A New Model for Automated Examination Timetabling

Barry McCollum, Paul McMullan

School of Computer Science, Queen's University, Belfast,
University Road, N. Ireland, BT7 1NN, UK,
{ b.mccollum, p.p.mcmullan }@qub.ac.uk,

Edmund K. Burke, Andrew J. Parkes, Rong Qu

School of Computer Science, University of Nottingham,
Jubilee Campus, Nottingham, NG8 1BB, UK,
{ ekb, ajp, rxq }@cs.nott.ac.uk

April 29, 2008

Abstract

Automated examination timetabling has been addressed by a wide variety of methodologies and techniques over the last ten years or so. Many of the methods in this broad range of approaches have been evaluated on a collection of benchmark instances provided at the University of Toronto in 1996. Whilst the existence of these datasets has provided an invaluable resource for research into examination timetabling, the instances have significant limitations in terms of their relevance to real-world examination timetabling in modern universities. This paper presents a detailed model which draws upon experiences of implementing exam timetabling systems in universities in Europe, Australasia and America.

This model represents the problem that is presented in the 2nd International Timetabling Competition (ITC2007). In presenting this detailed new model, this paper describes the examination timetabling track introduced as part of the competition. In addition to the model, the datasets used in the competition are also based on current real world instances introduced by EventMAP Limited. It is hoped that the interest generated as part of this competition will lead to the development, investigation and application of a host of novel and exciting techniques to address this important real world search domain. Moreover, the motivating goal of this paper is to close the currently existing gap between theory and practice in examination timetabling by presenting the research community with a rigorous model which represents the complexity of the real-world situation. In this paper we describe the model and its motivations, followed by a full formal definition.

1 Introduction

Building on the success of the First International Timetabling Competition in 2002 [1], the second competition (ITC2007)[1] was introduced with the overall aim of attracting researchers to develop and test leading edge techniques within a competitive arena. It also aimed to generating further interest in the research area by providing various formulations of the timetabling problems encountered within educational institutions and therefore based on a 'real world' perspective. Particular emphasis was placed on 'real world' scenarios with the objective of encouraging the production of techniques which have the potential to solve practical instances of the problem. The competition therefore had, and will continue to have, an important part to play in bridging the current gap which exists between research and practice in this area [2].

[1] `http://www.cs.qub.ac.uk/itc2007/`

In order to tackle the main variations which exist within the practical area of educational timetabling, the competition was divided into three sections or tracks.

1. Examination Timetabling (Exam TT).
2. Post-Enrolment Course Timetabling (Post-Enroll CTT)
3. Curriculum-Based Course Timetabling (Curriculum CTT).

The competition introduced three tracks along with associated benchmark datasets. From a research perspective this division is important in that it provides a framework to capture the main types of educational timetabling research currently taking place within the academic community. Although the tracks all fall under the general 'umbrella' of educational timetabling, the identified problem areas have significant differences which are discussed in detail at the Competition website and in the associated article [3]. In addition, technical reports for all areas are available from the official website and can be found under each track [4, 5, 6].

In this paper, we present the real-world model that was used in the Examination Timetabling Track. The information presented here is self-contained, but can also be regarded as a detailed and rigorous description of the problem presented in that track and incorporates and extends the content on the ITC2007 website. In addition to the presentation of the model, we also give a description of the competition's hierarchical solution evaluation methods.

1.1 The Examination Timetabling Problem: Modelling

Previous research has concentrated on the Toronto benchmark dataset[2] introduced by Carter et al [7] at the University of Toronto, but collected from various universities in North America. These benchmarks and the various issues associated with them are discussed in more detail in Qu et al [8]. However, the Toronto dataset uses only a very limited model, and so the complexity of modelling of timetabling problems continues to pose a series of important and challenging issues in the timetabling research area [2].

There is some research in the literature on building general timetabling languages and tools in an attempt to model real world instances of the problem. Tsang, Mills and Williams [9] developed a high level language to specify exam timetabling problems as constraint satisfaction problems. Di Gaspero and Schaerf [10] built a software tool called EASYLOCAL++ for the easy implementation of local search algorithms on general timetabling problems.

De Werra, Asratian and Durand [11] in 2002 presented a simple model and some possible extensions for class-teacher timetabling problems. The computational complexity of these problems was also studied, showing some variants of the problem as NP-complete. However, significant further work is required to address these and related issues. Moreover, it is needed to provide fundamental support for a better understanding and development of real-world exam timetabling research. There is a very wide variety of institutional practices, for example, Al-Yakoob et al [12] give an integer programming model of an examination timetabling problem that includes "idiosyncrasies of the problem related to gender-based policies and having multiple exam centers."

Generally, these modelling issues have not been widely discussed over the last ten years, and given the wide variety of institutional practice it is not surprising that there are no

[2] `ftp://ftp.mie.utoronto.ca/pub/carter/testprob/`

universal complete models. However, the point of this paper is to provide a model that is substantially more general and reusable than existing frameworks. Our intention is that it will further improve the current development and justify clear and meaningful scientific comparisons.

1.2 The Examination Timetabling Problem: Algorithms

The purpose of this paper is not to present algorithms however for context and completeness we briefly discuss some of the existing algorithms. The last ten years have seen a significant amount of research; for a survey, see [8]. Naturally, meta-heuristics have been widely studied, for example, Tabu Search [13, 14, 15], Simulated Annealing [16, 17, 18], Genetic Algorithms [19, 20], Memetic Algorithms [21], Ant Algorithms [22, 23], Multi-objective Evolutionary Algorithms [24]. There is a substantial body of work where hybridisations between meta-heuristics are studied. Indeed, these hybridisations are currently the most effective approaches on the standard Toronto benchmarks. This includes the effective integration of early timetabling techniques such as graph heuristics [25, 26, 27, 28] and constraint based techniques [17, 29, 30].

Along with these main themes of algorithm research there are also a number of new trends including more effective design of neighborhood structures (e.g. variable neighborhood search for timetabling [31, 32]). Flexibility of search is thus improved to tackle more complex problems with a wider range of constraints. Some research motivated by the objective of raising the generality of timetabling approaches has also obtained promising results, hyper-heuristics [33] being one of the areas that is attracting significant research attention [26, 27, 28, 31, 34, 35].

1.3 Overview of the Paper

The Examination Timetabling Problem addressed here introduces a practical formulation of the problem which represents the situation as it exists in many modern universities and which will set the agenda for future efforts in the area by establishing new challenging benchmarks which are relevant to the needs of the user community. An additional goal, it is intended, is that the interest generated by the release of this model will lead to the development, investigation and application of a host of novel and exciting techniques not previously tried within this important real-world search domain.

The problem model can be described as post enrollment. That is, students enrolled on particular courses which have associated exams are considered to be enrolled on or 'taking' those exams. Although other approaches to the problem are taken within institutions, this is by far the most common from a practical perspective as well as being the most widely reported model of the problem within the academic literature.

The model presented in this paper not only significantly adds to the research field by the introduction of a more 'real' representation of the problem, but is also associated with real benchmarks. All the datasets used as part of the competition were taken from real institutions (with their permission) and were anonymised so that they could be publically released for the benefit of the research community.

