Date __/__/__

Page ____

LAB - 6

A* Search Algorithm for 8 queens

```
def heuristic (state)
    row= sum
      if row>=8:
          return 0

    free_space=0
      for col in range(8):
          if   safe ( state, row, col):
                  free_space = free_space+1;
          return -free_space

        current
def  safe (state, row, col):
          for  r  in  range (row ):
                 c = state [r]
                 if c== col  or  (c - col) = (r - row):
                      return False
          return True
```

1) Check if queen exists in same column
2) check if queen is in same diagonal (top left
bottom right
3) check if queen is in bottom left to top right

1. Initialize by placing one queen per column, along with initialzing priority queue and heuristic

2. Dequeue the the node with lowest f value (the one with least conflicts)

3. Generate successors by moving each queen to any row within that column
Calculate f value (g + h) for every step by calculating number of conflicts

4. Push the successors into the priority queue and add f = g+h into the priority queue
g = cost to reach

5. Continuous expansion & exploring successors until solution is found.

				Q			
Q							
							Q
					Q		
		Q					
						Q	
	Q						
			Q				

[1, 6, 4, 7, 0, 3, 5, 2]

Hill climb Search Algorithm

Place one queen in each column
Count the number of conflicts between two queens
If heuristic is 0, solution is found
Starting with current solution and its conflicts
generate neighbors by moving each queen to different rows in its column
select neighbor with least conflicts, and lowest heuristic
If no neighbor improves the current state, the program terminates.

		Q					
					Q		
	Q						
				Q			
							Q
Q							
						Q	
			Q				

[5, 2, 0, 7, 3, 1, 6, 4]