Multi-Site Timetabling

Ruben Gonzalez-Rubio

Département de génie électrique et de génie informatique
Université de Sherbrooke,
Sherbrooke, Québec, J1K 2R1
Canada
Ruben.Gonzalez-Rubio@USherbrooke.ca

Keywords : Timetable Construction, Timetable algorithms, Timetable software, DIAMANT.

1 Introduction

Timetable construction in Universities is a moving target. De Werra [1] indicates that educational methods are driving changes in models and software applications, helping or constructing timetables. In this article we present a new timetable model which came from a new program in one Faculty[1] at the Université de Sherbrooke.

The basic difference of this new program compared with other instances of timetabling constructions in the University is that the courses in a term are taught at different sites instead of only one. In the case of one site all rooms are located in a single building, whereas in this new department the courses take place in three different sites. These sites are located in three cities (the longest distance between two sites being about 150 km). Some professors can teach at one site, some others can teach at two sites. For the moment nobody teaches at the three sites, but it could change, which means that the algorithm and the program must handle this possibility.

One last special constraint is that in two sites there is one teleconference room, which means that in some cases a professor can give a lecture in both sites at the same time. This implies that the lecture can be taught to two sections at the same time.

The article is organized as follows: First we introduce the current model for courses timetabling, then we show the new model to make clear where the differences are. Next, we present the steps that we follow to integrate the functionality of multi-site problem. After that we present how DIAMANT was modified to take into account the new data and the new algorithm, followed with a discussion of the experience. We end with a conclusion.

[1] A program in this context means an entity that can deliver a degree, the entity is a department, in this case a program can deliver degrees in various specialities.

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 416–419. ISBN 80-210-3726-1.

2 The Multi-site model

2.1 The Université de Sherbrooke timetabling model

Currently DIAMANT is used to schedule timetables at Université de Sherbrooke. DIAMANT [2] is an iterative application developed at the University which helps in scheduling courses and exams timetables. In this article we only talk about how to schedule courses.

Various definitions of the timetabling problem exist [5], [6] and [4]. We prefer that of fixing in time and space a sequence of meetings between teachers and students, in a prefixed period of time.

At Université de Sherbrooke there are many entities preparing timetables, each entity producing its own timetable using DIAMANT. Some entities work as demand driven systems, the students select courses from a list, then the schedule is prepared. Other entities prepare a master timetable, then the students can select the courses they want to take.

A more detailed definition of the problem could be the scheduling of a set of lectures for each course with a given number of rooms and time periods. For each set of lectures there is a professor. A professor can teach more than one course. Each entity has a prefixed period of time.

We formulate the problem following the presentation in [1], the we introduce changes to handle unavailabilities and preassignments. We have also the multiple sections and grouping subproblem and the classroom assignment subproblem. The periods are of variable length. The period sizes are different from one entity to an other.

2.2 The Université de Sherbrooke multi-site timetabling model

The multi-site problem came from an entity at the University, this entity offering courses (lectures) in three (sites) cities, the distances between cities being about 150 km, 90 km and 60 km. A set of courses is assigned to each site. That means a lecture of a course will take place on the specified site. A course can have multiple sections, sections can have lectures in the same site or in different sites. The students are assigned to each site, a set of rooms is defined for each site, a subset of professors can teach at one site, another subset can teach at two sites. For the moment nobody teaches at the three sites, but it could happen in a future.

The professors are assigned to a course or a section. The schedule must respect all hard constraints (no two classes in the same room, and so on). A new special constraint came from the fact that the travel time from one city to the other makes it impossible to teach two classes in a row at two different sites.

3 Steps to solve the multi-site problem

In order to solve de problem we proceed as follows:

1. Simplified solution with no changes to the application, we recall that the application can work manually or automatically.
   (a) We divide the problem into three problems (three sites). This allow to use the application three times to build three independent schedules.
   (b) As the three problems are dependent, a professor assigned to a period becomes unavailable elsewhere and the next, to let the professor travel from one site to the other. Changing the professor availability is done manually, when the schedule of the other site is prepared.
2. Solution with changes into the application.
   (a) One problem instead of three. The resources (courses, rooms, and students) are assigned to the corresponding site. As a professor is assigned to a course, he is indirectly assigned to a site. Inside the application we can have three sub problems. A menu allows the user to consult the state of a schedule of each site.
   (b) Professor availability can be calculated for each site by taking into account if he has a lecture in another. The unavailabilities are displayed in different colors according to their site.
   (c) The user can fix the of building sequence of the schedules of each site. The sequence is executed automatically.
   (d) Teleconference room problem is solved by the new class Teleconference-Room which is a derived from of class Room.

4 Changes in the application DIAMANT

DIAMANT was build using object-oriented technologie and design patterns [3] in order to simplify modifications. There are classes representing a set of resources (courses, professors, rooms, and students). We modify each class representing a resource to take into account the new data, for exemple we add a data member to Student Class to indicate the site where the student follows a course.

While reading the data the application detects that a multi-site problem must be solved. Then the menus are updated to work in multi-site mode.

Some dialogs where also modified to take into account the new data members.

We implemented the pattern Strategy to chose dynamically the algorithm to be executed. When there is more than one site the multi-site algorithm is executed. The multi-site algorithm follows the sequence of schedules fixed by the user.

5 Experience

When we saw the requirements for the multi-site problem, we though that it would be difficult to change the application. Using an iterative development process and taking avantage of our object-oriented design, we succeeded in a short period of time to satisfy our users. The whole development took about four months.

6 Conclusions

We introduce the problem of multi-site timetabling. We presented how we modified the application in order to treat a new the problem. Our the decision of developing using object-oriented programming really pay off. This allows us to add new functionalities without disturbing users that do not use these new functionalities. The new multi-site functionality was added in a short period of time.

References

1. D. de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19(2):151–162, 1985.
2. R. Gonzalez Rubio. Generating university timetables in a interactive system: DIAMANT. In *PATAT'00*, Konztanz, Germany, August 2000.
3. R. Gonzalez Rubio and Y. Syam. A design pattern: "TestConditions" which could be used in timetable construction software. In *PATAT'02*, Gent, Belgium, August 2002.
4. Luís Paulo Reis and Eugenio Oliveira. A language for specifying complete timetabling problems. In Edmund K. Burke and Wilhelm Erben, editors, *PATAT*, volume 2079 of *Lecture Notes in Computer Science*, pages 322–341. Springer, 2000.
5. A. Schaerf. A survey of autamated timetabling. *Arificial Intelligence Review*, 13(2):87–127, 1999.
6. Anthony Wren. Scheduling, timetabling and rostering - a special relationship? In Edmund K. Burke and Peter Ross, editors, *PATAT*, volume 1153 of *Lecture Notes in Computer Science*, pages 46–75. Springer, 1995.