Section 2 gives a non-mathematical description of the problem together with some motivating discussion. Section 3 gives a mathematical programming formulation of the problem. For ease of exposition, we have kept it separate and compact. Section 4 presents some remarks about the penalties. Section 5 presents some concluding comments.

2 The Model

The fundamental problem involves assigning exams to a given number of periods within a defined examination session while satisfying various hard constraints. As with other areas of timetabling, a feasible solution is one in which all but the soft constraints are satisfied. The quality of the solution is measured in terms of soft constraint satisfaction.

Compared to previous benchmark models, new and additional information is provided on constraints, resources and the examination session. For example, in terms of hard constraints, room numbers and sizes are provided. In addition, information on the structure, duration and number of individual periods is also given. In terms of soft constraints, much more practical information is provided in terms of how an organisation measures the overall quality of a solution.

We remark that our goal in this paper is to give the semantics of the model, and not to delve into the particular syntax in the instance files; such details can be found on the competition website.

2.1 Problem Description

The problem consists of the following:

- An examination session is made of a number of periods over a specified length of time: The number of periods and duration of each period are provided.
- A set of exams that are to be scheduled into periods. (Exam codes are not provided, instead we assume sequential numbering beginning with 0.)
- The set of students enrolled on each individual exam: Each student is enrolled on a number of exams. Students enrolled on an exam are considered to 'take' that examination.
- A set of rooms with individual capacities.
- A definition of the constraints that any feasible solution must satisfy.
- Specific instance-dependent constraints on feasible solutions.
- Soft Constraints (i.e. those which are desirable but not essential) are outlined: the violation of these constraints contribute to a penalty which is used to evaluate feasible timetables.
- Details including a 'weighting' of particular soft constraints.

A feasible timetable is one in which all examinations have been assigned to a single period and room, and in which all the following constraints are satisfied:

- No student sits more than one examination at the same time.
- The capacity of individual rooms is not exceeded at any time throughout the examination session.
- Period durations are not exceeded.
- Period related hard constraints (e.g. Exam_A after Exam_B).

- Room related hard constraints (e.g. Exam_A must use Room 101).

Notice that, unlike course timetabling, exams are generally permitted to share rooms.

The soft constraints and their violation penalties are discussed in detail later, but can be outlined as follows;

- Two exams in a row: The number of occurrences when students have to sit two exams in a row on the same day.
- Two exams in a day: The number of occurrences when students have to sit two exams on the same day. This constraint only becomes important when there are more than two examination periods in the same day.
- Specified spread of examinations: This is the number of occurrences when students have to sit more than one exam in a time period specified by the institution. This is often used in order to try to satisfy the preference by students that their exams should not be too close together.
- No Mixed duration of examinations within individual periods: This corresponds to the number of occurrences of exams timetabled in rooms along with other exams of differing time duration.
- Larger examinations appearing later in the timetable: This represents the number of 'large' exams appearing in the 'later portion' of the timetable. Both 'large' and 'later portion' are user defined.
- Period related soft constraints: These represent the number of times a period is used which has an associated penalty. This is multiplied by the actual penalty as different periods may have different associated weightings.
- Room related soft constraints: These correspond to the number of times in which a room is used which has an associated penalty. This is multiplied by the actual penalty as different rooms may have different associated weightings.

These constraints can effectively be split into two groups i.e. those which are resource specific and those which can have a global setting. Period related and room related constraints are resource specific i.e. settings can be established for each period and each room. This allows control of how resources would be used in constructing a solution. Values for these can be found after the introduction of periods and rooms in the datasets. All other soft constraints can be set relative to each other. These are referred to as global settings.

Institutions may weight these soft constraints differently (relative to one another) in order to produce a solution which is appropriate for their particular needs. This is known as building the 'Institutional Model' and the associated weighting is defined here as the Institutional Model Index. This is a relative weighting of the soft constraints which effectively provides a quality measure of the solution to be built which suits the particular institution in which it is to be built. These weightings are provided within each benchmark dataset released on the competition webpage[3].

It should be noted that when formulating a solution, it is common place for an institution to 'play' with various settings of soft constraints in an attempt to produce solutions which they judge satisfactory to all the end users. Indeed, this is why we provided the soft constraint weightings in the data as opposed to the problem definition. In addition,

[3] `http://www.cs.qub.ac.uk/itc2007`

including the weights in the data rather than requiring them to be hard coded into the solver enabled us to set different weightings for each dataset. We hope that this encourages the development of solvers that are robust rather than potentially over-tuned to one particular set of weights for a dataset. Once again, this is motivated by our experience that different institutions do indeed have different weights, and so no one set would be completely useful across a range of problems.

The details provided here significantly add to the model of the problem commonly used within the research arena. Of course, how individuals judge that particular solutions are 'satisfactory' is an interesting open research problem and is currently being tackled in a number of novel ways (for example, see Asmuni et al [36]).

2.2 Framework for Evaluation of Solutions

Generally, in real-world applications, not all constraints are of equal importance to the end-user. Often, this is captured by separating constraints into "hard" constraints that must be satisfied, and "soft" constraints that can be violated at the cost of contributing a penalty to the objective function. However, such a two-level approach can easily be generalised to multiple levels to give "constraint hierarchies" [37, 38]. The basic idea of such a hierarchy is that constraints are separated into multiple levels, and constraints of a higher level are always to be preferentially satisfied over those of a lower level.

Although not presented as such, the models in the three tracks of ITC2007 effectively used three levels of constraints which we denote here as follows:

1. **Required**: "never broken". Constraints that absolutely can *never* broken. Any assignment violating such constraints cannot be used, and so the corresponding solution is worthless.
2. **Hard**: "exceptionally broken". Constraints that are strongly preferred not to be violated, but for which violations can be tolerated *exceptionally*, that is, only under special and unusual circumstances and then only when necessary.
3. **Soft**: "normally broken". The sets of constraints that we prefer to satisfy but we expect that it will *normally* be partially broken, that is, it will not be possible to satisfy them all.

(Of course, the names of the three levels are somewhat ad hoc; it might help, in some circumstances, to think in terms of the corresponding breakage of constraints as being 'never', 'exceptional', or 'normal.')

The hard and soft constraints can be broken, and so it is natural to assign separate penalty scores measuring the extent to which they are broken. Hence each solution satisfying the required constraints is given a score of a pair of numbers, (d, s), with

d : Penalty measuring the breakage of the **Hard** constraints. We will informally refer to this as the "*distance to feasibility*" (though it is not necessarily any form of distance in the solution space).

s : Penalty measuring the breakage of the **Soft** constraints.

The notion of a hierarchy is that improving the solution at any level takes precedence over improving it at lower levels. In terms of the scores this means that comparison

is first in terms of the value of d, only if the d scores are equal then the s-scores are compared. That is, the (total) ordering is defined by

$$(d_1, s_1) > (d_2, s_2) \quad \text{if and only if} \quad d_1 > d_2 \text{ or } (d_1 = d_2 \text{ and } s_1 > s_2) \tag{1}$$

This is also equivalent to taking a single objective score $Md+s$ with M being any weight that is large enough such that any improvements in d always improve the overall value independent of the change in s.

By a "feasible solution" we will mean one that satisfies all required and hard constraints. Thus all feasible solutions have $d = 0$ and the comparison of their scores reduces to the standard comparison of their soft violation scores, s.

