Solving a Time Tabling Problem by Constraint Satisfaction

F.P.M. Dignum [*] W.P.M. Nuijten [†] L.M.A. Janssen [‡]

Abstract

The time tabling problem is well known and much research has been done about it. This paper shows how a difficult instance of the time tabling problem with many parameters and loose constraints can be handled using constraint satisfaction. To this end the constraints have to be modeled in a format that can be handled efficiently. It is shown how several modelling problems can be solved in a generic way and that the results are satisfactory.

1 Introduction

This paper is one in a series of papers in which we investigate the potentials of constraint satisfaction techniques to handle deterministic scheduling problems. Our research is focussed on two main lines of investigation. The first line concentrates on theoretical studies in the area of job-shop scheduling. See e.g.[6].
The second line of research investigates whether the improved performance characteristics of our constraint satisfaction algorithms also hold for problems in practical planning and scheduling situations. To this end we performed some case studies on actual instances of time tabling and production planning problems. In [7], a first account of these studies is given. The present paper also reports on this line of the research and presents the results of the constraint satisfaction techniques, as we use them, on a real-life school time tabling problem.

The main characteristics of a time tabling problem are a large number of variables, a high level of complexity and a varied nature. In [4], it is shown that all common time tabling problems are NP-complete. Due to the variety in problems most approaches differ to a large extend. A general applicable time tabling model does not exist. Furthermore, it is difficult to compare the solutions presented in the literature. An extensive overview of early studies on time tabling can be found in [8].

After 1980 several heuristic methods have been tried. Gans presents a heuristic approach to solve the time tabling problem of secondary schools in the Netherlands in [2]. Werra gives a clear overview of handling the basic class teacher problem and course scheduling problem by introducing models based on graphs and networks in [3]. Some constraints can be formalized within the models, others, which are also found in many practical situations, can not. However, the exact models presented form a useful basis to derive heuristic methods. In [5], Monfroglio describes a class teacher problem as a case study of the application of constraint satisfaction in a logic programming environment. Although the approach is similar to ours it appears that it is not possible to solve any large realistic problem using this method. A different approach in solving time tabling problems is the use of *simulated annealing*, as done by e.g. Abramson [1].

[*]Eindhoven University of Technology, Department of Mathematics and Computing Science, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

[†]Ilog S.A. 2 Avenue Gallieni BP 85, 94253 Gentilly Cedex, France

[‡]CSB-System GmbH, P.O. Box 1293, D-5 25 02, Geilenkirchen, Germany

In our approach to the time tabling problem we translate the problem to a more general time and resource constrained scheduling problem for which we have developed very efficient constraint satisfaction techniques. This makes it possible to use the general method of constraint satisfaction while retaining a high degree of efficiency. In the next section we will introduce the CVA time tabling problem. In section 3, we will describe the TRCSP as the model to which the CVA time tabling problem is mapped. Section 4 briefly describes how this translation is effectuated. In section 5, we will show some results. Section 6 concludes the paper with some concluding remarks.

2 The CVA Time Tabling Problem

In this section we describe *The CVA Time Tabling Problem* (CVATTP). We will first give an informal description of the problem, then we will make this description formal to facilitate a proper translation to the CSP formulation in the next section.

The CVA is an educational establishment attached to the employment bureau of Maastricht. It provides refresher courses, focused on administrative topics, to people seeking employment. The courses are given during periods of four weeks, which we will refer to as *cycles*. The start and end times of the courses are predefined, but they do not have to coincide with the start and end times of the cycle, i.e., courses do not have to take four weeks.

A course consists of lectures given by a teacher (the *theory part* of the course) and of practical work referred to as the *practice part*. While theory parts are tought by teachers, during the practice parts no teacher has to be present. A teacher can teach more than one course during a cycle. Furthermore, theory parts can be tought by several teachers, i.e. several teachers each teach a part of the course. For most courses there are also several persons who are able to teach that course. However, we assume that the teacher assignment to courses is given at the time that the time table has to be made. Therefore, it is not a part of the time tabling problem.

Both the theory part as well as the practice part of all courses are split up into a number of *periods* with pre-specified, and possibly different, lengths (ranging between 45 minutes and two hours with units of 15 minutes). The periods of one course do not have to be all of the same length! The periods of a theory part are called *theory periods*, the periods of a practice part *practice periods*. All periods of a course form a chain with a predefined shape. Per week a fixed number of periods of the course has to be scheduled. Two theory periods of the same course are not scheduled at the same day.

The students are divided into *course groups*. All students taking the same courses during a cycle form a course group. Courses are given to one or more groups. Different course groups are permitted to take the same theory part of a course in different classrooms at different times, while they follow the practice part of that course together. Furthermore, course groups taking practice periods of different courses may be simultaneously present in the same classroom.

