THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

COMPSCI 366 S1 C 2006
Foundations of Artificial Intelligence

Computer Science Department

Assignment 2: A* Search in a Maze

Worth: 10% [40 marks]

Due Date: Sat 3rd June 2006 11.59 pm

Submission: Assignment Drop Box

2.1 Overview

Search is an essential part of AI, appearing in many different flavours like A* search, hill climbing, simulated annealing, constraint satisfaction, genetic algorithms, and many more. Many subareas of AI use search in their problem solving procedures. This assignment applies A* search to a maze problem.

The maze problem we are dealing with in this assignment is finding a path through a maze, given a start position and a goal position in the maze. Implement the A* algorithm in Prolog with two different heuristics, test each of them with several examples, and compare the results. Instead of implementing the A* algorithm from scratch, you can use Bratko's code for the A* algorithm.

2.2 Datastructure

A maze can be viewed as a two-dimensional structure consisting of walls and paths. The easiest representation of such mazes in Prolog is a nested list structure. However, accessing an item in a list structure (i.e., a position in the maze) is not very efficient, since it takes time proportional to the length of the list. An array structure, which enable direct access to a required item, would be more suitable.

There is no array facility in Prolog, but arrays can be simulated very easily by using the built-in predicate `arg/3`. A goal `arg(N, Term, A)` is true if `A` is the `N`th argument in `Term`. In our case, `Term` has a two-dimensional structure, such as `my(mx(...), mx(...), ...)`. Inside the inner terms, you could use the atoms `w` and `p` to indicate walls and paths, respectively. To state that this structure is a maze, you could put the following into your database:

```
maze(my(mx(w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,p,p,p,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(p,p,p,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,p,p,p,p,p,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,p,w,w,p,w,p,w,w,w,w,p,p,p,p,p,p,p,p,p,p,p,p,w,w,w),
        mx(w,w,w,p,p,p,p,w,p,w,w,w,w,p,w,w,w,w,p,w,w,w,w,w,p,w,w,w),
        mx(w,w,w,w,w,w,w,w,p,w,w,w,w,p,w,w,w,w,p,w,w,w,w,w,p,p,p,p),
        mx(w,w,w,w,w,w,w,w,p,p,p,p,p,p,w,w,w,w,p,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,w,w,w,w,w,w,w,w,w,w,p,p,p,p,p,p,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,w,w,w,w,w,w,w,w,w,w,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,w,w,w,w,w,w,w,w,w,w,p,w,w,w,w,w,w,w,w,w,w,w,w,w,w),
        mx(w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w,w))).
```

The above data structure can be used for various queries. For example, to test whether there is a path at some given coordinates, you could define a predicate `ispath/1` as follows:

```
ispath([X, Y]) :- maze(Maze), arg(Y, Maze, Row), arg(X, Row, Pos), Pos = p.
```

2.3 Heuristics

Use the following two heuristics in your search algorithm:

- The first heuristic, called the Euclidean or straight-line heuristic, takes the Euclidean distance between the current position (x, y) and the goal (x_g, y_g) to estimate the cost for getting from the current position to the goal:
$$h((x,y)) = \sqrt{(x - x_g)^2 + (y - y_g)^2}$$
- The second heuristic, called the Manhattan heuristic, takes the Manhattan distance between the current position (x, y) and the goal (x_g, y_g) to estimate the cost for getting from the current position to the goal:
$$h((x,y)) = |x - x_g| + |y - y_g|$$

2.4 Testing

Test each heuristic with 5 different mazes (ranging from simple mazes to more complex ones with cycles) and measure how long it takes in each case to find a path from the starting point to the goal. Does it make a difference if you use the Manhattan heuristic instead of the Euclidean heuristic? Explain your answer!

2.5 Marking

Marking is based on the following material:

- Printout of the documented Prolog code [20 marks].
- A listing of the test cases with an explanation for the choice of mazes [8 marks].
- Test results and their interpretation [12 marks].