EDT^{2004} : an open interactive timetabling tool

Sylvain Piechowiak, Jingxua Ma, and René Mandiau

LAMIH-CNRS, University of Valenciennes, France
sylvain.piechowiak@univ-valenciennes.fr

Abstract. This article deals with the analysis and design of an interactive decision support system for timetable management. This tool will be able to take hierarchical data organization into account and to maintain coherence of the constraints on this data. Our research which has led to the creation of the EDT^{2004} tool[1] has two aims. The first aim is to provide an open, generic tool which can be developed in many different ways. In order to achieve this aim, we have followed an object-oriented approach and we have defined object classes dedicated for the modeling of the timetabling problem. The second aim is to analyze the needs in timetable manipulation and to provide a generic organization so that the tool can be used in many situations. To achieve this aim, both user based and automated technics are used.

keywords: generic timetables model, interactive timetabling, multi points of view, decision support tool

1. Introduction

Every year, the task of the university educational managers is to organize timetables for the various courses or branches, whilst trying, as far as possible, to meet the "human" constraints of the teachers and students, along with the pedagogical constraints imposed by teaching progression and the "physical" constraints linked to material resources (rooms, equipment, etc.).

Up until now, managers have explored two different ways: graphical tools and full automated tools.

The graphical tools (like tabler or dedicated tools) are easy to use. They are powerful to draw pretty timetables. Unfortunately these tools give very poor help especially to find conflicts in timetables. They give no help to resolve these conflicts.

The full automated tools [4], [13], [25] are able to find a timetable. But generally, these tools require powerful machines. They are better feet the needs of secondary schools. Their timetables are built on a weekly basis. The development of a timetable for a year only duplicates the weekly structure. These automated generation tools are also well suited to the examinations [6], [14] or conferences [10].

[1] http://edt2004.free.fr

Fully automated tools are not efficient when the constraints cannot lead to a valid solution (impossibility of building a clash-free timetable). In these situations, the tools do not provide any support in explaining the causes for the lack of solution. Nothing is given to determine which constraints must be relaxed to bring about a solution. The quality of these timetables also depends on the exhaustiveness of the constraints. In an university, it is impossible to collect and to formalize all this information. Expertise of timetablers is the key.

Besides, more advanced techniques have been explored by the constraint programming community for the design of timetables [18]. These techniques allows users to give some guidelines to the system and get quicker or more adequate solution. However this approach is still focused on systems. We have chosen a different point of view since our work is dedicated to the end user.

The tool required must above all be interactive, adaptable and open and must have ergonomic qualities.

We will describe first the current organisation at the University of Valenciennes and Hainaut-Cambrésis (UVHC) and more specifically at the Institute of Sciences and Techniques of Valenciennes (ISTV). We will then present the `EDT`^{2004} tool, written in DELPHI object-oriented language and we will describe the different objects defined. Finally, we will give results we have obtained using `EDT`^{2004} and we present perspectives to get the highest level of genericity for this kind of tool and to make it designed for a multi-user configuration.

2. Modeling of the timetable problem

In an abstract view, the timetable problem consists in distributing over time pedagogical activities which require resources (teachers, groups, rooms and equipment). In order to respect the usual vocabulary, these activities are called teaching modules and each occurrence of an activity is called a session. The sessions have a duration. The timetable problem therefore requires the modeling of the activities, the resources and the time.

2.1 Modeling of the pedagogical activity

The pedagogical activity is modeled using one entity : **teaching module**.

A teaching module is characterized by a name, a **subject**, a total duration, a default duration (the duration suggested by default for each teaching module session) and a set of resources. Experience in the field led us to separate the resources into two sets : imposed resources and necessary resources. The distinction made between these two types of resource makes it possible to set certain constraints as soon as the teaching module is described.

In addition to these characteristics, each teaching module must be situated in relation to the others. For example, the E2 teaching module may only take place once the E1 has been completely finished.

2.2 Modeling of resources

The resources considered are the physical entities necessary for the preparation of timetables, indead rooms, teachers, groups, students and equipment. In order to take in most of the configurations imaginable, the resources are organized according to a treillis structure. Each resource R has daughter resources and can be the daughter of several other resources. Figure 1 shows an example.

Each reference to a resource R also concerns all the daughters of R, as well as all its "granddaughters", down to the lowest level of the hierarchy. Moreover, as soon as a session concerning R is placed, this limits the degrees of freedom of all its mother resources.

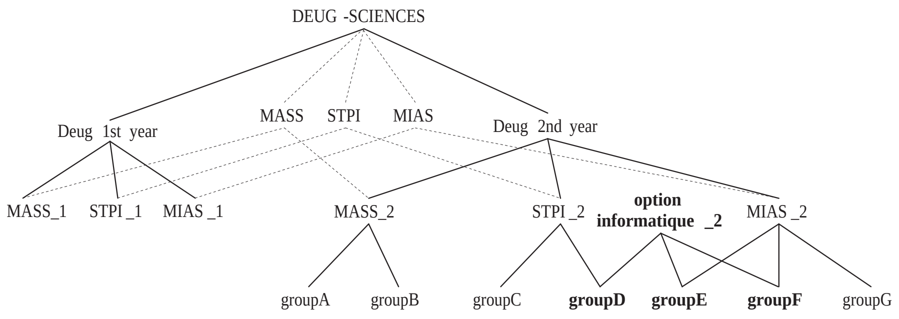


Fig. 1. Example of group hierarchy in the ISTV.

In the ISTV, this organization of the resources into treillis structures is used for the groups and for the equipment.

Five categories of resources are considered : the rooms, the teachers, the groups, the students and the equipment. Each resource is described with data to identify it (like its name) and data to characterize it (like capacity, speciality, etc).

2.3 Modeling of time

Two fields are used in order to represent time: the instant and the duration. The duration of each session is distinguished from the length of time separating the sessions. It should be noted that this representation of time is more flexible than that generally chosen in secondary schools. In classical timetabling problems each day is divided into a set of time slots which are fixed [5]. In theses cases, each session can only take place in one or several consecutive slots. Here a session can begin at each time in a day.

The granularity of time makes it possible to link representations of time with various degrees of sharpness. At the lowest level, we find the basic temporal representation (for example 15 minutes).