Course groups and teachers have three breaks per day. These breaks are not at fixed times, but must be scheduled within fixed intervals. During the breaks the pupils stay in the canteen. The capacity of the canteen is, like the capacity of the classrooms, limited, i.e. the canteen is not big enough to fit all the students at the same time. Classrooms, teachers and course groups have specified availabilities. Teachers may teach part-time at the CVA and thus not be available at every moment during the week. The CVA also has part-time students that are only available at specific times during the week. Classrooms also can be partially unavailable, i.e., a portion of the capacity of a classroom may be occupied due to an external activity. The canteen is always available during breaks.

Some courses require special equipment (e.g. language courses) and can only be given in specific classrooms. For some courses all the theory periods have to be given in the same classroom (although which classroom that is, can still be chosen).

Informally, the CVATTP can be formulated as follows. Given are a set of course groups, a set of teachers, a set of classrooms, a set of theory periods, a set of practice periods, a set of breaks of teachers and a set of breaks of course groups. The problem is to find a *time table*, i.e., an assignment of start times and classrooms to all course periods and an assignment of start times to all breaks, that satisfies the following.

1. The courses are scheduled between the first allowed start time and the last allowed completion time.
2. The capacities of the classrooms and the canteen are never exceeded.
3. Course groups are assigned to one period or break at a time, teachers are assigned to one theory period or break at a time.
4. A start time can only be assigned to a period if there is a classroom with enough capacity left and if the course groups and the teacher of that course period are available.
5. All periods of a course are scheduled in their pre-defined order.
6. Periods of a course are scheduled in appropriate classrooms during the weeks they have to be scheduled, theory periods are scheduled in classrooms which are not already occupied by other periods. Breaks are scheduled during the predefined intervals.
7. Never schedule more than one theory period of a course on the same day.
8. For some courses all theory periods have to be given in the same classroom.

We will now define the CVATTP formally.

First we define the *time tabling data* and *time tables*.

Definition 2.1 *An instance of the* time tabling data *is a 20-tuple* $\langle\, \mathcal{G}, \mathcal{T}, \mathcal{RO}, \mathcal{TB}, \mathcal{PB}, \mathcal{BG}, \mathcal{BT}, gr, ro, te, cap, Cap, pup, ln, av, av_{occ}, rt, dl, \prec, \mathcal{OS}\rangle$, *with*

- $\mathcal{G}$, *a set of course groups,*
- $\mathcal{T}$, *a set of teachers,*
- $\mathcal{RO}$, *a set of classrooms,*
- $\mathcal{TB}$, *a set of theory blocks,*
- $\mathcal{PB}$, *a set of practice blocks,*
- $\mathcal{BG}$, *a set of breaks of course groups,*
- $\mathcal{BT}$, *a set of breaks of teachers,*
- $gr : \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \rightarrow \mathcal{P}(\mathcal{G})$, *giving the course groups taking a block or a break, where* $|gr(b)| = 1$ *if* $b \in \mathcal{PB}$ *or* $b \in \mathcal{BG}$,
- $ro : \mathcal{TB} \cup \mathcal{PB} \rightarrow \mathcal{P}(\mathcal{RO})$, *giving the appropriate classrooms for a block,*
- $te : \mathcal{TB} \cup \mathcal{BT} \rightarrow \mathcal{T}$, *giving the teacher for a theory block or the teacher taking a break,*

- $cap : \mathcal{RO} \rightarrow \mathbb{N}^+$, *giving the capacity of a classroom,*
- $Cap \in \mathbb{N}^+$, *representing the capacity of the canteen,*
- $pup : \mathcal{G} \rightarrow \mathbb{N}^+$, *giving the number of pupils of a course group,*
- $ln : \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT} \rightarrow \mathbb{N}^+$, *giving the lengths of blocks and breaks,*
- $av : \mathcal{G} \cup \mathcal{T} \rightarrow \mathcal{P}(\mathbb{N})$, *expressing the availabilities of course groups and teachers,*
- $av_{occ} : \mathcal{RO} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}^+)$, *expressing the availability and degree of occupation of classrooms, where for all $r \in \mathcal{RO}$ and for all $t \in \mathbb{N}$*

  $$av_{occ}(r)(t) = 0$$

  if room r is available on moment t, and

  $$1 \leq av_{occ}(r)(t) \leq cap(r)$$

  in all other cases.
- $rt : \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT} \rightarrow \mathbb{N}$, *giving release times of blocks and breaks,*
- $dl : \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT} \rightarrow \mathbb{N}$, *giving deadlines of blocks and breaks, and*
- *a binary relation $\prec$ on $\mathcal{TB} \cup \mathcal{PB}$ such that for all $o, o', o'' \in \mathcal{TB} \cup \mathcal{PB}$,*

  $$o \not\prec o,$$
  $$o \prec o' \wedge o' \prec o'' \Rightarrow o \prec o''.$$
- $\mathcal{OS} \subseteq \mathcal{P}(\mathcal{TB})$ *a set of all sets of theory blocks which have to be scheduled in the same classroom, where*

  $$\forall_{os, os' \in \mathcal{OS} | os \neq os'} os \cap os' = \emptyset.$$

  Furthermore, for all $os \in \mathcal{OS}$ and all $o, o' \in os$,

  $$o \neq o' \Rightarrow o \prec o' \vee o' \prec o.$$

The set of all instances of the time tabling data is denoted by $\mathcal{TTD}$.

