Analysing Similarity in Examination Timetabling

E. K. Burke[1], A. J. Eckersley[1], B. McCollum[2], S. Petrovic[1], and R. Qu[1]

[1] University of Nottingham, Nottingham, NG8 1BB, UK
{ekb, aje, sxp, rxq}@cs.nott.ac.uk,
[2] Queen's University of Belfast, Belfast, BT7 1NN, UK
b.mccollum@qub.ac.uk

Abstract. In this paper we carry out an investigation of some of the major features of exam timetabling problems with a view to developing a similarity measure. This similarity measure will be used within a case-based reasoning (CBR) system to match a new problem with one from a case-base of previously solved problems. The case base will also store the heuristic or meta-heuristic technique(s) applied most successfully to each problem stored. The technique(s) stored with the matched case will be retrieved and applied to the new case. The CBR assumption in our system is that similar problems can be solved equally well by the same technique.

1 Introduction

1.1 Examination Timetabling

Exam timetabling is a subset of the general timetabling problem, proved to be NP-hard by Cooper & Kingston [17]. A set of exams must be scheduled to a set of timeslots such that every exam is located in exactly one timeslot within the timetable, subject to certain constraints. In most real life problems there will also need to be some allocation of resources (e.g. rooms, invigilators and special equipment) to each exam. The constraints on the problem can be divided into *hard constraints* and *soft constraints.* Hard constraints are those which must be satisfied in order for the resulting timetable to be considered feasible. Soft constraints are those which are desirable to be satisfied, but which in general cannot all be wholly satisfied. The quality of a resulting timetable is measured according to an objective function which weights the violation of the different soft constraints. The aim of the timetabling problem is to create a timetable which minimises this objective function, meaning that as many soft constraints are satisfied as possible. If a timetable cannot be found which violates no hard constraints then the problem is said to be infeasible. In such cases, some hard constraints must be relaxed in order to produce an acceptable timetable.

Both hard and soft constraints vary between institutions, as can be seen from the survey of British universities carried out by Burke et. al. [1]. The most common hard constraints can be summarised as follows:

– Every exam must be scheduled in exactly one timeslot
– Every exam must be assigned to a room(s) of sufficient size and assigned an invigilator(s)
– No student must be scheduled to be in two different exams at the same time
– There must be enough seats in each period for all exams scheduled
– Certain exams must be scheduled into specific timeslots or rooms
– Certain exams must take place simultaneously

Depending on the institution in question, some of these hard constraints may not be relevant. Some universities have very tight constraints on room space whereas others have a large amount of rooms usable for exams so that the constraint on room availability is unnecessary. Soft constraints tend to be a lot more varied and much more dependent on the peculiarities of each institution since it is these which measure, amongst a given set of feasible timetables, which are the most desirable. For some exam timetabling problems it is difficult to find a feasible solution at all, whereas for other problems there are a large number of feasible solutions and the focus of the problem solving is very much directed to the minimisation of soft constraint violations. Soft constraints often encountered include the following:

– Exams for each student should be spread as far apart as possible
– A student should not be required to take x exams in y periods
– Time windows for certain exams
– No more than x exams taking place simultaneously
– No more than y students scheduled to sit exams at any one time
– Exams should not be split across rooms
– No more than one exam in a room at a time
– Teacher or student preferences
– Distance between rooms holding a given exam should be minimised (when the exam is split across two or more rooms)
– The total number of periods should be minimised

Some constraints may be soft in one problem, but hard in another, depending on the requirements of the universities. In some cases, the total number of periods is fixed *a priori* and is taken as a hard constraint, whilst in others this is taken into account in the objective function with one of the aims being to minimise the length of timetable produced. It may also be the case that different institutions with the same basic constraints place differing weights on the soft constraints in the objective function to define what they consider a *good* timetable. It can often be the case that two or more soft constraints on a given problem are conflicting, such as the wish to spread exams as far apart as possible and the wish to minimise the total length of the timetable.

Due to its NP-hard nature, the exam timetabling problem lends itself well to being tackled by a variety of different meta-heuristic techniques. These are essentially techniques which work in the solution space of exam timetables, having been seeded with an initial solution which may be either feasible or infeasible. Initial solutions are often found using a fast graph colouring or bin packing heuristic which sequentially assigns each exam into an initially empty timetable until all exams are assigned to a timeslot [3]. Such construction heuristics can themselves produce solutions of an acceptable quality and are often combined with a second, improvement phase [13, 14, 18].

A range of different meta-heuristic techniques have been applied to exam timetabling in recent years. Amongst those successfully applied include Simulated Annealing [27, 28], Tabu Search [25, 29, 26], Great Deluge [7] and Memetic algorithms [2, 4]. Amongst the current methods being applied to exam timetabling are Ant Colony optimisation and Variable Neighbourhood Search. For a wider view on exam timetabling techniques, consult the Selected Papers volumes from the past PATAT conferences. Burke et al. [8] propose a case-based reasoning (CBR) technique for selecting amongst a set of previously used heuristics and meta-heuristics a suitable one to apply to a new problem. In this paper, we examine this idea further, looking more closely at the key features involved in measuring the similarity of exam timetabling problems.