In order to model time, the following entities are defined : date, time, duration, slot and calendar.

A date refers to a triplet (day, month, year). Using this triplet, the value associated to it on the day axis is defined. A time is a whole number included between the minimum value $HMin$ (8h00) and the maximum value $HMax$ (19h00). These two numbers correspond to times in relation to a date.

A duration is a number included between $DMin$ and $Dmax = Hmax - HMin$. $DMin$ represents the smallest temporal unit available.

A time slot refers to a temporal interval during a day. It is characterized by a couple (H, D) in which H represents the starting time of the slot and D its duration.

A calendar is a set of dates. Each date is associated with a state (available or non available).

Each teaching module is also associated with a calendar to precise when it can be planed during the year. For example we can decide that a teaching module must be planed on Friday and an other from September 1st to December 15th.

2.4 Sessions, schedules and bookings

A session corresponds to a temporal event of a teaching module on a given date, during a specific slot. The characteristics of a session are : its teaching module, its date, its slot, its equipment and its room. The schedule of a resource is the set of sessions in which this resource appears. In this way, it is possible to consider the schedule for a room in the same way as that of a teacher.

A booking corresponds to an option placed on the occupation of a resource. In relation to a session, a booking is not associated to a teaching module. Bookings give flexibility to the use of the tool, especially during the timetable creation stage performed by the various managers. For example, a room can be booked by a manager for a group, without knowing exactly which teaching module will take place in it. The other managers are aware of the booking and will avoid using the room. If they do use the room, the room manager will have to find an alternative by suggesting another room.

2.5 Description of constraints

The constraints to be respected can be classified in two groups : physical constraints (also called hard constraints) and preference constraints (or soft constraints) which are linked to the pedagogical quality of the timetables. The physical constraints make it possible to be sure that any particular timetabling problem has a solution (resolvable) whereas the preference constraints are generally taken into account when a solution is being improved (optimization).

Physical constraints. These constraints cannot be violated, otherwise clashing situations would arise. There is a physical resource clash between two sessions s1 and s2 if these sessions have a common resource for a non null duration.

The physical constraints integrated into `EDT`^{2004} are the following:

- No resource r may be occupied by two different sessions s_1 and s_2 at the same slot (time and date): $\forall s_1 \forall s_2 \forall r[((s_1 \neq s_2 \wedge r \in resources(s_1) \wedge r \in resources(s_2)))] \Rightarrow [slotDateTime(s_1) \cap slotDateTime(s_2) = \emptyset]$
  where $resources(s_i)$ is the set of all the resources affected by s_i; $slotDateTime(s_i)$ indicates date, time and duration of s_i.
  As soon as a resource appears in a session, all its daughter resources are also occupied by the same session recursively:
  $\forall s \forall r(r \in resources(s) \Rightarrow (\forall x(x \in subresources(s) \Rightarrow x \in resources(s))))$
  In the same way, as soon as the resource appears in a session, none of its mother resources may be occupied at the same time (because all of these resources have resources in common). These are constraints linked to the hierarchical structure.
- It is forbidden to put more students than there are places in a room
- The total duration of the sessions in a teaching module may not exceed the volume planned.
- The calendars are respected for all the resources (sessions may only be placed during resource "working" days).

These constraints are part of EDT^{2004} : they cannot be modified by the users but they can be parmetrized or can be ignored. Thus, in all the orders available to the user, only actions which respect the constraints can be performed. For example, when a user wants to allocate a room to a session, only the rooms which respect all the constraints (free, suitable size and type) are suggested. This way of proceeding makes it possible, amongst other things, to avoid clashes appearing. We will discuss this further in the paragraph concerning the treatment of clashes. The user can decide to use the automatic timetabler and then to modify the result.

Preference constraints. Against physical constraints, preference constraints may be violated: in this case, the timetable obtained is of a lower quality. Typically, these constraints are used to express what a "good" timetable should be for the students and for the teachers.

In EDT^{2004} only 2 preference constraints are used. The first one is used to limit the movement of resources between rooms. The second one allocates to a session the room for which the capacity is the near that the number of student. For example EDT^{2004} avoid to allocate a very large room for a small set of students.

3. Visualization of the timetables and points of view

The two types of view most commonly used are : the weekly graphic view and the yearly graphic view. In both cases, visualizing a timetable consists in representing sessions of one or several resources on a plan. On every view of a timetable, two axes are represented : the resource axis and the time axis. It should be noted that there are no restrictions concerning the resources which appear on the resource axis. In other words, it is possible to visualize on one screen the timetables of several groups, teachers, rooms and elements of equipment. This has two major advantages. Firstly, during the modification phase, the user is informed on one screen about the availability or occupation of the resources he/she is dealing with. Secondly, it enables the user to understand why the tool refuses to place a session in the slot he/she has chosen (role of explanation).

The manipulation orders are given via screens. They are invariably available in each view. The orders allow the user to "navigate" around the timetables and modify them whilst controlling clashes (and avoiding them as far as possible). In order to make the user's task easier, the constraints are taken into account in a dynamic fashion. Thus, for example, when a session is being moved, the tool begins by finding the moves allowed and then suggests them to the user. This approach limits the user's search space to the areas in which sessions can be placed without creating clashes.

We defined the notion of points of view in a previous research on cooperative reasoning and its application to interactive diagnosis [19]. The basic idea consists in representing a model according to different angles of vision, called points of view. Each point of view only concerns a part of the model, but the set of points of view covers the entire model. The reasoning performed using each point of view is propagated to the other points of view, and this guarantees the coherence of the set of points of view.

The notion of point of view is used here to represent various elements of information for the same indicator (time, resource) without overloading the screen. The points of view retained here are : the resource point of view (teacher, group, room and equipment), the teaching module point of view and a specific point of view in which the user chooses the information he/she wishes to see. In this last point of view, the user can see for example the names of the rooms, their capacity and their occupation rate as well as the names of the teachers who use the rooms in question. A button enables the user to switch from one point of view to another whilst keeping the general appearance of his/her view along with the same orders (addition, removal, etc.).

