



TDDC17 ARTIFICIAL INTELLIGENCE :

Lab 2 : Search Algorithms

Arnaud PECORARO

September 2015

1 Implementing CustomGraphSearch

1.1 Aim

This second lab session explores the concept of graph search algorithms. The agent is still a Vacuum cleaner evolving in a simple grid. Each square of that grid is either clean filled with dirt, or a wall. The vacuum cleaner uses search algorithm to locate the dirt squares. The purpose of this lab is to write our own implementation of *Breadth First Search* and *Depth First Search* algorithms which inherit from a *CustomGraphSearch* class.

Using the Java GUI included in the project files, it is possible to compare the efficiency of different search algorithms : BFS, DFS, IDS, A*.

1.2 Implementation

```
~/D/C/T/L/1/1/src >>> ./run.sh
Breadth-First Search (Z)
Agent: Changed Search Method to 0
Agent: planning from (01,01) to (22,01)
Agent: starting Breadth First Search Method (BFS)
        Needed 11 msec, PathLength: 33, NumExpNodes: 460
Custom Breadth-First Search (S)
Agent: Changed Search Method to 4
Agent: planning from (01,01) to (22,01)
Agent: starting Breadth First Search Method (BFS)
        Needed 11 msec, PathLength: 33, NumExpNodes: 481
```

(a)

```
Creating new World!
Agent: Changed Search Method to 1
Depth-First Search (X)
Agent: Changed Search Method to 1
Agent: planning from (01,01) to (24,11)
Agent: starting Depth First Search Method (DFS)
        Needed 11 msec, PathLength: 211
Custom Depth-First Search (D)
Agent: Changed Search Method to 5
Agent: planning from (01,01) to (24,11)
Agent: starting Depth First Search Method (DFS)
        Needed 9 msec, PathLength: 211
```

(b)

FIGURE 1 – Comparison between builtin and custom implementation (a) BFS - (b) DFS

2 Theory

Question 1

Question 2

Question 3

Question 4

Question 5

Question 6

Question 7