Project Report

Constraint Programming

Mohamed Ahmed Ali
28-2229

Omar El-Sabrout
28-11398

Mohamed Ahmed Zaki
28-11392

Inputs variables

```
schedule(Curr, History, Oblig, ObligCH, CredLimit, Schedule,
                                Prob, Sched_Days, VarSchedule):-
```

Curr:

Stands for curriculum and is a list of tuples. Each of which contains the name, credit hours and prerequisites of a certain course.

History:

Is a list of tuples of course names and grades, which represents the history of the student.

Oblig:

Stands for Obligatory and is a list of obligatory course names.

ObligCH:

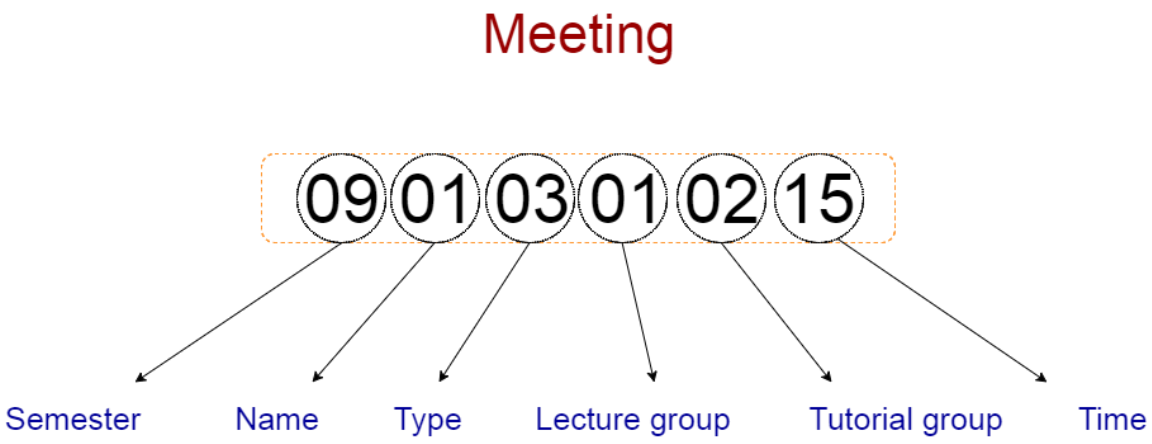
Stands for Obligatory Credit hours and is a number representing the the credit hours of the Obligatory courses.

CredLimit:

Stands for credit hours limit and is the maximum credit hours that the student may take.

Schedule:

Is a list of a serial numbers representing meetings as shown in the following diagram:



Prob:

Stands for Probation and is a boolean showing whether the student is on probation or not.

Sched_days:

Is a number representing the maximum number of days that contains meetings.

Output Variable

VarSchedule:

As shown below, VarSchedule is a list containing the chosen meetings that meets all of the requirements and tries to satisfy the soft constraints as much as possible. It is the only output of this project.

```
VarSchedule = [110603020344, 110602020343, 110601020042, 110503020334, 110502020333,
               110501020032, 110203020324, 110202020323, 110201020022, 110103020314,
               110102020313, 110101020012]
```

Important Variables

TCN:

Is a list of the taken course names.

TCH:

Is a list of the taken course credit hours.

Gaps:

Is the number of gaps found in a given schedule. This variable is minimized while labeling to prefer schedules with less gap slots.

TGroupVar:

Is a number representing the difference between the lab group and the tutorial group of the same course. This variable is minimized to prefer schedules with the same tutorial group for the labs and the tutorials of the same course.

SGroupVar:

Is a number representing the difference between the lecture groups and tutorial groups of all of the courses in the same lecture. Minimizing this variable will allow the program to prefer schedules with meetings that have same lecture and tutorial groups.

Labels:

Is a binary list that has a domain between 0..1. This list is labeled to choose courses with the maximum credit hours less than the credit limit.