Definition 2.2 *Let $TTD \in \mathcal{TTD}$ be an instance of the time table data. A time table for TTD is a function $tt = \langle st, roa \rangle$, where*

$$st : \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT} \rightarrow \mathbb{N}$$

assigns a start time to course blocks and breaks, and

$$roa : \mathcal{TB} \cup \mathcal{PB} \rightarrow \mathcal{RO}$$

assigns a classroom to course blocks.

The next definition introduces the CVA Time Tabling Problem.

Definition 2.3 *An instance of the* CVA Time Tabling Problem *(CVATTP) consists of an instance of the time tabling data. The problem is to find a time table tt that satisfies the following constraints (cf. informal description).*

1. *All blocks and breaks are scheduled after they are released, i.e., for all $b \in \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT}$,*

$$rt(b) \leq st(b).$$

All blocks and breaks are completed before they are due, i.e., for all $b \in \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT}$,

$$st(b) + ln(b) \leq dl(b).$$

2. *The capacities of the classrooms are never exceeded. Therefore, for all $bl \in \mathcal{TB}$,*

$$\sum_{g \in gr(bl)} pup(g) \leq cap(roa(bl)) \wedge$$

$$\textit{for all } (n_1, n_2) \in av_{occ}(roa(bl)) \textit{ where } st(bl) \leq n_1 < (st(bl) + ln(bl)), n_2 = 0,$$

for all $r \in \mathcal{RO}$ let $PB_{r,t}$ be the set of all practice blocks which are scheduled in classroom r at time t, then for all $r \in \mathcal{RO}$ and all $t \in \mathbb{N}$,

$$\sum_{pb \in PB_{r,t}} pup(gr(pb)) \leq (cap(r) - n) \textit{ where } (t, n) \in av_{occ}(r).$$

The capacity of the canteen is never exceeded. Let BG_t be the set of all course group breaks which are scheduled at time t, then for all $t \in \mathbb{N}$,

$$\sum_{bg \in BG_t} pup(gr(bg)) \leq Cap.$$

3. *A course group can only be assigned to one block or one course group break at a time, i.e., for all $b, b' \in \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG}$ where $b \neq b'$,*

$$gr(b) \cap gr(b') \neq \emptyset \Rightarrow [st(b), st(b) + ln(b)) \cap [st(b'), st(b') + ln(b')) = \emptyset. \tag{1}$$

A teacher can only be assigned to one theory block or one teacher break at a time, i.e., for all $b, b' \in \mathcal{TB} \cup \mathcal{BT}$,

$$te(b) = te(b') \Rightarrow [st(b), st(b) + ln(b)) \cap [st(b'), st(b') + ln(b')) = \emptyset.$$

4. *For all blocks and breaks the needed resources are continuously available from the start time until the release time, i.e., for all $bl \in \mathcal{TB}$,*

$$[st(bl), st(bl) + ln(bl)) \subseteq \{t | av_{occ}(roa(bl))(t) = 0\} \wedge$$

$$[st(bl), st(bl) + ln(bl)) \subseteq av(te(bl)) \wedge$$

$$\textit{for all } g \in gr(bl), [st(bl), st(bl) + ln(bl)) \subseteq av(g),$$

for all $bl \in \mathcal{PB}$,

$$[st(bl), st(bl) + ln(bl)) \subseteq \{t | av_{occ}(roa(bl))(t) + pup(g) \leq cap(roa(bl))\} \wedge$$

$$[st(bl), st(bl) + ln(bl)) \subseteq av(g), \textit{with } \{g\} = gr(bl),$$

for all $br \in \mathcal{BG}$,

$$[st(br), st(br) + ln(br)) \subseteq av(g), \textit{with } \{g\} = gr(br),$$

for all $br \in \mathcal{BT}$,

$$[st(br), st(br) + ln(br)) \subseteq av(te(br)).$$

5. *All theory and practice blocks satisfy the precedence relation* $\prec$*, i.e., for all* $bl, bl' \in \mathcal{TB} \cup \mathcal{PB}$,

$$bl \prec bl' \Rightarrow st(bl) + ln(bl) \leq st(bl').$$

6. *All blocks are scheduled in proper classrooms, i.e., for all* $bl \in \mathcal{TB} \cup \mathcal{PB}$,

$$roa(bl) \in ro(bl).$$

A classroom can only be assigned to one theory block, or one or more practice blocks at a time, i.e., for all $bl \in \mathcal{TB}$ *and* $bl' \in \mathcal{TB} \cup \mathcal{PB}$ *with* $bl \neq bl'$,

$$roa(bl) = roa(bl') \Rightarrow [st(bl), st(bl) + ln(bl)) \cap [st(bl'), st(bl') + ln(bl')) = \emptyset.$$

