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

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS**

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

**I Semester 2019-20**

**Programming Assignment-5**

**Coding Details**

**(November 17, 2019)**

*Instruction: Type the details precisely and neatly*

1. ID \_\_\_\_\_\_\_\_\_\_\_\_\_2017A7PS0093P\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_AYUSH JAIN\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Mention the names of Submitted files :
   1. agents.py
   2. ass5.py
   3. csp.py
   4. file.txt
   5. flag.png
   6. KB-ruleFile1.txt
   7. logic.py
   8. output-{fname}
   9. output-query1.txt
   10. output-query2.txt
   11. predicateFile1.txt
   12. predicateFile2.txt
   13. predicateFile3.txt
   14. query1.txt
   15. query2.txt
   16. ruleFile1.txt
   17. ruleFile2.txt
   18. ruleFile3.txt
   19. search.py
   20. utils.py
   21. wumpusWorld-actions.txt
   22. wumpusWorld.txt
2. Total number of submitted files: \_\_\_\_\_22\_\_\_\_\_\_
3. Name of the folder :\_\_\_\_\_2017A7PS0093P\_ASS5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
4. Have you checked that all the files you are submitting have your name in the top?(yes/no) YES
5. Have you checked that all the files you are submitting are in the folder as specified in 4 (and no subfolder exists)?(yes/no) YES
6. Techniques implemented
   1. Forward Chaining (Yes/No) : \_\_YES\_\_\_\_\_\_\_\_
   2. Resolution (yes/ no):\_\_\_\_\_YES\_\_\_\_\_\_
7. Logic Problems solved manually
   1. Logic Problem 1 (Yes/No) : \_\_\_YES\_\_\_\_\_\_\_

List all queries handled by you:\_\_\_All the queries were handled but manually The problem is that many of the statements cannot be written in horn clauses which is a strong requirement for the forward chaining and resolution algorithms.

* 1. Logic Problem2 (Yes/No) : \_\_\_\_\_\_YES\_\_\_\_

List all queries handled by you:\_\_All the queries were handled but manually The problem is that many of the statements cannot be written in horn clauses which is a strong requirement for the forward chaining and resolution algorithms.

* 1. Logic Problem3 (Yes/No) : \_\_\_\_\_\_\_YES\_\_\_

List all queries handled by you:\_\_\_Based on the percept it can take the relevant action. \_\_\_\_\_

Are you able to show the path found by logical reasoning of the agent? (yes/no) : \_\_YES\_\_\_\_\_\_\_

1. Data structures used
   1. For defining predicate: (predicate\_name, list of args)
   2. Rules: rules taken as strings, parsed with predicates and symbols. Form\_cnf function parses and stores it in the form of logical expressions. The logic expressions given by python were overwritten.
   3. Knowledge base: Takes in definite clauses and parses them to convert into CNF Form.
2. Implementation Details for Forward Chaining technique
   1. Name the functions/ modules implemented by you: modified fc\_ask in logic.py, parsing done through form\_cnf and finally evaluated in query function.
   2. Which data structure(s) did you use to implement forward chaining for FOL? Explain. I used a list which is initially initialized to None. If any more inferences could be derived, I keep adding it until no clause is added or my rules are exhausted.

What is the time complexity of the technique? O(n \* m), where n are the number of clauses and m are the number of literals per clause because in each step atleast one literal should be removed from rules. The algorithm is semi-decidable and in the worst case can run into infinite loop due to the presence of functions, I am presently ignoring that case.

* 1. What is the space complexity of the technique? Space complexity is polynomial. The list gets bigger and bigger as I keep adding rules. I can add atmost n \* m rules.

1. Implementation Details for Resolution () algorithm
   1. How have you tagged variables, constants, lists and compound statements? All the variables are defined as op, var and const object. Hence they are checked using the isInstance functionality provided by python. List is also checked using isInstance method. Compound statements are just a list of lists and hence passed accordingly.
   2. Name the functions/modules that you implemented : modified pl\_resolution in logic.py and tested in query function.
   3. Which data structure(s) did you use to implement the algorithm? Explain. I used a list. From the rules in CNF, it identifies complementary literals and a new clause is added to the knowledge base, till no new clauses could be added or empty clause is produced.
   4. What is the time complexity of the algorithm? Polynomial time complexity, if we have literals in CNF.
   5. What is the space complexity of the algorithm? Polynomial space complexity
2. Files
   1. Submitted the files consisting the predicates description (yes/no):\_\_\_Yes\_\_\_\_\_\_
   2. Submitted the files consisting the rules in FOL (yes/no):\_\_\_\_\_\_\_\_Yes\_\_\_\_\_\_\_\_\_\_
3. Graphics: Created the graphics (yes/no)\_\_\_\_\_\_\_\_\_\_Yes\_\_\_\_\_\_\_\_
4. Compilation Details:
   1. Code Compiles (Yes/ No):\_\_\_\_\_\_Yes\_\_\_\_\_\_\_\_
   2. Mention the .py files that do not compile:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   3. Any specific function that does not compile:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   4. Ensured the compatibility of your code with the specified Python version(yes/no)\_\_Yes\_\_\_\_\_\_\_\_\_\_
   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 ass5.py
5. Driver Details: Does it take care of the options specified earlier(yes/no):\_yes
6. Execution status (describe in maximum 2 lines) Since the logic of many of the text was not convertible to first order logic even after hours of effort, and removing those lines were completely distorting the results (specially for wumpus world), I hardcoded the solutions I manually derived. Rest all is complete.
7. Declaration: I, \_\_\_\_\_\_\_Ayush Jain\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (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\_\_\_\_\_\_\_\_2017A7PS0093P\_\_\_\_\_ Name:\_\_\_\_AYUSH JAIN\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_17 NOV, 2019\_\_\_\_\_\_\_\_\_

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Should not exceed three pages