|  |
| --- |
| ARTIFICIAL INTELLIGENCE AND KNOWLEDGE ENGINEERING |
| Game “The parking lot maze” |
| First assignment |
| **Álvaro Martín Orive 78955100**  **Sonia Piérola Abraldes 78907376** |
|  |
| **17/04/2011** |

|  |
| --- |
|  |

Index

Question 1---------------------------------------------------------------------------------------------------

Question 2---------------------------------------------------------------------------------------------------

Question 3---------------------------------------------------------------------------------------------------

Question 4---------------------------------------------------------------------------------------------------

Question 5---------------------------------------------------------------------------------------------------

Problem Formulation and Design

## Problem’s environment properties and Performance measure

**According to the problems description we have determined the following properties:**

* **Fully observable system: the system can see the board and the marks of each position at each time. The initial state of the board is going to be also the final one.**
* **Single system: there is just one system operating by itself in our environment as we have a single car and board.**
* **Deterministic system: the next state is completely determined by the current one and the executed action; the current movement of the car will affect the following one**
* **Sequential system: the current decision can affect all the future ones.**
* **Static system: the environment is unchangeable while the system is deliberating.**
* **Discrete system: the number of states is discrete as well as the number of percept and actions.**
* **Know system: the system knows the outcomes of all actions.**

**The problem´s environment consists on a board of a specific number of rows and columns where it can be found some different elements such as crosses, circles, a car or a flag.**

**Those elements indicate the possible movements that the car can do in order to drive it to the parking lot marked by the flag. We have to bear in mind that there is only one single entrance to the lot as it is surrounded by three walls. When implementing this problem we have declared a dual array of cells to be able to know the position of the element in the board and the type of cell because, as we have mentioned before, it can be a cross or a circle.**

**The class which represents the flag has a special field (“entrance”).This let us know the side we have to take to enter in the lot (remember the constraint of the surrounding walls). The class that defines the car has a field (“direction”) that indicates the direction that the car has in a specific moment; north, south, west or east. Bearing in mind this position, the car will be able to be moved through the board by using the different operators. The possible movements are “move straight”, “turn left” and “turn right”.**

**With all this classes we have the needed information in order to solve the problem with a mechanism of comparison. The performance measure of the problem will be to minimize the number of movements of the car in order to find a path from the initial position to the final one.**

## Problem formulation

**Initial state: the car is at the entrance of the board, the flag is already in a specific position and each of the cells has their associated mark (circle or cross).**

**Goal test: is the car in the parking lot?**

**State: the car is at any other position different from the one of the flag.**

**Actions: Move straight if the position is a circle**

**Make a right turn if the position is a cross.**

**Make a left turn if the position is a cross.**

**Transition model: the car changes its position.**

**Path cost: number of actions. Therefore the cost of each action would be 1 and the total path cost will be the number of actions in the path**

## Relaxed Problem Technique to define an admissible and consistent heuristic

We consider it is possible to use the relaxed problem technique in the parking lot maze problem. The relaxed problem results from eliminating the restrictions from the actions defined. So in this specific problem we have eliminated restrictions such as the actions to be taken when finding a particular mark on the board (X or O) or the walls that surround the flag. According with the theorem, the cost of an optimal solution to a relaxed problem is an admissible heuristic for the original problem. So we can define

H(n)= Manhattan Distance from the position of the car to the flag

So by using the heuristic of the manhattan Distance...

**public** **class** ManhattanDistance **extends** EvaluationFunction {

**public** **double** calculateG(Node nodo) {

**return** 0;

}

**public** **double** calculateH(Node nodo) {

Board b = (Board) nodo.getState().getInformation();

**int** distance = 0;

distance = (**int**) **this**.getManhattanDistance(b.getFlag().getPosition(), b.getCar().getPosition());

**return** distance;

}

**private** **double** getManhattanDistance(Cell f, Cell c) {

**return** Math.*abs*(f.getX() - c.getX()) + Math.*abs*(f.getY() - c.getY());

}

}

We can solve the problem with the philosophy of the relaxed problem.

## Search Method

We have different search methods in which we can trust to solve this problem such as the blind and the informed algorithms. The local search algorithms are not valid here because we are interested in the path of the solution, not the solution itself. Lets classify them in two groups:

### Blind search

#### Breadth-First

It is a valid algorithm because it is Complete and is Optimal, because all the actions have the same cost.

#### Uniform-Cost

This method will act as Breadth-First because all the same actions have the same cost.

#### Depth-First

If we consider that it could be a grid of 2x2 that it just have X (loop) this algorithm is not valid because is not complete and not Optimal. The completeness can be solved if we keep in memory a list of repeated states, but it will be still not optimal.

#### Depth-Limited

Considering that we will use an l < d, this method will not ensure the completeness, because the solution can be in a depth between l +1 and d.

#### Iterative-Deeping

Iterative-Deeping is a good choose in case that we use a blind search because it ensures the completeness and the optimality (all the action have the same cost).

As we have saw, there are some methods such as Breadth-First, Uniform-Cost and Iterative-Deeping that are valid for this problem, but as we have observed in previous questions, the problem is fully observable and also, it is an offline problem, because the environment is static and the board does not change from the initial state, so it will be better to use an informed search algorithm that makes use of the heuristic obtained in the previous question.

### Informed Search

#### Greedy Best First –Search (Based on GraphSearch)

It will make use of the heuristic defined (Manhattan Distance) as the evaluation function in order to sort the frontier, so it is expected to have a better performance than any blind search algorithm.

#### A\*

This algorithm is one of the best algorithms to solve this problems because the evaluation function will use also a function that calculates the actual cost of the path that is being covered ( g(n) ). The function just have to count the number of movement done to calculate g(n) and the add them to h(n) to obtain our evaluation function f(n)

However, this method is not implemented in the jar file provided in the subject, so we will make use of the Best-First search algorithm provided in the practice, which is based in an algorithm that only uses a Heuristic as Evaluation Function.

## UML Class Diagram

To see the package diagram open the file “Packages.pdf”

To see the UML Class Diagram open the file “ParkingLotMaze.pdf”

Note: Board has a matrix of cells, but the software does not know how to associate that with the Cell class.