1.2 The Automation of Exam Timetabling

One of the major goals in the area of timetabling research is to develop new techniques able to solve problems in a given domain (such as exam timetabling) and test these techniques on benchmark problems to see how they compare against other methods. However, another major aim of the automated timetabling community is to apply these techniques to real world problems, not simply as test data, but as part of an actual system which will be used in one or more institutions to produce their regular timetables. The survey conducted by Burke et. al. [1] of British Universities in 1996 gives a very good overview of the varying constraints between different universities and of their methods for dealing with these. Of perhaps most interest though were the somewhat surprising results in response to the question of computer usage and automation of the timetabling process:

> 'Fifty eight percent of those universities that responded use a computer at some stage in the timetabling process. Of these, eleven (21%) have a facility to perform scheduling, although these may, in some cases, still require a certain amount of manual input and previous knowledge of the particular timetabling problem... Four other universities stated that they are either currently developing their own or customising a commercial package'

The authors also report that, of those universities which don't schedule by computer, half use the previous year's timetable as a template for the new one and half construct a new timetable each year. The time taken by the latter group to produce their timetables tends to be around four weeks, indicating that some form of automation could certainly be of benefit. It is also noted that 10 of the 11 universities which use a scheduling system make no use of the previous year's timetable. A number of different techniques were employed by those not using a computer for their timetabling process, but many of these relied on assumptions which, in many cases, are becoming increasingly less valid. One of the main examples of this is the technique of building up the global timetable by putting together individual exam timetables produced by each department. With the increase in modular courses which have a large inter-departmental dependence, this technique becomes much less viable. For the manual timetabler, this results in a potentially huge change in the way timetables must be constructed and the lack of any kind of automation can make the timetabling process employed very inflexible to the changing requirements of a modern university. In many cases, a manual equivalent to the kind of case-based reasoning approach applied to course timetabling by Burke et al. [5, 6] is used - some parts of the previous year's timetable are re-used with slight alterations while the rest is rebuilt using the new partial timetable as a starting point.

Often, re-using parts of previous timetables is not a very efficient or useful method for creating a new timetable. However, the use of the same technique as has been used either for the same institution the previous year (or in earlier years) or by a different institution whose timetabling needs are similar can be very effective. For this, a case-base of problem solving methodologies would be stored rather than parts of actual problem solutions. These can then be retrieved and applied to the timetabling problem at hand. Such a system would require some method of deciding which technique to retrieve to apply to the new problem.

In case-based reasoning (CBR), this process is carried out using a similarity measure between problems. Key features of exam timetabling problems are stored in the case-base as individual cases, together with the successful technique(s) used on each problem forming part of each case. The key features of any new problem can then be matched against those in the case-base and the best match found. The assumption from CBR is that the technique used for that retrieved problem should be able to provide a high quality solution to the new similar problem also. With a large enough case-base of problems and their associated techniques, such a system could be used very effectively by universities to solve their timetabling problems using knowledge stored from the problem specifications of other universities. Acting as a "black box", there would be no personal data about any university's timetabling problem within the system, only contextualised knowledge as to the overall problem structure which may be similar to that in many other institutions. In this paper we examine many of the key features which will form a part of a similarity measure between exam timetabling problems. Using this within a CBR framework, we can match a new problem with a previously solved problem to retrieve a technique which should work well on the new problem.

1.3 Issues in Automated Timetabling

Given the findings of the survey of British universities, one obvious question to consider is why so few universities used (in 1996, at least) an automated or at least partially automated system to produce their timetables, given that those who do can save many weeks of time on the scheduling process. This wasn't one of the questions specifically asked by the survey, however there are a number of indicators from the results of the survey which may help to explain. Most obviously, in the case of those universities which use the previous year's timetable to construct the new one, there is no current need for automation. So long as the requirements are not changing significantly from year to year, it is relatively easy for an experienced timetabler to account for these changes whilst adapting the previous timetable. Also, in doing so, the timetabler can get a much better feel for the structure of the timetable and how it was put together than would be possible with the use of an automated system. Having said that, a semi-automated process involving a human timetabler manually moving exams around a partial or fully completed timetable could serve the same purpose provided the interface is user-friendly. This was one of the points covered by the conclusions of the survey when considering the requirements for a general timetabling system which may be used by non-experts.

For those institutions which can feasibly split their exam timetabling problem into departmental blocks, there may also be deemed to be no need for an automated timetabling package as each of these sub-problems may be relatively small and easy to solve by the individual departments. With the increase in modularised courses however, this technique becomes less feasible as mentioned earlier. The piecing together of the individual parts of the timetable from each department into a global timetable may require a lot of moving around of exams, which may result in a final timetable unacceptable to some departments.

Perhaps one of the main reasons behind the seeming unwillingness to automate the timetabling process in many universities is the sheer size and complexity of the problem. Hand in hand with this may come a lack of trust or belief by human timetablers in the ability of meta-heuristic methods to be able to solve such problems to an acceptable quality. In many cases, timetabling is carried out by one or more experts who have an in-depth knowledge of the overall timetabling problem and the various constraints included. For many experts, whatever their area of expertise, the idea that their years of experience at dealing with the problem could be satisfactorily replaced by an automated system is understandably greeted with some scepticism. In building up their experience, experts will invariably develop a vast knowledge of their subject area, the techniques used to solve the problems, solutions and parts of solutions which are acceptable or not and many other small pieces of information which are used.

In many cases expert timetablers can look at the timetable they produce and decide 'by feel' how good it is and what areas need changing. It is perhaps that issue of 'feel' which is key - something which cannot easily be represented purely by numbers and equations. The ability to take a general overview of a timetable you have constructed and decide how good it is without any kind of numerical evaluation is one which expert manual timetablers rely heavily on. To rely on a fully automated system to do the same job would require all that knowledge of the expert to be adequately transferred to a computer system dealing with exact numerical inputs. Many universities attempt to copy the role of a human expert by creating what are known as ‘expert systems’ which incorporate into a rule-based system the methods which the human expert relies on to construct his or her timetables. Such automated methods are perhaps far more acceptable and trusted by those in charge of timetabling at universities than the somewhat more abstract methods employed by meta-heuristics.

