Setting the Research Agenda in Automated Timetabling: The Second International Timetabling Competition

Barry McCollum[1], Andrea Schaerf[2], Ben Paechter[3],
Paul McMullan[1], Rhyd Lewis[4],
Andrew J. Parkes[5], Luca Di Gaspero[2],
Rong Qu[5], Edmund K. Burke[5]

[1] School of Electronics, Electrical Engineering and Computer Science, Queen's University, Belfast, BT7 1NN, UK. { b.mccollum , p.mcmullan }@qub.ac.uk

[2] Department of Electrical, Management and Mechanical Engineering, University of Udine, 33100 Udine, Italy. { schaerf, l.digaspero }@uniud.it

[3] Centre for Emergent Computing, Napier University, Edinburgh, EH10 5DT, UK. b.paechter@napier.ac.uk

[4] Cardiff Business School, Cardiff University, Cardiff, CF10 3EU, UK. LewisR9@cardiff.ac.uk

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

29 August 2008

Abstract

The 2nd International Timetabling Competition (ITC2007) was opened in August 2007. Building on the success of the first competition in 2002, this sequel aimed to further develop research activity in the area of educational timetabling. The broad aim of the competition was to create better understanding between researchers and practitioners by allowing emerging techniques to be developed and tested on real world

models of timetabling problems. To support this, a primary goal was to provide researchers with models of problems faced by practitioners by incorporating a significant number of real world constraints. Another objective of the competition was to stimulate debate within the widening timetabling research community. The competition was also divided into three tracks to reflect the important variations which exist within educational timetabling within Higher Education. As these formulations incorporate an increased number of 'real world' issues, it is anticipated that the competition will now set the research agenda within the field. After finishing in January 2008, final results of the competition were made available in May 2008. Along with background to the competition, the competition tracks are described here together with a brief overview of the techniques used by the competition winners.

1 Introduction

Timetabling within a university context has long been recognised as difficult from both a theoretical and practical perspective; for a survey see Schaerf (1999), Lewis (2008). Whether it be for courses or examinations, much effort in the real-world is spent producing timetables which are both workable and of a sufficiently high quality. In response to these practical and theoretical demands, in 2002, the European Metaheuristic Network organized the First International Timetabling Competition (ITC 2002). Based on a specific problem model, the formulation presented contained characteristics of the course timetabling problem found in many Universities. Details on applied techniques and results can be found at the webpage (ITC 2002). More recently, this formulation has become somewhat of a standard within the research area with many researchers using it together with the associated generated datasets, for example, see: Abdullah et al. (2007); Chiarandini et al (2006); Di Gaspero and Schaerf (2006); Kostuch (2005); and Lewis et al. (2007b). ITC2002 was therefore successful in generating common ground for cross-fertilisation of ideas for research groups within the timetabling community.

The Second International Timetabling Competition (ITC 2007) followed the main goals of the first and further aimed to provide a basis on which research in timetabling can progress. Detailed information can be found at the competition website (ITC 2007). An important aim of this competition was the generation of new approaches for the timetabling problems it introduced by attracting users from all areas of research. An additional important aim was to narrow the gap which exists between research and practice within this important area (McCollum 2007a). To this end, the competition introduced various formulations of the timetabling problems encountered within educational institutions and based them on a 'real world' perspective as far as possible within the practical requirements of a competition. In providing these formulations, we considered it important to balance the inclusion of all known aspects of real world problems with the required competitive element. Importantly, we built on the success of the first international timetabling competition by introducing significantly more depth and complexity in not one but three distinct variations of the timetabling problem, called *tracks.* Competitors were encouraged to enter one, two, or all of the tracks.

Although sufficient overlap exists between tracks for it to make sense to put them all into a single competition, they still clearly represent distinct problems within the area of educational timetabling both from a research and practical perspective. From a research perspective, it was felt by the organisers that this division was important because it provided a framework for capturing the main types of educational timetabling research currently taking place within the academic community. From a practical perspective, the tracks also provided more details of the models experienced in real world situations. Although, some hard and soft constraints do occur in more than one track, no specific effort was made to enforce this; it was considered more important to allow tracks to be representative of their area, and let any commonalties emerge naturally.

The three tracks considered in the competition cover the main formulations of both examination timetabling (ETT) and course timetabling (CTT) problems. As for the course timetabling problem, this area was subdivided into two tracks. Both of these CTT tracks are distinct and represent methods of course timetable construction which are used in various forms within many institutions, namely post enrolment based course timetabling and curriculum based course timetabling.

Technical reports (McCollum et al (2007b), Lewis et al (2007), Di Gaspero et al. (2007)) are available on the competition web site for each track detailing the formulation offered. In this work, following a description of the main aspects of the competition, an overview of each track is provided along with results on the datasets released as part of the competition.

An observant reader might well remark that many of the individual aspects of the formulations have already appeared at some point in the literature, however, such occurrences in the literature are rarely associated with publicly available data or a comparison of many algorithms. Unfortunately, the timetabling literature is littered with papers that present results of one unpublished implementation on one unpublished set of instances. Hence, this paper and competition offer a unique combination of realistic public instances, and a comparison of many algorithms, with the likelihood that many more will be developed.

2 Competition Rules

The competition was officially started on August 1st, 2007. On this date, sets of benchmark problem, the "early" instances, for each track were released to the public. Entrants were then invited to design algorithms, and associated implementations that produced solutions to these problems according to the competition criteria. On the 11th of January 2008, two weeks before the end of the competition, a second set of "late" instances was also released for each track. Competitors were then required to submit their best solutions for these instances to the organisers by the 25th of January. Further sets of "hidden" instances were also used by the organisers for evaluating entries.

