An Adaptive Tie Breaking and Hybridisation Hyper-Heuristic for Exam Timetabling Problems

Burke, E.K. and Qu, R. and Soghier, A.

Abstract Graph colouring heuristics have long been applied successfully to the exam timetabling problem. Despite the success of a few heuristic ordering criteria developed in the literature, the approaches lack the ability to handle the situations where ties occur. In this paper, we investigate the effectiveness of applying tie breakers to orderings used in graph colouring heuristics. We propose an approach to construct solutions for our problem after defining which heuristics to combine and the amount of each heuristic to be used in the orderings. Heuristic sequences are then adapted to help guide the search to find better quality solutions. We have tested the approach on the Toronto benchmark problems and are able to obtain results which are within the range of the best reported in the literature. In addition, to test the generality of our approach we introduced an exam timetabling instance generator and a new benchmark data set which has a similar format to the Toronto benchmark. The instances generated vary in size and conflict density. The publication of this problem data to the research community is aimed to provide researchers with a data set which covers a full range of conflict densities. Furthermore, it is possible using the instance generator to create random data sets with different characteristics to test the performance of approaches which rely on problem characteristics. We present the first results for the benchmark and the results obtained show that the approach is adaptive to all the problem instances that we address. We also encourage the use of the data set and generator to produce tailored instances and to investigate various methods on them.

Burke,E.K.
Automated Scheduling, Optimisation and Planning (ASAP) Group School of CSIT, University of Nottingham, Nottingham, NG8 1BB, U.K., e-mail: ekb@cs.nott.ac.uk

Qu, R.
Automated Scheduling, Optimisation and Planning (ASAP) Group School of CSIT, University of Nottingham, Nottingham, NG8 1BB, U.K., e-mail: rxq@cs.nott.ac.uk

Soghier, A.
Automated Scheduling, Optimisation and Planning (ASAP) Group School of CSIT, University of Nottingham, Nottingham, NG8 1BB, U.K., e-mail: azs@cs.nott.ac.uk

1 Introduction

The Operational Research and Artificial Intelligence research communities have been addressing timetabling problems and research issues since the 1960s [8]. Exam timetabling has been particularly well studied. Survey papers and overviews of different approaches developed are presented in [5, 8, 17]. In [5], the importance of graph colouring methods when applied to timetabling problems is highlighted. A survey which includes the different approaches developed in exam timetabling from 1986 to 1996 is presented in [8]. Furthermore, a recent survey is presented in [17] summarising the recent approaches developed in the past decade.

An exam timetabling problem involves assigning a number of exams to a certain number of timeslots taking into account different constraints, some of which have to be satisfied (hard constraints) and some of which we would like to satisfy if possible (soft constraints). A solution without hard constraint violations is called a feasible solution. The quality of a feasible solution is determined by measuring the level of soft constraint violation.

A wide variety of search methods have been investigated for exam timetabling over the years. Examples include Tabu Search (e.g. [11, 20], Simulated Annealing (e.g. [12, 19]), Evolutionary Algorithms (e.g. [10, 15, 21]), Case Based Reasoning (e.g. [6]) and Fuzzy Methodologies (e.g. [1]). In particular, graph colouring heuristics have been widely used to solve exam timetabling problems by underpinning constructive heuristic solution methods (e.g. [5, 9]).

Many of the approaches mentioned above involve the fine-tuning of parameters and so may not work as effectively on different problems and even different instances. This observation has motivated the development of hyper-heuristics [4, 18]. One of the goals of hyper-heuristics is to increase the level of generality at which search methods can automatically solve a range of problems. A hyper-heuristic can choose the appropriate heuristics to be applied to the problem in hand. Hence, it is concerned with the exploration of a search space of heuristics rather than directly acting upon a problem to find a solution. In the literature, several hyper-heuristic techniques have been developed to solve exam timetabling problems (e.g. [7, 13]).

Graph heuristics have been hybridised in an iterative adaptive approach developed in [3]. In each iteration, the exams are ordered using different graph heuristics and the exam to be scheduled is chosen. In addition, the exams that are found to be difficult to schedule in previous iterations are adaptively moved forward. Another approach has been developed in [16] where different graph heuristics are adaptively hybridised. Different ordering strategies in graph heuristics are adaptively called at a higher level. However, a random exam is selected in the situation where a tie appears. Hence, instead of choosing these exams randomly, tie breaking is a way to help guide a search in scenarios where two or more exams have the same weight in a specific ordering.

