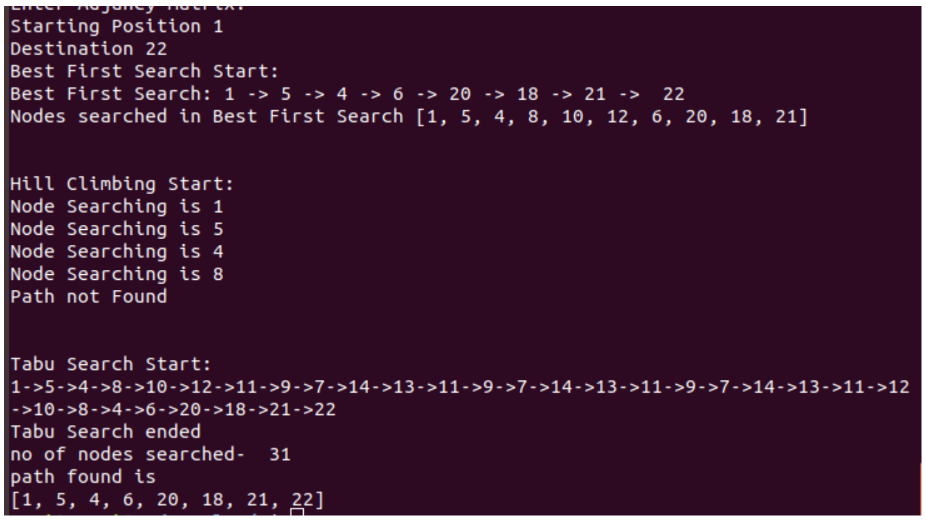
Algorithm	Nodes Visited	Height Traversed
Best FS	10	8
Hill Climb	4	Path Not Found
Tabu Search	31	8

1 Heuristic Function:-

As our domain is travelling of passengers from one station to another station by the means of rail transport. Given Start state as the Boarding Station, the Goal state as the Destination Station and the link between available stations, we have to find the viable route between them(if achievable). So, we have used the euclidean distance of the node from the goal node given the co-ordinates of the nodes(stations). Example of our Rail Network looks like this:-



The output for Starting Station as (1) and Destination Station as (20)



```
Enter Adjacency Matrix:
Starting Position 1
Destination 22
Best First Search Start:
Best First Search: 1 -> 5 -> 4 -> 6 -> 20 -> 18 -> 21 ->  22
Nodes searched in Best First Search [1, 5, 4, 8, 10, 12, 6, 20, 18, 21]


Hill Climbing Start:
Node Searching is 1
Node Searching is 5
Node Searching is 4
Node Searching is 8
Path not Found


Tabu Search Start:
1->5->4->8->10->12->11->9->7->14->13->11->9->7->14->13->11->9->7->14->13->11->12
->10->8->4->6->20->18->21->22
Tabu Search ended
no of nodes searched-  31
path found is
[1, 5, 4, 6, 20, 18, 21, 22]
```

The above Output shows the traced path of Tabu Search from Start State to Goal State. In our Tabu Search Algorithm, We had set the Tabu Tenure to 4. We only store last 4 nodes visited in tabu nodes and increase the frequency of current node, then generate the negibours of current node. if frequency of neighbour is 4 then remove it from negihbour list. If there are no nodes to go to i.e all nodes visited for more than 3 times then that means it is stuck in loop and we end the Tabu Search and print **Tabu Search Failed**.We also find eVal value for all valid neighbours and get minimum of eVal value and use it.
Below is pseudo code for Tabu Search, for code refer to python code attached with it.

```
def tabusearch(mat,cordinatesList,curr,dest):
    currentBest = curr
    bestCandidate = curr
    tabuList = []
    tabuList.append(curr)
    while (no Negihbours to visit)
      neighbours = MoveGen(bestCandidate)
      for i in neighbours
        if ((not tabuList.contains(i)) and (heuristic(i) < heuristic(bestCandidate)))
          bestCandidate = i
        end
      end
      if (heuristic(bestCandidate) < heuristic(currentBest))
        currentBest = bestCandidate
      end
      tabuList.append(bestCandidate)
      if (tabuList.size > maxTabuSize)
        tabuList.front()
      end
    end
    return
```