In order to ensure a degree of fairness and consistency in the competition, a number of rules were also imposed, which we now outline. Refer to the official website (ITC 2007) for a full listing of these rules. When implementing their algorithms, competitors were free to use the programming language and operating system of their choosing; however, all implementations were required to run on single processor machines and adhere to a strict run-time limit, determined for different machines via the use of a benchmarking program (see Section 3). The submitted algorithms were obliged to produce solutions to the given problems such that a number of hard constraints were satisfied (i.e. a feasible timetable was achieved), whilst also minimising a weighted sum of a function reflecting the number of soft constraint violations.

In cases where feasibility could not be reached, infeasible solutions were evaluated using a "Distance to Feasibility" measure (see Section 4).

An important characteristic of the competitors' algorithms was that they were not permitted to "recognise" which problem instance they were trying to solve in order to alter their run characteristics. The same version of the algorithm therefore had to be used for all instances. It was, however, perfectly acceptable for an algorithm to analyse the features of a given problem instance in order to make choices about how it would run. Both stochastic and deterministic algorithms were permissible, though in both cases participants needed to ensure that any claimed results were repeatable in the given run-time limit. That is, if the solvers were allowed to use pseudo-random numbers they then needed to be able to take a seed for the generator, and so guarantee reproducibility.

When the deadline of the competition was reached, competitors were required to submit their solutions (timetables) for each of the released instances, together with the random seeds used to generate these (if applicable). A document containing a description of their algorithm was also required. For each track, a set of five finalists was then chosen according to the quality of the submitted solutions using a ranking procedure (explained in Section 5). The finalists' programs were then tested by the competition organisers on their own machines using the publicly available instances together with the third set of hidden instances. The results of these trials were used to determine the official winner in each of the tracks.

One of the features of ITC2002 was that winners were chosen based on the quality of the solution provided. This meant that the competition was open to the criticism that participants could take advantage of the "Mongolian Horde" approach (Schaerf and Di Gaspero 2007): "Run as many trials as you can and report only the best of all of them" – although in practice the checks on unseen instances showed that this was not a problem. In ITC2007, the re-running of finalist solvers on organisers' machine (with new seeds) and the use of hidden instances in the rankings were introduced to improve the situation. Although not used for the place-list, some of the organisers plan to use principled statistical tools to analyse in more detail the performance of the solvers, especially for the stochastic ones.

3 CPU Time Limits and Benchmarking of the Machines

As with the first competition, the winner of each track was chosen based on the quality of the solutions produced by the proposed technique within a specific, pre-imposed time limit, measured in elapsed time (see Section 5 for further detail on winner designation). Note that although conforming to a time limit might not always be an important constraint in real world timetabling; using one did allow us to introduce a competitive element to the competition (which, it was hoped, would help to attract more researchers to enter).

In order to allocate a time limit to each of the competitors, a benchmarking program was created and distributed. This program was only suitable for individual, single processor machines, not for specialist parallel machines or clusters. Competitors would execute this on their own machines, whereupon the program would set about performing a number of computational operations of the sort involved in timetabling. When the program halted, the program then considered how long it had taken to execute, and used this figure in order to calculate an appropriate time limit for the machine at hand. Obviously, the speed of the benchmark program (and resultant time limit) on an individual machine depends on a number of factors including the memory, the type of processor, the clock speed, and the operating system. Note that in providing this benchmark program, it was not possible to provide perfectly equitable benchmarks across the various platforms, types of processor, and so on, and we acknowledge that the benchmark may well have been "kinder" to some people than others. However, one way that we attempted to counter this potential discrepancy, was by running all of the eventual finalists' algorithms on our own "benchmark machine" therefore creating more of a level playing field in the final stages of the competition.

The reason why it was decided to have a fixed running time was mainly to remove one degree of variability from the scoring system. We anticipate that future competitions will take into account in some principled way the trade-off between solution quality and running times. For the selection of the fixed amount of running time, the key question is concerned with establishing a realistically feasible running time for the actual timetabling. Given that the timetabling from both a course and exam perspective is performed usually a few times a year, one might think that a running time much longer than the 5-10 minutes granted for the competition would also be reasonable. In practical cases however, as many researchers and practitio-

ners have pointed out, the solution of a real case is an interactive process, during which it is necessary to solve a large number of instances. In fact, constraints and objectives are manually adjusted between runs of a working session for one single case (for various reasons: what-if scenarios, last minute changes, etc.). As a rule of thumb, based on experience, a running time longer than a few minutes makes the process very tiresome and difficult for the human operator.

4 Evaluation of Solutions

For all three tracks of the competition, solution quality was measured using two separate values: (1) the "Distance to Feasibility", DTF, and (2) the "Soft Cost", which indicated the level at which the hard and soft constraints, respectively, were adhered to. See McCollum et al (2007b), Lewis et al (2007), and Di Gaspero et al. (2007)) for details of how these values were calculated in each track. Of course, the "Distance to Feasibility" is not a distance in the sense of distance between solutions. Rather, more formally, the DTF is a score for violations within the higher level of constraints in a multi-level hierarchy of constraints; see Borning et al (1987) and McCollum et al (2007b), though we do not pursue such a view here.

As usual, a solution is said to be feasible if all the hard constraints are satisfied. Often, in the real world, cases arise where solutions are judged as 'good' even though some "so-called" hard constraints are violated. Indeed, there are many anecdotes of practitioners initially claiming a constraint is hard, only for the researcher to discover that it is sometimes, or even routinely, violated in real solutions. In the previous competition, ITC2002, in order to avoid the problems of measuring degrees of infeasibility and comparing these with number of broken soft constraints, it was decided that all hard constraints had to be respected, and so only feasible solutions were accepted. Consequently, all problem instances were constructed so that feasibility was not overly difficult to obtain. In this competition we wanted to create harder instances, with the consequent risk that some or even many submissions would fail to find feasible solutions. We therefore introduced the DTF measure to allow some hard constraints to be violated if no solution could be found otherwise, but still give the resulting solution some value.

