COMP2611
Artificial Intelligence

Assignment 1: Search Algorithms

Astu Dent — scs2000ad@leeds.ac.uk
Anne Otheren — scs2001ano@leeds.ac.uk

A. 8-Puzzle Search Investigation

A.1. Search Algorithm Test Sequence

After experimenting with various search options we found that the following sequence of tests gives an informative set of statistics regarding the performance of a wide range of search algorithms and options, when applied to the 8-Puzzle sliding tiles problem:

```
## Our testing code:
eight_puzzle_1 = ....

search( eight_puzzle_1, ..., ... ... )
 :
 :
 :
 :

eight_puzzle_2 = ....

search( eight_puzzle_2, ..., ... ... )
 :
 :
 :
```

For this answer you can just paste in the code that your ran to get your test results. But you may like to add a few sentences of explanation as well.

You need to decide on exactly which tests you do. But remember that the table of results needs to fit on one page. Try to get a set that covers the main options but not too many that it is difficult to understand the results.

A.2. Results

The results obtained for the first 'dum case' problem instance were as follows:

`dum_case_101` **Results**								
	splayed				pegged			
	tips	flips	d-flips	que?	tips	flips	d-flips	que?
Nose forward	100	1	2	Y	10	11	808	$*$
Nose forward	10,000	111	1,000,123	Y	200	22	2777	$*/?$
:	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:
Chin up	100		∞	N	0	-77	∞	N

Table 1: This is a stupid table. It just illustrates the kind of table structure you could use.

For a different test which was chosen to be floppier than the other, we got the following results:

`dum_case_102` **Results**								
	splayed				pegged			
	tips	flips	d-flips	que?	tips	flips	d-flips	que?
Nose forward	0	0	0	0	0	0	0	0
Nose forward	?	?	?	?	?	?	?	?
:	:	:	:	:	:	:	:	:
Chin up	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Table 2: This is other table shows another 'dum case' example.

You could have 3 or 4 tables, an you could have tables of different kinds. You need to try to make the tables show the results as clearly and informatively as you can. Don't ask the lecturers and assistants about what should be the exact contents and layout of the tables. Deciding on that is part of the assignment.

You need to make sure ***all the tables together fit on one page.*** *And if the meaning is not obvious from the table itself you will need to add some concise explanatory sentences on the same page. (For example: 'The symbol $\emptyset$' means my laptop ran out of battery.')*

A.3 Observations

Here are some silly observations. In your report you should list significant and/or interesting (not silly) observations that you have made regarding behaviour of the different search algorithms and options.

After examining our results, we gained deep understanding of search algorithms. Of the many interesting observations we made, the most flabbergasting were as follows:

- The observation that **Chin up** performed an infinite number of *d-flips* when pegged was truly flabbergasting, but is probably due to the flips being negative for this configuration. Negative *flips* are well-known to be problematic when applied to chin-based algorithms. This is because there are no nostrils to unblock.
- It can be seen that the **pegged** option reduced tips by at least a factor of 10 for all nose and chin positions. For this kind of problem it is clear that pegging is always beneficial since results without pegging can be seen to be worse in all cases.
- ...
- ...

You should list at least 4 observations, possibly up to 8 or more. But you need to make sure they are clear and distinct, and have some interest. Just adding many detailed small points will not gain extra marks and could actually lose marks as the overall list will become less informative.

It will be good to consider general patterns as well as individual results and also consider how the patterns and results are related to general characteristics of search strategies in relation to the problem.

A.4 Heuristics

The following gives some silly examples. It just shows the typical layout of how you should give the heuristics in your report. You paste in the code and give a very brief description of how it works.

Misplaced Tiles Heuristic

Our `misplaced_tiles` heuristic is defined as follows:

```
def misplaced_tiles( state ):
     drop.all( [tiles in state] )
     for tile on floor as it:
         grab it
     return num(grab.s)
```

Our algorithm simply drops all the tiles on the floor and counts how many grabs are required to pick them up.

Manhattan Distance Heuristic

Our `manhattan` heuristic is defined as follows:

```
def manhattan( state ):
     swing:
        from:
           skyscraper
        to:
           skyscraper
     return num(swing.s)
```

The algorithm uses a `swing` block to iterate over Manhattan. At the end of the swing transit the number of swings is returned.

B. Robot Worker Scenario

This part of the assignment is similar to Part A. However, it is more open ended and you are expected to use your own initiative of how exactly to do the task and write it up. But you still have a maximum of one page for each of the four. In fact, half a page for each will be sufficient for a reasonable mark, provided your answers are clear and satisfy the requirement.

B.1. My/Our Robot Scenario

Starting with the code given in the Search Exercise 5 notebook create your own robot worker scenario by adding extra items and/or rooms and/or doors. If you wish you can change completely change the types of items and rooms to your own theme. You can also make more technical changes such as adding actions for opening and/or locking doors; or more complex conditions on how objects can be moved. You could potentially even enable different types of goal to be specified.

It is strongly advised that you start by making only very minor changes and then experiment with the search algorithms in order to obtain an initial set of results. Then if you have time you could make the scenario more complex (and of course save a copy of your initial simple solution in case you want to go back to it).

In presenting your answer for this question ***do not include your code****. Instead you should give a clear concise description of the scenario. You should specify the items, the rooms, connecting doors and any other relevant details. You should also briefly describe the possible actions of the robot and the possible goals that could be specified. You could perhaps include a simple diagram of the room layout.*

B.2. Heuristic(s)

You should define and describe at least one heuristic that can guide the robot to solving planning problems in your scenario situation.

It is sufficient to implement a very simple heuristic. Considering how something like the `misplaced_tiles` *heuristic for 8-Puzzle might be applied to a Robot Worker situation should give you an idea for a basic heuristic.*

It is possible to devise much more sophisticated heuristics that can enable solution of more complex scenario. Remember that a heuristic should be an estimate of the remaining number of actions from a given state to the nearest goal state. In most cases it is best to have a heuristic that gives a number that is close to but does not exceed the actual number of remaining actions.

B.3. Results

You should present your results in a clear way that brings out the most import and things you have discovered. Probably this would be in the form of one or more tables with a few sentences of explanation.

Note that you do not need to include the code of the search test sequence use to carry out the tests that generated the data that you put in the table. However, it would be a very good idea to devise and maintain a test sequence similar to what you did for **A.1**. *This will enable you to easily rerun and/or modify the test sequence in order to check your results perhaps produce more informative results.*

	a	b	c	d	e	...
?	?	?	?	?	?	...
:	:	:	:	:	:	

B.4. Key Findings

This should be done in a similar way to **A.4***, but of course you need to identify key findings for how the different search algorithms and/or your heuristics behave when used to solve problems in your scenario.*

The most significant observations we made regarding the use of search algorithms applied to our robot worker scenario were as follows:

- A general observation is that ...
- We found that in the case of ...
- ...
- ...