**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

Batch No. :

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

**Artificial Intelligence (BITS F444/ CS F407)**

**I Semester 2018-19**

**Programming Assignment-1**

**Coding Details**

**(September 10, 2018)**

*Instruction: Type the details precisely and neatly*

1. ID : 2017H1030130P

Name : Santosh Kumar Desai

1. Mention the names of Submitted files :
   1. analysis.py
   2. api.py
   3. driver.py
   4. goalStateGen.py
   5. gui.py
   6. initialStateGen.py
   7. helper.py
   8. state.py
2. Total number of submitted files: 8 source code files + 112 pickle files.
3. Name of the folder : 2017H1030130P
4. Have you checked that all the files you are submitting have your name and ID (in comments) in the top?

YES

1. Have you checked that all the files you are submitting are in the folder as specified in 4 (and no subfolder exists)?

YES, no subfolder exists

1. Problem formulation
   1. State representation:

Array of boolean values. True represents presence of matchstick. False represents absence of matchstick.

* 1. How is the Initial state generated?

Programmatically through initialStateGen.py module.

* 1. What is the goal state?

Goal state is the required number of non -overlapping squares formed by the matchsticks derived from initial state through removal of one or more matchsticks and WITHOUT addition of matchsticks to the initial configuration.

* 1. Are there more than one goal states?

It is possible to have more than one goal state. The number depends on the size of the mesh configured initially.

* 1. How have you created the goal states for 1, 2 and 3 squares in the goal states? (manually/ automated)

It was automated via goalStateGen module.

* 1. Mention the numbers of goal states possible for 1, 2 and 3 squares separately.

The following is written w.r.t. 4x4 mesh

* + - For 1 square in the goal state: 30
    - For 2 squares in the goal state: 272
    - For 3 squares in the goal state: 1374

* 1. State representation in Python (name the construct and give one small example of a state)

For a 4x4 grid with 100% coverage,

state = [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]

1. coveragePercentage: In function initialStateGenerator (gridSize n, coveragePercentage p), what is your interpretation and usage of p? Does p refer to the percentage coverage area of the complete grid or percentage of maximum n2 squares?

I have taken percentage of maximum n2 squares as p. For example, a 4x4 mesh with p=50% will have 8 unit squares of area perimetered by the matchsticks.

1. Goal test: Describe the logic used in implementing goal test. Also describe any additional data structures used to store the goal states.

In the submitted implementation, the hash of current state array is checked whether it exists in pre-defined set of goal state hashes.

1. Time for goal test: Mention the time complexity of goal test implemented by you.

It is O(1).

1. Are you creating the goal states automatically every time you are applying the goal test? (yes/No) Why?

The goal state is prepared only once and pickled in a .pkl file for future use. When a program runs an algorithm, the .pkl is read once and fetched in a variable.

1. Define your understanding of a move in the given problem:

An action is performed on a given state, altering one parameter within the state, giving rise to new configuration of the state.

1. What is the branching factor (maximum)? 40
2. Successor function description

1. Uninformed Search Technique (T1) details
   1. Technique used for search:

DFS

* 1. Reason for selecting this technique over the other two:

Faster result when compared to others and gives an initial estimate of complexity involved. Also, DFS sometimes presents us with worst case depth of the search involved.

* 1. Is the search applied on match sticks, squares or on states?

It is applied on the states.

* 1. Error handling and reporting (yes/No):

Maximum care has been taken to avoid abrupt errors and release gracefully.

* 1. List the errors handled:

Improper initial configuration, improper goal selection, impossible event selection, session logging option on console.

* 1. Data Structure description for the tree node (in maximum two lines):

A class has been used to encompass all the necessary details. In the submitted code, it is called Node.

* 1. Code status (implemented fully/ partially/ not done)

Except for the generation of G3, every parameter has been coded.

1. Uninformed search Technique (T2) details:
   1. Technique used for search:

IDS

* 1. Reason for selecting this technique over others:

It is memory efficient when compared to BFS and DFS was already selected as T1.