In this paper, we investigate the effectiveness of applying tie breakers to heuristic orderings. Furthermore, we present an adaptive approach where tie breakers are applied to the Saturation Degree heuristic followed by a hybridisation with another heuristic to solve exam timetabling problems. In addition, we develop an exam

timetabling instance generator to test the adaptiveness of our approach on randomly generated data.

The remainder of this paper is structured as follows. Section 2 gives a short description of graph heuristics and previously developed approaches for exam timetabling problems. Section 3 presents the Toronto benchmark and the instance generator we developed to test our approach. In section 4, we present an investigation on the effectiveness of hybridisations and tie breaking in graph heuristics. We describe our approach and discuss our results in section 5. In section 6 we give some concluding remarks.

2 The Graph based Hyper-heuristic (GHH)

Graph heuristics are simple constructive approaches which have been widely applied in exam timetabling. Exam timetabling problems with only hard constraints can be represented as graph colouring models. Vertices in the graph represent exams in the problem, and edges represent the conflicts between exams. Colours correspond to time slots. Hence, if the exam timetabling problem is considered as a graph colouring problem, the aim is to find the minimum number of time slots which are able to accommodate all the exams without any clashes. Several heuristics (e.g. [2]) have been developed to solve graph colouring problems.

By analysing the student enrolment list, the exams that are in conflict (exams that have at least one common student) can be identified. The graph heuristics are usually used to order exams that are not yet scheduled according to different difficulty criteria. For example, by using Largest Enrolment (see Table 1), the exams are ordered in a descending order according to the number of students taking them and then they are scheduled one by one to construct a timetable. The objective of creating such orderings is to deal with the most difficult exams in the problem first, to construct good quality timetables.

Hyper-heuristic research is concerned with building systems which can automatically guide and adapt a search according to the structure of the problem in hand. This can be achieved through the exploitation of the structure of a problem, and the creation of new heuristics for that problem, or intelligently choosing from a set of pre-defined heuristics. In other words, hyper-heuristic research aims to automate the heuristic design process through automating the decision of which heuristics to employ to explore the solution space for a new problem. The aim here is to automate the heuristic design process and produce optimisation tools available to stakeholders who require quick and cheap solutions for their optimisation problems.

In our previous work [7], tabu search was used to search a set of heuristic sequences (rather than the solutions themselves) which consisted of the first five graph heuristics presented in Table 1 and a random ordering strategy. In every step of tabu search, a sequence of these heuristics is used to construct a solution. Therefore, the search space of the tabu search consists of all the possible sequences of the low-level graph heuristics described in Table 1. A tabu search move in this case was to create a

Graph Heuristic	Ordering Criteria
LD (Largest Degree)	Descending order by the number of conflicts each exam has with others (i.e. conflicted exams are the ones that have students in common)
LWD (Largest Weighted Degree)	Similar to LD but is weighted by the number of students involved in the conflict
LE (Largest Enrolment)	Descending order of the number of students taking each exam
SD (Saturation Degree)	Ascending order of the number of remaining valid timeslots to assign an exam without causing conflicts
CD (Colour Degree)	Descending order of the number of conflicts each exam has with the exams already scheduled
LUD (Largest Uncoloured Degree)	Descending order of the number of conflicts each exam has with the unscheduled exams
LUWD (Largest Uncoloured Weighted Degree)	Similar to LUD but is weighted by the number of students involved in the conflict

Table 1 Ordering Criteria in Graph Heuristics for Exam Timetabling

new sequence by changing two heuristics in the previous heuristic list. A parameter was used to decide the number of exams scheduled in every iteration of the solution construction. The list is then applied to the problem in hand and the solution obtained is evaluated. If a better solution is obtained it is saved and the heuristic list is stored in the tabu list or otherwise it is discarded. The process continues for a number of pre-defined iterations depending on the problem size. The exams are scheduled to the first timeslot that leads to the least cost. Figure 1 presents the pseudocode of the approach. This approach and another approach developed in [16] can adaptively search and hybridise different low-level heuristics.

