Shailesh Mani Pandey, Tushar Agarwal
2013CSB1069, 2013CSB1035
4 April 2016

README: AI, Lab-4

Planning

We provide a python script (p1.py) containing implementations for breadth-first forward search, A-Star forward search and goal stack planners. Actions are encoded in a way which makes modification easy.

How to execute:

python p1.py **p.txt**

Here, the text in bold refers to a filename with a blocks world problem instance as per the input format described in the problem description.

The output will be written to p_out.txt. In general, for input filename name.ext, the output will be written to name_out.ext. A three-letter extension is preferred.

Additional documentation is present (in HTML format) inside the documentation folder.

Discussion on A-Star heuristic:

To calculate a heuristic value for a state, the heuristic computation algorithm relaxes the problem and solves an easier version of the problem. The heuristic computation algorithm performs a breadth-first forward search with relaxation in the following two ways:

1) Delete lists are ignored when a state is expanded. Hence, monotonic progress is made towards the goal state.
2) When a state is expanded, all possible actions are applied at once, together. This helps to control the branching factor which otherwise, if only technique (1) was used, may result in the creation of extremely large number of states.

Performance of breadth-first forward search and A-Star forward search:

We compare the performance of breadth-first forward search and A-Star forward search on two planning problems. The two instances are given in files 1.txt and 2.txt in the "tests" folders of breadth-first forward search and A-Star forward search respectively.

We now compare the performance of these search techniques on some example instances of the blocks world.

In the instance with four blocks, the output of the two planners was:

Breadth-first forward search

```
.....................................................
Planner: f
Time: 11.3294079304
Plan length: 10
Nodes expanded: 11704
Output written to: "tests/test cases/fw test cases/1_out.txt"
.....................................................
```

A-Star forward search

```
.....................................................
Planner: a
Time: 1.17206811905
Plan length: 10
Nodes expanded: 101
Output written to: "tests/test cases/astar test cases/1_out.txt"
.....................................................
```

In the instance with four blocks, A-Star forward search completes significantly faster and expands less than 1 percent of the number of states expanded by breadth-first forward search.

In the instance with five blocks, the output of the two planners was:

Breadth-first forward search

```
.....................................................
Planner: f
Time: 259.07286787
```

```
Plan length: 14
Nodes expanded: 155008
Output written to: "tests/test cases/fw test cases/2_out.txt"
.....................................................
```

A-Star forward search

```
.....................................................
Planner: a
Time: 30.2981629372
Plan length: 14
Nodes expanded: 1275
Output written to: "tests/test cases/astar test cases/2_out.txt"
.....................................................
```

In the instance with five blocks, too, A-Star forward search completes significantly faster and expands less than 1 percent of the number of states expanded by breadth-first forward search.

In the two cases mentioned above, both techniques give plans of the same length. The results of A-Star forward search may vary each time the program is run because of the nature of heuristic. Over many runs, it was observed that the above run is representative of the overall performance of A-Star forward search, and the resultant plans are different but have the same length.

The test case with 12 blocks could not be computed on by A-Star forward search or breadth-first forward search in a reasonable span of time.

Performance of goal-stack planner:

In the instance with four blocks, the output of the **goal-stack planner** was:

```
.....................................................
Planner: g
Time: 0.00588917732239
Plan length: 24
Nodes expanded: N.A.
Output written to: "tests/test cases/gsp test cases/1_out.txt"
.....................................................
```

In the instance with five blocks, the output of the **goal-stack planner** was:

```
.....................................................
Planner: g
Time: 0.00870299339294
Plan length: 26
Nodes expanded: N.A.
Output written to: "tests/test cases/gsp test cases/2_out.txt"
.....................................................
```

In the instance with twelve blocks, the output of the **goal-stack planner** was:

```
.....................................................
Planner: g
Time: 0.0170328617096
Plan length: 36
Nodes expanded: N.A.
Output written to: "tests/test cases/gsp test cases/3_out.txt"
.....................................................
```

The goal stack planner performs significantly faster than both breadth-first forward search and A-Star forward search. There is almost imperceptible rise in time to solve a problem with a much larger number of blocks.

However, the goal stack planner output plans of a larger length. Also, the planner is not guaranteed to terminate. Moreover, a problem specific heuristic was utilized to decide on relevant actions to insert actions onto the goal stack. The generalized goal stack planner is also implemented in the present assignment, but performs worse than the specialized version.

The results of goal stack planner may vary each time the program is run because of the nature of heuristic and the randomness in the order when pushing subgoals onto the stack.

Over many runs, it was observed that the above run is representative of the overall performance of goal stack planner.

End