7. *Never schedule more than one theory block of a course on the same day. Let* X *be the set containing the first points in time of each working-day except the first one. Then, for all* $bl, bl' \in \mathcal{TB}$, *where* $bl \neq bl'$, $bl \prec bl'$,

$$\exists_{t \in X}\ st(bl) < t \wedge t \leq st(bl'). \tag{2}$$

8. *All theory blocks of some courses have to be given in the same classroom, i.e., for all* $os \in \mathcal{OS}$ *and for all* $bl, bl' \in os$,

$$roa(bl) = roa(bl'). \tag{3}$$

The set of all instances of The CVA Time Tabling Problem is denoted by $\mathcal{I}_{CVATTP}$.

In the next section we will describe the model to which the above problem is translated.

3 Time and Resource Constrained Scheduling Problems

Informally, for each instance of the TRCSP a set of *operations* and a set of *resources* are given. Each operation is given a set of *feasible* resource sets, together with a *processing time* for each feasible resource set. Each resource set has an integer *capacity* and each operation has an integer *size*. A *schedule* specifies both a start time and a resource set for each operation. Formally, this is defined as follows:

Definition 3.1 *An instance of the* scheduling data *is an 8-tuple* $(\mathcal{O}, \mathcal{R}, \mathcal{RS}, fr, cp, sz, pt, H)$ *where* $\mathcal{O}$ *is a set of operations,* $\mathcal{R}$ *a set of resources, and* $\mathcal{RS} \subseteq \mathcal{P}(\mathcal{R})$ *a set of* resource sets. *Furthermore,* $fr : \mathcal{O} \to \mathcal{P}(\mathcal{RS})$ *gives for each operation the set of* feasible resource sets, $cp : \mathcal{RS} \to \mathbb{N}^+$ *gives the* capacity *of the resource sets, and* $sz : \mathcal{O} \to \mathbb{N}^+$ *gives the* size *of the operations. The function* $pt : \mathcal{O} \times \mathcal{RS} \nrightarrow \mathbb{N}^+$ *gives the* processing time *for each combination of an operation and a feasible resource set. Finally,* H *is the* scheduling horizon. *The set of all instances of the scheduling data is denoted by* $\mathcal{SD}$.

A schedule for this scheduling data is defined as:

Definition 3.2 *Let* $SD = (\mathcal{O}, \mathcal{R}, \mathcal{RS}, fr, cp, sz, pt, H) \in \mathcal{SD}$ *be an instance of the scheduling data. A* schedule s *for* SD *is a function* $s : \mathcal{O} \to \mathcal{RS} \times \mathbb{N}$ *which gives for each operation a resource set and a start time. The set of all schedules for* SD *is denoted by* $\mathcal{S}_{SD}$.

Let $SD \in \mathcal{SD}$ be an instance of the scheduling data and $s \in \mathcal{S}_{SD}$ a schedule for SD. The following functions are used throughout this paper. If no confusion is possible the index s is omitted. Furthermore, with $(x_1, \ldots, x_n) \in X_1 \times \ldots \times X_n$, we use $\pi_i(x_1, \ldots, x_n) = x_i$, for $1 \leq i \leq n$.

$ra_s : \mathcal{O} \to \mathcal{RS}$ gives the *resource set assignments* of the operations. For each operation $o \in \mathcal{O}$, $ra_s(o) = \pi_1(s(o))$.

$st_s : \mathcal{O} \to \mathbb{N}$ gives the *start times* of the operations. For each operation $o \in \mathcal{O}$, $st_s(o) = \pi_2(s(o))$.

$ct_s : \mathcal{O} \to \mathbb{N}$ gives the *completion times* of the operations. For each operation $o \in \mathcal{O}$, $ct_s(o) = st_s(o) + pt(o, ra_s(o))$.

The problem is not just to find a schedule, but to find a schedule that satisfies two kinds of constraints, viz., *mandatory constraints*, i.e., constraints that are defined for all instances of the TRCSP and so-called *additional constraints*, i.e., constraints that may differ from one instance to another. A schedule should satisfy the following mandatory constraints:

Definition 3.3 *Each operation is scheduled between time* 0 *and scheduling horizon* H *on a feasible resource set for that operation, i.e., for all* $o \in \mathcal{O}$,

$$st(o) \geq 0 \;\wedge\; ct(o) \leq H \;\wedge\; ra(o) \in fr(o). \tag{4}$$

At each point in time no resource set processes a set of operations whose cumulative sizes exceed the capacity of the resource set, i.e., for all $rs \in \mathcal{RS}$ *and* $t \in \mathbb{N}$,

$$\sum_{o \in \mathcal{O} | ra(o) = rs \;\wedge\; st(o) \leq t < ct(o)} sz(o) \leq cp(rs), \tag{5}$$

