Programming Assignment 3: Constraint Satisfaction Problem

Problem Description

A college offers three programs: Program-A, Program-B and Program-C. In a given semester, a number of courses C_01, C_02, C_03… etc. is offered to the students. Each student in a program should pick up 6 courses out of which three have to be the 'discipline core' (DC), one 'general elective' (GE) and remaining two are 'Discipline Electives' (DE). Classes can be only from Monday to Saturday. Each Day, except Saturday, has 7 one hour slots: slot-1, slot-2, slot-3 ... etc. (Table 1). Only forenoon classes are held on Saturday.

	Forenoon					Afternoon		
	Slot-1	Slot-2	Slot-3	Slot-4	Lunch Break	Slot-5	Slot-6	Slot-7
From Time	09:00	10:00	11:00	12:00		02:00	03:00	04:00
To Time	09:50	10:50	11:50	12:50		02:50	03:50	04:50

Table 1: Time slots

The professors deliver the lectures and conduct labs. Each course is conducted for the specified number of lectures, tutorials and laboratory sessions (Table 2). The university needs to automate the time table scheduling so that it can allocate the courses to slots in such a way that all professors can teach and students attend classes without any clash.

Courses	Program A	Program B	Program C	L T P
C_01	DC	NA	DE	3 0 3
C_02	DC	NA	NA	3 0 3
C_03	DE	DC	NA	3 0 2
C_04	NA	DC	DE	3 0 2
C_05	NA	NA	DC	3 0 3
C_06	DE	NA	DC	3 0 3
C_07	DE	NA	DE	3 1 0
C_08	NA	DC	DE	3 1 0
C_09	DC	DE	NA	3 1 0
C_10	NA	DE	DC	3 1 0
C_11	GE	GE	GE	2 0 0
C_12	GE	GE	GE	2 0 0

Table 2: Course details

The above table depicts the courses with their categorization as DC, DE and GE for different programs. NA refers to not applicable, implying that the corresponding course is not included in the program. Also the information LTP refers to the number of lectures, number of tutorials and number of lab sessions per week. The college has 5 lecture halls H-1, H-2, H-3, H-4 and H-5. The laboratories are also 5 named as L-1, L-2, L-3, L-4 and L-5.

The professors teaching the courses are

Professor	List of courses taught by the professor			
Prof-1	C_01	C_08	NA	NA
Prof-2	C_09	NA	NA	NA
Prof-3	C_04	C_05	NA	NA
Prof-4	C_10	C_11	NA	NA
Prof-5	C_02	C_07	NA	NA
Prof-6	C_03	NA	NA	NA
Prof-7	C_06	C_12	NA	NA

Table 3: Professors and their courses

In this table, NA refers to not applicable, implying that the professor is not going to teach courses other than the ones assigned. The test case will be provided to you as a CSV file containing data as shown below.

```
$COURSES$,,,,
Courses,Program A,Program B,Program C,L T P
C_01,DC,NA,DE,3 0 3
C_02,DC,NA,NA,3 0 3
C_03,DE,DC,NA,3 0 2
C_04,NA,DC,DE,3 0 2
C_05,NA,NA,DC,3 0 3
C_06,DE,NA,DC,3 0 3
C_07,DE,NA,DE,3 1 0
C_08,NA,DC,DE,3 1 0
C_09,DC,DE,NA,3 1 0
C_10,NA,DE,DC,3 1 0
C_11,GE,GE,GE,2 0 0
C_12,GE,GE,GE,2 0 0
$PROF$,,,,
Professor,List of courses taught by the professor,,,
Prof-1,C_01,C_08,NA,NA
Prof-2,C_09,NA,NA,NA
Prof-3,C_04,C_05,NA,NA
Prof-4,C_10,C_11,NA,NA
Prof-5,C_02,C_07,NA,NA
Prof-6,C_03,NA,NA,NA
Prof-7,C_06,C_12,NA,NA
```

Students are advised to generate more test cases with number of courses as much as 50, number of rooms as 10, number of professors as 20 and so on, in which the details vary and you need to test the developed code for producing correct schedule.