In figure 2 we give an illustration: it is part of the schedule for the $DESS\_ICHM$ class and its sub-groups $ICHM1$ and $ICHM2$, from the point of view of the teachers and the rooms for a period chosen by the user. It is a yearly view where each week is represented by one line. Each line has as many sub-lines as resources selected (here the resources are the groups: $DESS\_ICHM$, $ICHM1$ and $ICHM2$). The name of the teacher for each session visualized is displayed (teacher's point of view).

3.1 Action possible using the graphic interface

We have defined three types of user: the designers (pedagogical managers), the analyzers (administrative managers) and the consultors (teachers, students). They have specific needs which lead to the definition of specific action classes. This action classes are described in [20].

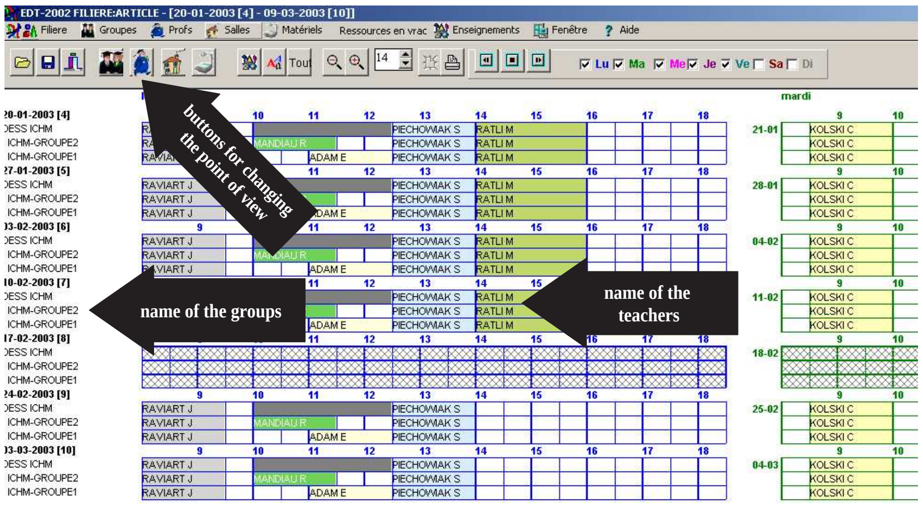

EDT-2002 FILIERE:ARTICLE - [20-01-2003 [4] - 09-03-2003 [10]]
Filiere Groupes Profs Salles Matériels Ressources en vrac Enseignements Fenêtre Aide
buttons for changing the point of view
name of the groups
name of the teachers
mardi

Fig. 2. A timetable seen from the teacher point of view.

3.2 Allocation of resources in an existing schedule : example of the rooms

The allocation of rooms is performed by the room manager once the timetables have been created by the educational managers. The rooms are divided into several types : lecture theaters, ordinary rooms, specialized rooms. They are characterized by their capacity as well as by their geographical location. The location is limited to proximity zones because the important thing is to know whether one room is close to another or not, more than the actual distance between the two.

Up to now, the manual allocation of rooms has been carried out without too many problems thanks to the experience acquired over the past twenty years by the various managers. Moreover, the allocation of rooms rarely leads to clashes which bring about considerable modifications in the timetables. This can be explained mainly by the applied approach. Before the educational managers even begin developing the timetables, they meet together to pinpoint the critical teaching modules (i.e. those which need critical resources - rooms or equipment). These teaching modules are placed on the various timetables in such a way as to satisfy all requirements and avoid clashes. For example, the lecture theaters are allocated to the various courses so as to cover global needs. This is done using the booking notion explained previously. Thus, for example, for a specific group, a lecture theater is reserved for four half-days every week from 15th September to 30th May.

When the rooms are actually allocated to sessions, the user must choose a room from all the rooms so that the constraints are respected: the room must be free and have a sufficient capacity to accommodate the students concerned by the session. Extra constraints (preference constraints) must also be taken into account not only for the quality of the timetables but also in order to anticipate needs which are unknown when the timetables are developed (for example, organization of work meetings or seminars).

We suggest helping the user to decide which room to allocate to which session in the following manner. Firstly, a list is drawn up of the free rooms which have a sufficient capacity to accommodate the group of students. Then these rooms are classified using a function which depends on preference constraints or heuristics. Thus the user can either trust the tool and choose the first room in the classified list, or choose another room on the list for reasons which are not known to the tool. The parameters of the function used to classify the rooms in the list can be chosen by the user who can decide to apply all of the preference constraints or just some of them.

The preference constraints currently in use were defined in collaboration with the room managers: regularity of room allocation, limitation of group movement between rooms and anticipation of unknown events.

The user can decide whether to take preference constraints into account or whether to ignore them, as is shown in the dialogue window in figure 3. In addition, this user can decide to use the automatic allocation module. This module is based on intelligent backtracking algorithms like *backjumping* and *forward checking* (we know that these exact methods are not the best for a full automated approach, but they are sufficient in our situation) [24].

4. Treatment of clashes

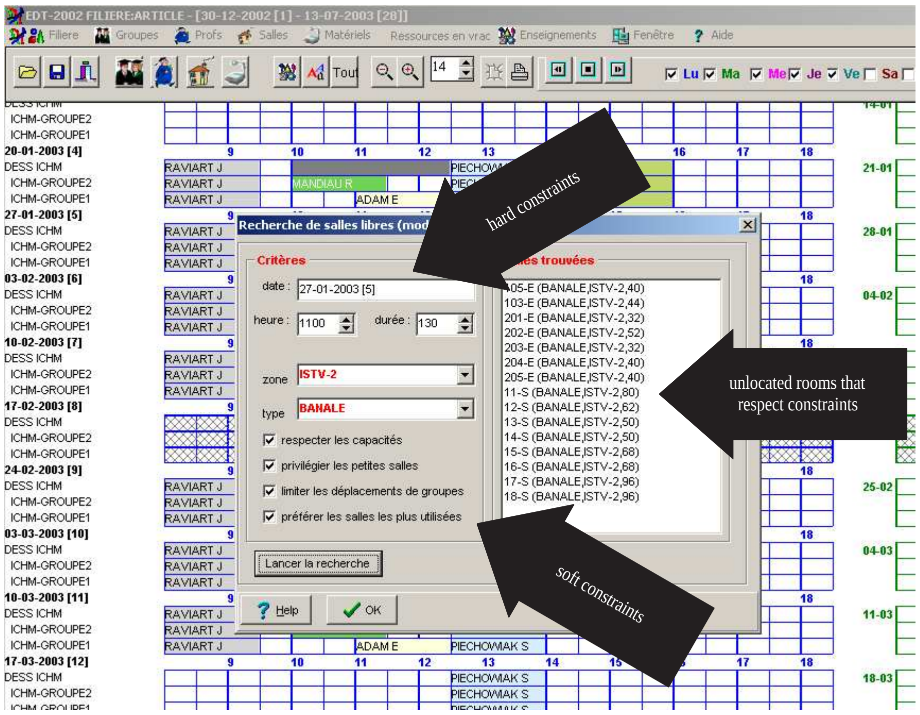
In the section concerning constraints, we defined the notion of clashes: a clash is characterized by the sharing of resources between several sessions at the same moment in time: $\forall s_1 \forall s_2, clash(s_1, s_2) \Leftrightarrow ((s_1 \neq s_2) \wedge (overlaped(date(s_1), date(s_2))) \wedge (resources(s_1) \cap resources(s_2) \neq \emptyset))$.