Details of how the DTF and soft costs are calculated depend on the track. The definitions of the problems in each track arose independently from the requirements of the system under consideration; however, some broad properties are shared by all three.

Hard constraints generally include:

1. Need to allocate events to a single timeslot and a single room
2. There is a conflict matrix specifying which events cannot be allocated to the same timeslots. That is, all the problems contain an underlying Graph Colouring Problem and so belong to a class of NP hard problems.
3. There can be room related hard constraints that limit room availability
4. Events have sizes, and rooms have capacities and events should be allocated to a single room and it should be large enough. (In one track this is relaxed to be a soft constraint.)

Soft Constraints are quite varied but might loosely be classified into:

1. Period-related Pattern Penalties: The timeslots can be softly constrained to avoid sequences that would be inconvenient for the participants. This is probably the area in which the ETT and CTT differ most greatly. In ETT we attempt to spread out exams, but in CTT we tend to do the opposite and for example want to avoid events that are isolated in time. The tracks also differ in whether violations of such penalties are counted “per event” or “per affected student”.
2. Room-related Penalties: Some patterns of room usage might be penalised. For example, some room might only be used at a particular time if necessary, subject to a penalty.

The details of these penalties depend on the individual track e.g. some constraints might even be hard in one track and soft in another. However, there are sufficient characteristics in common for competitors to be compelled into entering all three tracks with similar/identical versions of their techniques.

5 Designation of the Winners

The adjudication process of the competition was divided into two phases. First, in each track five "finalists" were selected based on the results that were provided by the competitors. Second, the algorithms of these five finalists were then directly compared against one another by the competition organisers on their own machines, using the publicly available instances, together with a hidden set of problem instances.

Candidate solutions were then compared in the following way. First, the "Distance to Feasibility" was considered, and the solution that was seen to have the lowest value for this was considered superior. However, if two or more solutions were equal in this respect, then the best solution was judged to be the one among these that has the lowest soft cost. This method of using a pair of values means that solution quality is a type of ordinal data, meaning that we are able to rank solutions, but we cannot calculate distances between solutions (except, of course, in cases where solutions have an equal DTF).

	Results achieved instances #1 to #5. ("Distance to feasibility" and soft cost (in parenthesis))					*Rank on each instance*					*Av*	*Position*
Algorithm	#1	#2	#3	#4	#5	#1	#2	#3	#4	#5		
A	5 , 3	0 , 0	20 , 34	0 , 0	0 , 0	2	2	3	2.5	2	2.3	2nd
B	0 , 10	0 , 0	0 , 49	0 , 0	0 , 0	1	2	1	2.5	2	1.7	1st
C	5 , 1023	0 , 89	0 , 10	0 , 0	0 , 34	3	4	2	2.5	4	3.1	4th
D	10 , 102	0 , 0	35 , 200	0 , 0	0 , 0	4	2	4	2.5	2	2.9	3rd

Table 1. Example demonstration of the process used to rank the algorithms.

Given the above, in order to choose the finalists for the competition, a simple ranking process was used which judged how well the algorithms performed *in relation to one another* on the

publicly available instances. For this description, let n represent the number of problem instances, and m the number of algorithms being compared. An example for $n = 5$ and $m = 4$ using some arbitrary data is shown in Table 1. First, the results achieved by each algorithm on each of the instances are recorded and verified. For each instance, the algorithms are then assigned rank, one through to m, indicating the position of the algorithms solution for this instance compared to the others. In cases of ties, the average of the corresponding ranks are assigned to each of the algorithms. To determine the final positions of the algorithms, the average (arithmetic mean) of the rank is then taken across all instances. Note that the minimal average rank is thus 1.0, and the maximum is m. For the competition, the finalists were the algorithms with the five lowest rank averages.

Of course, this procedure is a matter of "instances" acting as voters on a selection of algorithms acting as candidates in an election, and so potentially is subject to the standard and unavoidable voting paradoxes. For example, in a bad case, it is possible that removing the algorithm placed 5th could change the order of the others. Such potential anomalies are well-known to be inevitable, and competition organisers have no choice but to pick one system and then stick to it. However, it turned out that the results were relatively clear and so we do not believe that such paradoxes played a role in selecting the winner.

In each track final, the winners of the competition were also determined using this ranking process. In this case, however, results were generated by performing a random sample of ten runs with each algorithm on all available instances, including the hidden set. For each problem instance, fifty solutions were thus generated (ten for each of the five algorithms) which were then ranked from one through to fifty (again, mean ranks were assigned for ties). The 10 scores of each algorithm are then averaged. The best rank-average obtainable is thus 5.5 (if the algorithm has all the best ranks 1, …, 10), the worst was 45.5 (if the algorithm has all the worst 41, …, 50).

6 Competition Tracks

In this section a description of the formulation is provided for each track along with information on the chosen track finalists. Furthermore, commentary is provided on issues specific to individual tracks that proved important to the running of the competition. As the tracks relat-

ing to the examination timetabling and curriculum based course timetabling tracks are based on real world data, information is provided on further criteria which are considered during implementation. As discussed in the introduction, the authors felt is essential to balance the complexity of the formulation provided and the competitive element required to meet the competitions overall objectives

6.1 Track 1: The Examination Timetabling Problem

From a practical perspective, much work is required in establishing a generic examination timetable model which is applicable across a wide range of scenarios. The problem formulation proposed as part of ITC2007 significantly adds to current models used within research and provides a basis for further real world constraints to be described. The problem model can be described as 'post enrolment'. That is to say, students enrolled on courses which have associated exams are considered to be enrolled on or 'taking' those exams. Although other approaches to the problem are taken within some 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. Recent research has concentrated on a number of benchmark datasets introduced by Carter et al. (1996). These benchmarks and the problems associated with them are discussed in more detail elsewhere (Qu et al. 2007). This particular track of the competition significantly adds to the research field by the introduction of a more 'real' model of the problem in terms of data, constraints and evaluation. All datasets used as part of this competition are taken from real institutions and have been anonymised for the purpose of competition use.