* 1. Does this technique look at a square or a match stick?

This technique also looks at state.

* 1. Does this technique use a state?

Yes.

* 1. Code status (implemented fully/ partially/ not done)

Except for generation of G3, every thing else is coded.

1. GUI details
   1. Created the GUI : YES
   2. Have created it according to the specifications? YES
   3. Which module of Python is used for creating graphics? PYTURTLE/turtle graphics module.
   4. Is this under the standard Python library or not? YES
   5. If not, why?
   6. Are the window panes working independently? YES
2. Graphics details:
   1. Is turtle graphics working fine for removal of the match stick? YES, a red line denotes removal of matchstick.
   2. How are you creating the environment of the intelligent agent?

Initially a grey mesh is shown to inform a layout. Then blue mesh is created to show the initial configuration. Each removal of matchstick is denoted by a red line that swipes over blue line. A line, here, indicates a matchstick.

* 1. How are you showing the matchsticks?

In the initial configuration, each blue line indicates a matchstick in that position.

* 1. Are you showing the removal of a match stick graphically as per the action path produced by T1 ? Describe the turtle actions appropriately.

Yes. Every time a matchstick is removed, a blue line is replaced by a red line through swipe animation. It is present in the section labelled G1.

* 1. Are you showing the removal of a match stick graphically as per the action path produced by T2 ? Describe the turtle actions appropriately.

Yes. Every time a matchstick is removed, a blue line is replaced by a red line through swipe animation. It is present in the section labelled G2.

1. Compilation Details:
   1. Code Compiles : YES
   2. Mention the .py files that do not compile: There are no such files. However, only driver.py , analysis.py, goalStateGen.py run with main function. Rest are considered helper files.
   3. Any specific function that does not compile: No such function found during last test.
   4. Ensured the compatibility of your code with the specified Python version(yes/no) Yes. It runs with Python 2.7. However, it is recommended to check if the installed version contains pickle module as some versions of python 2.7 do not dispatch it with the compiler. Also, pyturtle MUST be installed.
   5. Instructions for compilation of your files mentioning the multi file compilation process used by you (We may use the replica of these for compiling your files while evaluating your code)

`python driver.py` is sufficient provided that current folder submitted is not altered with, initial state has less than 5 as grid size, goal states are 1,2 or 3 squares.

1. Driver Details: Does it take care of the options specified earlier(yes/no): YES
2. Execution status (describe in maximum 2 lines)

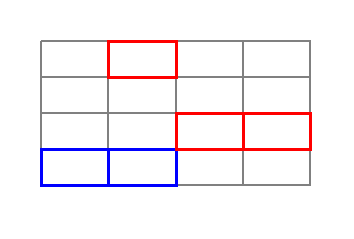
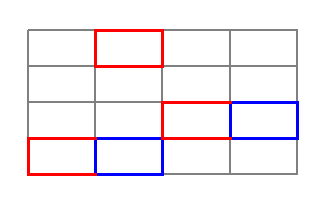
There is a show Progress option and I recommend using it to track progress.

The files are executing properly and exiting gracefully during the last integration test.

1. Output Details

The following is published for 4x4 mesh size with percentage of coverage as around 30%, goal state as 2 squares. It may differ for different initial configuration or different parameter setting (size,percentage of coverage and number of squares in goal state.)

* 1. Copy and paste the output of three graphs G1-G3 here

 G1 (after DFS) G2 (after IDS)

G3 couldn’t be generated programatically.

Write some more details here for the above graphs, if needed

They show that more than one solution is possible for a given config.