The basic idea of our work is to design an interactive decision support tool which will allow the planong of sessions on a schedule whilst guaranteeing that there are no clashes. Nevertheless, several reasons oblige us firstly to accept that there will be clashes and secondly to resolve them.

Firstly, during our on-site tests with the users, it appeared that during the development of timetables, some users prefer to have the possibility of placing clashing sessions temporarily and then correcting these clashes, rather than only being able to consider non clashing situations.

Secondly, in reality, the timetables are created locally for the branches, without knowing the schedules for other branches. When all the schedules for all these branches are grouped together, clashes appear generally because of teachers who work in several branches for example.

Finally, our tests revealed that even for users who wish avoid clashes when placing sessions, it is important for them to be able to be informed on the reasons for the impossibility of placing certain sessions. For example, when the user clicks with the mouse on a grid in order to place a session, the tool begins by searching



Recherche de salles libres (mod

Critères

date : 27-01-2003 [5]

heure : 1100 durée : 130

zone ISTV-2

type BANALE

- [✓] respecter les capacités
- [✓] privilégier les petites salles
- [✓] limiter les déplacements de groupes
- [✓] préférer les salles les plus utilisées

Lancer la recherche

es trouvées

105-E (BANALE,ISTV-2,40)
103-E (BANALE,ISTV-2,44)
201-E (BANALE,ISTV-2,32)
202-E (BANALE,ISTV-2,52)
203-E (BANALE,ISTV-2,32)
204-E (BANALE,ISTV-2,40)
205-E (BANALE,ISTV-2,40)
11-S (BANALE,ISTV-2,80)
12-S (BANALE,ISTV-2,62)
13-S (BANALE,ISTV-2,50)
14-S (BANALE,ISTV-2,50)
15-S (BANALE,ISTV-2,68)
16-S (BANALE,ISTV-2,68)
17-S (BANALE,ISTV-2,96)
18-S (BANALE,ISTV-2,96)

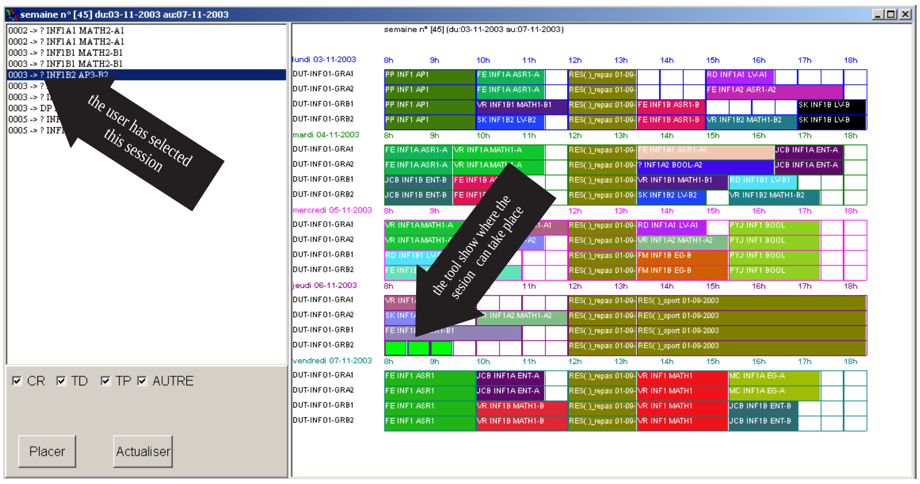
Help OK

Fig. 3. Dialogue window for the allocation of rooms with consideration of preference constraints.

for a list of sessions which may be placed without creating clashes. If there is nothing in this list, it is better to indicate why that is so. Two explanations can be given. The first is that all the sessions have been placed. The second explanation is that for all the sessions which still have to be placed, there is systematically one of the resources which is already used for another session at the same time. In this case, it is best to give the user the list of sessions which hinder the placing; the user can then decide whether to rethink the choice of place or whether to come back on sessions placed beforehand.

4.1 1st approach : avoiding clashes

The method which consists in avoiding clashes is well adapted when the timetable is available and the user wishes to modify it slightly, or when it is available on paper and the user wishes to enter it on a computer. Avoiding clashes then comes down to avoiding data entry errors. The users who appreciate this method are the room managers because they allocate the rooms to the sessions, once the sessions have already been scheduled. The figure 4 shows a weekly view for sessions placement. In its left part there is the list of the sessions that should be placed in this week. In the right part, the user can see where to place the session he has selected.