No two resource sets that have a non-empty intersection can be active simultaneously, i.e, for all $o, o' \in \mathcal{O}$,

$$(ra(o) \cap ra(o') \neq \emptyset \;\wedge\; ra(o) \neq ra(o')) \Rightarrow (ct(o) \leq st(o') \;\vee\; ct(o') \leq st(o)). \tag{6}$$

Observe that according to (6) two different resource sets that have a non-empty intersection cannot be active simultaneously. This prevents a situation in which e.g. an operator has to simultaneously operate two machines that are far apart. Obviously, also, scheduling situations occur in which two operations can be scheduled simultaneously on two different resource sets that have a non-empty intersection. We can model such situations as follows. Let $r, r', r'' \in \mathcal{R}$, and suppose that operation $o1$ needs $\{r, r'\}$ for its processing and operation $o2$ needs $\{r', r''\}$. Furthermore, $o1$ and $o2$ can be processed on r' simultaneously. By introducing two extra operations $o1'$ and $o2'$, defining $fr(o1) = \{\{r\}\} \ \wedge \ fr(o1') = \{\{r'\}\} \ \wedge \ fr(o2') = \{\{r'\}\} \ \wedge \ fr(o2) = \{\{r''\}\}$, and introducing the additional constraints $st(o1) = st(o1')$ and $st(o2) = st(o2')$, we can model this situation adequately. The extra introduced operations are called *virtual operations*.

Besides the set of mandatory constraints, each instance of the TRCSP contains a set of additional constraints. An additional constraint is either a *time constraint*, restricting the start time and completion time assignments of operations, or a *resource constraint*, restricting the resource set assignments of operations. See [6] for a formal description of the language in which these constraints can be expressed. Here we suffice to state that the language is powerfull enough to define all possible relations between two time intervals (in which possibly two operations are processed). The resource constraints can express all relations concerning resource set assignments.

Now we can define an instance of the Time and Resource Constrained Problem as follows:

Definition 3.4 *An instance of the* Time and Resource Constrained Scheduling Problem *(TRCSP) is a pair (SD, C) where $SD = (\mathcal{O}, \mathcal{R}, \mathcal{RS}, fr, cp, sz, pt, H) \in \mathcal{SD}$ is an instance of the scheduling data and $C \subseteq \mathcal{C}_{\text{add}}$ is a set of additional constraints. The problem is to find a schedule $s \in \mathcal{S}_{SD}$ for which the mandatory constraints defined in definition3.3 hold and for which all additional constraints are satisfied.*

It is easy to see that the TRCSP is a special case of the CSP. For each operation $o \in \mathcal{O}$, a variable is introduced. Furthermore, for each operation $o \in \mathcal{O}$ a domain $D(o)$ equal to

$$\{(rs, t) \in \mathcal{RS} \times \mathbb{N} \mid rs \in fr(o) \ \wedge \ cp(rs) \geq sz(o) \ \wedge \ t \in [0, H - pt(o, rs)]\}$$

is defined. This definition of the domains ensures that the constraints of (4) are satisfied and that no operation is scheduled on a resource set whose capacity is exceeded by the size of the operation. Note that with $\mathcal{O} = \{o_1, \ldots, o_N\}$, an assignment $(a_1, \ldots, a_N) \in D(o_1) \times \ldots \times D(o_N)$ corresponds to a schedule $s : \mathcal{O} \rightarrow \mathcal{RS} \times \mathbb{N}$ for which

$$\forall_{1 \leq i \leq N} \ s(o_i) = a_i.$$

For each capacity constraint of (5), each resource conflict constraint of (6), and each additional constraint of C, a corresponding constraint is introduced. This correspondence is straightforward. It can clearly be seen that a solution of the CSP instance corresponds to a solution of the TRCSP instance.

The framework that is used to handle the TRCSP consists of the traditional CSP elements. While no schedule has been found and no proof of infeasibility can be given the following actions are repeated.

First, the consistency is checked. Here we make use of the structure of the TRCSP to do a more extensive and efficient checking than what is done in the general CSP framework. A full description of the consistency checking can be found in [6].

If a dead end is detected we first perform a chronological backtrack step. If no solution is found, after a reasonable number of backtrack steps, we perform a complete restart of the search, combined with a random selection of a new operation in order to prevent following the same path again.

If we are not in a dead end, an operation and assignment for the operation are selected. The selection strategy for operations and assignments we used for the time tabling problem are discussed in section 5, where we discuss the test results.