In order to specify such a 3-level model one first needs to catalogue the relevant constraints, then need to make decisions as to which level they should be placed. (Potentially, each type of constraint could be used at any level, and so in general we could decide separately where to place every instance of each constraint type. Each constraint instance could be assigned to a level and given a weight. However, we have not found such generality to be necessary.) Soft constraints are easiest to identify because solutions routinely violate them. We are then left with constraints that might well be classified as either required or hard. However, if feasible solutions are always used then all hard and required are satisfied and so the distinction is not important.

One of the simplest choices for a *distance to feasibility* would to allow exams to remain unassigned to a period and room. The *distance to feasibility* could then, for example, be the number of such unscheduled exams, or a sum weighted by the number of students in each examination. (This is essentially the choice taken by the post-enrollment CTT track.) It would have the advantage of being simple, however, it is unrealistic: of all constraints surely the one that every exam must take place is the one that is surely "required" rather than merely "hard."

Also, in real-world exam timetabling, at the stage of initial planning, infeasibility will typically be met with addition of extra resources such as extra rooms or extra periods. That is, the incumbent timetabler would normally restore feasibility by introducing another period or indeed another room and set a high penalty for their use. Typically, infeasibility tends to arise at later stages in the planning process; for example, if the data changes after creating provisional timetables. Such cases then can correspond to the "exceptional" circumstances under which some hard constraints might be allowed to be broken.

Such issues require a detailed investigation and discussion and as such are beyond the scope of this paper. They will be investigated in future work. In this paper, our focus is on presenting a new and realistic model for the soft constraints; those that are broken normally.

Hence, in the exam track for the competition, and in this paper, we do not make a distinction between the "hard" and "required" constraints but instead simply call them all "hard". Any solution violating one or more hard constraints is called infeasible. Furthermore, for the competition only feasible solutions were acceptable. Equivalently, all infeasible solutions received the same score of $(1, 0)$. This made all infeasible solutions equally bad, and all worse than any feasible solution.

Accordingly, in order to describe our new model we will only need to define the hard and soft constraints.

2.3 The Hard Constraints

As already outlined in section 2.1, feasible solutions must satisfy the constraints discussed below.

Complete-Allocation: All exams must actually take place. Scheduling problems are never a good reason to fail to run an exam!

No-Conflicts: Conflicting exams cannot be assigned to the same period. Exams conflict whenever they have some student taking them both.

No-Period-Splitting: Exams cannot be split between periods.

No-Room-Splitting: Exams cannot be split between rooms.

Room-Occupancy: For every room and period no more seats are used than available for that room

Period-Utilisation: No exam assigned to a period uses more time than available for that period.

Period-Related: All of a set of ordering requirements are obeyed.

Room-Related: Room requirements are obeyed.

Note that many of these hard constraints are also potentially breakable in exceptional circumstances; or even potentially, normally broken in some institutions. However, for concreteness of the model in this paper they are treated as hard.

We emphasise that although we have presented a particular choice in the model presented here for which constraints are hard and which are required; however, this choice is made to represent the views of typical institutions and it is quite possible that other choices are also useful for specific institutions. For example, a lack of large rooms might lead institutions to treat **No-Room-Splitting** as a soft constraint, or to soften the **Room-Occupancy** by allowing a few extra seats when really needed.

2.4 The Evaluation Function: Soft Constraints

Within the model presented here, (and within the competition) feasible solutions are classified based on the satisfaction of the soft constraints. The following provides a description of how each soft constraint is calculated, and also discusses its motivations based on real-world experience. We also give pointers to the appropriate subsections in the mathematical formulation of section 3.

2.4.1 Two Exams in a Row

This calculation considers the number of occurrences where two examinations are taken by students one straight after another, i.e. back to back. Once this has been established, the number of students are summed and multiplied by the number provided in the 'two in a row' weighting within the 'Institutional Model Index'. Note that two exams in a row are not counted overnight: i.e. if a student has an exam in the last period of one day and another the first period the next day, this does not count as two in a row. (See section 3.8.1).

2.4.2 Two Exams in a Day

In the case where there are three periods or more in a day, the number of occurrences of students having two exams in a day which are not directly adjacent, i.e. not back to back, are calculated. The total number is subsequently multiplied by the 'two in a day' weighting provided within the 'Institutional Model Index'. Therefore, two exams in a day are considered as those which are not adjacent i.e. they have at least a free period between them. This is done to ensure a particular exam placing within a solution does not contribute twice to the overall penalty. For example, if Exam A and Exam B were in adjacent periods in the same day, the penalty would be counted as part of the 'Two exams in a row penalty'. It should be noted that where the examination session contains days with 2 periods, this component of the penalty, although present for continuity, always is zero and hence superfluous. (See section 3.8.2).

2.4.3 Period Spread

This soft constraint enables an organisation to 'spread' an individual's examinations over a specified number of periods. This can be thought of as an extension of the two soft constraints previously described. Within the 'Institutional Model Index', a figure is provided relating to how many periods the solution should be 'spread' over. The higher this figure, potentially the better the spread of examinations for individual students. In the author's commercial experience, constructing solutions while changing this setting has led to timetables with which the institution is much more satisfied. If, for example, PERIODSPREAD within the 'Institutional Model Index' is set at 7, for each exam we count all the occurrences of enrolled students who have to sit other exams afterwards but within 7 periods i.e. the desired period spread. This total is added to the overall penalty. It should be noted that the occurrences here will have contributed to the penalty calculated for the 'two exams in a row' and 'two exams in a day' penalties. Although, a single occurrence within the solution is effectively penalised twice, it is often necessary due to, as indicated above, many institutions requiring certain spreads to be minimised as an indication of solution quality. (See section 3.8.3).

2.4.4 No Mixed Durations

This applies a penalty to a Room and Period (not Exam) where there are mixed durations. The intention here is to try and ensure that exams occur together which are of equal duration. In calculating this portion of the penalty, the mixed duration component of the 'Institutional Model Index' is calculated by the number of violations detected. (See section 3.8.4).

2.4.5 Front Load

It is desirable that examinations with the largest numbers of students are timetabled at the beginning of the examination session. In order to take account of this the FRONT-LOAD expression is introduced. Within the 'Institutional Model Index' the FRONT-LOAD expression has three parameters e.g. $(100, 30, 5)$. The first of these is the number of largest exams that are to be considered. Largest exams are specified by class size. If there are ties by size then exams occurring first in the data file are chosen. The second parameter is the number of last periods to take into account and which should be ideally avoided by the large exams. The third parameter is the penalty or weighting that should be added each time the soft constraint is violated. This allows the institution to attempt

to ensure that larger exams occur earlier in the examination session. This constraint is very popular in practice as exams with more students enrolled take longer to mark. (See section 3.8.5).

2.4.6 Room Penalty

It is often the case that organisations want to keep certain room usage to a minimum. As with the 'Mixed Durations' component of the overall penalty, this part of the overall penalty should be calculated on a period by period basis. For each period, if a room used within the solution has an associated penalty, then the penalty for that room for that period is calculated by multiplying the associated penalty by the number of exams using that room. (See section 3.8.6).