the user has selected this session
the tool show where the sesion can take place
☑ CR ☑ TD ☑ TP ☑ AUTRE
Placer
Actualiser

```
semaine n° [45] du:03-11-2003 au:07-11-2003

0002 -> ? INF1A1 MATH2-A1
0002 -> ? INF1A1 MATH2-A1
0003 -> ? INF1B1 MATH2-B1
0003 -> ? INF1B1 MATH2-B1
0003 -> ? INF1B2 AP3-B2
0003 -> ?
0003 -> ? I
0003 -> DP
0005 -> ? INF
0005 -> ? INF

semaine n° [45] (du:03-11-2003 au:07-11-2003)

lundi 03-11-2003     8h 9h 10h 11h 12h 13h 14h 15h 16h 17h 18h
DUT-INFO1-GRA1  PP INF1 AP1 | FE INF1A ASR1-A | RES( )_repas 01-09- | RD INF1A1 LV-A1
DUT-INFO1-GRA2  PP INF1 AP1 | FE INF1A ASR1-A | RES( )_repas 01-09- | FE INF1A2 ASR1-A2
DUT-INFO1-GRB1  PP INF1 AP1 | VR INF1B1 MATH1-B1 | RES( )_repas 01-09- | FE INF1B ASR1-B | SK INF1B LV-B
DUT-INFO1-GRB2  PP INF1 AP1 | SK INF1B2 LV-B2 | RES( )_repas 01-09- | FE INF1B ASR1-B | VR INF1B2 MATH1-B2 | SK INF1B LV-B
mardi 04-11-2003     8h 9h 10h 11h 12h 13h 14h 15h 16h 17h 18h
DUT-INFO1-GRA1  FE INF1A ASR1-A | VR INF1A MATH1-A | RES( )_repas 01-09- | FE INF1A1 ASR1-A1 | JCB INF1A ENT-A
DUT-INFO1-GRA2  FE INF1A ASR1-A | VR INF1A MATH1-A | RES( )_repas 01-09- | ? INF1A2 BOOL-A2 | JCB INF1A ENT-A
DUT-INFO1-GRB1  JCB INF1B ENT-B | FE INF1B A | RES( )_repas 01-09- | VR INF1B1 MATH1-B1 | RD INF1B1 LV-B1
DUT-INFO1-GRB2  JCB INF1B ENT-B | FE INF1B | RES( )_repas 01-09- | SK INF1B2 LV-B2 | VR INF1B2 MATH1-B2
mercredi 05-11-2003  8h 9h 12h 13h 14h 15h 16h 17h 18h
DUT-INFO1-GRA1  VR INF1A MATH1-A | 1-A1 | RES( )_repas 01-09- | RD INF1A1 LV-A1 | PYJ INF1 BOOL
DUT-INFO1-GRA2  VR INF1A MATH1-A | -A2 | RES( )_repas 01-09- | VR INF1A2 MATH1-A2 | PYJ INF1 BOOL
DUT-INFO1-GRB1  RD INF1B1 LV- | RES( )_repas 01-09- | FM INF1B EG-B | PYJ INF1 BOOL
DUT-INFO1-GRB2  FE INF1 | RES( )_repas 01-09- | FM INF1B EG-B | PYJ INF1 BOOL
jeudi 06-11-2003     8h 11h 12h 13h 14h 15h 16h 17h 18h
DUT-INFO1-GRA1  VR INF1 | RES( )_repas 01-09- | RES( )_sport 01-09-2003
DUT-INFO1-GRA2  SK INF1 | R INF1A2 MATH1-A2 | RES( )_repas 01-09- | RES( )_sport 01-09-2003
DUT-INFO1-GRB1  FE INF1 R1-B1 | RES( )_repas 01-09- | RES( )_sport 01-09-2003
DUT-INFO1-GRB2  RES( )_repas 01-09- | RES( )_sport 01-09-2003
vendredi 07-11-2003  8h 9h 10h 11h 12h 13h 14h 15h 16h 17h 18h
DUT-INFO1-GRA1  FE INF1 ASR1 | JCB INF1A ENT-A | RES( )_repas 01-09- | VR INF1 MATH1 | MC INF1A EG-A
DUT-INFO1-GRA2  FE INF1 ASR1 | JCB INF1A ENT-A | RES( )_repas 01-09- | VR INF1 MATH1 | MC INF1A EG-A
DUT-INFO1-GRB1  FE INF1 ASR1 | VR INF1B MATH1-B | RES( )_repas 01-09- | VR INF1 MATH1 | JCB INF1B ENT-B
DUT-INFO1-GRB2  FE INF1 ASR1 | VR INF1B MATH1-B | RES( )_repas 01-09- | VR INF1 MATH1 | JCB INF1B ENT-B
```

Fig. 4. A weekly view for sessions placement.

This method is also advantageous for the creation of timetables because it constantly limits the space of the sessions which can be placed in a given slot. In this case, this method alone is not sufficient. We noted that when wanting to place a session in a slot, the user is put off if the session he/she thinks he/she can place is not on the list of sessions possible. To solve this problem, the user would like to be given the reasons for this absence (teacher, room, group or one of the descendants or ascendants already occupied). Another way to tackle this problem is to accept clashes and to resolve them afterwards. The danger of this method is that there could be a lot of clashes which would be impossible to resolve without starting the work again from the beginning.

By allowing the manipulation of incomplete sessions (no teacher, no room), a different approach is adopted which consists in creating a timetable without resources (or the fewest resources possible) and then allocating the resources afterwards. The problem with this approach is that the constraints are not integrated early enough. At the ISTV, it is this approach which is used, since the rooms are allocated once the timetables have been created.

These two approaches can be compared to the prospective and retrospective approaches which are well known in the field of constraint programming [24]. It is shown that the approaches which involve forward checking through constraints are more efficient than those which check if constraints have been respected after the choices have been made.

4.2 2nd approach : interactive resolution of clashes

A clash can be resolved in several different ways.

The first way is to delete certain clashing sessions and to request the user to place them elsewhere on the schedule. This method is only viable if the dependencies between clashes are taken into account and if the number of clashes is reasonable. For example, let us consider the case shown in figure 5 in which the sessions $S1$, $S2$ and $S3$ are scheduled on the same day in the same slot.

