CS255 Artificial Intelligence Coursework: The show must go on (at the scheduled time)

October 30, 2020

In this assignment, you are tasked with organising the timetable for an up-and-coming Comedy Club. You will be given a list of comedians and their typical themes (politics, family, satire etc.), and you must organise them over the course of a week. Each week, the comedy club wants to appeal to a number of demographics, which each have their own particular topics they like to see in comedy, and you must assign comedians and demographics to slots in the timetable, based on matching the comedian's themes to the demographic's topics.

You will produce three timetables, one that considers demographics and comedians, another that introduces test shows and a final one that considers the cost of hiring these comedians to perform their sets. This assignment document will discuss the requirements of the timetable and how to submit your solution. Your solution will be written in Python 3 and must run on the DCS systems. **The deadline for this coursework is 12 noon, Monday the 18th of January.**

1 Getting Started

To begin, download the zip file from the CS255 web page, it can be found underneath the heading 'Coursework'. Inside of this zip file, there are a number python files and a folder of text files, labelled 'ExampleProblems'.

These python files provide everything needed to load a set of resources, and develop your own strategies for creating a valid timetable. The ExampleProblems folder contains several text file representation of comedians and demographics that can be used to test your solutions.

There is also a readme.txt, which explains the purpose of each file. Please read through the readme.txt file carefully.

The two main python files that you will use in this assignment, runScheduler.py and scheduler.py, both have preambles and comments, discussing how to use them.

We have also provided a method that creates a random schedule, but does not check if it is legal. This demonstrates how to use the main methods and objects needed for this task.

1.1 runScheduler.py

This file is provided to allow you to test your strategies. Several pre-designed problems are included, allowing you to test your solution in several settings. To run this file, you will need to use the command `.\runScheduler.py` or `python3 runScheduler.py`.

This will load the Scheduler class found in scheduler.py and use the createSchedule method in that class to provide a legal assignment of comedians to demographics and show slots. It will output a Timetable object, which will then be checked to see that it satisfies every constraint.

In addition to changing the problem number that is loaded when tested, you may also wish to change the scheduler method called from createSchedule to createTestShowSchedule or createMinCostSchedule, to allow you to test your solution for all three tasks.

1.2 scheduler.py

This file contains the Scheduler class, and the createSchedule, createTestShowSchedule and creatMinCostSchedule methods. These methods are where you will create your schedulers to fulfil the tasks described below. The preamble provided discusses the methods from other classes you are allowed to use in your solution. Importantly, you may only use the modules already imported into the file, which includes the python math and random libraries. Use of any other libraries, or methods from the provided files not explicitly stated in the preamble, will result in a penalty.

Your createSchedule, createTestShowSchedule and createMinCostSchedule methods should assign a comedian and a demographic to each slot available. This timetable object should then be returned at the end of your methods, so that the timetable can then be evaluated to ensure that it follows the requirements discussed below.

2 Timetable Scheduling Programming Tasks

You will be given a list of comedians and a list of demographics. Each comedian consists of a name, and a list of themes that they cover in their shows. Each demographic consists of a reference code (two capital letters followed by a 3 digit number), and a list of topics they like to see in comedy shows. The timetable covers the 5 weekdays, Monday, Tuesday, Wednesday, Thursday and Friday. For task 1, each day has 5 slots, numbered 1 to 5. For task 2 and 3, this is extended to 10 slots, numbered 1 to 10.

You will be making three different schedulers, as described below.

At the top of each of the three schedule methods (createSchedule, createTestShowSchedule and createMinCostSchedule), you should include a short preamble that describes your approach. These preambles should explain the theory behind your approach and briefly justify its use. These should not be very long, but should present a strong case for the approach, and highlight the principles of artificial intelligence that underpin the decisions you have made.

2.1 Task 1: Basic Scheduling (20% of assignment)

In this task, you must complete the createSchedule method in scheduler.py, to produce a schedule that adheres to the following requirements:

- Each demographic must be assigned to a slot.
- Each slot requires a comedian to be assigned to it. Assigning a comedian and a demographic to the same slot means that the comedian will perform a show in that slot, and the show will be marketed toward that demographic.
- A comedian can only be marketed toward a demographic (and thus, can only be assigned to the same slot) if their themes contain every topic that demographic is interested in.
- A comedian can only perform a maximum of two shows a week.
- A comedian cannot perform more than one show a day.

Hint: Consider the techniques discussed in the lectures, such as backtracking and how they may help with this task.

2.2 Task 2: Introducing Test Shows (20% of assignment)

Thanks to your packed timetable of events produced previously, the comedy club can afford to put on extra shows. Each day can now have 10 shows. Furthermore, there are now 'main' shows and 'test' shows. Main shows are two hour long comedy sets with the comedian's best material, while the test shows are an hour long set, where a comedian may try out new jokes.

Test show tickets are, naturally, cheaper and the audience does not go in expecting every joke to be to their taste. However, people like to attend them for a chance to see new material early. The comedy club is cautious about spamming their demographics with marketing, and so wants to limit marketing to one main show and one test show for each demographic each week. You must now create a scheduler that adheres to the following requirements:

- Each demographic must be assigned one main show and one test show.
- Each show, main or test, requires a comedian to be assigned to it.
- A comedian can only be assigned to the main show for a demographic if the comedian's themes contain every topic the demographic likes.
- A comedian can only be assigned to the test show for a demographic if the comedian's themes contain at least one topic the demographic likes.
- A comedian can only perform in 4 hours of show a week. Main shows are 2 hours long, test shows are 1 hour long.
- A comedian cannot perform in more than 2 hours of show a day.

- A comedian can perform only main shows, only test shows, or a combination of the two.
- The shows marketed to a demographic can either involve two different comedians, or the same comedian.

You must complete the createTestShowSchedule method in scheduler.py with your solution to this problem.

2.3 Task 3: Cost-effective Scheduling (60% of assignment)

In this task, you must now consider the cost of a schedule. To hire a comedian to perfom in a single main show costs £500. If that comedian is hired for a second main show, the second main show will only cost £300. If the two main shows are on consecutive days, the second main show only costs £100. As such, it is preferable to hire a comedian to perform in two main shows, and place them on consecutive days.

To hire a comedian to perform a single test show costs £250. Each subsequent test show a comedian performs costs £50 less than the preceeding one, so the second show will cost £200, the third £150 and the fourth £100. Furthermore, a test show that is performed on the same day as something else the comedian is performing, main show or test show, has its cost halved. This means if a comedian is performing in two test shows on the same day, they would cost £125 and £100 respectively instead of £250 and £200.

You must complete the createMinCostSchedule method in scheduler.py, to produce a schedule of the minimal possible cost, while adhering to the constraints laid out in task 2.

Hint: You should consider a heuristic based approach, think of what you can use to measure closeness to the optimal solution.

3 Submission

You will submit one file to tabula. It should contain the Scheduler class, with the createSchedule, createTestShowSchedule and createMinCostSchedule methods defined. Please rename the file to uniID.py, e.g. 1003685.py.

Ensure that your submitted file can solve each task, does not use any methods that are not permitted in the preamble and contains no additional imports.

Please make sure you submit your coursework by 12 noon, Monday the 18th of January.