2.4.7 Period Penalty

It is often the case that organisations want to keep certain period usage to a minimum. As with the 'Mixed Durations' and the 'Room Penalty' components of the overall penalty, this part of the overall penalty should be calculated on a period by period basis. For each period, the penalty is calculated by multiplying the associated penalty by the number of exams timetabled within that period. (See section 3.8.7).

Also the website provides a "validator" that is available to help check solutions and give the expected scores. On the website, in order to explain the calculation of the penalty, a simple example is used allowing individual components of the overall penalty to be explained. And also various data files and solution files are provided, and these can also be used to verify any intended implementations.

2.5 Remarks

These constraints are relatively complex, and certainly more complicated than those in previous models. Hence, a practical issue is that implementations are more likely to be error-prone. To help users of the model cope with these issues the competition website provides a "validator" to check solutions and give their scores. Also, to reduce potential ambiguities due to the above natural language description, in the next section, we provide mathematical definitions for the above descriptions.

3 Integer Programming Formulation

The mathematical formulation given here is intended to provide a comprehensive and rigorous definition of the new model. Accordingly, it was designed for compactness and (relative) clarity, Consequently, the encodings we give are not optimised towards giving the best results when presented to an integer programming solver. However, they are still usable, albeit only on instances somewhat smaller than those provided as challenges in the competition.

The problem description captures real-world constraints, and so is inherently rather long and complicated. This inevitably has the effect that ambiguities and misunderstandings can easily creep into implementations. Accordingly, we implemented three separate solution validators and cross-checked their results. Two of these validators were coded directly and independently, the third was based on encoding the formulation given in this

section into Ilog's OPL/CPLEX[4]. It is then straightforward to create a solution validator by using a given solution to fix the primary decision variables specifying the period and room assignments. In this case, integer programming solvers easily (i.e. without branch-and-bound, but just preprocessing and the linear relaxation) determine whether the solution is feasible, and simultaneously produce the values for the various penalties. We extensively cross-checked the results of such direct implementations with the integer programming formulation. In any model as complicated as the one presented here, the chances of ambiguity, misunderstanding, or simple human-error increase dramatically, and so such extensive testing is tedious but necessary. As discussed earlier, we also directly implemented solution validators, and one of these was made available on the competition website.

To help render the following formulation more readable we will follow the following conventions:

- Sets (of exams,etc) are upper case
- Parameters (quantities whose value is known or easily derivable from the input files) are lower case
- Variables (quantities whose value is to be determined by the search) are upper case.
- When quantities such as size are associated with two different types, such as exam size and room size, then, rather than increase the usual plethora of symbols, we'll indicate the "type" with a superscript. For example, s^E and s^R are used for exam and room sizes respectively.

3.1 Sets and Parameters

The following sets and parameters are either directly present in the input file, or are straightforward to derive from it.

Exam Related:

E : set of exams

s_i : size of exam $i \in E$

d_i^E : duration of exam $i \in E$

f_e^E : a boolean that is 1 if exam e is subject to the Front-Load penalties, 0 otherwise

D : the set of durations used $\cup_i d_i^E$

u_{id}^D : a boolean that is 1 if and only if exam i has "duration type" d which an index for set D. We call them "types" because the duration values don't matter, only whether they are equal.

In the competition input file format, the "FRONTLOAD" entry specifies 3 parameters. The first parameter is the number of largest exams with the largest exams being specified by class size. This is used to select which exams are subject to the front load penalty, that is, the exams for which $f_e^E = 1$. To be precise, the exams should be sorted by largest-first, with a secondary sort by earliest-index-first in the case of equal sized exams. The specified number of exams are then taken from the front of this sorted list and given $f_e^E = 1$, the remaining exams (if any) are given $f_e^E = 0$. (The other two parameters are used later).

[4] `http://www.ilog.fr`

The duration type, u^D_{id}, is used for the no-mixed-duration penalty. For example, suppose all exams might have durations of either 120 or 180 minutes, then there would be two duration types, and we could have $d \in \{0, 1\}$ with $u^D_{i0} = 1$ if and only if exam i has duration 120, $u^D_{i1} = 1$ if and only if exam i has duration 180.

Students:

S : set of students

Student enrollments are encoded by:

t_{is} : 1 if student s takes (is enrolled in) exam i, 0 otherwise

Room Related:

R : set of rooms

s^R_r : size of room $r \in R$

w^R_r : a weight that specifies the penalty for using room r

Period Related:

P : set of periods

d^P_p : duration of period $p \in P$

f^P_e : a boolean that is 1 if period p is subject to the FrontLoad penalties, 0 otherwise. The second parameter of the FRONTLOAD line in the input file is used to fix this. Starting with the latest period, the required number of periods are given $f^P_p = 1$.

w^P_p : a weight that specifies the penalty for using period p

y_{pq} : a boolean that is 1 if periods p and q are in the same day, 0 otherwise.

3.2 Period-Related Hard Constraints

AFTER:

H^{aft} : a set of pairs of exams.

For every pair $(e_1, e_2) \in H^{aft}$ exam e_1 must occur strictly after exam e_2

EXAM_COINCIDENCE:

H^{coin} : a set of pairs of exams

For every pair $(e_1, e_2) \in H^{coin}$ exams e_1 and e_2 must occur in the same period (though not necessarily the same room)

EXCLUSION:

H^{excl} : a set of pairs of exams

For every pair $(e_1, e_2) \in H^{excl}$ exams e_1 and e_2 must *not* occur in the same period.

3.3 Room-Related Hard Constraints

EXCLUSIVE:

H^{sole} : a set of exams

For every exam $e \in H^{sole}$, if exam $e1$ is assigned to period p and room r then e must be

the sole occupier, i.e. no other exam can be assigned to both p and r. (Unless specified by an EXCLUSIVE rule, then, as standard in exam timetabling, exams are allowed to share rooms.)

3.4 Institutional Weights and Parameters

w^{2R} : weight for "two in a row"

w^{2D} : weight for "two in a day"

w^{PS} : weight for period spread (defaults to one as not currently specified in the input format, but included here for completeness)

w^{NMD} : weight for "No mixed duration"

w^{FL} : weight for the Front load penalty

The PERIODSPREAD line of the input format itself just specifies the parameter:

g : the period spread, the preferred minimal "gap" between exams for a student

3.5 Variables

3.5.1 Primary Decision Variables

The binary (boolean) decision variables that fix the assignment are simply

$$X^P_{ip} = \text{ 1 if exam } i \text{ is in period } p, \text{ 0 otherwise} \tag{2}$$

$$X^R_{ir} = \text{ 1 if exam } i \text{ is in room } r, \text{ 0 otherwise} \tag{3}$$

3.5.2 Secondary Variables

By secondary variables we mean those whose values will be directly forced given any legal assignment to primary variables. They are used to write the constraints and to compute the objective function.