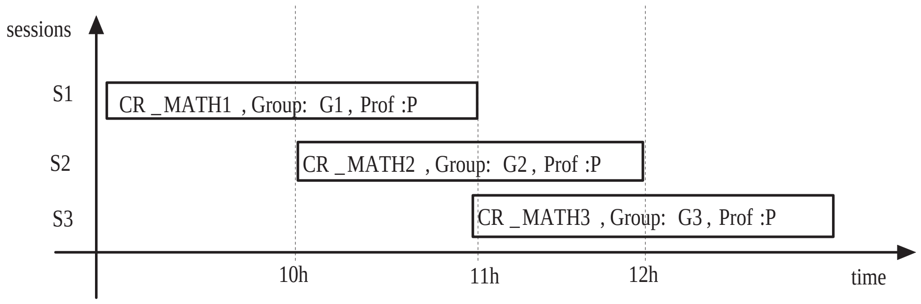

sessions
S1
S2
S3
CR _MATH1 , Group: G1 , Prof :P
CR _MATH2 , Group: G2 , Prof :P
CR _MATH3 , Group: G3 , Prof :P
10h
11h
12h
time

Fig. 5. Example of clash with groups.

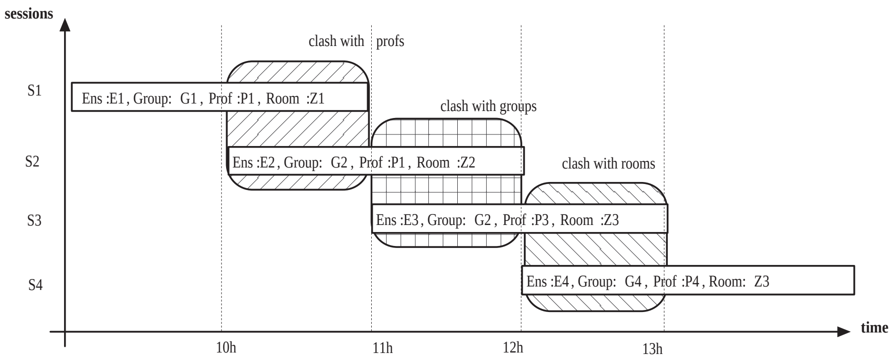

sessions
S1
S2
S3
S4
clash with profs
clash with groups
clash with rooms
Ens :E1, Group: G1 , Prof :P1 , Room :Z1
Ens :E2, Group: G2 , Prof :P1 , Room :Z2
Ens :E3, Group: G2 , Prof :P3 , Room :Z3
Ens :E4, Group: G4 , Prof :P4 , Room: Z3
10h
11h
12h
13h
time

Fig. 6. Example of a multiple clash.

In this case, two clashes are linked : $S1$ with $S2$ and $S2$ with $S3$. The clashes can be resolved by deleting the three sessions and trying to place them elsewhere, by deleting $S1$ and $S3$ and placing them elsewhere, or by deleting $S2$ only and by placing it in another slot. The choice of solution can depend on various criteria. It is possible to choose the solution which modifies the fewest sessions (just $S2$

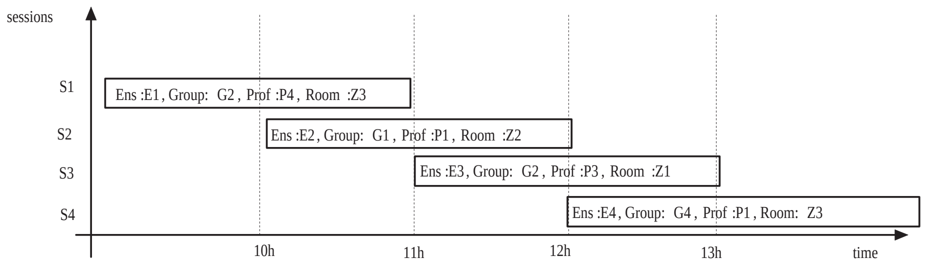

sessions
S1
S2
S3
S4
Ens :E1, Group: G2 , Prof :P4 , Room :Z3
Ens :E2, Group: G1 , Prof :P1 , Room :Z2
Ens :E3, Group: G2 , Prof :P3 , Room :Z1
Ens :E4, Group: G4 , Prof :P1 , Room: Z3
10h
11h
12h
13h
time

Fig. 7. The clashes have been resolved.

in the example). It is also possible to choose the solution which preserves the quality of the timetables.

The second way consists in grouping together the resources involved in different sessions. The teaching modules may be shared by several groups in different branches. When the data relating to the different branches is merged, these teaching modules are declared several times. This can cause clashes with the teachers, for example, because for these shared teaching modules, the same teacher gives a lesson to different groups at the same time. The resolution of these clashes is easy: the groups in the two teaching modules have to be merged.

The third way of resolving clashes is to reduce the duration of the sessions so that the time interval associated to the clashes is eliminated. In the previous example, the two clashes take place in the time intervals $[10.00 - 11.00]$ and $[11.00 - 12.00]$. There are several possibilities : reduce session $S1$ to the $[09.00 - 10.00]$ interval and session $S3$ to the $[12.00 - 13.00]$ interval or reduce session $S1$ to the $[09.00 - 10.30]$ interval, session $S2$ to the $[10.30 - 11.30]$ interval and session $S3$ to the $[11.30 - 13.00]$ interval.

The fourth way is to eliminate the "cause" of the clash, that is the teacher, the group or the room. Once the clash has been eliminated, an attempt can be made to allocate the sessions to other teachers, groups or rooms.

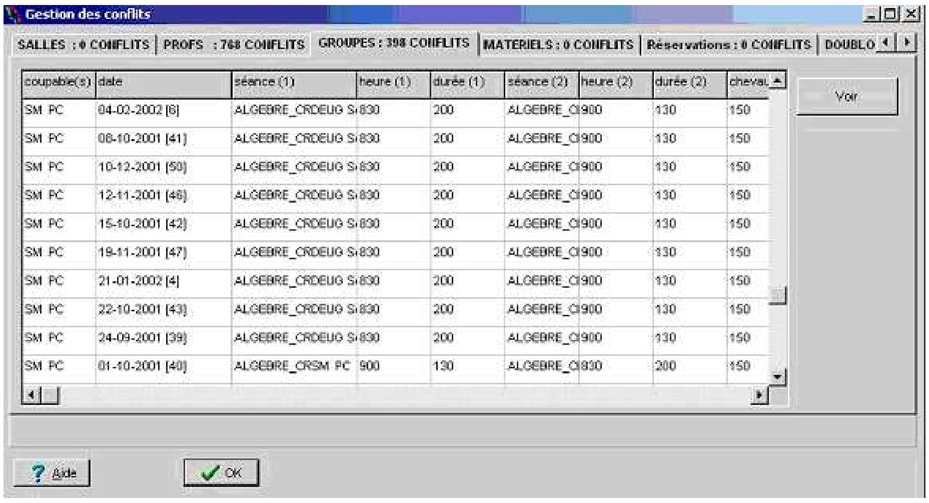
This method can be extended so that it is capable of exchanging the resources for various sessions. For example, the teachers of two teaching modules can be interchanged. More complicated exchanges can also be envisaged, involving more than two teachers. However, searching for these exchanges with an aim to resolving clashes is a laborious process. In figure 6, an example of a multiple clash is shown.