4 The time tabling problem modelled as a time and resource constraint scheduling problem

Formulating the CVATTP as a special case of the TRCSP is not a trivial task. Although for some elements of a CVATTP instance a straightforward translation into elements of the scheduling data and additional constraints can be found, other elements can not be directly translated. First, an overview is given of the translation of those parts of the CVATTP instance that can be directly translated into the TRCSP format. Next, we extend the resulting TRCSP instance with extra scheduling data elements and additional constraints until it forms a complete representation of a CVATTP instance.

Let I be an instance of the CVATTP. We define a corresponding instance $I' = \langle SD, C \rangle$ of the TRCSP , with $SD = \langle \mathcal{O}, \mathcal{R}, \mathcal{RS}, fr, cp, sz, pt, H \rangle$ as follows:

1. For each theory period, practice period and break of I, an operation is introduced in I'. The corresponding operation types are called *theory operations*, *practice operations*, *course group break operations* and *teacher break operations*.
   Formally, $\forall o \in \mathcal{TB} \cup \mathcal{PB} \cup \mathcal{BG} \cup \mathcal{BT}$ introduce an operation o' in $\mathcal{O}$.
2. For all course groups, teachers, classrooms and the canteen a resource is introduced in $\mathcal{R}$.
   Formally, $\forall r \in \mathcal{G} \cup \mathcal{T} \cup \mathcal{RO} \cup \{canteen\}$ introduce a resource r' in $\mathcal{R}$.
3. The scheduling horizon H is set on four weeks (and actually plays no active role in the time tabling problem).

For each of the operation types we will now indicate the feasible resource set and the size. The processing time of each operation is equal to the length of the corresponding period or break.(I.e. $pt(o') = ln(o)$).

For theory operations the feasible resource set consists of a capable teacher, an appropriate classroom and one or more course groups. The size of the operations is put to one. We do not take the size of the coursegroups into account, because we assume that the coursegroups fit into every (appropriate) classroom.
For practice operations the resource set consists of a classroom and one or more course groups. The size of these operations is equal to the size of the course group taking the practice.
The feasible resource sets of course group break operations consist of the canteen and the course group. The size of these operations is also equal to the size of the course group taking the break.
The feasible resource set of teacher break operations consists of the teacher only.(They do not use the canteen). The size of these operations is equal to one, indicating the teacher taking the break.

Because two practice operations may be scheduled in the same classroom they may have intersecting resource sets, which is not possible within the TRCSP framework. The same holds for the breaks of course groups. Therefore we have to use the general construction, described in section 3, to prevent intersecting (but not equal resource sets) by splitting up the resource sets and introducing virtual operations. In this case, we split off the course groups from the feasible resource sets.
All this leads to the following resource sets in the TRCSP model:

1. $\{\{\tau', \rho'\} | \tau \in \mathcal{T} \text{ and } \rho \in \mathcal{RO}\}$
2. $\{\{\rho'\} | \rho \in \mathcal{RO}\}$
3. $\{\{canteen\}\}$
4. $\{\{\gamma'\} | \gamma \in \mathcal{G}\}$

5. $\{\{\tau'\}|\tau \in \mathcal{T}\}$

The capacity of the resource sets consisting of a classroom is equal to the capacity of the classroom, the capacity of the resource set consisting of the canteen is equal to the capacity of the canteen. The capacity of the other resource sets is equal to 1.

As said above, we have to introduce virtual operations corresponding to theory operations, practice operations and course group break operations to make it possible to split up the resource sets. The feasible resource set of each of these operations consists of the course group taking that period or break. The size of these operations is equal to one, which is equal to the capacity of the corresponding feasible resource set. The reason to set both equal to one and not the size of the course group is the fact that consistency checking is easier with resource sets having capacity one.

The processing times of the virtual operations is equal to the processing time of the corresponding period or break. An additional constraint is introduced for each virtual operation which forces the start time of the virtual operations to be equal to the start time of the cooresponding theory, practice or course group break operations.

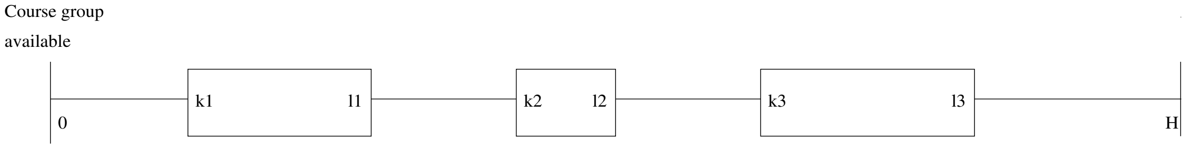
Now we have established the basic elements of the CVA time tabling problem in the TRCSP model, we will indicate how the constraints 1-8, given at the end of section 2, are modelled within the TRCSP framework.

1. Start times and completion times, In conformity with the elements of the scheduling data it is not possible to indicate release times or deadlines which differ from the beginning and the end of the period (given by H). Therefore, we have to use additional constraints to model different release times and deadlines for the operations o corresponding with the periods and breaks. If the release time of o is x then the constraint $st(o) \geq x$, is added to C. And if the deadline of o is y then the constraint $ct(o) \leq y$, is added to C.