Another issue with the automated timetabling process is that of defining and modeling the problem for the automated system to solve. Given a set of constraints and their weights forming a single objective, a meta-heuristic technique will attempt to create a timetable by minimising this objective function. However, the final timetable produced will only be as good as the constraint weightings fed into the system by the human timetabler. Given two very different timetables, each with the same penalty evaluated by the objective function, will the human timetabler consider the two to be equally good or will he/she regard one as a lot better? If so then why is one better than the other and how can this information be included into the automated system so that it will be evaluated as such? Since there is rarely, if ever, such a thing as the perfect timetable, the ability of an automated system to output a variety of different timetables, often very quickly can be extremely useful. Depending on the technique used, the timetables produced may all be quite similar with just small differences or have a large amount of diversity whilst still having similar evaluation by the objective function. Provided the system runs relatively fast, the weightings on the constraints can easily be altered to change the focus of the timetables produced also. This provides a much greater degree of flexibility than is usually possible for a human timetabler and is one of the biggest strengths of an automated system.

In the following section we examine the various features of exam timetabling problems in a qualitative manner with a view to developing a measure of similarity between problems based on their most important features for use within our CBR system. With a number of potentially very different techniques within the case-base, we aim to develop a case-based reasoning system which will allow the user to input their needs and can, if desired, return a selection of complete timetables of similar quality from which the human timetabler can select the best one. If none of the timetables produced are acceptable, the user can change the focus of the constraints and ask the system to produce a new set of solutions from which to choose. By learning from its past experiences, the system will select from its case-base of heuristic and meta-heuristic techniques the one which best fits the new problem. Feedback from the human user as to the quality of the timetable(s) produced is also important for the system to learn whether the match was a good one or whether re-indexing is required to avoid the chosen heuristic being selected again for the given problem.

2 Studying the features of Exam Timetabling problems

2.1 The requirements of a CBR system

As mentioned earlier, our aim is to produce a case-based reasoning (CBR) heuristic/meta-heuristic selector which will intelligently select, from a variety of techniques, the one best suited to a new problem given to the system. This is done by a matching process which employs a similarity measure between exam timetabling problems. Within the case base, each case is made up of a set of feature-value pairs representing a given problem, together with the heuristic or meta-heuristic technique(s) which give the best results for that problem. When presented with a new problem, its set of feature-value pairs is matched with those of all cases within the case-base and the most similar case(s) will be retrieved. The similarity measure employed will calculate a weighted sum of the difference in value for each feature between the two problems under comparison and is shown in equation 1

$$S(C_x, C_y) = g(\sum_{i=1}^{n} h_i(|fx_i - fy_i|)) \tag{1}$$

where:

- n is the number of features in the similarity measure
- fx_i and fy_i are the values of the ith feature of cases C_x and C_y respectively
- $h_i(a) := w_i a^2 + 1$; w_i representing the weight of the ith feature
- $g(b) := 1/\sqrt{b}$

The similarity, $S(C_x, C_y)$, between two cases will be in the interval $(0, 1]$, with 1 representing two identical cases and results closer to 0 indicating cases with a low degree of similarity.

Clearly the key elements of this similarity function are the features together with their weights representing the importance of each feature to the overall similarity measure. To decide exactly which features should be used in the case-base and how to weight them is a near impossible task to perform purely by hand. Instead, we use knowledge discovery techniques in two stages, as reported by Burke et al. [11]. An initially large set of features and cases is systematically trimmed leaving only those cases which contribute positively to the knowledge of the system and identifying which combinations of features give the best system performance. In [11], the system performance is measured as being the number of successful retrievals of one of the best two heuristics as pre-calculated for a set of training cases and a set of test cases. Using these knowledge discovery techniques the system can tune the weightings on the features used as well as selecting the best subset of features themselves to improve the performance of the case base. Tabu Search and Hill Climbing techniques are both used within the knowledge discovery to obtain the best feature vector from the search space of all possible feature vectors. It is found that the best system performance comes from having a relatively small number of features within the case-base (usually between 3 and

7). Fewer features give too little information from which to accurately measure similarity, whilst too many features reduce system performance by diluting the impact of the most important features. In the following subsection we examine the initial list of simple features with which we begin the knowledge discovery process. Ratios between all pairs of simple features are also included within the features search space as many of these will provide far more meaningful measures of similarity than single features.

2.2 Qualitative analysis of features used within the CBR system

Whilst Burke et al. [11] concluded that between 3 and 7 features gives best performance for a CBR system, we initially wish to identify as many features as possible to use in the knowledge discovery process. In this way, a more thorough search can be performed to find the best feature vector to be used for our similarity measure given in equation 1. An analysis of the features used is given below, including combinations of features which may prove important.

Number of Students By itself, the number of students within an exam timetabling problem can be extremely misleading as a measure of problem difficulty and as such needs to be carefully considered before being input as a feature to be considered. In [9], the authors examine the role of the 'students' in the definition of an exam timetabling problem. It is evident that for real life problems with tight room capacity constraints resulting in a capacity constraint on each period of the timetable, the number of students is a crucial factor. However, it was noted that for the problems considered in that paper (no capacity constraints), the number of students was irrelevent. In such cases, the only role of the 'students' is to define the conflict matrix via their enrolments.

Having set up the conflict matrix, the students play no further part in the problem and how many of them there are is unimportant. Indeed, there are a huge number of student enrolment sets which could define exactly the same conflict matrix - of particular note are the set of 2-enrolment students in which every student has just two enrolments and contributes to exactly 1 edge on the conflict graph. In this case the number of students would be equal to the number of edges in the conflict graph (with weights on edges counting as multiple edges). At the other end of the spectrum would be the minimum set of 'students' which would define the same conflict matrix. This set would be obtained by assigning the largest clique of exams in the conflict graph to a student and then continuing, assigning the remaining largest clique to a new student until all exams and edges are assigned to a student. The relevence of this as a measure for similarity is considered later under the heading of "Cliques".

In those problems where capacity constraints are important, the number of students becomes more important, but only as a ratio to the total capacity available over the period of the timetable and in other ratios concerning enrolments for individual exams. Purely by itself, the number of students in a problem does not represent a feature which can contribute to a similarity measure.

