CSI4106: Introduction to Artificial Intelligence
Fall 2016

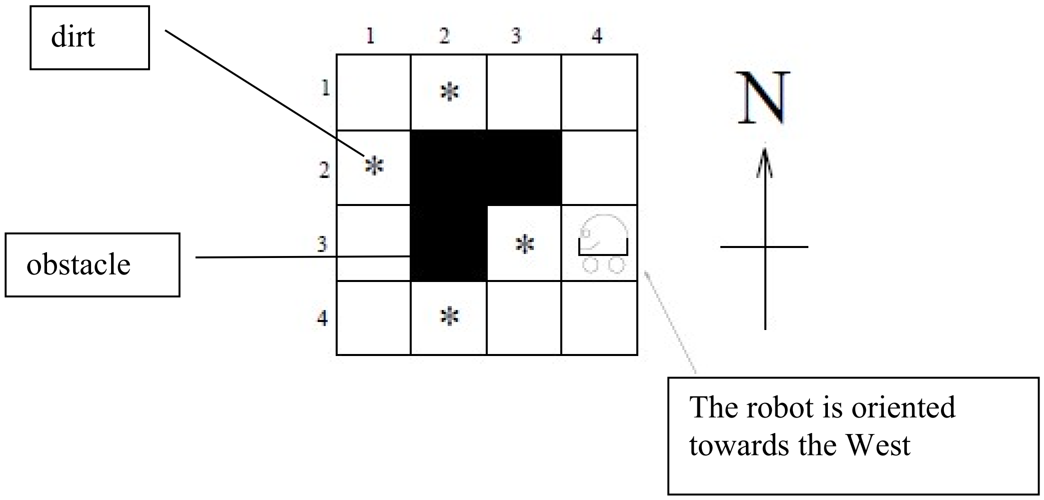
Assignment 1

Handed in on: September 19th, 2016
Due on: October 3rd, 2016 (midnight)

Learning objective: program blind and informed search in the vacuum cleaner robot problem
Requirements: The assignment MUST be done in a group of two.

1. Description

The objective of this work is to program in **Java** a blind and heuristic search.
The problem is the following. A robot is located in a room that contains obstacles and dirt. The robot must clean the room by sucking the dirt while minimizing the energy used.
The environment is represented in the following manner. The room is a square and all possible positions are represented by a *n x n* grid. Positions (1,1) and (n,n) correspond to the upper left square and the bottom right square respectively.
An obstacle occupies a certain number of squares in the grid. Of course, the robot cannot move to a square occupied by an obstacle and there is no dirt to suck in such a square. The following figure shows an example of the environment and robot.



In the figure we can see that the size of the grid is 4 x 4 and that the obstacle occupies the positions (2,2 ; 2,3 ; 3,2).

The robot can make 4 actions:

1. **Suck**: sucks the dirt at the position where the robot is located. This costs the robot an energy of 10.
2. **Right**: turns right on a 90 degree angle. This costs the robot an energy of 20.
3. **Left**: turns left on a 90 degree angle. This costs the robot an energy of 20.
4. **Move**: moves to the next square in front of it, if the square is not occupied by an obstacle. This costs the robot an energy of 50.

The objective is to obtain a room without any dirt.

Here is an example of the required output (for any search strategy), which displays the actions from the start state to the goal, the path cost and the time required to obtain the solution:

OUTPUT for the grid given as example:

```
pos(4, 3), West, start
pos(3, 3), West, move
pos(3, 3), West, suck
pos(3, 3), South, left
pos(3, 4), South, move
pos(3, 4), West, right
pos(2, 4), West, move
pos(2, 4), West, suck
pos(1, 4), West, move
pos(1, 4), North, right
pos(1, 3), North, move
pos(1, 2), North, move
pos(1, 2), North, suck
pos(1, 1), North, move
pos(1, 1), East, right
pos(2, 1), East, move
pos(2, 1), East, suck

total cost: 520
Depth: 10
Time : 1 ms
```

Note that depth and time are just examples.

To Do List
You must return a zip file containing your complete **<u>code</u>** and a **<u>report</u>** entitled *yourname1_yourname2_A1.zip*. We must be able to import your zip file in Eclipse.
Requirements:

Code:

1. The code must contain a class called *RobotApp.java* which includes the *main* method. The main method must generate a grid according to appropriate parameters and launch a search method.

2. RobotApp.java must provide a method *generateGrid* that generates a grid. The method must take as input the size of the grid, the obstacles positions, the dirt positions, the robot position and its orientation.
3. RobotApp.java must also contain a *search* method that takes as input an integer parameter (1=DFS, 2=BFS, 3=A*), and the grid returned by the method generateGrid and returns a solution.
4. RobotApp.java must implement a *printSolution* method that takes as input the solution returned by *search* and print it to the console as shown in the description section (OUTPUT).
5. The code must solve the problem described above using the following search methods:
    a. A depth-first search implementation
    b. A breadth-first search implementation
    c. A* implementation with an appropriate heuristic

Report:
The report must have a front page with your names, the course and the assignment number. It should contain the answer to the following questions:

General questions
1. How do you represent a state in this problem?
2. What is the initial state, goal state and path cost in this problem?
3. What is the branching factor in this problem?
4. What is the proposed heuristic? Justify your choice.
5. Describe the environment properties. Justify your answers.

Implementation questions
1. How do you represent the room/grid?
2. How do you represent the dirt and obstacles?
3. How do you represent a state in your implementation? (note that this is different from the Q1 in general questions)

Code description
Provide a brief description of your implementation and the classes in your various packages.

Comparative analysis
Make a comparative table describing, for each search method (BFS, DFS, A*), the depth, path cost, the time taken, and the <u>worst</u> time and space complexity. What is your conclusion?

Marking guidelines
1- Working implementation following the guidelines
2- Good programming practices (appropriate use of classes, methods, parameters, **<u>comments</u>**, etc.)
3- Correct answers to questions

Code	
DFS	12
BFS	12
A*	12
Heuristic	10
Problem representation (grid, state, etc.)	15
Report	
General questions	10
Implementation questions	9
Code description	10
Comparative analysis	9
total	100