*Polina Minina, Group B19-02*

**AI Assignment 1**

**PEAS model**

**Agent**: actor

**Performance measure**: shortest path to the home without being infected by covid.

**Actuators**: ability to see what is in the adjacent cells (standing next to the cell/standing 1 cell away to see covid), ability to move on 8 directions, ability to be vaccinated by doctor, ability to wear the mask.

**Sensors**: information about location of the home and agent coordinates, size of the map

**Environment**: map with doctor, mask, covid and actor.

*Partially Observable* – agent cannot see full map; it can only see adjacent and visited cells.

*Single agent* – there is only one agent on the map.

*Deterministic* – we can determine the next state of the environment given the previous state and the action we are applying into it.

*Episodic* – we allow to make atomic actions not dependent upon previous.

*Static* – environment does not change as the actor is thinking about action.

*Discrete* – there is states in the environment.

*Known* – designer of the agent has full knowledge of the rules of the environment.

**Algorithms**

**Backtracking**:

The algorithm chooses which direction to go and performs it recursively for each cell. If it turns out that at some point the choice was wrong, the algorithm returns to this point and makes another choice (prolog works on principle of backtracking, so we do not need to implement it). Thus, our algorithm "simultaneously" starts several branches of the path finding. Those branches that reached the home are returned and compared with each other. Branch that has a longer length are cut off. At the very beginning, we set the maximum path length to N\*N, so that the rest of the branches can compare length with this value. If there are no branches that could reach the home, we lost the game.

In the assignment two different variants of the vision of covid cells were given. In case of backtracking, it doesn't matter how far we can see the covid, because by running this algorithm on the first and second options, we can understand that they are no difference.

**BFS**:

BFS algorithm considers all neighboring cells and recursively searches for a path from these cells. In breadth first search, we need to store the visited vertices, so each time we must check whether the vertex is subset of the visited vertices or not. Also, after considering one vertex, we put it into list of visited vertices. In the 'rule' of algorithm we must also store a queue, following it we consider vertices, but in the case of using the Prolog, we do not need to implement this, because language works according to this principle.

In the assignment two different variants of the vision of covid cells were given. In case of breadth first search, it doesn't matter how far we can see the covid, because by running this algorithm on the first and second options, we can understand that they are no difference.

**Statistical comparison**

|  |  |  |
| --- | --- | --- |
|  | Backtracking | BFS |
| Execution time (size is 6) | 2371 ms | 110 ms |
| Execution time (size is 9) | 3363 ms | 205 ms |
| Execution time (size is 10) | 19905 ms | 415 ms |

So, BFS is faster that backtracking.

**Input map**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  | C | C | C |  |  |  |
| 4 |  |  |  | C | C | C | H |  |  |
| 5 |  | D |  | C | C | C |  |  |  |
| 6 | M | C | C | C |  |  |  |  |  |
| 7 |  | C | C | C |  |  |  |  |  |
| 8 |  | C | C | C |  |  |  |  |  |
| 9 | A |  |  |  |  |  |  |  |  |

**Input**:

size(9).

home(7,4).

covid(3,7).

covid(5,4).

doctor(2,5).

mask(1,6).

**Output** of the algorithms (also displayed by yellow in the map):

WIN

Number of steps: 8

Path: [cell(1,9),cell(2,9),cell(3,9),cell(4,9),cell(5,8),cell(5,7),cell(6,6),cell(7,5),cell(7,4)]

Execution took 3363 ms.

**Impossible map**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 |  |  |  |  |  | H |
| 2 |  | D | M | C | C | C |
| 3 | C | C | C | C | C | C |
| 4 | C | C | C | C | C | C |
| 5 | C | C | C |  |  |  |
| 6 | A |  |  |  |  |  |

**Input**:

size(6).

home(6,1).

covid(5,3).

covid(2,4).

doctor(2,2).

mask(3,2).

**Output** of the algorithms:

false.