By performing the following exchanges : $prof(S1) \leftrightarrows prof(S4)$, $group(S1) \leftrightarrows group(S2)$ and $room(S1) \leftrightarrows room(S3)$, the three clashes are resolved and the configuration shown in figure 7 is obtained.

Trials were carried out in real situations. We have worked with the manually made timetables for the academic year 1999-2000. The search for clashes amongst the 2500 sessions took 3 seconds using a PC with a Pentium III 500Mz processor, 256M ram. This search revealed 350 group clashes, 50 teacher clashes and 30 room clashes.

It should be noted that this test was performed using real data from the manually-made timetables used for the university year, which should in theory have been clash-free! This mean that clashes was not detected during the conception phase of timetables: they have been treated during the year by teachers themselves.

Figure 8 shows how the clashes are displayed (group clashes in the case of the figure). There is a 9-column chart (the 9th column is hidden because of the size of the window). Each line represents a clash. For example, the first line shows a clash involving the $SMPC$ group on 04-02-2002. In fact there are two overlapping sessions. The first one, ALG_CR_DEUG, starts at 08.30 and lasts for 2 hours, while the second, ALG_CR_DEUG, starts at 09.00 and lasts for 1.30 hours. The overlap is therefore 1.30 hours. By using the $Voir$ button, the user sees the schedule for this group for the entire week which shows up these two sessions. The user can then decide how to correct the clash (move the sessions, eliminate them, etc.).



coupable(s)	date	séance (1)	heure (1)	durée (1)	séance (2)	heure (2)	durée (2)	cheval
SM PC	04-02-2002 [6]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	08-10-2001 [41]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	10-12-2001 [50]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	12-11-2001 [46]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	15-10-2001 [42]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	19-11-2001 [47]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	21-01-2002 [4]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	22-10-2001 [43]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	24-09-2001 [39]	ALGEBRE_CRDEUG S	830	200	ALGEBRE_C	900	130	150
SM PC	01-10-2001 [40]	ALGEBRE_CRSM PC	900	130	ALGEBRE_C	830	200	150

Fig. 8. Window displaying the clashes detected.

Figure 9 shows the view a user can have when he/she has decided to request a graphic view of a clash (the first clash in our example). The two clashing sessions are shown up by a color. It is necessary to modify (eliminate, move or reduce) the duration of one of the two sessions to resolve the clash.

The user can view the schedule for a set of resources, showing the teachers as well as the groups, rooms and equipment. This type of view is particularly well-suited to understanding why the choice of a place or move of session is impossible. For example, let us suppose that the graphic view shown concerns the timetable of a group G1 and that the user wishes to move a session S for this group. By clicking on the session, the user can choose the instruction move in

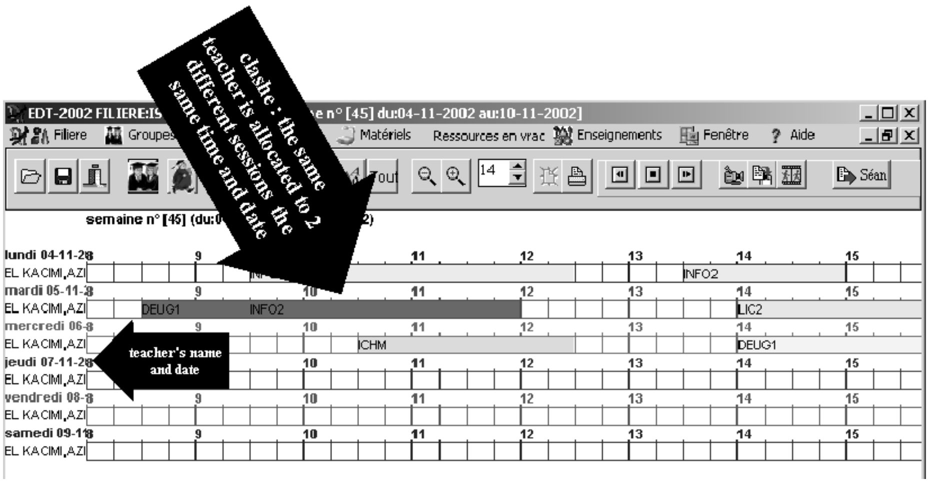


Fig. 9. Visualization of one of the clashes.

the day. EDT^{2004} calculates the list of slots corresponding to the possible moves. If the user does not find the slot he/she had been thinking of a priori on this list, the tool enables him/her to understand the reasons why by providing the occupation of each resource involved in the session.

4.3 Semi-automatic resolution of clashes

The problem of resolving clashes can be tackled in a generic way, but experience has led us to the same remark as that made by Foulds and Johnson [11]: the clashes generally concern few sessions and can be resolved by backtracking a little way.

Clashes are detected according to each type of resource (teacher, group, room and equipment) and are classified according to different criteria. These criteria were defined on the basis of the experience of the various educational managers.

– The clash represents a **double**. Two sessions are a double (session duplication) if they are sessions in the same teaching module (so they have the same teachers and the same groups, but not necessarily the same equipment or the same rooms) or if they take place on the same day, at the same time and for the same duration. This problem is solved by deleting one of the sessions. If the total volume of sessions of this teaching module is not equal to the total volume planned, the user will have to place the deleted session elsewhere.
– Sharing of a teaching module over several groups. For example, conferences are organized for two courses represented by G1 and G2. The managers of these courses create their timetables. Then, when all the timetables are collated for the allocation of rooms and detection of clashes, the conferences

are detected as being clashes because they share the same teacher, on the same date, at the same time and for the same duration. Experience shows that these cases of teaching modules shared over several courses is done with small groups and according to the level (1st, 2nd or 3rd cycle). The problem is solved by deleting one of the two teaching modules and adding the teaching module groups from the deleted module to the groups of the remaining module.
– Resolution of clashes by giving preference to moving sessions involving small groups and short sessions (in duration).