In order to encode the room capacity constraints we will use the variable

$$X^{PR}_{ipr} = 1 \text{ if exam i is in period p and room r, 0 otherwise} \tag{4}$$

and so the intended meaning is that

$$\forall\, i \in E.\ \forall\, p \in P.\ \forall\, r \in R.\ X^{PR}_{ipr} \equiv X^P_{ip} X^R_{ir} \tag{5}$$

The penalties for violations of the various soft constraints are encoded as non-negative variables as follows

C^{2R}_s = "Two in a Row" penalty for student s

C^{2D}_s = "Two in a Day" penalty for student s

C^{PS}_s = "Period Spread" penalty for student s

C^{NMD} = "No Mixed Duration" penalty

C^{FL} = "Front-Load" penalty

C^{P} = "Soft Period" penalty

C^{R} = "Soft Room" penalty

These variables are all secondary in the sense that their values are forced by the primary decision variables and the constraints. (It follows that many need not be forced to be integer but can be relaxed to be real values if desired.)

We also remark that many of these variable are not strictly necessary, as the terms they will represent could be included directly into the objective. However, we keep them separate for the purposes of clarity, and because their values should correspond to values given by the validator on the competition website.

We also use the following variable:

$$U^{D}_{dpr} = \text{ 1 if duration type } d \text{ is used in period } p \text{ and room } r\text{, 0 otherwise} \tag{6}$$

3.6 Objective and Constraints

3.6.1 Objective

Minimise

$$\sum_{s \in S} \left(w^{2R} C^{2R}_s + w^{2D} C^{2D}_s + w^{PS} C^{PS}_s \right) + w^{NMD} C^{NMD} + w^{FL} C^{FL} + C^{P} + C^{R} \tag{7}$$

Notice that there are no separate weights for the room and period penalties C^R and C^P as the associated weights were already included in their definitions. Of course, the problem is inherently multi-objective, but this weighted sum approach is used here for simplicity.

One can also see that the objective represents a compromise between the various interested parties or stakeholders. Roughly speaking:

- Student interests for a good individual timetable are represented by $w^{2R} C^{2R}_s + w^{2D} C^{2D}_s + w^{PS} C^{PS}_s$ for each student s. (Notice, that it would also be straightforward, at least in concept, to encourage assignments that are fair between students by using the standard technique of including nonlinear terms, such as $(C^{2R}_s)^2$, to suppress penalties above the average.)
- Interests of exam invigilators (and students) are represented by C^{NMD}.
- The front load C^{FL} represents the desire of the exam markers to receive the largest exams as soon as possible so as to give more time for marking.
- The estate management has interests, represented by terms such as C^P and C^R, in avoiding the (presumably expensive) use of some rooms and periods.

We also define the penalties for the entire set of students:

$$C^{2R} = \sum_{s \in S} C^{2R}_s \tag{8}$$

$$C^{2D} = \sum_{s \in S} C^{2D}_s \tag{9}$$

$$C^{PS} = \sum_{s \in S} C^{PS}_s \tag{10}$$

Minimisation is subject to the following hard constraints defining feasibility, and also to the constraints defining the soft penalties.

3.7 Hard Constraints

Every exam is allocated to at least one room, and to at least one period:

$$\forall\, i \in E. \quad \sum_{r \in R} X^R_{ir} \quad \geq \quad 1 \tag{11}$$

$$\forall\, i \in E. \quad \sum_{p \in P} X^P_{ip} \quad \geq \quad 1 \tag{12}$$

Every exam is allocated to at most one room (exams cannot be "split"):

$$\forall\, i \in E. \quad \sum_{r \in R} X^R_{ir} \quad \leq \quad 1 \tag{13}$$

Every exam is allocated to at most one period:

$$\forall\, i \in E. \quad \sum_{p \in P} X^P_{ip} \quad \leq \quad 1 \tag{14}$$

To link the various X decision variables we enforce

$$\forall\, i \in E.\ \forall\, p \in P. \quad X^P_{ip} \quad = \quad \sum_{r \in R} X^{PR}_{ipr} \tag{15}$$

$$\forall\, i \in E.\ \forall\, r \in R. \quad X^R_{ir} \quad = \quad \sum_{p \in P} X^{PR}_{ipr} \tag{16}$$

Room capacities are always respected:

$$\forall\, p \in P.\ \forall\, r \in R. \quad \sum_{i \in E} s^E_i X^{PR}_{ipr} \quad \leq \quad s^R_r \tag{17}$$

Notice that this is the only time that the variable X^{PR}_{ipr} is used.

Period durations are respected:

$$\forall\, p \in P.\ \forall\, i \in E. \quad d^E_i X^P_{ip} \quad \leq \quad d^P_p \tag{18}$$

In any period, any student is taking at most one exam:

$$\forall\, p \in P.\ \forall\, s \in S. \quad \sum_{i \in E} t_{is} X^P_{ip} \quad \leq \quad 1 \tag{19}$$

This enforces the usual conflict matrix between exams.

The hard period constraints are enforced by:

$$\forall (i,j) \in H^{aft}\ \ \forall p, q \in P, \text{ with } p \leq q \quad X^P_{ip} + X^P_{jq} \quad \leq \quad 1 \tag{20}$$

$$\forall (i,j) \in H^{coin} \;\; \forall p \in P, \qquad X^P_{ip} = X^P_{jp} \tag{21}$$

$$\forall (i,j) \in H^{excl} \;\; \forall p \in P, \qquad X^P_{ip} + X^P_{jp} \;\leq\; 1 \tag{22}$$

The hard room constraints are enforced by

$$\forall i \in H^{sole} \;\; \forall j \in E, j \neq i \;\; \forall p \in P, \; \forall r \in R, \quad X^P_{ip} + X^R_{ir} + X^P_{jp} + X^R_{jr} \leq 3 \tag{23}$$

3.8 Soft Constraints

Finally, we cover the penalty terms in the objective corresponding to violations of the soft constraints. We will use the term "pattern penalties" for those terms arising from restrictions on sequences of enrolment for each student; that is, the two-in-a-row C^{2R}, two-in-a-day C^{2D}, and period-spread C^{PS} penalties. These pattern penalties are particularly awkward to encode and susceptible to misunderstanding. Hence, to enhance clarity we will first present them as non-linear constraints. Such non-linear constraints are of course still mathematically precise and so do meet our primary goal which is to present the new model. However, such non-linearities would present a greater challenge to integer programming solvers, so in section 4 we give alternative ways to write the pattern penalties, and in particular give one way to write them as linear terms. Note that of course such non-linearities do not affect the mathematical precision of the encoding. Also, they do not prevent the practical use of the encoding as a solution validator, because the non-linear terms are fixed to constant values when given a (complete) solution.

3.8.1 Two in a Row

If a student s is enrolled into two distinct exams i and j, and j occurs on the same day in the period immediately after the period used for i, then the penalty C^{2R}_s receives an increment of 1. Hence,

$$C^{2R}_s \;=\; \sum_{\substack{i,j \in E \\ j \neq i}} \; \sum_{\substack{p,q \in P \\ q = p+1 \;\&\; y_{pq}=1}} t_{is} t_{js} X^P_{ip} X^P_{jq} \tag{24}$$

Notice that there is no double counting. The condition on the periods is $q = p+1$ rather than $|p-q| = 1$ and so the condition $j \neq i$ is needed rather than $j > i$. That is, we separately capture the cases 'i is before j' and 'i is after j'.

This is not a particularly easy equation to interpret; see Section 4 for other ways to write it.