2. Capacities, The capacities of the classrooms and the canteen are never exceeded due to the modelling of the standard data elements and the mandatory constraints.

3. Unique assignments, By modelling course groups and teachers as resource sets of periods and breaks, it is not possible to assign a course group or teacher to more than one period or break.

4. Availability of teachers and course groups and capacity of classrooms, To ensure the satisfaction of constraints concerned with availabilities of students and teachers, we introduce a new (virtual) operation type called: "*availability operation*". The availability operations occupy the teachers and course groups during the periods that they are not available. Suppose that a teacher or course group (represented by the resource set $\{\{r\}\}$) is available at the following intervals:

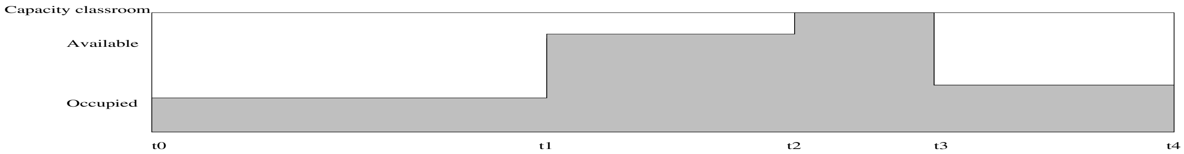


then we introduce operations $o_{i\,av}$ $i \in \{0,\ldots,m\}$ such that

$$fr(o_{i\,av}) = \{\{r\}\},\ sz(o_{i\,av}) = 1,\ pt(o_{i\,av},\{r\}) = k_{i+1} - l_i - 1,\ \text{and}\ st(o_{i\,av}) = l_{i-1} + 1$$

where l_0=-1 and k_{m+1}=H.

To account for the partial availability of classrooms, we introduce (virtual) operations that occupy exactly that part of the classroom that is not available. Suppose the availability of a classroom (represented by the resource set $\{\{\rho\}\}$) looks as in the following picture:



Then we introduce availability operations
$o_{i\,av}$ $i \in \{j \mid \text{classroom not completely available between } t_j \text{ and } t_{j+1}\}$, such that

$$fr(o_{i\,av}) = \{\{\rho\}\},\ sz(o_{i\,av}) = av_{occ}(\rho)(t_i),\ pt(o_{i\,av}, \{\rho\}) = t_{i+1} - t_i + 1 \text{ and } st(o_{i\,av}) = t_i$$

5. Precedence relation, To express the fact that the periods of a course have to be scheduled in a pre-defined order we introduce the following constraint in C

$$ct(o) \leq st(o')$$

for every o, o' such that o and o' are periods from the same course and o should be scheduled before o'.

6. Appropriate classrooms and times, This constraint can be modelled in the same way as constraint 1, using start and completion times. The fact that courses are scheduled in appropriate classrooms can be enforced by using appropriate feasible resource sets for the course periods.

7. One theory period of a course per day, This constraint states that all theory periods of a course should be scheduled on a different day. The TRCSP model does not provide constraints to spread operations over time. This type of constraints can be formulated in terms of the TRCSP as follows. First we introduce a so-called *hedge* resource: "*night*". The night is only available *between* successive working-days. Furthermore, for each pair of theory operations o_{t1}, o_{t2} such that o_{t1} has to be scheduled before o_{t2} and there is no o_{t3} that has to be scheduled between o_{t1} and o_{t2}, a so called *different day* operation o_{dd} is introduced, where

$$fr(o_{dd}) = \{\{night\}\},\ sz(o_{dd}) = 1, \text{and } pt(o_{dd}, \{night\}) = 1.$$

The night takes one time unit. Furthermore, the constraints

$$ct(o_{t1}) \leq st(o_{dd}), \text{ and } ct(o_{dd}) \leq st(o_{t2}),$$

are introduced in C. The set $\mathcal{RS}$ is extended with $\{\{night\}\}$, and $cp(\{night\}) = \mid \mathcal{O} \mid$. It is obvious that this construction forces the theory periods to be scheduled on different days.

8. All theory periods in one classroom, To express the fact that all theory periods of certain courses have to be scheduled in the same classroom, we introduce the following additional constraints. Let o and o' be two periods that have to be scheduled in the same classroom, then if o and o' are tought by the same teacher the resource assignments have to be completely equal and thus:

$$ra(o) = ra(o'),$$

If o and o' are tought by different teachers, the resource assignments are not completely equal but we can formulate the following constraint:

$$\{\rho\} \not\subseteq ra(o) \vee \{\rho\} \subseteq ra(o')$$

This concludes the formulation of the CVATTP as a special case of the TRCSP. In the next section we will present the computational results of the generalized constraint satisfaction algorithm.