TotalCH:

Is a number with domain (0..40) representing the total credit hours of a chosen number of courses. Maximizing this number will allow the program to choose the courses with the maximum total credit hours less than or equal the limit.

Predicates

is_attended:

Loops on the given history of the student and removes the courses that have a grade of "FA".

member_list:

Checks if the first list is a subset of the second list.

curr_filter:

Filters a given curriculum by looping on its courses. For each course curr_filter does:

1. Add obligatory courses to the filtered curriculum.

2. If the course is not obligatory, the prerequisites of the course is compared to the student's history.

3. If the prerequisites is a subset of the history, the course is added to the filtered curriculum.

4. else, the course is disregarded.

course_names:

Loops on given list of courses and returns a list of names of these courses.

sum_list1:

Loops on a given list of integers and returns the sum of these integers.

corr_courses:

Given a binary list and a list of courses, corr_courses returns a list of courses that have a corresponding value of 1 in the binary list, and omits the courses with a corresponding binary value of 0.

all_sum:

Given a list of courses' credit hours, all_sum creates a randomized list of binaries that represents the courses taken, and calculates the credit hours of the taken courses.

max_credits:

Aims to choose the courses that maximizes the value of credit hours without exceeding the credit limit given. It does so by introducing a binary array called Labels, with the same length of the courses list, which represents the courses taken by value of 1 and courses neglected by value of 0. It then calculates the Total credit hours of the chosen courses. The clpfd predefined method Labeling is used to assign values to the unbound binary list (Labels), while maximizing the total credit hours of the chosen courses.

cal_oblig:

Removes the obligatory courses from the Course names list , the course credit hours list and credit limit.

sub_list:

Removes the first list elements from the second list and output the resultant list.

sched_filter:

Given the schedule of all of the courses and a list of the chosen courses, sched_filter predicate filters the schedule by removing all the courses not included in the chosen courses list.

analyze:

Divides a given meeting into Semester, Course name, Lecture group, Tutorial group and Time(Day, Slot).

divmod1:

Divides a given integer by a given divisor, then outputs the Quotient and the remainder resulted.

divide_sched:

Divides schedule into list of lists. Each list contains all the meetings of a single course with a certain type.

place:

Places a given meeting in a given list of lists resulting in an updated list of lists.

create_PSched:

Randomly choose a meeting from every list.

constraints:

Applies different constraint on the final schedule.

unique_times:

Assures that the timings of every meeting in the final schedule are distinct.

get_times:

Gets the timings of every meeting in the final schedule.

day_off:

Assures that the schedule contains at least one day off.

get_days:

Returns a list containing the occupied days in the final schedule.

day_helper:

Assigns 1 in the list L for every day in the schedule.

tut_lab:

Assures that each tutorial precedes labs of the same course.

tut_lab_helper:

Compares a given meeting with a list of meetings to assure that the timing of each tutorial precedes each lab of the same course.

same_group:

Adds a soft constraint, which prefers schedules with same tutorial group.

same_group_helper:

Compares a given meeting with a list of meetings to calculate the difference between the tutorial groups in a given schedule.

get_gaps:

Gets the number of gap slots in a given schedule.

same_sem_grp:

Adds a soft constraint, which prefers schedules with less difference in the tutorial and lecture groups in a specific semester.

same_sem_group_helper:

Compares a given meeting with a list of meetings to calculate the difference between the tutorial groups and lecture groups in a given schedule.

schedule:

The main Predicate that calls all of the aforementioned predicates. It takes as input the following: the curriculum, history of the student, obligatory courses, obligatory credit hours, credit hours limit, schedule containing all the courses, whether the student on probation or not , the maximum number of days and outputs the final schedule.

Final Note: While running the final tests of the project, we found that the library “pyswip” is not thread-safe, and creates an error while running a prolog query through it inside a web app (Django). To sum up, the viewing of the results of the prolog file is not running right now, however it can be viewed in python console.