List of Constraints

The time table follows certain constraints listed below:

1. **Laboratory** sessions cannot be held in the forenoon.
2. **Laboratory** sessions are consecutive for the given number of hours.
3. **Tutoria**l class can be conducted in forenoon or afternoon anytime, but cannot be held on the same day of the lecture of the corresponding course.
4. No two **disciplinary** core courses of a program can be in succession. For example, for students of program-B, the courses C-03 and C-08 cannot be scheduled at 10:00 am and 11:00 am respectively on the same day.
5. The **general** elective courses cannot be scheduled on the same day.
6. At the most one **lecture** for a course can be scheduled on a day.
7. The student strength in each program is divided in two batches for their lab sessions. Therefore, a **laboratory** session is required to be scheduled twice a week for two batches of a course.
8. **Professor** Prof-04 does not want to teach on Thursday while Professor Prof-01 can only teach during 9:00 AM to 11:00 AM.
9. No **professor** should have two or more lectures/ lab sessions in succession. This means that lectures or lab sessions to be taught by a professor should have at least an hour gap.
10. No **student** should suffer a clash of time slots for chosen course package.
11. No **professor** should suffer a clash of time slots for the courses to be taught by her or him.

Represent the above time table scheduling problem as a Constraint Satisfaction Problem (CSP) and solve the same using two techniques taught in the class namely- DFS (Depth First Search) with backtracking (DFS+BT) and DFS with constraint propagation (DFS+CP).

List the variables to represent the above problem and define their value domains. Draw the constraint graph for the given problem and specify the constraints as edges (or graphs as applicable) and use appropriate data structure to use the constraint graph for constraint checking. Design the modular functions for constraint checking.

Implementation

Use Python version 2.7.13 (Windows 10) for implementing your solution. Only standard Python Libraries should be used. Support from external sources or libraries such as github will not be accepted in your submissions. Each student must design own solution and write own code. [Refer handout to understand the malpractice policies.]

Modules

A CSP is represented by the list of courses, list of programs, list of categories, list of number of lectures, tutorials and labs, list of professors etc. Students are given below some prototypes to start with, but will have the flexibility of designing the constraint modules and corresponding prototypes on their own.

Preprocessing modules

1. **package generate_packages (testcasefile):** This function generates all possible packages for students of all programs. The same function must read the file to extract information about the list of courses and professor teaching the corresponding courses. The file provided to you is a CSV file which has two tags `$COURSES$` and `$PROF$` which start the contents of Tables 2 and 3 respectively. The output package must be a collection of two sets: one of students and another of professors. Example of student packages in the given test case file for program A is given below in the order of DC, DE and GE:

   {C_01, C_02, C_09}, {C_03, C_06}, {C_11}
   {C_01, C_02, C_09}, {C_06, C_07}, {C_11}
   {C_01, C_02, C_09}, {C_03, C_07}, {C_11}
   {C_01, C_02, C_09}, {C_03, C_06}, {C_12}
   {C_01, C_02, C_09}, {C_06, C_07}, {C_12}
   {C_01, C_02, C_09}, {C_03, C_07}, {C_12}

   Similarly other packages for other programs can be obtained.

2. **CG Constraint_Graph (CSP, package):** This function takes as input the generated packages and the complete details of the CSP such as the data given in table 1, 2 and 3. The function uses the data to generate a constraint graph. The graph can have edges defining binary constraints. If the constraints are involving more variables, you can represent the constraint hypergraph while the constraint checking later will be through the different functionalities. The function populates the constraint graph CG and returns it for further processing.

Constraint modules

Each constraint module takes as input the CSP details, the constraint graph CG and must return a boolean value

1. **boolean Laboratory_constraint_1 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-1 above is violated on assignment of a value to from_node for the given CSP.
2. **boolean Laboratory_constraint_2 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-2 above is violated on assignment of a value to from_node for the given CSP.
3. **boolean Tutorial_constraint_3 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-3 above is violated on assignment of a value to from_node for the given CSP.
4. **boolean Disciplinary_constraint_4 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-4 above is violated on assignment of a value to from_node for the given CSP.
5. **boolean General_constraint_5 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-5 above is violated on assignment of a value to from_node for the given CSP.

6. **boolean Lecture_constraint_6 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-6 above is violated on assignment of a value to from_node for the given CSP.
7. **boolean Laboratory_constraint_7 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-7 above is violated on assignment of a value to from_node for the given CSP.
8. **boolean Professor_constraint_8 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-8 above is violated on assignment of a value to from_node for the given CSP.
9. **boolean Professor_constraint_9 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-9 above is violated on assignment of a value to from_node for the given CSP.
10. **boolean Student_constraint_10 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-10 above is violated on assignment of a value to from_node for the given CSP.
11. **boolean Professor_constraint_11 (CSP, CG, from_node, to_node)**: This returns a false value if the constraint-11 above is violated on assignment of a value to from_node for the given CSP.