3.8.2 Two in a Day

If a student s is enrolled into two distinct exams i and j and these exams occur in non-consecutive periods on the same day, then the penalty C^{2R}_s receives an increment of 1. Hence,

$$C^{2D}_s \;=\; \sum_{\substack{i,j \in E \\ j \neq i}} \; \sum_{\substack{p,q \in P \\ q > p+1 \;\&\; y_{pq}=1}} t_{is}\, t_{js}\, X^P_{ip}\, X^P_{jq} \tag{25}$$

This is the same as for two-in-a-row except the $q = p+1$ condition changed to $q > p+1$.

3.8.3 Period Spread

If a student s is enrolled into two distinct exams i and j and these exams occur in distinct periods such that j is after i but is within the gap g, then the penalty C_s^{PS} receives an increment of 1. Again, double counting is prevented by putting a time order on the exams that contribute. Hence,

$$C_s^{PS} = \sum_{\substack{i,j \in E \\ j \neq i}} \sum_{\substack{p,q \in P \\ p < q \leq p+g}} t_{is} t_{js} X_{ip}^P X_{jq}^P \tag{26}$$

This is the same as for two-in-a-row or day except that the conditions on the two periods p and q changed to $q \in \{(p+1), \ldots, (p+g)\}$.

3.8.4 No Mixed Durations

We need to force U_{dpr}^D to be non-zero whenever some exam with duration type d uses period p and room r:

$$\forall d \in D.\ \forall i \in E, \text{ with } u_{id} = 1.\ \forall p \in P.\ \forall r \in R.$$
$$U_{dpr}^D \geq X_{ip}^P + X_{ir}^R - 1 \tag{27}$$

The period-room pair pr receives a non-negative penalty, C_{pr}^{NMD}, which is the maximum of zero and the excess above one of the total number of duration types assigned to it:

$$\forall\, p \in P.\ \forall\, r \in R.$$
$$1 + C_{pr}^{NMD} \geq \sum_{d \in D} U_{dpr}^D \tag{28}$$
$$C_{pr}^{NMD} \geq 0 \tag{29}$$

The overall penalty is

$$C^{NMD} = \sum_{p \in P} \sum_{r \in R} C_{pr}^{NMD} \tag{30}$$

The minimisation in the overall objective will force U_{dpr}^D and C_{pr}^{NMD} to be the intended minimal values consistent with the assignment.

3.8.5 Front Load

The front load penalty applies to exams i with $f_i^E = 1$ and assigned to periods with $f_p^P = 1$, and so

$$C^{FL} = \sum_{i \in E} \sum_{p \in P} f_i^E f_p^P X_{ip}^P \tag{31}$$

3.8.6 Soft Period Penalties

These are enforced by:

$$C^P = \sum_{p \in P} \sum_{i \in E} w_p^P X_{ip}^P \tag{32}$$

3.8.7 Soft Room Penalties

Finally, the soft room penalties are enforced by:

$$C^R = \sum_{r \in R} \sum_{i \in E} w_r^R X_{ir}^R \tag{33}$$

4 Reformulating the Pattern Penalties

The pattern penalties are important yet rather troublesome to handle. Hence, in this section, we give a different way of writing them and crucially an associated linear formulation of the constraints. We also discuss the relationship to the pattern penalties usually used with the Toronto dataset.

4.1 Reformulating the Two-in-a-Row Penalty

Note that, using (24), the contribution to the objective from the two in a row penalty, $C^{2R} = \sum_{s \in S} C_s^{2R}$, can be re-written in many different ways. For example,

$$C^{2R} = \sum_{s \in S} C_s^{2R} = \sum_{\substack{i,j \in E \\ j \neq i}} \left[\left(\sum_{s \in S} t_{is} t_{js} \right) * \left(\sum_{\substack{p,q \in P \\ q=p+1 \ \& \ y_{pq}=1}} X_{ip}^P X_{jq}^P \right) \right] \tag{34}$$

The first parenthesised sub-expression $\sum_{s \in S} t_{is} t_{js}$ can be now recognised as 'the number of students taking both i and j'. The second parenthesised expression is 1 or 0 depending on whether or not exams i and j are allocated to periods such that the "two in a row" penalty should be applied.

Alternatively,

$$C^{2R} = \sum_{\substack{i,j \in E \\ j > i}} \left[\left(\sum_{s \in S} t_{is} t_{js} \right) * \left(\sum_{\substack{p,q \in P \\ |q-p|=1 \ \& \ y_{pq}=1}} X_{ip}^P X_{jq}^P \right) \right] \tag{35}$$

which has the clearest connection to descriptions of the penalty in terms of pairs of adjacent same-day exams weighted by the number of students taking both.

One could of course directly linearise such terms by directly introducing a new variable for non-linear terms. For example, we could simply introduce a new variable, Y with intended meaning that $Y_{ipjq} \equiv X_{ip}^P X_{jq}^P$. This is a standard procedure (see for example [39]), but generally has the severe disadvantage of introducing too many new variables.

To give one conversion to linear form that introduces fewer new variables, firstly observe that such a term as $X_{ip}^P X_{jq}^P$ is only relevant when the two exams i and j have students in common. That is, we can restrict attention to pairs of exams that conflict with each other. This suggests using edges of the associated (student-generated) conflict graph. That is, define the conflict graph $G_C = (E, A)$ with vertices being the set E of exams. The set A of undirected edges, contains an edge (i, j) if and only if exams i and j conflict. Without loss of generality, and to prevent double counting, we assume $i < j$ for all $a = (i, j) \in A$.

Define a weight, w_a^C, for each edge in A, to be the number of students shared between the two exams:

$$\forall\, a = (i, j) \in A. \quad w_a^C = \sum_{s \in S} t_{is} t_{js} \tag{36}$$

At this point, we could linearise by introducing new variables $Y_{apq} \equiv X^P_{ip} X^P_{jq}$ for each edge a in the conflict graph. However, this would still give an unnecessarily large encoding: Since each conflict edge can only contribute once to the penalty, it suffices to only introduce a new variable to encode the total penalty at that edge. Specifically, we introduce $|A|$ new variables

$$C^{2R}_a = \begin{cases} 1 & \text{if edge } a = (i,j) \text{ for exams } i \text{ and } j \text{ incurs a two-in-a-row penalty} \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

Then the overall penalty, as from equation (35), is the sum of such occurrences weighted by the number of students (encoded as the edge weight)

$$C^{2R} = \sum_{a \in A} w^C_a C^{2R}_a \tag{38}$$

Notice there is no double counting because the conflict edges are unordered, and so written as a set of two exams; each pair occurs only once in the sum.

The minimisation of C^{2R} within the overall objective will automatically force C^{2R}_a towards zero. Hence, it only remains to force it to be one when necessary. This can be achieved using the following *linear* constraints:

$$\begin{aligned} \forall\, a = (i,j) \in A.\ \ \forall\, p, q \in P.\ \ |q-p| = 1\ \ \&\ \ y_{pq} = 1 \\ X^P_{ip} + X^P_{jq} \leq 1 + C^{2R}_a \end{aligned} \tag{39}$$

The non-linear version in equations (24) can now be replaced with the above linear version above.

(Note that there is no claim here that this is the 'best' linear encoding. We have used it to solve some small problems, but we strongly encourage work towards better formulations.)

The two-in-a-day and period-spread penalties can also be re-written in the following similar fashions.