Number of Events (Exams) Always reported together with the number of students to give an idea of the size of a given data set, the exams form the core of the problem, being far more influencial than the student numbers in defining the problem structure. The exam timetabling problem is concerned, of course, with assigning these exams to timeslots within a timetable with the main constraint being that two clashing exams are not scheduled in the same slot. In a graph colouring model of exam timetabling, the exams form the nodes of the graph with the edges, defined by the student enrolments, representing the conflicts. It is the structure of this graph representation which is one of the key aspects in how well a given heuristic or meta-heuristic technique will perform when optimising the problem. As a similarity feature by itself, results from the literature suggest that 'no. of exams' is a simple, yet effective indicator as to the potential best technique. The Great Deluge technique with adaptive initial solution generation presented by Burke and Newall [7] provides best known results at the time of writing on the 3 largest problems[3] from the Carter benchmark data sets [13] (see tables 1 and 2), each with > 500 exams. On some of the smaller problems, however, this technique proves less effective when compared to those of Caramia [12], Merlot et al. [20] and Casy and Thompson [16] (see table 3).

One major area which can be affected by the number of exams in the problem is the run time of a particular technique. In a majority of cases, more exams in the problem equates to longer running time of the algorithm or fewer iterations than would be possible in the same time on a smaller problem. As such, techniques which can converge to a good result in fewer iterations will have an advantage on larger problems. Other noteworthy features based on the number of exams include exams per student and exams per period averages. In the case of the Carter benchmarks, the STA-F-83 data set has a very large number of exams per student and exhibits very different behaviour from other problems - The GRASP technique of Casey & Thompson [16] gives a best known solution to this problem notably better than results reported from most other techniques. Clearly there are other factors which make this problem fairly anomalous amongst the 11 data sets considered however, as EAR-F-83 and YOR-F-83 also have a relatively large exam/student ratio whilst not exhibiting the same behaviour when optimised.

Number of Periods The number of periods in a timetable can be either fixed *a priori* or may be a variable to be minimised as part of the objective function or seperately. In this paper, we consider the case where the number of periods is fixed. For those problems in which number of periods is not fixed, similarity can only be measured against other problems in which this is also the case. As a feature by itself, the number of periods assigned to a problem tells us next to nothing since it is only relevant in conjunction with other features to determine how highly constrained the problem is. In particular, the average number of exams per period and the ratio of number of clashes to number of periods may be of importance. Certainly, the number of exams per period gives a simple measure of one aspect of the *difficulty* of the problem and when combined with

[3] measured by number of exams

Table 1. Simple features for Carter Data Sets

Data Set	No. of exams	No. of students	No. of enrolments	Graph Density	No. of periods
CAR-S-91	682	16925	56877	0.13	35
CAR-F-92	543	18419	55522	0.14	32
EAR-F-83	181	1125	8109	0.27	24
HEC-S-92	81	2823	10632	0.42	18
KFU-S-93	486	5349	25113	0.06	20
LSE-F-91	381	2726	10918	0.06	18
STA-F-83	139	611	5751	0.14	13
TRE-S-92	261	4360	14901	0.18	23
UTA-S-92	622	21267	58979	0.13	35
UTE-S-92	184	2750	11793	0.08	10
YOR-F-83	190	941	6034	0.29	21

Table 2. Further features for Carter Data Sets

Data	enrolments per student		Exams per	enrolments per exam		Largest no. of clashes
Set	average	max (num)	period	average	max	for 1 exam
CAR-S-91	3.36	9 (1)	19.5	83.40	1385	472
CAR-F-92	3.01	7 (29)	17.0	102.25	1566	381
EAR-F-83	7.21	10 (9)	7.5	44.80	232	134
HEC-S-92	3.77	7 (1)	4.5	131.26	634	62
KFU-S-93	4.70	8 (11)	24.3	51.67	1280	247
LSE-F-91	4.01	8 (3)	21.2	28.66	382	134
STA-F-83	9.41	11 (209)	10.7	41.37	237	61
TRE-S-92	3.42	6 (20)	11.4	57.09	407	145
UTA-S-92	2.77	7 (23)	17.8	94.82	1314	303
UTE-S-92	4.29	6 (20)	18.4	64.09	482	58
YOR-F-83	6.41	14 (1)	9.0	31.76	175	117

Table 3. Results from the literature (best results given)

Data Set	Carter et al.	Caramia et al.	Burke & Newall	Di Gaspero	Casey & Thompson	Merlot et al.
CAR-S-91	7.1	6.6	4.6	5.7	5.4	5.1
CAR-F-92	6.2	6.0	4.0	-	4.4	4.3
EAR-F-83	36.4	29.3	36.1	39.4	34.8	35.1
HEC-S-92	10.6	9.2	11.3	10.9	10.8	10.6
KFU-S-93	14.0	13.8	13.7	-	14.1	13.5
LSE-F-91	10.5	9.6	10.6	12.6	14.7	10.5
STA-F-83	161.5	158.2	168.3	157.4	134.9	157.3
TRE-S-92	9.6	9.4	8.2	-	8.7	8.4
UTA-S-92	3.5	3.5	3.2	4.1	-	3.5
UTE-S-92	25.8	24.4	25.5	-	25.4	25.1
YOR-F-83	36.4	36.2	36.8	39.7	37.5	37.4

the conflict matrix density[4] this measure is potentially a very important one. The number of periods used for the benchmark data sets is slightly higher in each case than the minimum number of periods found to schedule all the exams in, however the constraints are still very high since the objective function used for these problems is concerned with spreading clashing exams as far apart as possible in the timetable. Clearly, the more periods available in the timetable over the minimum required to obtain a feasible solution, the more the exams can be spread out. As can be seen from table 2, the ratio of exams to periods varies hugely between problems. The main reason for this is the difference in the conflict graphs for the problems. More highly conflicting problems will result in lower exams per period since the large number of conflicts often makes it hard to schedule all the exams in fewer periods. One of the key strengths of the knowledge discovery techniques we use to select which are the important features is that they can combine pairs of simple features which we select and chose amongst the resulting large number of features the most promising feature vector which may include combinations of two simple features which we wouldn't have otherwise considered. In problems were the number of rooms forms a tight constraint on the problem, the ratio between exams per period and the number of available rooms per period will also form an important characteristic of the problem.