Techniques modules
Implement the following techniques. The techniques apply DFS on the list of variables as discussed in the class. Read the chapter on CSP to revise the concepts and relate to the time scheduling problem solved in the class.

1. **Schedule DFS_BT (CSP, variable_list, constraint_graph)**: Takes as input the CSP problem description, a list of variables and the constraint graph and returns a schedule of allotted time slots and room numbers for lectures, tutorials and lab sessions.
2. **Schedule DFS _BT_ Constraint_Propagation (CSP, variable_list, constraint_graph)**: Takes as input the CSP problem description, a list of variables and the constraint graph and returns a schedule of allotted time slots and room numbers for lectures, tutorials and lab sessions.

Analysis Module
Produce the following analyses and display the resultant values.

(a) DFS+backtracking algorithm based analysis

  i. Compute the number of nodes generated till the problem is solved. **[R1]**
  ii. Compute the amount of memory allocated to one node. **[R2]**
  iii. Compute the maximum growth of the implicit stack (if recursion is used) or of explicit stack used with the search tree. **[R3]**
  iv. Compute the total time to compute the schedule. **[R4]**

v. Use an appropriate heuristic such as MRV or degree heuristic and compute the number of nodes generated till the problem is solved. **[R5]**

vi. Write the schedule in file names as testcase#_DFS_BT.txt where # is the testcase file number. The format for the output is specified below.

(b) DFS+ backtracking using constraint propagation algorithm based analysis

i. Compute the number of nodes generated till the problem is solved. **[R6]**

ii. Compute the ratio (R1 - R6)/R1 as saving using constraint propagation. **[R7]**

iii. Compute the total time to compute the schedule. **[R8]**

iv. Write the schedule in file names as testcase#_DFS_BT_CP.txt where # is the testcase file number. The format for the output is specified below.

(c) Comparative analysis

i. Compare R4 and R8.

Sample format for the output file should be as follows (sample entries not validated below, given only as an example). Create this file as a csv file with an extension of .csv, readable using MS excel software.

Courses	Professor	Day	Time	whether Lecture/ Tutorial/Lab ?	Room number (H-# or L-#)
C_01	Prof_1	Monday	10:00AM	Lecture	H_1
C_04	Prof_3	Monday	11:00AM	Lecture	H_2
C_10	Prof_4	Tuesday	12:00PM	Tutorial	H_2
C_01	Prof_1	Monday	2:00 PM	Lab	L_2
C_12	Prof_7	Wednesday	12:00PM	Lecture	H_2

Driver

The driver must integrate all functionalities and execute the functions appropriately using these options

Option 1: Display the packages.

Option 2: Display the constraint graph appropriately.

Option 3: Execute DFS+BT with and without the heuristic. Prompt for the choice of use of heuristic first. Show the output timetable in the format specified above on the console.

Option 4: Execute DFS+BT+Constraint_Propagation with and without the heuristic. Prompt for the choice of use of heuristic first. Show the output timetable in the format specified above on the console.

Writeup, evaluation and submission

Write up details will be made available two days before the submission. Evaluation will be out of 15 marks (5% weight). Students are advised to inform me immediately, if any discrepancy exists in this document. The assignment is due for submission on October 26, 2017 (Thursday) by 7:00 p.m. The countdown for the 10 days time for this assignment will start from October 16, 2017 (Monday) after the mid semester tests. However, the document is made available early for the students to clarify all their doubts pertaining to the problem specification, explanations given above, conceptual understanding, doubts related to constraint graph or the data structure to be used, and the doubts relating other aspects of implementation. Students are advised to work out the details, decide upon the variables and corresponding value domains, draw the constraint graph on paper first and then dry run the algorithms as discussed in the class. All the doubts must be clarified before October 8, 2017 (Sunday), i.e. before the start of the mid semester test. Please note that I will not be available in campus during October 15, 2017 to October 22, 2017.

Please feel free to write me an email for a mutually convenient time for discussion.

Vandana
October 5, 2017