4.2 Reformulation of the Two-in-a-Day Penalty

Introduce new variables

$$C^{2D}_a = \begin{cases} 1 & \text{if edge } a = (i,j) \text{ for exams } i \text{ and } j \text{ incurs a two-in-a-day penalty} \\ 0 & \text{otherwise} \end{cases} \tag{40}$$

and constraints

$$C^{2D} = \sum_{a \in A} w^C_a C^{2D}_a \tag{41}$$

$$\begin{aligned} \forall\, a = (i,j) \in A.\ \ \forall\, p, q \in P.\ \ |q-p| \neq 1\ \ \&\ \ y_{pq} = 1 \\ X^P_{ip} + X^P_{jq} \leq 1 + C^{2D}_a \end{aligned} \tag{42}$$

This linear version can be used to replace non-linear constraints (25).

4.3 Reformulating the Period-Spread Penalty

We introduce the new variable

$$C_a^{PS} = \begin{cases} 1 & \text{if edge } a=(i,j) \text{ for exams } i \text{ and } j \text{ incurs a period spread penalty} \\ 0 & \text{otherwise} \end{cases} \tag{43}$$

and the constraints:

$$C^{PS} = \sum_{a \in A} w_a^C C_a^{PS} \tag{44}$$

$$\forall\, a=(i,j) \in A.\quad \forall\, p,q \in P.\quad (p < q \leq p+g) \text{ or } (q < p \leq q+g)$$
$$X_{ip}^P + X_{jq}^P \leq 1 + C_a^{PS} \tag{45}$$

This linear version can be used to replace non-linear constraints (26).

Remark about the reformulation: We had to introduce $|A|$ new variables for each pattern, giving a total of $3|A|$ variables. Also the variable set X_{ipr}^{PR} has $|E||P||R|$ booleans. Together these tend to dominate the numbers of variables. As might be expected current exact solvers cannot handle the size of problem that results from the competition dataset. However, the formulation here has been used on smaller instances, and of course does provide the desired mathematically precise definition of our new examination timetabling model.

4.4 Generalising the Period Spread Penalty

We could also define a "Gap-k" penalty C_{sk}^{Gk} associated with a specific gap

$$C_{sk}^{Gk} \quad = \quad \sum_{\substack{i,j \in E \\ j \neq i}} \sum_{\substack{p,q \in P \\ q=p+k}} t_{is} t_{js} X_{ip}^P X_{jq}^P \tag{46}$$

With this definition, the previous spread penalty is simply

$$C_s^{PS} = \sum_{k \in K} C_{sk}^{Gk} \tag{47}$$

with $K = \{1, \ldots, g\}$.

This is of interest because it is then straightforward to generalise this to a sum with a weight, w_k^{Gk}, dependent upon k:

$$C_s^{PS} = \sum_{k \in K} w_k^{Gk} C_{sk}^{Gk} \tag{48}$$

The choice

$$w_k^{Gk} = 2^{g-k} \text{ for } k \in K \text{ and } 0 \text{ otherwise} \tag{49}$$

then reproduces the spread penalties, or proximity costs [7] commonly used in the literature with the Toronto data. (Though it is worth mentioning that such penalties were

not part of the data itself: essentially, the Toronto data only contains the enrollments, and so is equivalent to just t_{is} for each instance.)

Obviously, many other weighting choices could easily be incorporated into the formulation. The choices made for the competition were made because they correspond to choices that have been successful in practical experience.

5 Conclusion and Discussion

We have presented a formal model of the examination timetabling problem as it appears in many institutions. As used in the the examination track of the 2nd International Timetabling Competition, the formulation introduces many aspects of practical real world implementations. This work therefore serves to provide a firm basis for setting the research agenda for continual development in the field. In doing so the authors believe the formulation significantly addresses the gap which exists between the efforts made in research and the requirements of institutions.

In providing this formulation, it is pointed out that minimising the number of periods is not considered to be an objective, because in the authors' experience, educational institutions manage the process by using set times for the examination session. That is not to say, of course, that this is not a major issue in relation to planning examination sessions. Also, we have delayed until future work, the full investigation and explanation of which constraints are absolute, and which can be broken to give a "*distance to feasibility*". Such an investigation will allow the formulation provided here to cover not just the typical institutions and instances but also less common instances of the problem. For example, when feasibility is not possible for a given problem, institutions make decisions relating to either the structure of the time periods, the availability of the resources or indeed the events to be timetabled. It is envisaged that issues relating to this issue of "repairing infeasibility" will also be analysed in future work.

We emphasize that definition of these soft constraints is based on real-world experience, and their introduction to the academic community, provides the core contribution of this paper.

In relation to solution measurement, although a 'weighted sum' evaluation function is not ideal e.g. it may have adverse side effects for certain individual students. It is the chosen method as part of this formulation due to ease of implementation and for comparison purposes. As part of ITC2007, we specifically decided to include the weights in the data format itself rather than solvers having to hard code them. The purpose of this was to allow variations of the weights to enable exploration of the multi-objective properties.

References

[1] Metaheuristics Network, "International timetable competition 2002," 2003. http://www.idsia.ch/Files/ttcomp2002/ Organised by the Metaheuristics network, http://www.metaheuristics.net/ and PATAT 2002 http://www.asap.cs.nott.ac.uk/patat/patat02/patat02.shtml, accessed April 2008.

[2] B. McCollum, "A perspective on bridging the gap between theory and practice in university timetabling," in *Revised Selected Papers of PATAT 2006, Proceedings of*

the 6th International Conference on the Practice and Theory of Automated Timetabling, vol. 3867 of *Lecture Notes in Computer Science (LNCS)*, pp. 3–23, 2007.

[3] B. McCollum, P. McMullan, B. Paechter, R. Lewis, A. Schaerf, L. D. Gaspero, A. J. Parkes, R. Qu, and E. K. Burke, "Setting the research agenda in automated timetabling: The second international timetabling competition." Submitted to INFORMS Journal on Computing, April 2007, 2008.

[4] B. McCollum, P. McMullan, E. K. Burke, A. J. Parkes, and R. Qu, "The second international timetabling competition: Examination timetabling track," Tech. Rep. QUB/IEEE/Tech/ITC2007/Exam/v4.0/17.2007, Queen's University Belfast, sep 2007. http://www.cs.qub.ac.uk/itc2007/.

[5] R. Lewis, B. Paechter, and B. McCollum, "Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition," Cardiff Working Papers in Accounting and Finance A2007-3, Cardiff Business School, Cardiff University, August 2007.

[6] L. D. Gaspero, B. McCollum, and A. Schaerf, "The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (track 3)," Tech. Rep. QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1 August 2007, Queen's University Belfast, aug 2007. http://www.cs.qub.ac.uk/itc2007/.

[7] M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: Algorithmic strategies and applications," *Journal of Operational Research Society*, vol. 47, no. 3, pp. 373–383, 1996.

[8] R. Qu, E. K. Burke, B. McCollum, L. Merlot, and S. Lee, "A survey of search methodologies and automated system development for examination timetabling," *Journal of Scheduling*, 2008. to appear.

[9] E. Tsang, P. Mills, and R. Williams, "A computer aided constraint programming system.," in *The 1st International Conference on the Practical Application of Constraint Technologies and Logic Programming (PACLP)*, pp. 81–93, 1999.