Conflict Matrix Density Hand in hand with the analysis of number of periods given above comes the conflict matrix density. The conflict matrix is an *exams* x *exams* size matrix in which pairs of clashing exams are noted by a 1 with all non-clashing pairs being denoted by 0. The conflict matrix density gives the ratio of 1's as a fraction of the total matrix. Therefore, a high conflict matrix density represents a high probabilty of conflict between any two exams, e.g. a density of 0.5 implies that on average each exam will clash with half of the other exams in the problem. From tables 1 and 2, it can be seen that there is a strong correlation between exams per period and conflict matrix density. As mentioned above, this is due to the fact that a densely conflicting exam graph tends to mean that a higher number of periods are needed than sparesly conflicting graphs of the same size and therefore the average number of exams per period is correspondingly lower.

The conflict matrix is one of the most important aspects of any exam timetabling problem, representing both the hard constraints and some major soft constraints. Problems with a very high conflict matrix density, such as HEC-S-92 tend to be more difficult to find feasible solutions to in the first place. Also, for meta-heuristics which work only in the space of feasible solutions, this can lead to the search space being very disconnected with respect to certain move neighbourhoods. This can have a major impact on how successfully a particular meta-heuristic can traverse the search space to find a high quality solution. Also, the more disconnected the search space is for a given local search technique, the more focus is placed on the initial solution fed to the local search - whilst

[4] discussed in more detail later on

some techniques are relatively independent of initial solution and can connect the majority of the search space very effectively, others rely heavily on a good initialisation.

Conflict matrix density is one of the simplest metrics to be taken from the conflict matrix and also one of the most effective in measuring problem *difficulty*. However, it does only give an average on the percentage of other exams that each exam will clash with. Two problems with the same conflict matrix density can still be very different in structure. We discuss below some of the other measures, which when combined with conflict matrix density, should give a better indication of problem structure for the pruposes of measuring similarity.

Largest Degree The *degree* of an exam is defined as the number of other exams in the problem with which it conflicts through having students in common. One of the most common graph heuristics for constructing initial solutions for local search techniques uses a largest degree ordering of the exams to assign sequentially to the timetable. In this way, the most conflicting exams are assigned first as these are deemed to be the most difficult to schedule. The largest degree of an exam timetabling problem would be the exam of largest degree. As another measure to be obtained from the conflict matrix, this gives us some more information on the problem structure, but by itself of course isn't enough to be of use. The number of exams of largest degree could provide useful information, but is in most cases equal to one.

Of more interest to us as a feature of the timetabling problem are statistical measures resulting from the conflict matrix. The density mentioned earlier provides an average degree, with the largest degree giving us a maximum. In order to distinguish between two very different problems of equal conflict matrix density we need to consider statistical measures such as the variation in degree from the average and the degree of the exam at different percentiles[5]. We could also consider, as a variation, the number of exams or the percentage of the total exams whose degree is within a given percentage of the largest degree. As with all areas of statistical analysis, there are a potentially infinite number of different statistical measures we could take based on the structure of the conflict matrix which could each provide some useful information relevent to measuring the similarity between two problems. For our work, we will concentrate on just a small number of these to be fed into the knowledge discovery process.

All of the above statistical measures can, of course, be combined as a ratio with our other features and of particular note would be those ratios to the total number of periods in the timetable. Also, a relatively simple measure which could provide useful knowledge is the percentage of the total exams whos degree is strictly less than the number of periods. Depending on the neighbourhood used, exams which clash with another exam in every period of the timetable may be unable to move within the local search process if the search is conducted within the set of feasible solutions only. For instance, many techniques successfully utilise the most simple move neighbourhood which selects a single exam and

[5] with exams ordered in decreasing order of degree

moves it to a new period of the timetable, selected either at random or by some deterministic method. Using this neighbourhood, only those exams which do not have a clash in every other period of the timetable can be moved which can lead to a large amount of disconnectivity in the search space. On the other hand, using a neighbourhood such as the Kempe Chain neighbourhood employed by Thompson & Dowsland [27] and Casey & Thompson [16] ensures that every exam within the timetable can move to any other timeslot. In doing so, a series of other exams will also often have two be exchanged between the two periods in question. This is investigated further in the following paragraph.

Fluidity Analysis In table 4, we present an analysis of the fluidity of the benchmark problems studied when optimised using Simulated Annealing with the simplest move neighbourhood[6] over 100 runs, each with a different initial solution. The initial temperature and cooling schedule were set extremely high and slow respectively for these experiments since our aim was to examine how many exams within the timetable never moved from their initial position as given by our largest degree construction heuristic. With such a high temperature and slow cooling schedule, we can say with a relatively high degree of certainty that any exam which is capable of moving within the neighbourhood used would do so at some point over the course of the 100 seperate runs. Of course, there will be exams which may have a small window of opportunity to move in this neighbourhood, when a period briefly becomes available that they can move to, but given the random nature of the move selection, the exam wasn't selected to be moved during this window. However, such exams will be very few across 100 runs of the algorithm. Our main objective was to examine how different the data sets are with respect to fluidity for this commonly used neighbourhood.

