Fr. Conceicao Rodrigues College of Engineering Fr. Agnel Ashram, Bandstand, Bandra (W), Mumbai - 400050
Department of Computer Engineering
Academic Term II: 23-24

Class: B.E (Computer), Sem – VI Subject Name: Artificial Intelligence Student

Name: Sumit Sanjay Rai **Roll No: 9570**

Practical No:	**6**
Title:	Implementation of AO* algorithm
Date of Performance:	**11/03/2024**
Date of Submission:	**18/03/2024**

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Marks
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Logic/Algorithm Complexity analysis (03)	03(Correct)	02(Partial)	01 (Tried)	
3	Coding Standards (03): Comments/indention/Naming conventions Test Cases /Output	03(All used)	02 (Partial)	01 (rarely followed)	
4	Post Lab Assignment (03)	03(done well)	2 (Partially Correct)	1(submitted)	
Total					

Signature of the Teacher:

Post Lab Assignment:

1. What is the difference between A* and AO* algorithm?
2. Why AO* algorithm only works when heuristic values are underestimated?

Name : Sumit Sanjay Rai

Roll no : 9570

Class : TE COMPS A

PAGE NO. :
DATE : / /

Q.1. What is the difference between A^* and AO^* algorithm?

Ans.

Aspect	A^* Algorithm	AO^* Algorithm
Optimality	Guaranteed optimal solution	Not guaranteed optimal solution.
Heuristic Quality	Requires admissible heuristic	Works with under-estimated heuristics.
Solution Quality	Always provides optimal solution	May not provide optimal solution.
Exploration	Efficient guided search	Iterative refinement of estimates.
Performance	More efficient with admissible heuristic	Better in scenarios with underestimated heurishics.

Q.2. Why AO algorithm only works when heuristic values are under-estimated?

Ans. 1. Improper Heuristic Handling : AO^* may coverage to suboptimal solutions if heuristic values are overestimated.

2. Convergence Issues : Overestimated heurristics hinder AO^*'s Iterative refinement process, leading to unreliable solutions.

3. Futility Condition Violation : Overestimated heuristics may prevent AO^* from terminating early, resulting in unnecessary exploration.

4. Inefficient Exploration : AO^* might waste resources exploring unnecessary parts of the search space with overestimated heuristics.

5. Performance Degradation : AO^*'s performance suffers with overestimated heuristics, resulting in longer convergence times and sub-optimal solutions.