Constraint-Based University Compensation Reservation for the German University in Cairo

Baher Abdelmalek, Nada Ibrahim, and Hagar Mosaad

German University in Cairo, Cairo, Egypt

Abstract. Reserving a compensation for a skipped session at the German University in Cairo has been a cumbersome process for all instructors. Previously, an instructor needed to manually collect information about the set of suitable times, the group size and try to reserve a room given that information. In this paper, we provide a model for the corresponding constraint satisfaction problem and demonstrate an implementation using *SWI Prolog* and the library *clpfd* which applies both hard and soft constraints in order to reach an optimal solution. Given a certain instructor, group, and a set of preferences, the user is provided with a reservation entry with the greatest possible similarity with the preferences. On top of the constraint solver, we also implement a java program that includes both a Web service and a data processor.

Keywords: constraint programming, clpfd, scheduling, prolog, logic, declarative programming

1 Introduction

Reserving a compensation slot for a skipped or an unattended class has been a tedious problem at the German University in Cairo for the past previous years ever since it started. Generic solutions or systems that were previously implemented, or used in other universities do not really satisfy the full variety of constraints our problem has. Therefore, until now the compensation system at the German University in Cairo is done manually, where teacher assistants as well as professors have to search for a free room or a lecture hall that could be available at a certain slot of a certain day and date. Moreover, that assigned slot must fulfill the teacher assistants or the professors schedule as well as their students schedules.

The searching task is a very hard and tedious task to do and the output in many cases is unreliable and could result in many conflicts; especially for a huge number of students with different schedules and timetables. Accordingly, the main objective behind this project is to attempt to implement a tailored system that suits the requirements and constraints of the compensation system at the German University in Cairo.

The paper is organized as follows. Section 2 introduces the compensation system problem description at the German University in Cairo, explains it and discusses all constraints associated with it. In Section 3, we describe how the

problem can be modeled as a constraint satisfaction problem. In Section 4, we give an overview of our implemented system. Finally, we conclude with a summary and directions for future work.

2 Problem Description

Coming close to the date of any official holiday, or even a class that should be skipped at its original time and to be compensated in any other time; teaching assistants and doctors should go through the tough process of finding a specific time for that compensation class. At the German University in Cairo ever since it started, the process of finding a compensation slot was done by the teaching assistant or the professor manually. Therefore, the teaching assistant or the professor should check their schedules for a free slot, then find an empty room, lab or lecture hall that could be free at that certain day and slot in that specific week. Moreover, the teaching assistant or the professor should check that the selected slot would fit in all the scheduled of all the groups that he/she would like to make a compensation for. Hence, making scheduling a compensation manually to a large group of student, a group of several different tutorials and handling the special cases such as the different starting semester time and exam times for first year student and having to deal with the schedules of the advising students that take different courses with different studying years would be a very hard and tedious task to accomplish. Moreover, the output of that process are in most cases unreliable and inconsistent, which could result in many clashes between different courses or different schedules.

Consequently, an attempt to implement a tailored system that suits the requirements and constraints of the compensation system at the German University in Cairo is the main aim of this project. This aim is accomplished through modelling the compensation system problem as a constraint satisfaction problem using mainly the prolog Constraint Logic Programming over Finite Domains (CLPFD) library. The system should consider the hard constraints of the problem that must be satisfied as well as soft constraints including the preferences of the teaching assistant or the professor, in order to output the best solution that satisfies both as much as possible.

3 Implementation and Modeling

The design of our model is based on an input to the constraint solver, then should result with the best output according to given input and constraints. The input consists of the given data which include the occupied slots of every staff, tutorial and group. Also, the occupied slots of every room and the officials holidays are given to the solver in the form of list of inputs.
To search for a compensation slot, the staff member should enter his ID, the name of the group and the preferred days, slots and location. Then using the backend, the occupied slots of the staff member, the group are processed and

given to solver in the form of list of inputs. The resulted output is in the form of tuple of the possible week, day, slot and room to be reserved.

3.1 Design of the Input

The data of the staff member is given in the form of tuple consist of list of the occupied slots of this staff member, list of the his compensated slots and list of the off days. The occupied slots list consists of tuples of week, day and slot. Similarly, the compensated slots is list of week, day and slot. The days off list is a list of integers which imply the number of the days.
The data of the group is formed as a tuple of list occupied slots, list of compensated slots for this group, list of the off days and the size of this group. The data of the rooms are given in the form of list of lists, the lists are list of the IDs of the rooms, list of the corresponding locations of every room, list of their capacities, list of their types list of the occupied slots and list of the compensated slots. The list of the occupied slots of the rooms consists of integers each one represent the mapping of a binary representation of the occupied slots where every bit represent a slot in the week.
The preferences of the staff are given as a tuple of list of preferred times, preferred room type and list of preferred locations. The holidays are given as list of week, day and slot. Also the desired start week and day are given to the solver.

