Date ______
Page ______

Assignment - II

1. Discuss these following env. types.

a) Fully observable (vs partially observable)

→ If an agent's sensor give it access to the complete state of the environment at each point in time then we say that the task environment is fully observable. An environment might be partially observable because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data.

b) Deterministic (vs stochastic)

→ If the next state of the environment is completely determined by the current state & the action executed by the agent then we say the environment is deterministic otherwise it is stochastic.

c) Episodic (vs sequential)

→ The episodic task environment, the agent experiences is divided into atomic episodes. Each episode consists of the agent perceiving & then performing a single action. Crucially the next episode does not depend on the actions taken in previous episode. In sequential the choice of the action in could affect all future decision.

d) Static (vs dynamic)

→ If the env. can be change while an agent is delibrating then we say the environment is dynamic for that agent otherwise it is static.

e) Discrete vs continuous

→ The discrete continuous distribution can be applied to the state of the environment, to the way time is handled & to the percepts and actions of the agents.

f) Single (vs multiagent)

→ Single agent & multiagent is differential by observing no. of agents in the environment.

3) You were given 2 jugs, a 4-gallon & 3 gallon another. Neither have any measuring markers on it. A pump can be used to fill the jugs with water. How can you get exactly 2 gallons of water into 4-gallon jug?. Give the soln to the above problem showing all the production rules.

→ Here,

Initial state ⇒ (0,0)

Operator ⇒
- Fill 3 gallon jug from pump; fill 4 gallon jug from pump.
- Fill 3 gallon jug from 4 gallon jug
- " 4 " " " 3 " "
- Empty 3 " " into 4 " "
- " 4 " " " 3 " "
- Dump 3 " " down drain
- " 4 " " " "

Final state ⇒ (0,2)

3 missionaries & three .....

Left	River	Right
MMMCCC	-	1000000
MM CC	MC →	MC
MMMCC	M ←	C
MMM	CC →	CCC
MMMC	C ←	CC
MC	MM →	MMCC
MMCC	CM ←	MC
CC	MM →	MMMC
CCC	C ←	MMM
C	CC →	MMMCC
CC	C ←	MMMC
000000	CC →	MMMCCC

Model Tic-tac-toe problem as a production system.
The main characteristic of a production system from the aspect of their storage in a computer system. The production system can be of various type but the most popular system is the monotonic system. A monotonic production system is a production system in which the application of one rule never prevent the later application of another rule.

- Board position = { 1, 2, 3, 4, 5, 6, 7, 8, 9 }
- An element contain the value ~~of~~ if the corresponding square is blank 1 if it is filled with "o" & "2" if filled with "x".

Hence starting state is { 0,0,0,0,0,0,0,0,0 } . The goal state or winning comb^n will be board position having "O" or "X" seperately in the comb^n of ({1,2,3}, {4,5,6}, {7,8,9}, {2,5,8}, {3,6,9}, {1,5,9}, {3,5,7}) element values. Hence two goal state can be {2,0,1,1,2,0,0,0,2} & {2,2,0,1,0,1,0,0}

6) Explain the production systems with help of 8-puzzle example

→ Initial state ⇒ Any state can be defigned to be initial

Goal " ⇒ To get at shortest past

Legal move ⇒ Blank can move (left, right, up, down)

Initial state

1		2
4	5	3
7	8	6

Final state

1	2	3
4	5	6
7	8	

Move 2 to left

" 3 to up

" 6 to "

8)

→ Well defined problem is defined as a problem which contains a clear specification of 3 element of the problem space. Initial state, the set of operator to solve the problem & goal state.

Goal test is the function which is created to check if the initial state is the final state or not. It is a criteria or cond^n used to determine whether a particular state or configuration of a problem solving agent satisfies the overall objective or goal of the problem.

Successor function is a function that generate the set of all possible state that can be reached from a given state in a search space. The successor function takes a current state as i/p & produces a set of successor states representing the possible next states the system or agent can transition to from the current state.

10) State & discuss problem formulation of Man, Lion, Goat & Cabbage problem.

→

West	River	East
{M, C, L, G}		{ϕ}
L, C	(M, G) →	(M, G)
M, L, C	M ←	G
L	M, C →	M, C, G
M, L, G	M, G ←	C
G	M, L →	C, M, L
M, G	M ←	C, L
{ϕ}	M, G →	M, L, G, C

11) State & explain DFS algorithm. Mention its completeness, optimality, time complexity & space complexity.

→ DFS progresses by expanding the first child node of the search tree that appears & thus going deeper & deeper until a goal node is found or until it hits a node that has no child. Then search back-trees returning to the most recent node it hasn't finished exploring.

completeness: NO, DFS isn't complete
optimality: NO, DFS isn't optimal
Time complexity: $O(bm)$
space " : $O(bm)$

Here,

b → branching factor
m → max^m depth of search tree.

12) State & explain BFS algorithm. Mention its properties.
→ BFS is a simple strategy in which the root is expanded first then all the root successor are expanded next, then their successor. The search tree is visited level by level that all nodes are expanded at a given depth before any node at the next level are expanded.

completeness: If the shallowest node is at some fine depth, the BFS will find out goal node. Yes, complete

optimality: Yes

Time complexity: $O(b^{d+1})$

space " : $O(b^{d+1})$

Here,

b → branching factor
d → depth of shallowed goal.

13) State & explain ~~DFS~~ DLS (depth limit search). Write its advantages.

→ It is similar to DFS with a predetermined limit. DLS can solve the drawback of the infinite path in DFS. Here, the node at the depth limit will be treated as it has no successor node further.

Completeness: Complete iff the solⁿ is above the depth limit

Optimality: NO, not optimal

Time complexity: $O(b \times l)$

Space ": $O(b^l)$

b → branching factor

l → depth level of tree

→ **Advantage:**

- Memory efficient
- Always terminates

14) Compare & contrast DFS, BFS, DFID.

→

criteria	DFS	BFS	DFID
completeness	NO	Yes	Yes
optimality	NO	Yes	Yes
Time comp.	$O(b^m)$	$O(b^d)$	$O(b^d)$
Space comp.	$O(bm)$	$O(b^d)$	$O(bd)$

Here,

b → branching factor

m → max^m depth of search tree

d → shallowest depth