Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Experiment # 5: Implement stochastic Hill climbing Algorithm for R... path finding

Aim/Objective:

Implementing a stochastic Hill climbing Algorithm for Robot path finding

Description:

Students will learn the Stochastic Hill Climbing (SHC) which is a local search algorithm commonly... in optimization problems, including pathfinding for robots in dynamic environments.

Pre-Requisites:

- Students 'll be well-equipped to understand, implement, and customize the Stochastic Hi... Climbing algorithm for robot pathfinding effectively.
- Familiarity with programming concepts and basic knowledge of Python programming language

Pre-Lab:

1. What is Stochastic Hill Climbing?

Stochastic Hill climbing is an optimization technique. where instead of always choosing the best nearby option the algorithm random. picks a neighbour. It can help avoid getting stuck in local peaks by introducing randomness into search.

2. How does Stochastic Hill Climbing differ from regular Hill Climbing?

Stochastic Hill Climbing differs from regular hill climbing in that is randomly selecting a neighbouring. Solution instead of always picking the best one. This randomness can help avoid getting stuck in loop

Course Title	Artificial Intelligence and Machine Learning	ACADEMIC YEAR: 2024-
Course Code(s)	23AD2001O	Page

Experiment #	<TO BE FILLED BY STUDE
Date	<TO BE FILLED BY STUDE

3. What are the advantages of using S...

Stochastic hi... for pathfinding b... getting stuck in... introducing ran... to explore ...

4. What are the key components ... pathfinding?

1. Initial Point

2. Step Size

3. Random...

4. Sideway...

5. Avoid ...

Course Title	Artificial In...
Course Code(s)	23AD2001...

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

3. What are the advantages of using Stochastic Hill Climbing for robot pathfinding?

Stochastic hill climbing is useful for robot for pathfinding because it helps the robot avoid getting stuck in sub optimal path by introducing randomness. This allows the robot to explore more potential routes.

4. What are the key components required to implement Stochastic Hill Climbing for robot pathfinding?

1. Initial Point: Start with an initial point in the search point
2. Step Size: Define a step size, which represents the distance within the search space.
3. Randomness: Generate a new point within the step size distance of the initial point
4. Sideways Moves: Allow exploration across flat regions by accepting different points
5. Avoid Infinite loops: Limit consecutive sideways moves to prevent infinite loops

Experiment #	<TO BE
Date	<TO BE

In-Lab:

A robot is located in a grid of size nXm. The robot can move up, down, left, or right, but ca move through obstacles. The goal is to find a path from the start position to the goal positio using a stochastic hill climbing algorithm.

Stochastic Hill Climbing:

1. **Initial State**: Start at the initial position.
2. **Neighbor Selection**: Randomly choose one of the neighboring states.
3. **Evaluation**: Move to the chosen neighboring state if it improves the evaluation function.
4. **Termination**: Stop when the goal is reached or no better neighbors are found.

Evaluation Function

The evaluation function for the hill climbing algorithm can be the Manhattan distance to the goal: Manhattan Distance=|x1−x2|+|y1−y2| where (x1,y1)(x2,y2) are the coordinates of the current position and are the coordinates of the goal.

Procedure/Program:

```
import random
def distance(a,b):
    return abs(a[0]-b[0])+ abs(a[1]+b[1])

def stochastic_hill(start, goal, n,m, obstacles):
    current = start
    while current != goal:
        neighbors = [(current[0]+ curnew[1]]
for dx, dy in [(-1,0),(0,1),(0,-1),(1,0)]
neighbors = [pos for pos in neighbors if
    0 <= pos[0] < n and 0 <= pos[1] and
        pos not in obstacles].
```

- Data and Results
- Analysis and

Course Title	Artificial Intelligence and Machine Learning	ACADEMIC YEAR: 2024
Course Code(s)	23AD2001O	Page

Course Title	
Course Code(s)	

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

```
next_move = random.choice(neighbors)
if distance(next_move, goal) < distance(current, goal):
    current = next_move
else:
    break
return current
```

- **Data and Results:**

The output of the above program tells that the robot has reached the goal position (4,4) by using this technique.