The fundamental problem involves timetabling exams into a number of timeslots within a defined examination session while satisfying a number of hard constraints. The quality of the solution is measured in terms of soft constraints satisfaction. Importantly, with respect to previously studied models, new and additional information is provided on constraints (hard and soft), resources and the examination session.

From experience, the authors have found that, in general, gaining feasibility within examination timetabling is not as important an issue as with some cases of course timetabling. How to implement a timetable solution when feasibility cannot be found is usually decided by the institution. Actions to be taken include, extending the session, introducing another room, allow-

ing more capacity within particular rooms, holding students over the lunch break etc. ITC2007 does not deal with these institutional idiosyncrasies and therefore researchers were expected to gain feasibility. All instances for this track were real problems encountered during commercial work and selected on the basis that they had feasible solutions and were representative of the area. They were also selected to be representative in terms of their sizes and to give a variety of them so as to provide some easier and some harder instances for the competition.

An examination session is made of a number of periods over a specified length of time, i.e., examination session. Period lengths within which a set of examinations can be allocated are provided. A set of students are enrolled on individual examinations, where each individual student may be enrolled on a number of exams. A set of rooms with individual capacities are provided. Details including a 'weighting' of particular soft constraints are provided within an Institutional Model Index. A feasible timetable is one in which all examinations have been assigned to a period and room so that the following hard constraints are satisfied:

Exams. No student sits more than one examination at the same time.

Rooms. The capacity of individual rooms is not exceeded at any time throughout the examination session; In addition, room related hard constraints were imposed e.g. *Exam_A* must use *Room 101*.

Periods. Period lengths are not violated. Similar to Rooms, period related hard constraints e.g. *Exam_A* after *Exam_B*.

A candidate timetable is penalised for each occurrence of the following soft constraints:

Period Spread. These include two exams in a row, two exams in a day and specified spread of examinations over the entire examination session.

Exam Duration. The Mixing of duration of examinations within individual periods within a single room;

Exam Position. Larger examinations appearing later in the timetable. It is common practice to attempt to place these earlier in the session to maximise the time available for marking.

Period Priorities. Period related soft constraints i.e. avoid particular periods at certain times.

Room Priorities. Room related soft constraints. This allows targets to be met in relation to room utilisations during the examination session

These constraints can effectively be split into two groups; those which are resource specific and those which can have a global setting. Resource specific constraints can be set for each period and each room (Period and Room Priorities) . This allows control of how resources would be used when constructing a solution. The remaining 'Global Setting' constraints can be set relative to each other (i.e. constraints a-e). and g). Institutions may weight these soft constraints differently relative to one another in an attempt to produce a solution which is appropriate for their particular needs. This is known as building the 'Institutional Model' and 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. Within the datasets provided a number of variables are given with values.

Table 2 provided the best results obtained using the submitted competitor's techniques. All results reported were independently achieved by the organisers strictly under the competition rules. The result in each case is composed of a pair "DTF , Soft Cost". That is, the first number is 0 for feasible and 1 for unfeasible. If feasible, the overall soft constraint violation score is subsequently provided. In cases where infeasible solutions were obtained, a score of 0 was recorded for the soft constraint violation. This differs to the other two tracks as it was originally intended that competitors should be able to find feasibility. In the end this was not the case. On realising this, the organisers felt it would have been wrong to change the original conditions as stated on the competition web site.

Algorithm: Instance #	*1. Müller*	*2. Gogos*	*3. Atsuta et al*	*4. De Smet*	*5. Pillay*
1	**0 , 4730**	0 , 5905	0 , 8006	0 , 6670	0 , 12035
2	**0 , 400**	0 , 1008	0 , 3470	0 , 623	0 , 3074
3	**0 , 10049**	0 , 13862	0 , 18622	1 , 0	0 , 15917
4	**0 , 18141**	0 , 18674	0 , 22559	1 , 0	0 , 23582
5	**0 , 2988**	0 , 4139	0 , 4714	0 , 3847	0 , 6860
6	**0 , 26950**	0 , 27640	0 , 29155	0 , 27815	0 , 32250

7	**0 , 4213**	0 , 6683	0 , 10473	0 , 5420	0 ,17666
8	**0 , 7861**	0 , 10521	0 , 14317	1 , 0	0 , 16184
9	**0 , 1047**	0 , 1159	0 , 1737	0 , 1288	0 , 2055
10	0 , 16682	1,0	0 , 15085	**0 , 14778**	0 , 17724
11	**0 , 34129**	0 , 43888	1 , 0	1 , 0	0 , 40535
12	0 , 5535	1,0	**0 , 5264**	1 , 0	0 , 6310

TABLE 2. Examination Timetabling Track.

Table key

(Müller) Tomas Müller (Purdue University, USA). Local search-based algorithm using routines taken from the Constraint Solver Library. Various neighbourhood search algorithms are also used to eliminate violations of hard and soft constraints.

(Gogos). Christos Gogos (Greece). GRASP procedure with a combination of other local search metaheuristics.

(Atsuta et al):Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Japan). Constraint satisfaction problem solver incorporating tabu search and iterated local search.

(De Smet) Geoffrey De Smet (Belgium). Tabu-based approach.

(Pillay) Nelishia Pillay (South Africa). Nature-inspired heuristic approach.

