Final project document for the Artificial Intelligence class

Pablo Gonzalez

December 6, 2016

1 Contents

This submission contains:

- *.java and *.bin: source and compiled files for all the classes used in the project.

  Among them are:

  - An abstract class for problems (Problem) with its corresponding implementations for each problem
    - BananagramsProblem
    - NPuzzleProblem
    - MazeProblem
    - HanoiProblem
  - An abstract class for search engines (SearchEngine) along with classes that extend it for each strategy
    - BFS_Engine
    - DFS_Engine
    - Greedy_Engine
    - Astar_Engine
    - SMAstar_Engine
  - The Node class to be used by all problems.
  - Three runnable classes that perform different uses of the program
    - Tester: a small program to test node generation and goal testing (see instructions below)
    - SearchTester: Runs a specified kind of search on a preset version of the problem (see instructions below)
    - RunExperiment_Bananagrams: runs a larger experiment with several problems (specified in input files)
- data/dictionary.txt: a text file with the scrabble dictionary of all valid words to be used by the Bananagrams problem
- data/n−gramfreq.txt: a series of text files with the collected frequencies of n-grams (for n between 2 and 15) as they appear in dictionary.txt
- data/ngramcounter.py: the python code to generate the n-gram files
- data/problem/n−bananagramProblems:files with the tilesets for randomly generated order n bananagrams problems highly probably to be solvable
- data/bananagramgen.py: the python code used to generate the problems
- Readme.pdf: this file

2 Running instructions

2.1 Tester

To run the tester for node generation navigate to the containing directory within the terminal and run:

```
$java Tester <problemnanme>
```

The <problemname> variable can take four values, it chooses which problem to test, the allowed values are:

```
Bananagrams
Hanoi
NPuzzle
Maze
```

The response is printed to console standard output so it will appear in the terminal. If modifications are made the code will have to be recompiled for them to take effect, run:

```
$javac Tester
```

The Rester program makes three tests and post the results:

- It first generates and prints all successors of the root node
- It then generates nodes at depths 2 through 4 (by taking always the first available action) and prints then
- It then takes a solution to the problem prints it and applies the goal test to it.

All tests worked as desired.

2.2 SearchTester

To run the Search tester to do search on an instance of one of the problems navigate to the containing directory and type:

```
$java SearchTester <problemnanme> <strategy> <extra args>
```

The <problemname> argument can have the values:

```
Bananagrams
Hanoi
NPuzzle
Maze
```

The <strategy> argument can have the values:

- BFS : Performs Breadth firs search
- DFS : Performs Depth fist search with depth cutoff
- Greedy : Performs Greedy Search
- Astar : Performs A* search
- SMA : Performs simplified memory bound A* search

The extra arguments are only needed for 3 of the problems If <problemname> is Bananagrams then one extra argument with the order (an integer between 2 and 15) must be provided (please note that uninformed search with orders bigger than 6 and heuristic search with orders larger than 10 will take a long time)

If <problemname> is Hanoi then the number of pegs followed by the number of disks should be provided.

If <problemname> is Npuzzle then an extra argument: 3, 8 or 15 must be provided for the size of the puzzle.

Some examples to run searches are

```
$java SearchTester Bananagrams Astar 8

$java SearchTester Hanoi Greedy 3 5

$java SearchTester NPuzzle SMA 8
```

2.3 RunExperiment_Bananagrams

The third way to run an experiment is to use the RunExperiment_Bananagrams, this will bulk run the problems stored in /problems for a given order, it is run with the command:

```
$java RunExperiment_Bananagrams <order>
```

Where <order> is the size of the bananagrams you want to run

The /results folder contains the ourput for runs of all problems from the search tester and from running the bananagrams bulk experiment for orders 4 and 5.