From these previously developed approaches, it was observed that Saturation Degree performs the best in most cases. The saturation degree of each unscheduled exam is calculated in each iteration and the exams are ordered according to the number of remaining timeslots where an exam can be scheduled without violating any hard constraint. However, all the exams have the same saturation degree at the beginning of the solution construction as the timetable is empty. Ties could also occur anytime during the construction process when the same number of timeslots is available for two exams. In this work, we analyse the effectiveness of applying different tie breakers to the Saturation Degree heuristic (section 4). In addition, we develop an intelligent adaptive approach (section 5) which determines the heuristic to use in a hybridisation with Saturation Degree and the amount of hybridisation required to achieve the best solutions. First, we describe the data we used to test our approach in the next section.

```
Create a heuristic sequence h1 = {h1,h2,h3……hk}  //k: number of exams/exams scheduled by each heuristic
//Tabu Search on Heuristic Sequences
for i = 1 to i = number of iterations
        h =  change two heuristics from the sequence h1
        if h is not in the tabu list
                Construct a solution using h
                if a feasible solution (sc) is obtained & sc < s  //s: best solution so far
                        save the best solution, s = sc
                        update the tabu list
                        h1 = h
        //end if
//end of Tabu Search
Output the best solution s
```

Fig. 1 The pseudo-code of the Tabu Search graph based hyper-heuristic [7]

3 Benchmark Exam Timetabling Problems

3.1 The Toronto Benchmark Exam Timetabling Problem

This data set is widely used in exam timetabling research by different state-of-the-art approaches and can therefore be considered as a benchmark [9]. It consists of 13 real-world problems from 3 Canadian high schools, 5 Canadian, 1 American, 1 UK and 1 mid-east universities. The problem has one hard constraint where conflicting exams cannot be assigned to the same timeslot. In addition, a soft constraint is present where conflicting exams should be spread throughout the timetable as far as possible from each other. The goal here is to minimise the sum of proximity costs given as follows:

$$\sum_{i=0}^{4}(w_i \times N)/S$$

where

- $w_i = 2^{4-i}$ is the cost of assigning two exams with i time slots apart. Only exams with common students and are four or less time slots apart cause violations
- N is the number of students involved in the conflict
- S is the total number of students in the problem

The characteristics of the problem instances used are presented in Table 2. Conflict density represents the ratio between the number of exams in conflict to the total number of exams.

Problem	Exams I/II	Students I/II	Enrolments I/II	Density	Timeslots
car91	682	16925	56877	0.13	35
car92	543	18419	55522	0.14	32
ear83 I	190	1125	8109	0.27	24
ear83 II	189	1108	8057	0.27	24
hec92 I	81	2823	10632	0.42	18
hec92 II	80	2823	10625	0.42	18
kfu93	461	5349	25113	0.06	20
lse91	381	2726	10918	0.06	18
sta83 I	139	611	5751	0.14	13
sta83 II	138	549	5417	0.19	35
tre92	261	4360	14901	0.18	23
uta92 I	622	21266	58979	0.13	35
uta92 II	638	21329	59144	0.12	35
ute92	184	2750	11793	0.08	10
yor83 I	181	941	6034	0.29	21
yor83 II	180	919	6002	0.3	21

Table 2 Characteristics of the Toronto Benchmark data set [17]

3.2 The Exam Timetabling Instance Generator

Different exam timetabling benchmark problems were collected and widely studied over the years due to the high-level of research interest in this area [17]. The data was collected from real world problems present in several institutions over the world. The benchmarks provide a means of comparison and evaluation to the different approaches developed. In addition to the Totonto benchmark described in the previous section, the international timetabling competition (ITC2007) exam timetabling track was recently introduced [14]. It is a data set which is more complex as it contains a larger number of constraints. As the focus of this work is on tiebreaking and hybridisations, the approach developed in this paper was not applied to the ITC2007 data set which requires different heuristics to schedule exams in rooms following certain constraints.

Since the Toronto benchmark is collected from real-world problems present in several institutions, they do not provide a full coverage of all possible problem scenarios. For example, the benchmark data set does not contain instances with conflict density in the range from 19%-27% and 29%-42% (see Table 2).

In this paper, we introduce an exam timetabling instance generator and a benchmark exam timetabling problem data set that consists of 18 different problem instances. The 18 problems consist of 9 small and 9 large problems. To be more precise, a problem is considered small if it does not contain more than 100 exams while a large problem exceeds 500 exams. The problems are generated with conflict density values starting from 6% to 47% using 5% intervals. The number of students and their enrolments are variables according to the problem size and conflict density.

The problems have similar constraints and use the same objective function as the Toronto benchmark data set stated in the previous section.