A detailed breakdown of the results an be found on the competition website. The winner, Tomas Müller, produced 9 best results employed a constraint based framework incorporating a series of algorithms based on local search techniques. Gogos achieved 9 feasible solutions. Atsuta produced 11 feasible solutions and recorded the best result on instance 12. De Smet uses achieves seven feasible solutions and the best recorded score for instance 10. Pillay achieved 12 feasible solutions.

We do not consider minimising the number of periods as part of this formulation as, in our 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. It is acknowledged that a full investigation and explanation of “Distance to feasibility” is required if the formulation provided here is to be useful for such purposes.
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 here due to the ease of implementation for purposes of comparison. It is hoped that the interest generated by efforts here will lead to true multi-objective evaluation of potential solutions. In particular, we specifically decided to include the weights in the data format itself rather than solvers having to hard code

them. This at least ought to easily allow variations of the weights so as to explore multi-objective properties. Also, it is unlikely that every institution would have the same weights, and so fixing them in the solver seems inappropriate.

6.2 Track 2: Post Enrolment based Course Timetabling

The second track of the competition modelled the situation where students are given a choice of 'events' to attend, with the timetable then being constructed such that the number of students able to attend their chosen options is maximised. Essentially, this problem version is an extension of the model used in ITC 2002, though for our purposes extra hard constraints were also included to help move the research further in the direction of real-world timetabling. These extra hard constraints were also intended to make finding feasibility more difficult – thus shifting the emphasis from soft constraints onto hard constraints.

The basic problem involves assigning the events (lectures, tutorials, and so on) to a fixed number of timeslots and rooms in accordance with a set of constraints. The hard constraints for the problem are as follows. First, for each event there is a set of students who have enrolled to attend; thus events need to be assigned to timeslots in such a way that no student is required to attend more than one event in any one timeslot. Next, each event also requires a set of room features (e.g. a certain number of seats, specialist teaching equipment, etc.), which will only be provided by certain rooms; thus each event needs to be assigned to a suitable room that exhibits the room features that it requires. The double booking of rooms is also prohibited. Hard constraints were also imposed stating that some events cannot be taught in certain timeslots (in a real world situation, perhaps the lecturer might be unavailable to teach at this time, or perhaps some school policy needs to be adhered to); and finally, certain precedence constraints – stating that some events need to be scheduled before or after others – were also stipulated.

The three soft constraints considered in this problem are all period pattern related, and were the same as the original competition: (1) Students should not be required to attend events in timeslots that occur at the end of a working day; (2) Students should not have to attend events in three or more consecutive timeslots in the same day; and (3) Students should not be required to attend just one event in a day. These are penalised 'per student'. A fuller description of this problem, including the precise methods of calculating solution quality are all given in

the official technical report of this problem [Lewis et al(2007a)], available on the ITC2007 website.

In all, sixteen problem instances were released for this track (eight early, eight late). A further eight hidden instances were then also used during the final for verification purposes. All instances for this track were created using an automated problem generator designed by the competition organisers, and all are known to feature at least one perfect solution – that is, a solution with no hard or soft constraint violations. The number of events in these instances ranges between 200 and 400, the number of rooms between 10 and 20, and the number of students between 300 and 1000. Such values were felt to be representative of practical timetabling problems, while also being of a manageable size for competitors to conveniently test and analyse their algorithms.

In all, fourteen entries were submitted to this track of the competition, though one of these was eventually withdrawn. Five finalists were then selected from the remaining thirteen entries. In Table 3 we summarise the best results that were obtained by each of these finalists on our own machines during the verification stage of the competition. Further details on all of these approaches and a full listing of all of the results can be found on the competition website.

Algorithm Rank *Instance #*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*
1	0 , 571	0 , 61	0 , 1482	**0 , 15**	0 , 1861
2	0 , 993	0 , 547	0 , 1635	**0 , 0**	39 , 2174
3	**0 , 164**	0 , 382	0 , 288	0 , 391	0 , 272
4	0 , 310	0 , 529	0 , 385	**0 , 239**	0 , 425
5	**0 , 5**	**0 , 5**	0 , 559	0 , 34	0 , 8
6	**0 , 0**	**0 , 0**	0 , 851	0 , 87	0 , 28
7	0 , 6	**0 , 0**	0 , 10	**0 , 0**	0 , 13
8	**0 , 0**	**0 , 0**	**0 , 0**	0 , 4	0 , 6
9	0 , 1560	**0 , 0**	0 , 1947	**0 , 0**	162, 2733
10	0 , 2163	**0 , 0**	0 , 1741	**0 , 0**	161, 2697
11	**0 , 178**	0 , 548	0 , 240	0 , 547	0 , 263
12	0 , 146	0 , 869	0 , 475	**0 , 32**	0 , 804
13	**0 , 0**	**0 , 0**	0 , 675	0 , 166	0 , 285
14	0 , 1	**0 , 0**	0 , 864	**0 , 0**	0 , 110
15	**0 , 0**	0 , 379	**0 , 0**	**0 , 0**	0 , 5
16	0 , 2	0 , 191	**0 , 1**	0 , 41	0 , 132
17	**0 , 0**	0 , 1	0 , 5	0 , 68	0 , 72
18	**0 , 0**	**0 , 0**	0 , 3	0 , 26	0 , 70

19	0 , 1824	267, 1862	0 , 1868	**0 , 22**	197, 2268
20	**0 , 445**	0 , 1215	0 , 596	655, 2735	0 , 878
21	**0 , 0**	**0 , 0**	0 , 602	0 , 33	0 , 40
22	0 , 29	**0 , 0**	0 , 1364	**0 , 0**	0 , 889
23	**0 , 238**	0 , 430	0 , 688	11 , 1275	0 , 436
24	**0 , 21**	0 , 720	0 , 822	0 , 30	0 , 372

Table 3: Post-enrolment CTT track. Best results produced in the final by the various competing algorithms. Also included are the organisers' results, which were not included officially as part of the competition.