5 Some results

In this section we present some computational results for real-life instances of the CVATTP. Although a number of experiments were done, due to space limitations,we will show the results of only one experiment. In this experiment we do not consider instances of a length of four weeks, because these result in instances of the TRCSP that are to big to fit into the internal memory of a SPARC-station ELC. Therefore, we split each full size instance into instances of one week. This is a valid choice because there are hardly any dependencies between the weeks and the number of periods and breaks to schedule per week is known and invariable.
The week we consider in the experiment embodies 10 courses, attended by 9 course groups and 9 teachers in 18 classrooms. The week instance is subdivided in five classes according to the manual schedule process at the CVA. The classes originate from the length of the intervals during which the course groups are available. The five classes are numbered from 1 to 5 with 5 being the class where the course groups (especially part-timers) are least available.

Operation selection. We apply four different methods of operation selection, called Greedy Greedy+Priority, OSSolve and OSSolve+Priority. When Greedy is applied, the algorithm randomly selects an operation from the set of unscheduled operations which can start first.
Greedy+Priority equals Greedy, with one difference, viz., the algorithm randomly selects first from the set of unscheduled theory operations and only after that from the set of unscheduled practice operations, with the idea that the latter can be scheduled easier and therefore should be selected later. When OSSolve is applied, first, the earliest minimal completion time of any unscheduled operation is determined. Next, the algorithm randomly selects one operation from the set of unscheduled operations which can start before this completion time. When OSSolve+Priority is applied, the algorithm uses the priorities as with Greedy+Priority.

The number of operations per instance class are as follows: class 1 contains 835 operations, class 2 contains 826 operations and class 3,4 and 5 contain 817 operations each.
The instance contains 38 resources, spread over 100 resource sets. The number of constraints equals 358, 126 constraints are due to precedences.

We performed a series of five runs, using different seeds for the random number generator. Each run is executed for a maximum of 900 seconds on a SUN workstation. If a solution is found within 900 seconds the run is successfully terminated. The computational results are presented in the following table, in which the rows represent the five instance classes. The columns of the tables correspond to the various operation selection methods. Each column is subdivided in three sub-columns giving the number of successful runs, the average number of restarts per successful run and the average computation time per successful run. From the viewpoint of the operation selection methods, the selection methods with priorities appear to perform better than the matching versions without priorities. These results confirm the experiences of the manual time tabling process. Furthermore, Greedy+Priority and OS-Solve+Priority do not diverge much. This indicates that the operation selection as done in OSSolve

	S			SP			G			GP		
class	suc	rst	t	suc	rst	t	suc	rst	t	suc	rst	t
1	5	1	74	5	1	74	5	1	72	5	1	73
2	5	1	73	5	1	73	5	1	71	5	1	72
3	5	1	73	5	1	74	5	1	72	5	1	71
4	5	3.4	273	5	6.4	430	5	4	312	5	3.2	241
5	3	8.4	598	5	8	570	1	1	78	5	6	467

Table 1: Results of the experiment.

is more beneficial for types of scheduling for which the left justification of operations is important (like in job shop scheduling).

6 Conclusions

In this paper we have shown how a fairly complicated time tabling problem can be solved using constraint satisfaction first translating the problem to a more general TRCSP model, which can be solved in an efficient way, because we can use the structure of that model.
Although the resulting instance of the TRCSP was solved, it should be recognized that the four week instance was to big to solve in one step. A breakdown into weeks was necessary, but proved to be feasible. A solution for the complete problem can be made by a simple combination of the one week solutions.
A topic for further research could be the possible extension of the TRCSP model with constraints expressing the spreading of operations over periods of time. If such constraints would be added and efficiently handled the TRCSP model would be considerably reduced in size, because many of the virtual operations that blow up the model are introduced through the modelling of these constraints.

References

[1] D. Abramson. Constructing school timetables using simulated annealing: sequential and parallel algorithms. *Management Science*, 37(1):98–113, January 1991.

[2] O.B. de Gans. A computer timetabling system for secondary schools in the netherlands. *European Journal of Operational Research*, 7:175–182, 1981.

[3] D. de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19:151–162, 1985.

[4] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5(4):691–703, December 1976.

[5] A. Monfroglio. Timetabling through a deductive database: a case study. *Data & Knowledge Engineering*, 3:1–27, 1988.

[6] W.P.M. Nuijten. *Time and Resource Constrained Scheduling: A Constraint Satisfaction Approach*. PhD thesis, Eindhoven University of Technology, 1994.

[7] W.P.M. Nuijten, G.M. Kunnen, E.H.L. Aarts, and F.P.M. Dignum. Examination time tabling: A case study for constraint satisfaction. In *Proc. ECAI '94 Workshop on Constraint Satisfaction Issues Raised by Practical Applications*, pages 11–19, 1994.

[8] G. Schmidt and T. Strohlein. Timetable construction-an annotated bibliography. *The Computer Journal*, 23(4):307–316, 1979.