Table 4. Percentage of total number of exams which never move in x runs out of 100 of Simulated Annealing using the simple move neighbourhood

Data	No. of runs in which x% of exams never moved					
Set	100	75-99	50-74	25-49	1-24	0
CAR-S-91	1.17%	3.96%	3.96%	3.81%	10.12%	76.98%
CAR-F-92	3.31%	3.31%	2.76%	3.50%	9.39%	77.72%
EAR-F-83	0.55%	6.63%	0.55%	2.21%	10.50%	79.56%
HEC-S-92	0.00%	1.23%	3.70%	2.47%	**50.62%**	41.98%
KFU-S-93	1.65%	2.88%	1.44%	3.70%	4.53%	85.80%
LSE-F-91	1.57%	4.99%	1.57%	1.31%	3.15%	87.40%
STA-F-83	**37.41%**	0.00%	0.00%	3.60%	1.44%	57.55%
TRE-S-92	0.77%	2.68%	0.38%	1.15%	7.66%	87.36%
UTA-S-92	3.05%	5.31%	3.05%	1.93%	7.07%	79.58%
UTE-S-92	6.52%	0.54%	1.63%	1.09%	5.43%	84.78%
YOR-F-83	0.53%	1.58%	0.53%	0.00%	8.42%	88.95%

[6] Move a single exam to a new timeslot whilst maintaining solution feasibility

It is clear that in the case of exams which never move throughout the local search process, their positioning in the initial solution is crucial to the quality of the final solution. For the majority of data sets in table 4, this percentage of exams is relatively small and generally below $\sim 3\%$. There are also a relatively small percentage of exams which fail to move in > 75 of the 100 runs. A higher percentage are imobile in < 25 runs, but in most cases $\sim 80\%$ of the exams in the data sets move at least once in every one of the 100 runs. Of course, our analysis does not tell us whether many of the exams moved just once during the local search or whether they moved hundreds of times, but in this analysis we are mostly interested in the boolean variable of whether an exam moved at all or not. Whilst 9 out of the 11 benchmark data sets presented share fairly similar fluidity analysis which doesn't add much to our similarity measurement, 2 of the data sets exhibit very different behaviour.

Standing out most obviously are the $\sim 37\%$ of exams in the STA-F-83 data set which never move across any of the 100 runs. Also noteable is that there are 0 exams in the 50-99 group and very few in the 1-49 group, indicating that if an exam can move at all in the STA-F-83 data set within this simple move neighbourhood, it will tend to do so in the vast majority of runs. Having over one third of its exams imobile relative to this simple move neighbourhood provides some important clues to the anomalous nature often displayed by this data set. The reliance on the initial solution becomes massive with so many exams being set in the positions they are originally placed in. Coupled with the fact that each student takes an average of 9-10 exams and these are spread across just 13 time slots, it is easy to see why this data set yields a very high penalty cost for all feasible solutions. Those exams which are imobile will, in general, be the ones with the most clashes and which therefore add the most penalty to the timetable. With this in mind, one of the reasons we believe Casey & Thompson's GRASP technique [16] is so successful on this problem relative to other techniques is the implementation of the Kempe Chain based neighbourhoods. As discussed earlier, the Kempe Chain neighbourhood allows any exam in the timetable to be moved to any other timeslot whilst always yielding a feasible solution. Using the simple neighbourhood, if the exam, e in timeslot $t_1$to be moved clashes with an exam in the chosen slot, t_2, it cannot be moved there and a new move must be selected. Kempe chains get around this problem by moving all those clashing exams from t_2 across to t_1. Any further clashes induced by this are resolved by moving the clashing exams across with the orginal exam e to timeslot t_2, with this process continuing until the two periods are conflict-free. Due to the fact that all periods are conflict-free before the first exam is moved, there will always exist a feasible resolution to the Kempe Chains; in the worst case scenario this would involve swapping all exams in t_1 with all exams in t_2.

The other data set of note is HEC-S-92 whose behaviour is perhaps even more interesting than that of STA-F-83 and also less easily understandable. Contrary to the other 10 data sets, HEC-S-92 does not have a single exam which never moves across 100 runs of Simulated Annealing from a different initialisation each time. The % of exams which fail to move in 25-99 runs is also very low, yet over half the exams (41) in the data set fail to move in 1-25 of the 100 runs. A deeper analysis of this behaviour concerning how much overlap there is in the runs during which these $> 50\%$ of exams do not move would be required to draw any firm conclusions on this behaviour, but it is worthy of note as being vastly different from all other data sets. This data set is also the only one which our Greedy Largest Degree with Backtracking initialisation technique fails to find a feasible initial solution to. This is due to the fact that our backtracking only searches 2 levels deep before giving up and restarting, but on this data set it always reaches the same irresolvable point. This is probably due to a combination of the high conflict matrix density and also this unusual behaviour indicated by our fluidity analysis. What this fluidity analysis seems to show is that whilst all exams in the data set can move around over the course of 100 runs from random initialisations, the actual fluidity of the data set from any given initialisation is not so high with a fair number of exams being fixed by their relative positions to other exams. Again, this behaviour can be attributed to the fact that the conflict graph is very dense. An analysis of cliques within the problem as discussed in the following subsection may also shed some light on this behaviour.

From this analysis it would seem that the fluidity of a given data set with respect to the neighbourhood used in a meta-heuristic can prove crucial to how successful the meta-heuristic will be. From the point of view of our similarity measure, this is an area which could prove very important with further analysis to be carried out.