[10] L. D. Gaspero and A. Schaerf, "Tabu search techniques for examination timetabling," in *Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference.* (E. K. Burke and W. Erben, eds.), vol. 2079 of *Lecture Notes in Computer Science (LNCS)*, pp. 104–117, 2001.

[11] D. de Werra, A. S. Asratian, and S. Durand, "Complexity of some special types of timetabling problems," *Journal of Scheduling*, vol. 5, pp. 171–183, 2002.

[12] S. M. Al-Yakoob, H. D. Sherali, and M. Al-Jazzaf, "A mixed-integer mathematical modeling approach to exam timetabling," *Computational Management Science*, 2007. Published online http://dx.doi.org/10.1007/s10287-007-0066-8. Awaiting volume assignment.

[13] L. D. Gaspero, "Recolour, shake and kick: A recipe for the examination timetabling problem," in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling.* (E. K. Burke and P. D. Causmaecker, eds.), (KaHo St.-Lieven, Gent, Belgium), pp. 404–407, 2002.

[14] G. M. White and B. S. Xie, "Examination timetables and tabu search with longer-term memory," in *Practice and Theory of Automated Timetabling: Selected Papers from the 3rd International Conference.* (E. K. Burke and W. Erben, eds.), 2001.

[15] L. Paquete and T. Stützle, "Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem," in *Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling.* (E. Burke and P. D. Causmaecker, eds.), 2002.

[16] E. K. Burke, Y. Bykov, J. Newall, and S. Petrovic, "A time-predefined local search approach to exam timetabling problems," *IIE Transactions*, vol. 36, pp. 509–528, Jun 2004.

[17] L. T. G. Merlot, N. Boland, B. D. Hughes, and P. J. Stuckey, "A hybrid algorithm for the examination timetabling problem," in *Practice and Theory of Automated Timetabling: Selected Papers from the 4th International Conference.*, vol. 2740 of *Springer Lecture Notes in Computer Science*, pp. 207–231, 2003.

[18] J. Thompson and K. Dowsland, "A robust simulated annealing based examination timetabling system," *Computers & Operations Research*, vol. 25, pp. 637–648, 1998.

[19] P. Ross, D. Corne, and H. Terashima-Marn, "The phase transition niche for evolutionary algorithms in timetabling," in *Selected papers from the First International Conference on the Theory and Practice of Automated Timetabling (PATAT 95)* (E. K. Burke and M. A. Trick, eds.), vol. 1153, pp. 309–324, Lecture Notes in Computer Science, Springer-Verlag, NY, 1996.

[20] T. Wong, P. Cote, and P. Gely, "Final exam timetabling: A practical approach," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 2002).*, vol. 2, pp. 726– 731, 2002.

[21] E. K. Burke, J. Newall, and R. F. Weare, "A memetic algorithm for university exam timetabling," in *The Practice and Theory of Automated Timetabling* (E. K. Burke and P. Ross, eds.), vol. 1153 of *Lecture Notes in Computer Science*, pp. 241–250, Springer, 1996.

[22] K. A. Dowsland and J. Thompson., "Ant colony optimization for the examination scheduling problem," *Journal of the Operational Research Society*, vol. 56, no. 4, pp. 426–438, 2005.

[23] Z. N. Azimi, "Hybrid heuristics for examination timetabling problem," *Applied Mathematics and Computation*, vol. 163, no. 2, pp. 705–733, 2005.

[24] C. L. Mumford, "An order based evolutionary approach to dual objective examination timetabling," in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched 2007), Honolulu, Hawaii 1-5th April.*, 2007.

[25] E. K. Burke and J. Newall, "Solving examination timetabling problems through adaptation of heuristic orderings," *Annals of operations Research*, vol. 129, pp. 107–134, 2004.

[26] E. K. Burke, M. Dror, S. Petrovic, and R. Qu, "Hybrid graph heuristics within a hyper-heuristic approach to exam timetabling problems," in *The Next Wave in Computing, Optimization, and Decision Technologies* (B. L. Golden, S. Raghavan, and E. A. Wasil, eds.), pp. 79–91, Springer, 2005.

[27] E. K. Burke, S. Petrovic, and R. Qu, "Case based heuristic selection for timetabling problems," *Journal of Scheduling*, vol. 9, no. 2, pp. 115–132, 2006.

[28] E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu, "A graph-based hyper-heuristic for educational timetabling problems," *European Journal of Operational Research*, vol. 176, pp. 177–192, 2007.

[29] P. David., "A constraint-based approach for examination timetabling using local repair techniques," in *Practice and Theory of Automated Timetabling: Selected Papers from the 2nd International Conference.* (E. K. Burke and M. W. Carter, eds.), vol. 1408 of *Springer Lecture Notes in Computer Science*, pp. 169–186, 1998.

[30] T. A. Duong and K. H. Lam, "Combining constraint programming and simulated annealing on university exam timetabling," in *Proceedings of the 2nd International Conference in Computer Sciences, Research, Innovation & Vision for the Future (RIVF2004)*, (Hanoi, Vietnam), pp. 205–210, February 2004.

[31] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu., "Hybrid variable neighbourhood approaches to university exam timetabling," Tech. Rep. NOTTCS-TR- 2006-2, School of CSiT, University of Nottingham, 2006.

[32] S. Ahmadi, R. Barone, P. Cheng, P. Cowling, and B. McCollum., "Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem.," in *Proceedings of Multidisciplinary International Scheduling: Theory and Applications (MISTA 2003)*, (Nottingham, August 13-16,), pp. 155–171, 2003. ISBN: 0-9545821-2-8.

[33] E. K. Burke, E. Hart, G. Kendall, J. Newall, P. Ross, and S. Schulenburg, "Hyper-heuristics: An emerging direction in modern search technology," in *Handbook of Meta-Heuristics* (F. Glover and G. Kochenberger, eds.), pp. 457–474, Kluwer, 2003.

[34] P. Ross, J. Marin-Blazquez, and E. Hart, "Hyper-heuristics applied to class and exam timetabling problems," in *Proceedings of the 2004 Congress on Evolutionary Computation (CEC2004)*, pp. 1691–1698, 2004.

[35] E. K. Burke, S. Petrovic, and R. Qu, "Case-based heuristic selection for examination timetabling," in *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL 2002)*, (Singapore), pp. 277–281, Nov 18-22 2002.

[36] H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, "A novel fuzzy approach to evaluate the quality of examination timetabling," in *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2006)*, (Brno, Czech Republic), pp. 82–102, August 30th-September 1st 2006.

[37] A. Borning, R. Duisberg, B. Freeman-Benson, A. Kramer, and M. Woolf, "Constraint hierarchies," in *Proceedings of the Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA)* (N. Meyrowitz, ed.), vol. 22, (New York, NY), pp. 48–60, ACM Press, 1987.

[38] A. Borning, B. Freeman-Benson, and M. Wilson, "Constraint hierarchies," *Lisp and Symbolic Computation*, vol. 5, pp. 223–270, Sep 1992.

[39] P. Hansen, B. Jaumard, and V. Mathon, "Constrained nonlinear 0-1 programming," *ORSA Journal on Computing*, vol. 5, no. 2, pp. 97–119, 1993.