Table Key
(1) Hadrien Cambazard, Emmanuel Hebrard, Barry O'Sullivan, and Alexandre Papadopoulos (Cork Constraint Computation Centre, Ireland): Mixed metaheuristic approach including tabu search and simulated annealing used in conjunction with various neighbourhood operators
(2) Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Kwansei-Gakuin University, Japan): Combination of a general purpose constraint satisfaction solver, tabu search and iterated local search techniques.
(3) Marco Chiarandini, Chris Fawcett, and Holger H Hoos (University of Southern Denmark): Hybrid algorithm comprising a constructive procedure for achieving feasibility, followed by applications of local search and simulated annealing for satisfying the soft constraints
(4) Clemens Nothegger, Alfred Mayer, Andreas Chwatal, and Gunther Raidl (Vienna University of Technology, Austria): Ant colony optimisation algorithm used in conjunction with a local improvement search routine.
(5) Tomas Müller (Purdue University, USA): Local search-based algorithm using routines taken from the Constraint Solver Library. Various neighbourhood search algorithms are also used to eliminate violations of hard and soft constraints.

Recall that in practice, the winner of each track was chosen according to the results obtained from ten runs on each instance, and so the results in this table do not necessarily correspond with the final rankings of the competition. However, this table does highlight some interesting features of the results. We see, for example, that three of the five finalists achieved feasibility on all instances at least once during the final. We also see that winner of the competition achieved the best results (tied or outright) in 13 of the 24 instances, which was more than any of the other entrants. One intriguing feature of these results is that instances 1, 2, 9, and 10 – which are known to be highly constrained instances with relatively small numbers of rooms into which events can to be scheduled – seem to be solved very well by algorithms (2) and (4), with the remaining algorithms seeming to struggle with these considerably. However, algorithms (2) and (4) do seem to find other instances more difficult in some cases. It would be interesting in the future to identify exactly what aspects of the various algorithms allow them to achieve success on certain instances, and to see if these could be combined to form an algorithm that is successful across a wider range of problem instance.

6.3 Track 3: Curriculum-Based Course Timetabling

The third track of the competition is concerned with Curriculum based Course Timetabling (Curriculum CTT). This problem consists of the weekly scheduling of lectures for several university courses within a given number of rooms and time periods, where conflicts between courses are set according to the curricula published by the University and not on the basis of enrolment data.

This formulation applies to the University of Udine (Italy) and also applies to many Italian and indeed International Universities; although it is slightly simplified with respect to the real problem to maintain a certain level of generality.

The problem consists of the following entities (a fuller description in the official technical report of this problem [Di Gaspero et al (2007)], available on the ITC2007 website):

Days, Timeslots, and Periods. We are given a number of teaching days in the week (typically 5 or 6). Each day is split in a fixed number of timeslots, which is equal for all days. A *period* is a pair composed of a day and a timeslot. The total number of scheduling periods is the product of the number of days and number of timeslots per day.

Courses and Teachers. Each course consists of a fixed number of lectures to be scheduled in distinct periods, it is attended by a given number of students, and is taught by a teacher. For each course there is a minimum number of days that the lectures of the course should be spread in, moreover there are some periods in which the course cannot be scheduled.

Rooms. Each room has a capacity, expressed in terms of number of available seats. All rooms are equally suitable for all courses (if large enough).

Curricula. A curriculum is a group of courses such that any pair of courses in the group have students in common. Based on curricula, we have the conflicts between courses and other soft constraints.

The solution of the problem is an assignment of a period (day and timeslot) and a room to all lectures of each course.

The set of hard constraints is the following:

Lectures: All lectures of a course must be scheduled, and they must be assigned to distinct periods. A violation occurs if a lecture is not scheduled or two lectures are scheduled in the same period.

Room Occupancy: Two lectures cannot take place in the same room in the same period. Two lectures in the same room at the same period represent one violation. Any extra lecture in the same period and room counts as one more violation.

Conflicts: Lectures of courses in the same curriculum or taught by the same teacher must be all scheduled in different periods. Two conflicting lectures in the same period represent one violation. Three conflicting lectures count as 3 violations: one for each pair.

Availabilities: If the teacher of the course is not available to teach that course at a given period, then no lecture of the course can be scheduled at that period. Each lecture in a period unavailable for that course is one violation.

There are four soft constraints as follows:

Minimum Working Days: The lectures of each course must be spread into the given minimum number of days. Each day below the minimum counts as 5 points of penalty.

Curriculum Compactness: Lectures belonging to a curriculum should be adjacent to each other (i.e., in consecutive periods). For a given curriculum we account for a violation every time there is one lecture not adjacent to any other lecture within the same day. Each isolated lecture in a curriculum counts as 2 points of penalty.

Room Capacity: For each lecture, the number of students that attend the course must be less or equal than the number of seats of all the rooms that host its lectures. Each student above the capacity counts as 1 point of penalty.

Room Stability: All lectures of a course should be given in the same room. Each distinct room used for the lectures of a course, but the first, counts as 1 point of penalty.

The minimum working days and curriculum compactness are both ‘period pattern related’, and similar to those in the other tracks, though differ in that penalties are assigned ‘per event’ whereas in the other tracks the penalties are ‘per student’. The room related penalty for room capacity is soft here, though for comparison, is considered as a hard constraint in the other tracks. The Room Stability is unique to this track, as the concept of a course being a set of events is not used in the other tracks.

Twenty one instances were released for this track: 7 for each set (early, late, and hidden). All instances are real data and come from the University of Udine. For all instances there exists at least one feasible solution, but at the time of release the optimal values for the soft constraints

was not known (though some of them have now been solved). Hidden instances have been released only after the conclusion of the competition.