5. Conclusion and future work

In this article, the problem of timetable management has been looked at from a viewpoint centred on the user, and not, as in many other research projects, centred on the problem (or on the algorithms available). This led us to design the EDT^{2004} decision support tool which helps the user when faced with solving a problem. The tool has been tested successfully at the University of Valenciennes and Hainaut-Cambrésis in large scale real situations.

EDT^{2004} is intended for different user profiles, going from the designers of timetables to users who have varied degrees of computing skills. It makes it possible to manage resources organized into a hierarchy (groups, teachers, rooms and equipment). It guarantees data coherence in a dynamic way. When clashing situations appear, various resolution approaches are suggested.

The aim was to design a generic interactive decision support tool for timetable problems on the basis of a real case study which was truly representative of the problem. An object-oriented approach was chosen and the generic classes have been defined like in [26].

Work is now continuing in a main direction : the aim will be to help the user to express constraints and to take these constraints into account in an interactive manner. For this, we intend to revise the work performed on constraint programming to provide support in the design and generating of timetables. In this field, most existing work deals with automated generation. On the other hand, very few research projects concentrate on support in the description of timetables [7], [3], [15] or [22]. We feel that an interesting project would be to combine the automated and interactive generation of timetables. This direction was explored successfully in [1] and [12] which deal with other problems based on scheduling. For example, we envisage performing an automatic allocation of rooms after the timetables have been created interactively with the users (thus making it easier to take into account the pedagogical constraints which are difficult to express and formulate). A project similar to [8] or [17] based on approaches from the constraint programming field has already been started on this subject.

References

1. S. Abdennadher, H. Schlenker, INTERDIP, an interactive constraint based nurse scheduler, *Proceedings PACLP'99*, London, 1999.

2. R. Bartak, H. Rudova, Integrated Modelling for Planning, Scheduling, and Timetabling Problems, *Proceedings of PLANSIG 2001*, Edinburgh, UK, 2001
3. E. K. Burke, J. H. Kingston, and P. A Pepper. A standard data format for timetabling instances. Practice and Theory of Automated Timetabling III, pp 213-222, 1997
4. P. Boizumault, Y. Delon, L. Peredy, Constraint logic programming for examination timetabling *Journal of logic programming* **26-2** (1996) 217–233
5. M. W. Carter A Comprehensive Course Timetabling and Student Scheduling System at the University of Waterloo. Practice and Theory of Automated Timetabling III, pp 64-84, 2000
6. M.W. Carter, G. Laporte, J.W. Chinneck, A general examination scheduling system, *Interfaces 24(3)* (1994) 109–120
7. T.B. Cooper, J. Kingston, A program for constructing high school timetables. Technical Report n° 496, University of Sydney, Australia, 1995.
8. S. Deris, S. Omatu, H. Ohta, Timetable planning using the constraint-based reasoning *Computers & Operations Research* **27** (2000) 819–840
9. M.Dimopoulo, P. Miliotis, Implementation of a university course an examination timetabling system *European Journal of Operational Research* **130** (2001) 202–213
10. R.W. Eglese, G.K. Rand, Conference seminar timetabling, *Journal of the Operational Research Society* **38** (1987) 591–98
11. L.R. Foulds, D.G. Johnson, SlotManager: a microcomputer-based decision support system for university timetabling, *Decision Support Systems* **27** (2000) 367–381
12. H.J. Goltz, D. Matzke, Combined interactive and automatic timetabling, *Proceedings PACLP'99*, London, 1999
13. H.J. Goltz, On methods of constraint-based timetabling, *Proceedings PACLP'00*, Manchester, 2000
14. D.G. Johnson, Timetabling university examinations, *Journal of the Operational Research Society* **41** (1990) 39–47
15. Jeffrey H. Kingston Modelling Timetabling Problems with STTL. Practice and Theory of Automated Timetabling III, pp 433-445, 2000
16. B.M.Lawal, D. Gilbert, A. Letichevsky, A web course scheduler in constraint logic programming: interactive computing with constraints, *Workshop on Constraint reasoning on the internet, 3rd International Conference on Principles and Practice of Constraint Programming (CP97)*, 1997
17. W. Legierski Search Strategy for Constraint-Based Class-Teacher Timetabling. Practice and Theory of AutomatedTimetabling IV, pp 247-261, 2002
18. T. Müller, R. Bartak, Interactive Timetabling, in Proceedings of ERCIM Workshop on Constraints, Prague, 2001.
19. S. Piechowiak, D. Jouglet, F. Vanderhaegen, A Multi-point of view for phone network system diagnosis, Proceedings ICSSSE'98, Bengjing, China, 1998.
20. S. Piechowiak, C. Kolski, Analyse et conception d'un outil interactif d'aide à la gestion des emplois du temps basé sur les points de vue (in french), Journal of Human-Computer Interaction, in Press
21. S. Piechowiak, C. Kolski, Towards a generic object oriented decision support system for university timetabling : an interactive approach, *International Journal of Information Technology and decision making*, in Press
22. L. P. Reis, E. Oliveira A Language for Specifying Complete Timetabling Problems. Practice and Theory of Automated Timetabling III, pp 322-341, 2000
23. A. Schaerf, A survey of automated timetabling. Technical Report n° CS-R9567, Centrum voor Wiskunde en Informatica, 1995.

24. E. Tsang, *Foundations of constraint satisfaction.* (Academic Press, Computation in Cognitive Science, 1993).
25. M. Yoshikawa, K. Kaneko, Y. Nomura, M. Watanabe, A constraint-based approach to high-school timetabling problems: a case study, *Proceedings of the 1995 International Joint Conference on AI (IJCAI'95)*, Montreal, Quebec, Canada, 1995.
26. K. Zervoudakis, P. Stamatopoulos A Generic Object-Oriented Constraint-Based Model for University Course Timetabling Practice and Theory of Automated Timetabling III, pp 28-47, 2000