Cliques One of the most significant features of exam timetabling problems when modelled as a graph is that there tend to exist large cliques and near-cliques, unlike the structure of a typical random graph in which any two nodes have an equal probability of being connected by an edge. In exam timetabling, many of the edges which contribute to the conflict matrix are clustered together representing exams which form part of a particular discipline. Students taking science-based subjects will generally take very few, if any, humanities exams, but will take a large number of science exams meaning that the density of clashes between science exams will be far greater than between science exams and humanities exams. Carter & Johnson [15] investigate the cliques to be found within the benchmark problems considered in this paper as well as looking at near-maximum cliques. An investigation of cliques gives another angle to measure similarity from based on the conflict matrix again. As well as considering the size of the maximum clique in each problem graph, Carter & Johnson investigate how many cliques there are of max size and also (max-1) size. As a problem feature, the number of maximum size cliques could provide invaluable information for measuring similarity between problems by giving a much more in-depth view of the structure of the conflict matrix.

The two largest data sets (CAR-S-91 and UTA-S-92) are found to have over 100 maximum size cliques whilst the majority of data sets have fewer than 5. Again the STA-F-83 data set exhibits very different behaviour to the other data sets, being the second smallest measured by exam size, but having 60 cliques of size 13, which is also the number of periods used to construct the timetable. Carter & Johnson also calculate the number of nodes occuring in all max cliques and in any max clique together with an analysis of the complement graphs. When considering cliques of of size (max-1), the number of these is significantly larger than max-size cliques in the majority of problems. The authors move on to consider Quasi-cliques, where all nodes in a quasi-clique, Q_k, have at most k missing edges from a true clique. Again these represent very dense areas of the conflict graph which have a far larger impact on the difficulty of the problem than the much less dense areas which balance these out to give the overall conflict matrix density.

There is a large amount of analysis which can potentially be done on cliques, with Carter & Johnson's work [15] providing a crucial backbone to this. How much of the clique analysis could be used as part of our similarity measure remains to be seen however, since finding the largest clique in a graph of size n is in itself an NP-hard problem. As such, it may not be feasible for our CBR system to calculate the required feature data for a new problem in order to compare with those in the case base. Having said that, cliques clearly form the basis of the core problem definition in most exam timetabling problems so cannot be ignored.

Side constraints & the Objective function So far we have considered features which are common to the core exam timetabling problem where the only hard constraints are that every exam must be scheduled to exactly one timeslot within the timetable and no two exams with students in common can be scheduled in the same time slot. The only soft constraint so far considered is that of spreading clashing exams around the timetable using the proxinity cost given by Carter et. al [13]. In reality, real world problems requiring to be solved will have a number of other constraints, both hard and soft. The hard constraints will determine the feasible solutions space, whilst the soft constraints will give a measure of how good the timetable is, either by being combined in a weighted single objective function or by forming a pareto front in a multi-objective optimisation.

When attempting to measure similarity between exam timetabling problems, it is required that the problems being compared have the same constraints. Attempting to compare two problems, one in which the number of periods in the timetable is fixed and one in which it is a variable to be minimised is clearly not sensible. For this reason, we need within the case-base a large variety of problems with different hard and soft constraints to give the system as wide applicability as possible. This can be done potentially by adding in constraints to the core problems and forming new cases with these additional constraints. In this way, any core problem[7] can be compared to any other core problem by adding in suitable side-constraints. Clearly there are a number of issues with this as regards how many constraints to add and how wide a set of constraints to include within the case base, but they are beyond the scope of this paper.

[7] The core problem being with the hard constraints we have so far considered

The author's previous work investigating the effect of the objective function on potential similarity measures [10] showed that even when the same soft constraints are employed within the problem, using a different set of weights on the constraints can have an effect on the performance of a given heuristic applied to the problem. This is to be expected since the objective function defines the height of the problem landscape at every point, therefore using different weights on the same set of constraints could change the structure of the landscape significantly causing a meta-heuristic which may previously have traversed the landscape very effectively to now get stuck more in local optima and provide a less high quality result.

From this analysis, it would seem that the matching process employed within our CBR system can only compare two problems whose definitions are the same regarding the hard and soft constraints included as well as the weightings applied to those soft constraints. It may be possible to adapt problems within the case base to use the weightings of a new problem given to the system with the same constraints, but this would likely require too much computational time for the system.

3 Conclusions and future work

In this paper we have discussed the needs and requirements of an automated timetabling system together with some of the many issues involved in using a fully or semi-automated system as oppose to a human timetabler. We consider the methods with which timetables are constructed by human timetablers and the areas in which automation could improve the process. One such area would be to store a case-base of exam timetabling problems together with the heuristic and/or meta-heuristic technique(s) successfully applied to give the best results amongst those tested. With a large base of past knowledge included within the case base, a timetabler could present a new timetabling problem to the system, which would then suggest the best technique to apply based on past experience. This would be done by matching the new timetabling problem with those in the case base to find the most similar case.

The biggest issue in such a system is how to measure the similarity between two exam timetabling problems in a meaningful way such that the general CBR assumption that similar problems have similar solutions holds true. In our case, this assumption would be that similar problems can be solved equally successfully by the same technique. Such a system could provide huge benefits by storing the information used in a large number of different problems so that new problems can be solved effectively without having to construct a new technique specifically for the problem. Our aim is not to beat the results produced by the more problem specific heuristics, but to provide a higher level of generality than would otherwise be possible to ensure that a suitable technique will be chosen to solve a new problem.

We carried out an investigation of a number of key features of exam time/-tabling problems, assessing their importance to the problem solving method and also how they can be combined to provide much better features by which to provide similarity. A knowledge discovery process will be applied, which will select from a large number of features (including ratios of all pairs of features included in the system), those which provide the best measure of similarity between exam timetabling problems. It has been shown [11] that between 3 and 7 features tend to give the best performance for a similarity measure and it is thought likely that most of these features will be ratios of the more simple features presented in this paper, some of which may not have been considered important, but which may be found to provide crucial knowledge of the problems studied.

Future work on the similarity measure may include investigating the use of Fuzzy Sets for indexing and retrieval of cases within CBR [19]. This technique has numerous advantages, allowing numerical features to be simplified into a number of fuzzy sets for comparison as well as making it possible to index a case multiple times on a given feature with different degrees of membership. The use of fuzzy sets would allow more flexibility in the comparison of features with matches being suggested which might not otherwise have come up as the most similar case.