3.2 Design and Domain of the Output

The output is in the form of tuple of the possible week, day,slot and the ID of the room. The week has domain from 1 to 16, the slot has domain from 1 to 6 and the domain of the slot is from 1 to 5. The IDs of the room has domain from 1 to the number of the rooms in the data. All the possible results are given by the solver, with the arrangements that satisfy the preferences as much as possible. A cost predicate is responsible for the arrangements the output results according to the preferences and the given input.

3.3 Design and Implementation of the Constraints

The constraints are designed so that no conflicts happens between the output compensated slots and the occupied and compensated slots of the rooms, the group and the staff member. There are hard constraints such as no conflicts between the output and possible given slots, the labs are not given in the class rooms. While there are other soft constraints such as preserving weekly off-days of the groups and the faculty members, make sure resources are used in an optimal way. For example, a tutorial should not be held in a lab unless that would make the instance solvable. The capacity should be suitable to the size of the group.

Group & Staff Member Session Overlap The group should not have two sessions at the same time, so the resulted compensation time should not be member of the compensation and occupied slots of this group. an implemented predicate check this constraint. Similarly, the output compensated time is constrained to be not member of the the compensations and occupied slots of the staff member.

Room Resources Constraint The output compensated room should be available at the output compensated time, this is checked using a predicate which check that the compensated week, day, slot tuple is not member in the list of the compensations and occupied slot of this room. The predicate use refined constraints which results zero if the this compensated time is not member in the occupied and compensations lists and one otherwise. The room type is also checked, so that if the type of required compensation room is lab, the type of the output room should be lab. Otherwise, the type of the room will be handled according to the preference by giving priority to the arrangement of the preferred types as a soft constraint.

Handling the Soft Constraints In order to handle the soft constraints, a cost predicate is implemented to calculate the cost of each possible compensated time according to the given preferences and the given data of the holidays and the weekly off-days.

The preferred times are having priority by giving them least cost. An implemented predicate, check if the compensated time is member of the preferred times list, so the index of the time in the list will be given as the cost of this time. Otherwise, the cost is calculated by checking if the compensated time is in the group or staff member days off and the difference between the compensation time and the start week and day.
The preferred locations are handled by adding the index of the location of the compensated room from the list of the preferred location to the cost of the compensation time. Also the cost of the group size capacity compared to the group capacity is added to the total calculated cost.
The labeling predicate is responsible for arranging the output according to the cost of every result. This can be done by unifying the variables and calculating the cost according to this unification and the output the results with the arrangements of the minimum cost.

3.4 Processing the Database

The data of the group, staff members and rooms are given in the form of CSV files. These files are processed using java, and the the information are stored in static hashtables and lists. Class is implemented to read the CSV files and return array lists of lines of these files. Then a processing class is used to process these lines by looping through these data and saving them in static hash tables and lists. For every, group, its ID, occupation slots, compensation slots, size and days

off are saved in the hash table. similarly the data of the staff members are saved in hash tables. The data of rooms are saved by linking every room ID with its compensation slots, occupied slots, capacity and locations. The for every group, its tutorials are saved in the hash tables.
By giving, the ID of the staff member, the name of the group and the preferences, their corresponding data are processed and given to the solver with the input format.

4 Results

By using a simple front end, the staff member can enter his ID, the group which need compensations and the preferred week, days, slots, room type and location. Then these data are retrieved from the front end, and by using server, the data are processed to java to generate the knowledge base to the prolog. The the server call the method which generate the solver query using system call. The output of the query are saved in a CSV file and processed to the front end to give the staff member the best possible compensation time and location. The chosen compensation time are updated in the compensation lists of the group, staff member and room to be used in the further queries.

5 Conclusion and Future Work

In this paper, we have represented how the compensation scheduling system at the German University in Cairo can be modeled as a constraint satisfaction problem. For implementation, we have used the constraint logic programming *SWI Prolog*. The use of the declarative logic paradigm provides the flexibility of further maintenance and modification.

Possible future enhancements include a better formulation of the cost function of each given solution. The cost could be calculated in such a way to satisfy the judgment criteria used by most instructors, based for instance on a survey methodology.

Moreover, the preferences list could extend to include other fields as the time preferences for the group, time slots to avoid, a dynamic number of time preferences for the instructor, etc.

The web-service could also encompass the option of rejecting a reservation and getting the solution with the next least cost. This could be simply done by a minor adjustment in the Prolog code, such that it executes all possible branches of the main predicate consecutively and save each solution in a separate xml file.

In order to ensure security, stability and reliability, a database is crucially needed in processing and manipulating the data. The given time constraint on hand did not allow us to go for such a better development choice.

Finally, the whole compensation system could be built upon better formulated concise data. The data should preferably also include the examinations times, the start and end study week for each group, and the official holidays list.