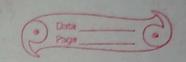
Assignment - I



a. Discuss these following env. types.

- a) fully observable (vs partially observable)
- state of the environment at each point in time

 then we say that the task environment is fully

 observable. An environment might be paralally

 observable because of noisy and inaccurate sensors

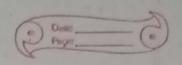
 or because parts of the state are simply missing

 from the sensor data.
- b) Deterministic (Vs stochastic)
 - of the next state of the environment is completely determined by the current state of the action executed by the agent then we say the environment is deterministic otherwise it is sto classic.

c) Episodic (vi sequential)

- experiences is divided into atomic episodes. Each episode consists of the agent perceiving then performing a single action. Crucially the hext 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)
 - delibrating then we say the environment is dynamic tor that agent otherwise it is static.



es discrete vs continuous

the state of the environment, to the way time is handled of to the percepts and actions of the agents.

t) single (vs multiagent)

- single agent & multiagent is differential by observing no of agents in the environment.

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 sold to the obove problem showing all the production rules.

Initial State => (0,0)

operator s. Fill 3 gallon jug from pump, fill 4 gallon

· Fill 3 gallon jug from 4 gallon jug

· Fmpty 3 11 11 into 4 11 11

. " 4 " " " " "

· Dump 3 " " down drain

. 11 4 11 11 11

Final state => (012)

| 5 | Date | -6) |
|---|------|-----|
| 1 | Page | |

| | | | - | | |
|-----|---------|------|---|--------|--|
| 2 m | iccina. | 2000 | 2 | three. | |

| Left | River | Right |
|--------|----------|---------|
| MMMCCC | | 1000000 |
| MM CC | Mc > | MC |
| MMMCC | Me | C |
| www | CC > | ccc |
| MMMC | C - | CC |
| MC | MM > | MMCC |
| MMCC | CM-O | MC |
| cc | MM -> | MMMC |
| ccc | @C | MMM |
| C | (ce > | MMMCC |
| CC | 608 C 4- | MMMC |
| 600000 | 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

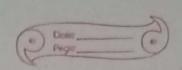
production system s is a production system in which the

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 I if the it is filled with "O" +
"2" If filled with "x".



Explain the production systems with help of 8-puzzle example

Tritical state => Any state can be designed to be initial

Goal " > To get at shortest past

Legal move => Blank can move (left, night, up, down)

R'nal steate

Initial steete

| | | | | | | | | - |
|---|---|---|---|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|---|---|-----|
| 1 | 1 | | 2 | 4-7-3-3-3 | 1 | 2 | 3 | 5 0 |
| | 4 | 5 | 3 | | 4 | 5 | 6 | |
| | 7 | 8 | 6 | The state of the s | 4 | 8 | | |
| | | | 1 | | | | - | |

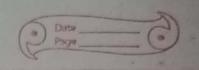
Move 2 to 18ft

11 3 to 4p

11 6 to 11

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 a goal state. Groundest is the function which is created to check if the initial state is the final state or hot. It is a criteria or conditused to determine whether a particular state or configuration of



or goal of the problem.

successor function is affunction 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 a produces a set of successor states representing the possible next states the system or agent can transition to from the current state.

2 cabbage problem.

| > | west | | River | Fast |
|---|--------------|---------|----------|---------|
| | fm, e, H, G3 | | (d) | 4 9 3 |
| | Lic | | (M/G) -> | (MIG) |
| 1 | M,L,C | | M | G |
| | L | 10 14 1 | M,C> | MIC, G. |
| | M,L,G | | MiG | C |
| - | G | | M, L > | CIMIL |
| - | M, G | | M < | CIL |
| 1 | 107 | | MIG-> | M12G,C |
| | | | | |

state 2 explain DFS algorithm. Mention its completeness.

optimality, time complexity 2 space complexity.

DFS progresses by expanding the first child node of

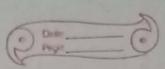
the search tree that appears 2 thus going deeper 2 deeper

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 comprexity: 0 (bm)) : 0(bm) , b > branching factor m + max m depth of search tree. State & explain BFS algorithm. Mention its properties. 12> BFS is a simple strategy in which the root is expanded first then all the not successor are expanded next, then their successor. The search tree is visited level by heres 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 comprexity: 0 (60) Space " : 0 (60) by branching factor d> depth of shallowed goal.

| | Date Page | | | | | | |
|-----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|--|--|--|--|
| 13> | State e explain des de DLS (depth limit fearch). White | | | | | | |
| | | | | | | | |
| -> | its advantages. It is similar to DFS with a predetermined limit. | | | | | | |
| | DLS can solve the drawback of the infinite path in DFS. 25 | | | | | | |
| | Here the node at the depth limit will be treated as it | | | | | | |
| | has no sucressor node further. | | | | | | |
| | Maz No sacretion was been a second and the second a | | | | | | |
| | completenose: complete iff the soln is above the depth limit | | | | | | |
| Maria ! | optimality: No , not optimal | | | | | | |
| | Time complexity: 0 (bx l) | | | | | | |
| The state of | Fine complexity: 0 (bx l) space " 0 (b²) | | | | | | |
| | by branching factor | | | | | | |
| 1. 3. | by branching factor 14 depth level of tree | | | | | | |
| | The transfer of the state of th | | | | | | |
| 29 | Advantage: | | | | | | |
| 2.40 | · Memory efficient | | | | | | |
| | · Always terminates | | | | | | |
| | | | | | | | |
| | the state of the s | | | | | | |
| 14) | Compare & contrast DFS, BFS, DFID. | | | | | | |
| - | criteria 270 BES DEID | | | | | | |
| | completeness NO Yes Yes | | | | | | |
| | optimality No yes Yes | | | | | | |
| | Time comp. 0(bm) 0(bd) | | | | | | |
| | space comp. o(bm) o(bd) o(bd) | | | | | | |
| | The state of the s | | | | | | |
| | Here, | | | | | | |
| | by branching feetor | | | | | | |
| 7.7. | my maxim depth of search tree | | | | | | |
| | d + shallowett depth | | | | | | |
| 100000000000000000000000000000000000000 | | | | | | | |