References

1. E. K. Burke, D. G. Elliman, P. H. Ford & R. F. Weare: Examination Timetabling in British Universities: a Survey. In [21], pages 76-90, 1996.
2. E. K. Burke, J. P. Newall & R. F. Weare: A Memetic Algorithm for University Exam Timetabling. In [21], pages 241-250, 1996.
3. E. K. Burke, J. P. Newall & R. F. Weare: Initialization Strategies and Diversity in Evolutionary Timetabling. Evolutionary Computation 6(1), pages 81-103, 1998.
4. E. K. Burke & J. P. Newall: A Multi-stage Evolutionary Algorithm for the Timetable Problem. IEEE Transactions on Evolutionary Computation, 3(1), pages 63-74, 1999
5. E. K. Burke, B. MacCarthy, S. Petrovic & R. Qu: Structured Cases in Case-Based Reasoning - Re-using and Adapting Cases for Time-tabling Problems. Knowledge-Based Systems, 13: 2-3 (2000) pages 159-165. Also in proceedings of ES'99 as one of the best papers.
6. E. K. Burke, B. MacCarthy, S. Petrovic & R. Qu: Case-based Reasoning in Course Timetabling: An Attribute Graph Approach. Case-Based Reasoning Research and Development, Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR-2001). Vancouver, Jul-Aug, Canada, 2001. LNAI 2080. pages 90-104
7. E. K. Burke & J. P. Newall: Enhancing Timetable Solutions with Local Search Methods. In [24], pages 195-206, 2003
8. E. K. Burke, S. Petrovic & R. Qu: Case-Based Heuristic Selection for Examination Timetabling. Proceedings of the SEAL'02 conference, Singapore, pages 277-281, 2002.
9. E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic & R. Qu: Similarity Measures for Exam Timetabling Problems. 1st Multidisciplinary Intl. Conf. on Scheduling: Theory and Applications (MISTA 2003), Vol. 1, pages 120-136, ISBN0-9545821-0-1, Nottingham, UK, Aug 13-16, 2003.

10. E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic and R. Qu: Using Simulated Annealing to Study Behaviour of Various Exam Timetabling Data Sets. Proceedings of the Fifth Metaheuristics International Conference (MIC 2003), paper MIC03 09, Kyoto Japan, Aug 2003.
11. E. K. Burke, S. Petrovic & R. Qu: Case-based Heuristic Selection for Timetabling Problems. University of Nottingham Technical Report NOTTCS-TR-2004-2, accepted for publication in the Journal of Scheduling, 2004.
12. M. Caramia, P. Dell'Olmo & G. F. Italiano: New Algorithms for Examination Timetabling. In S. Näher & D. Wagner (eds.): Algorithm Engineering 4th Int. Workshop, Proc. WAE 2000 (Saarbrücken, Germany). Lecture Notes in Computer Science, Vol 1982. Springer-Verlag, Berlin Heidelberg New York. pages 230-241, 2001.
13. M. W. Carter, G. Laporte & J. W. Chinneck: A General Examination Scheduling System. Interfaces 24, pages 109-120, 1994.
14. M. W. Carter, G. Laporte & S. Y. Lee: Examination Timetabling: Algorithmic Strategies and Applications. Journal of the Operational Research Society 47, No. 3, pages 373-383, 1996.
15. M. W. Carter & D. G. Johnson: Extended clique initialisation in examination timetabling. Journal of the Operational Research Society (2001) 52, pages 538-544.
16. S. Casey & J. Thompson: GRASPing the Examination Scheduling Problem. In [24], pages 232-244, 2003.
17. T. B. Cooper & J. H. Kingston: The complexity of timetable construction problems. In [21], pages 283-295, 1996.
18. E. Foxley & K. Lockyer: The Construction of Examination Timetables by Computer. The Computer Journal 11, pages 264-268, 1968.
19. B. C. Jeng & T-P. Liang: Fuzzy Indexing and Retrieval in Case-Based Systems. Expert Systems With Applications, Vol 8, No. 1, pages 135-142, 1995.
20. L. T. G. Merlot, N. Boland, B. D. Hughes & P. J. Stuckey: A Hybrid Algorithm for the Examination Timetabling Problem. In [24], pages 207-231, 2003.
21. E. K. Burke & P. Ross (eds): Practice and Theory of Automated Timetabling. Volume 1153 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 1996.
22. E. K. Burke & M. W. Carter (eds): Practice and Theory of Automated Timetabling. Volume 1408 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 1998.
23. E. K. Burke & W. Erben (eds): Practice and Theory of Automated Timetabling. Volume 2079 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 2001.
24. E. K. Burke & P. De Causmaecker (eds): Practice and Theory of Automated Timetabling. Volume 2740 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 2003.
25. L. Di Gaspero: Recolour, Shake and Kick: A recipe for the Examination Timetabling Problem. In: E. K. Burke & P. De Causmaecker (eds.): Proceedings of the Fourth International Conference on the Practice and Theory of Automated Timetabling, Gent, Belgium, August 2002, pages 404–407.
26. L. Di Gaspero & A. Schaerf: Tabu Search Techniques for Examination Timetabling. In [23], pages 104-117, 2001.
27. J. Thompson & K. Dowsland: Variants of Simulated Annealing for the Examination Timetabling Problem. Ann. Oper. Res. 63, pages 105-128, 1996.

28. J. Thompson & K. Dowsland: A Robust Simulated Annealing Based Examination Timetabling System. Comput. Oper. Res. 25, pages 637-648, 1998.
29. G. M. White & B. S. Xie: Examination Timetables and Tabu Search with Longer-term Memory. In [23], pages 85-103, 2001.