The number of courses in these instances ranges between 30 and 131, the total number of lectures from 138 to 434, the number of rooms between 5 and 20, and the number of curricula between 13 and 150.

The actual formulation used at the University of Udine, with respect to the one issued for ITC2007, has the following extra features:

- A cost component dealing with the lunch break for students: at least one free slot among those around the lunch time.
- The curriculum compactness feature is more complex, and specific patterns are more penalized than others.
- There is a maximum daily student load for each curriculum.
- Some specific lectures must be (must not be) in consecutive periods.
- Rooms might not be available in certain periods, and they must be not suitable for specific lectures.
- If a room is too big for a class, this is also penalized (this is not only for the unpleasant feeling that an empty room provokes, but also to save big rooms for unforeseen activities).
- Weight assigned to soft violations are more complex, and they depend also on the number of students in the curriculum.
- Teacher preferences on periods and rooms are only included as soft constraints.

The only reason for which we have decided to remove all the above features is to maintain a certain degree of generality, so as to do not inflict to the participant the burden to understand all the details of the formulation.

Competitor Rank / *Instance #*	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*
1	0, 5	0, 5	0, 5	0, 5	0, 10
2	0, 51	0, 55	**0, 50**	0, 111	0, 111
3	0, 84	**0, 71**	0, 82	0, 128	0, 119
4	0, 37	0, 43	**0, 35**	0, 72	0, 72

5	330	**0, 309**	0, 312	0, 410	0, 426
6	**0, 48**	0, 53	0, 69	0, 100	0, 130
7	**0, 20**	0, 28	0, 42	0, 57	0, 110
8	0, 41	0, 49	**0, 40**	0, 77	0, 83
9	0, 109	**0, 105**	0, 110	0, 150	0, 139
10	**0, 16**	0, 21	0, 27	0, 71	0, 85
11	**0, 0**	**0, 0**	**0, 0**	**0, 0**	0, 3
12	**0, 333**	0, 343	0, 351	0, 442	0, 408
13	**0, 66**	0, 73	0, 68	0, 98	0, 113
14	0, 59	**0, 57**	0, 59	0, 90	0, 84
15	0, 84	**0, 71**	0, 82	0, 128	0, 119
16	**0, 34**	0, 39	0, 40	0, 81	0, 84
17	**0, 83**	0, 91	0, 102	0, 124	0, 152
18	0, 83	0, 69	**0, 68**	0, 116	0, 110
19	**0, 62**	0, 65	0, 75	0, 107	0, 111
20	**0, 27**	0, 47	0, 61	0, 88	0, 144
21	**0, 103**	0, 106	0, 123	0, 174	0, 169

Table 4: Best results produced in the final by the various competing algorithms.

Table Key
(1) Tomáš Müller (Purdue University, USA): *see Table 4*
(2) Zhipeng Lü and Jin-Kao (Université d'Angers, France): Iterated Tabu Search with Kempe Chains
(3) Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Kwansei-Gakuin University, Japan):): *see Table 4*
(4) Martin Josef Geiger (University of Hohenheim, Germany): Threshold accepting local search
(5) Michael Clark, Martin Henz, Bruce Love (Metaparadigm Pte Ltd): Repair-based Local Search

In all, seventeen entries were submitted to this track of the competition. In Table 4 we summarise the best results out of 10 runs that were obtained by each of these finalists (ordered by ranking in the competition) on our own machines during the verification stage of the competition. All solutions are feasible. Further details on all of these approaches and a full listing of all of the results can be found on the competition website.

As already mentioned in Section 5, the winner of each track was chosen according to the results obtained from ten runs on each instance, and so the results in this table do not necessarily correspond with the final rankings of the competition. In any case, the ranking is preserved also in terms of number of best results obtained (in bold face): 11 for solver (1), 6 for solver (2), 5 for solver (3), 1 for solver (4), and none for solver (5).

7 Conclusions and Discussion

This paper has presented detailed information relating to the 2nd International Timetabling Competition. The competition rules have been outlined and discussed along with the major differences from the 1st Competition. In addition, limitations of each of the new formulations

presented have been discussed as a means of illustrating the need for further work in addressing the identified gap which currently exists between research and practice in relation to this research area. Results have been presented for each of the competition tracks along with descriptions of the associated algorithms.

It should be noted that although the competition has been highly successful in bringing the community together introducing new ideas and creating general interest in the field, the results should be treated with care due to programming issues and the possibility of competitors tuning their programs more accurately to the datasets. It is important to emphasise that the competition serves more as an encouragement to researchers as opposed to an identifier of best techniques. Just because an algorithm beats another on some instances, it certainly does not imply that it is a superior algorithm in general.

In order to help anyone that might be considering organising a similar competition we now briefly discuss some of the organisational difficulties that we encountered. The first difficulties were associated with the inevitable ambiguities in natural language descriptions of the problem constraints and objectives. Solution validators were provided to help resolve such ambiguities. However, although straightforward, this was surprisingly time-consuming, especially on the curriculum-based and examination timetabling with their many real-world objectives. (In contrast, the course-based timetabling had the advantage of building on the work of the previous competition.) A recommendation would be, independently and well before the start of the competition, to write separate validators. Also, the cross-checking of validators, and if possible, the creation of declarative but computer readable encodings, for example, using integer programming, and then to also use these as solution validators. Another practical difficulty was that submissions often had slightly different formats for how to run the executable. Although this gave no grave problems, it could lead to a very time-consuming adjustment for each submission. For the future, the authors would recommend that some form of test harness is provided in advance, that specifies precisely the required interface to the submitted solvers, and stringently demand that submissions respect the interface.

