MATH0462 - Discrete Optimization

In this project, the problem is related with the scheduling of exams in order to obtain the best possible schedule for the students. More specifically, we are given a list of students with the list of exams that each student is supposed to take. Simultaneously we are also given, for each exam, the type (written or oral) and, when applicable, the number of students maximum per day of oral exam. One hard constraint that needs to be considered is the fact that no student can have two exams on the same day.

Two problems are stated in this project. You can choose one on which to work. We now specify the two problems.

1 Series assignment

Before assigning days to exams, it is important to assign students to series of oral exams. Since the students can have very different lists of exams that they take, it is not trivial to assign students to series of exams in such a way that the series are as homogeneous as possible.

In this problem, the input is the list of students with their lists of exams that they take. The output that is expected is the same list of students, where each course code for which the exam is oral is replaced by the same course code, appended by two characters: an S and a number indicating the series assigned to the student.

The difficulty in this problem is to be able to formulate the right objective function, knowing that we want to make an assignment of students to series that will lead to the best exam scheduling.

If you choose this problem, you have to produce an integer programming formulation of the problem, be able to run the formulation with `julia` and `JuMP` with a MIP solver of your choice. If the formulation does not run to optimality in less than an hour, you must implement a trick in order to obtain the best possible solution in less than an hour. Besides the integer programming formulation, you must also implement a heuristic that must give the best possible solution to the problem in two minutes.

2 Computation of the schedule

Given the list of courses followed by each student, and for each oral exam, in which series the students are assigned to, the goal is to establish a schedule, i.e. to assign days to exams in such a way that no student has ever two exams on the same day. Another hard constraint is that two series of the same oral exam cannot occur on the same day. If these are the only constraints that are

imposed, it is very likely that the schedule that will come out will be extremely inconvenient for some students (like for example all exams in the beginning and nothing at the end).

Your task in this part is to provide a model that yields a schedule that is acceptable for most students. The main part of the work is to provide a criterion that both leads to a good schedule and to an integer programming model that can be solved in a reasonable amount of time.

If you choose this problem, you have to produce an integer programming formulation of the problem, be able to run the formulation with `julia` and `JuMP` with a MIP solver of your choice. If the formulation does not run to optimality in less than an hour, you must implement a trick in order to obtain the best possible solution in less than an hour. Besides the integer programming formulation, you must also implement a heuristic that must give the best possible solution to the problem in two minutes.

3 Files

Two instances are provided for you to test your program. The files that are provided are explained below.

- Files `cursus` are csv files. The first column of the file gives an integer that represent the students. The next two columns identify the courses. Finally, the fourth column indicates `Q1` if the exam is organized in January and `Q2` if the exam is organized in June.
  There are two cursus files. The large file should be used. The small file can be useful to test a smaller instance.

- Files `infoCourses` give information on the organization of exams. The first column is the identifier of the course. The fourth column indicates the type of exam. Only oral and written exams need to be organized. The fifth column is only useful for oral exams and indicates the maximum number of students that can take the exam on a given day. If no number is indicated, we can consider that the default value is 24 students maximum in a day. There is a file for the exams to be organized in January and a file for the exams to be organized in June. These two files therefore lead to two possible insatnces of the problem. Note that the session has to be organized in three weeks in January (with possible exams on Saturdays but no exams on Sundays) and four weeks in June (with possible exams on Saturdays but no exams on Sundays).

Note that a course that is present in the `cursus` file but not in the `infoCourse` file has no exam organized. Similarly a course that is present in the `infoCourse` file but that has no student following it has no exam organized.

4 Instructions

You work by groups of two. Groups of three or one can be accepted with a formal request.

You must choose one problem to work on. If you work on problem 1, your code should be able to produce outputs that can be read by the groups working on problem 2. Similarly, if you work on problem 2, your code should be able to read outputs of groups working on problem 1.

You must provide an integer programming model and a heuristic. The integer programming model must be written using `julia` and `JuMP`. The heuristic can be implemented either in `julia` or in `C`.

There is no formal report to be submitted. However, I ask you to provide a readable version of the integer programming model, namely a sheet with a short explanation of the variables and the constraints in mathematical form. This should not exceed a couple of pages.

The `julia` (or `C`) codes must be submitted by **December 12** by sending an archive by e-mail to `q.louveaux@ulg.ac.be`. You must show your code running on December 15.