- **Analysis and Inferences:**

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

The algorithm efficiently navigates the grid by moving towards the goal using randomness to avoid local minima.

It helps to stop eagerly if no better moves are available, leading to suboptimal paths.

This random restarts can improve the performance in complex grids.

VIVA-VOCE Questions (In-Lab):

1. What is the stochastic hill climbing algorithm, and how does it differ from deterministic climbing?
2. How do you ensure that the robot finds an optimal path using the stochastic hill climb algorithm?
3. What are the advantages and potential drawbacks of using stochastic hill climbing for pathfind in a robotic environment?
4. How do you handle local maxima, plateaus, and ridges in the context of the hill climb algorithm?
5. Can you explain a scenario where stochastic hill climbing would be more effective than ot pathfinding algorithms?

1. Stochastic Hill climbing randomly picks a neighbour to move to if its better unlike deterministic Hill Climbing which always chooses the best neighbour.

Course Title	Artificial Intelligence and Machine Learning	ACADEMIC YEAR: 2024-2
Course Code(s)	23AD2001O	Page 3

Experiment #	<TO BE FILLED BY ST
Date	<TO BE FILLED BY ST

2) Ensure optimal
annealing and he
locals

3) Advantage: St
Drawbacks: Slo
be

4) Handle local m
random restart
head techniqu

5) Stochastic Hill
obstacle filled e
methods might

Post-Lab:

Place N queens on an N×N chessboard
a random configuration and use stocha
reduces the number of conflicts.

Description: Stochastic hill climbing is a
problems. It introduces randomness into
maxima. In this problem move a queen to

- **Procedure/Program:**

import random
def conflicts(boa
return su

Course Title	Artificial Intelligence a
Course Code(s)	23AD2001O

2) Ensure optimal pathfinding by using random restarts annealing and heuristics to guide move and avoid locals

3) Advantage: Simple avoids local optima.
Drawbacks: Slower, possible, inefficiency, may miss best path.

4) Handle local maxima plateaus and ridges with random restarts. small perturbations and look a head techniques

5) Stochastic Hill climbing excels in complex, abstract obstacle filled environments wher determine methods might fail.

Post-Lab:

Place N queens on an N×N chessboard so that no two queens threaten each other by starting with a random configuration and use stochastic hill climbing to move a queen to a new position if it reduces the number of conflicts.

Description: Stochastic hill climbing is a variant of the hill climbing algorithm used in optimization problems. It introduces randomness into the selection of the next move, which can help avoid local maxima. In this problem move a queen to a new position so that it reduces the number of conflicts.

- **Procedure/Program:**

```
import random
def conflicts(board):
    return sum[ board[i] == board[j] or abs(
```

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

```
board[i]-board[j] == j-i  for j in range(i+1,len(board))
    for j in range(i,len(board)):
def stochastic_Hill_climb(n):
    board = [random.rand(0,n-1)]
    for in in range(n):
        while True:
            current = conflicts(board)
        if current = 0
            return board
        col = random.randint(0,n-1)
        mov = [ conflicts [board[:col]+[row]+
            board[col+i:]],row) for
            row in range(n)]
best_row = min(moves)
board[col] = best_row
```

Course Title	Artificial Intelligence and Machine Learning
Course Code(s)	23AD2001O

Experiment #	<T
Date	<T

- Data and Re

The
problem
iterativ

- Analysis and

board

Conflic

Evaluator Remark (if

Course Title	

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

- **Data and Results:**

The Stochastic hill climbing algorithm for the N-Queens problem starts with a random board configuration and iteratively moves a queen to reduce conflicts.

- **Analysis and Inferences:**

Calculates the number of conflicts for current board configuration

Moves a random queen to row that minimizes conflicts continuing until no conflicts remain

Evaluator Remark (if Any):	Marks Secured: 47 out of 50
	Signature of the Evaluator with Date