Overall it can be surprisingly difficult to create a set of benchmarks and then run a competition. However, it is important to do so because timetabling in the literature is beset by published peer-reviewed papers for which the constraints are specific to one institution, the algo-

rithms being ”one-off"s and the data not being available. This feature means that the scientific value of such papers to science is often reduced, and many such papers are quickly forgotten. In this paper, we described work that although not developing new algorithms in and of itself, we still believe will have longer term value, by supporting such algorithm development, and so enhancing science and pushing forward research.

References

Abdullah et al. (2007): S. Abdullah, E.K. Burke and B. McCollum, Using a Randomised Iterative Improvement Algorithm with Composite Neighbourhood Structures for the University Co2xurse Timetabling Problem, accepted for publication in Metaheuristics - Progress in Complex Systems Optimization (eds. K.F. Doerner, M. Gendreau, P. Greistorfer, W.J. Gutjahr, R.F. Hartl and M. Reimann), to appear in the Springer Operations Research / Computer Science Interfaces Book series, 2007.

Borning et al (1987): Alan Borning, Robert Duisberg, Bjorn Freeman-Benson, Axel Kramer and Michael Woolf . Constraint hierarchies. *ACM SIGPLAN Notices V*olume 22 , Issue 12 (December 1987) Pages: 48 - 60 Year of Publication: 1987. ISSN:0362-1340 http://portal.acm.org/citation.cfm?id=38807.38812&coll=GUIDE&dl=GUIDE

Burke et al. (2004): Burke, E.K., Bykov, Y., Newall, J.P., Petrovic, S., (2004) “A Time-Predefined Local Search Approach to Exam Timetabling Problems”, *IIE Transactions*, 36(6), 509-528.

Carter et al. (1996): M.W. Carter, G. Laporte and S.Y. Lee. (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of Operational Research Society*, 47(3): 373-383.

Chiarandini et al.(2006): M. Chiarandini, M. Birattari, K. Socha, and O. Rossi-Doria, An effective hybrid approach for the university course timetabling Problem, *Journal of Scheduling*, 9 (5): 403--432, 2006.

Di Gaspero et al. (2007), L. Di Gaspero, B. McCollum, and A. Schaerf. The second International timetabling competition (ITC2007): Curriculum-based course timetabling (track 3). Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1, School of Electronics, Electrical Engineering and Computer Science, Queen’s University, Belfast (UK), August 2007.

Di Gaspero and Schaerf (2006), L. Di Gaspero and A. Schaerf, Neighborhood portfolio approach for local search applied to timetabling problems. *Journal of Mathematical Modeling and Algorithms*, 5 (1) 65--89, 2006, DOI: 10.1007/s10852-005-9032-z.

Dueck, G., (1990), "Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing", J. Computational Physics, Vol. 90, 161-175.

ITC (2002): http://www.idsia.ch/Files/ttcomp2002/ Accessed July 2008.

ITC (2007): http://www.cs.qub.ac.uk/itc2007. Accessed August 2008.

Kostuch (2005): Ph. Kostuch, The university course timetabling problem with a three-phase approach. In Edmund Burke and Michael Trick, editors, Proc.of the 5th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT-2004), selected papers}, volume 3616 of Lecture Notes in Computer Science}, pages 109--125, Berlin-Heidelberg, 2005. Springer-Verlag.

Lewis (2008): Lewis, R. A Survey of Metaheuristic-based Techniques for University Timetabling Problems". OR Spectrum, vol 30 (1), pp 167-190.

Lewis et al (2007a): Lewis, R., Paechter, R., McCollum, B., 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. Prifysgol Caerdydd/ Cardiff University, Wales. ISSN: 1750-6658, v 1.0.

Lewis et al. (2007b): R. Lewis, B. Paechter, and O. Rossi-Doria. Metaheuristics for University Course Timetabling. In Evolutionary Scheduling (Studies in Computational Intelligence, vol. 49), K. Dahal, Kay Chen Tan, P. Cowling (Eds.) Berlin: Springer-Verlag, pp 237-272

McCollum(2007a), McCollum, B., A Perspective on Bridging the Gap between Theory and Practice in University Timetabling, Practice and Theory of Automated Timetabling VI, Springer LNCS Vol 3867, 2007, pp 3-23.

McCollum (2007b), McCollum, B., McMullan, P., Burke, E.K., Parkes, A.J., Qu, R., The second International timetabling competition (ITC2007): Examination Timetabling Track. Technical Report QUB/IEEE/Tech/ITC2007/Exam/v4.0/17, School of Electronics, Electrical Engineering and Computer Science, Queen's University, Belfast (UK), August 2007.

McMullan P. (2007) An Extended Implementation of the Great Deluge Algorithm for Course Timetabling, Computational Science – ICCS 2007, Springer LNCS Vol 4487, July 2007, pp 538-545.

Lewis et al. (2007), R. Lewis, B. Paechter, and O. Rossi-Doria. Metaheuristics for University Course Timetabling. In Evolutionary Scheduling (Studies in Computational Intelligence, vol. 49), K. Dahal, Kay Chen Tan, P. Cowling (Eds.) Berlin: Springer-Verlag, pp 237-272

Qu et al. (2007), R. Qu, E.K. Burke, B. McCollum, L.T.G. Merlot, and S.Y. Lee. The State of the Art of Examination Timetabling. Technical Report NOTTCS-TR-2006-4, School of CSiT, University of Nottingham.

Schaerf (1999), A. Schaerf., A survey of automated timetabling, Artificial Intelligence Review, 13 (2): 87--127, 1999.

Schaerf and Di Gaspero (2007), A. Schaerf and L. Di Gaspero, Measurability and Reproducibility in University Timetabling Research: Discussion and Proposals. In Edmund Burke and Hana Rudova, editors, Proc. of the 6th Int. Conf. on the Practice and Theory of Automated Timetabling (PATAT-2006), selected papers, volume 3867 of Lecture Notes in Computer Science, pages 40--49, Berlin-Heidelberg, 2007. Springer-Verlag.