* 1. Write the following values computed by you (refer the details of R1-R11 in the assignment document). Use appropriate units for the values

R1: 395 nodes

R2: 440 bytes

R3: 154

R4: 11 units

R5: 0.37 seconds

R6: 218790 nodes

R7: 440 bytes

R8: 126

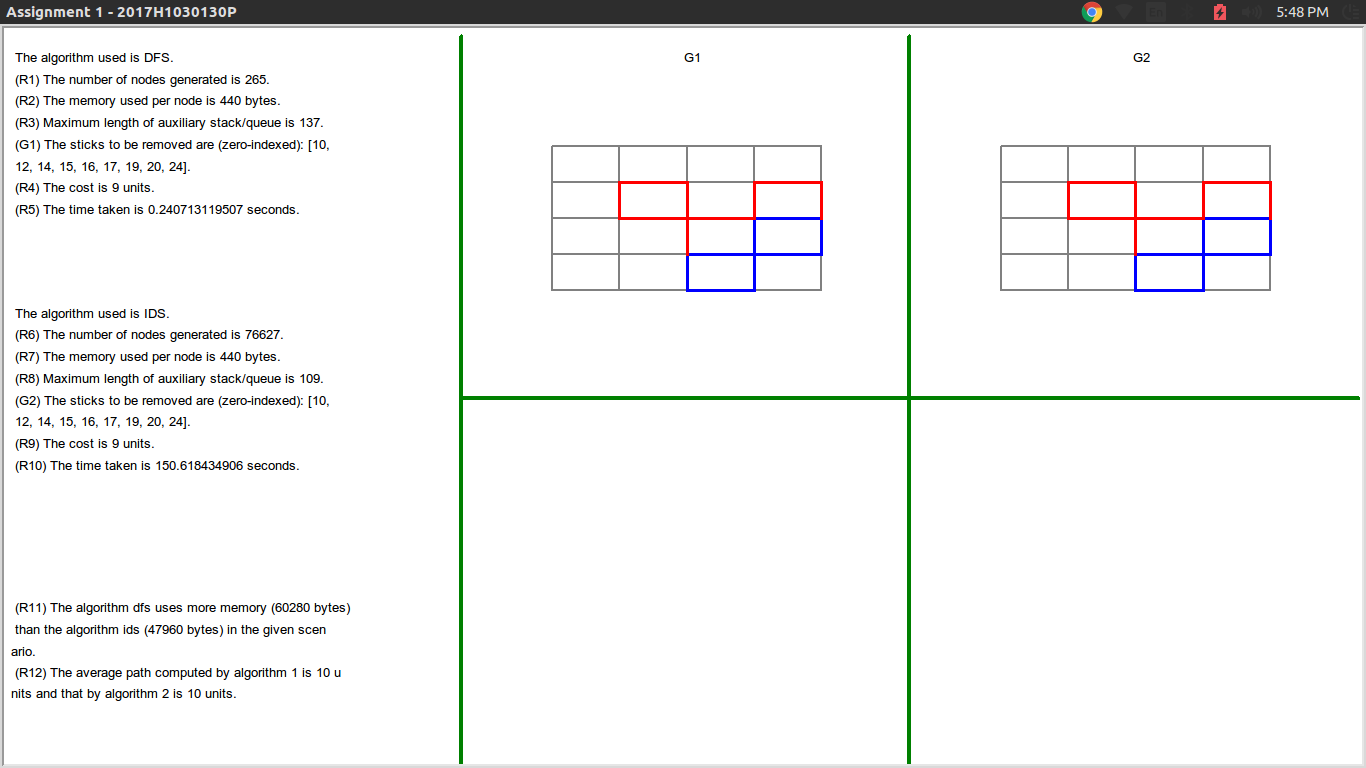
R9: 10 units

R10: 150.62 seconds

R11: DFS uses 67760 bytes of memory for nodes while IDS used 55440 bytes for nodes.

R12: Average cost in 10 random runs was 10.28.

Another run looked like this.



1. Declaration: I, Santosh Desai (name) declare that I have put my genuine efforts in creating the python code for the given programming assignment and have submitted only the code developed by me. I have not copied any piece of code from any source. If the code is found plagiarized in any form or degree, I understand that a disciplinary action as per the institute rules will be taken against me and I will accept the penalty as decided by the department of Computer Science and Information Systems, BITS, Pilani.

ID: 2017H1030130P Name: Santosh Desai

Date: 10-09-2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Should not exceed 5 pages