Table 3 presents the characteristics of the 18 problem instances in the data set. They vary in several characteristics, not only with respect to the number of exams (ranging from 80 - 567), the conflict density (ranging from 6%-47%), but also the number of enrolments (ranging from 194 - 60168) and the number of timeslots to assign the exams (ranging from 15-70). There are also different numbers of students (ranging from 66 - 15326) across different instances.

The data set is available at http://www.asap.cs.nott.ac.uk/resources/data.shtml where each problem consists of 3 text files. The file names describe the size (SP for small problem & LP for large problem) and conflict density of the problem instances. The representation is extendable to add new characteristics. At the same website, we also provide a solution evaluator to evaluate the quality of the timetables produced. In addition, the website presents the instance generator we developed to produce the data set.

We investigate the approach developed in this paper by applying it to the Toronto benchmark exam timetabling problems and the data set we generated in Table 3. Our approach is tested on the newly developed data set to highlight its generality on problems with a full range of conflict density. There is a large set of problem instances in the existing literature. However, none concerns the generation of a full range of problems for testing approaches which are dependent on problem characteristics.

Problem	Exams	Students	Enrolments	Density	Timeslots
SP5	80	66	194	0.07	15
SP10	100	100	359	0.11	15
SP15	80	81	314	0.17	15
SP20	80	83	344	0.19	15
SP25	80	119	503	0.26	15
SP30	80	126	577	0.32	15
SP35	100	145	811	0.36	19
SP40	81	168	798	0.42	19
SP45	80	180	901	0.47	19
LP5	526	1643	5840	0.06	20
LP10	511	2838	10659	0.12	20
LP15	508	3683	14026	0.16	24
LP20	533	5496	21148	0.21	30
LP25	542	7275	27948	0.26	35
LP30	550	8798	34502	0.31	35
LP35	524	9973	38839	0.37	50
LP40	513	10826	42201	0.41	60
LP45	567	15326	60168	0.46	70

Table 3 Characteristics of the Benchmark data set produced by our problem instance generator

4 Hybridising and Tie Breaking Graph Heuristics using the Conflict Density of Exam Timetabling Problems

As previously stated, many ties can appear in a Saturation Degree ordering of exams during solution construction. In previous work [3, 16], an exam is randomly chosen in such situations. The random choice in the earlier stages of the search has a great effect on the quality of the solutions produced. Therefore, a means of breaking these ties is essential. We initially developed a method of breaking ties in Saturation Degree using other graph heuristics. The Saturation Degree after breaking the ties is then hybridised with another heuristic in a sequence where random heuristic sequences with different percentages of hybridisation are generated. Each heuristic in the sequence is used to schedule a single exam. This technique was applied to four instances (HEC92 I, STA83 I, YOR83 I and CAR91 I) of the benchmark exam timetabling problems described in section 3.1 for off-line learning of the best tie breakers and heuristic hybridisations leading to the best solutions for problems with different conflict densities. These instances were chosen as they vary in size and cover a range of conflict densities. The effect of tie breaking and hybridising different low-level heuristics on the quality of the solutions produced is analysed. Figure 2 presents the pseudocode of the random graph heuristic sequence generator. The approach starts by constructing different heuristic sequences which consist of two graph heuristics (Saturation Degree and one of the other heuristics described in Table 1 using different pre-defined percentages of hybridisation). The sequences are then applied to the instances to construct a solution. For each percentage of hybridisation, 50 heuristic sequences are generated using different seeds to construct solutions. Only the sequences that produce feasible solutions are saved and the rest are discarded (see Figure 2). According to the quality of the solutions obtained for each instance, the best tie breaker and low-level heuristic for hybridisation are identified. Further analysis is performed to find the relation of conflict density with the low-level heuristics used and the amount of hybridisation required to guide the search to the best solutions.

```
Set h1 as the first graph heuristic to be used (SD, SD tb LWD, SD tb LUWD, SD tb LD)
Set h2 as the second graph heuristic to hybridise with h1 (LWD, LD, LUWD, LUD, CD)
for i = 0 to i = 1              // i : Percentage of hybridisation
{
    for n = 1 to n = 50
    {
        Initialise the heuristic sequence h = {h1 h1 ……h1}
        h = randomly change (i x size of h) heuristics in h from h1 to h2
        Construct a solution c using h
        if solution c is feasible
        {
            Calculate the penalty of the solution and save h
        }
        else
        {
            Discard the sequence h
        }
        n = n + 1
    }
    i = i + 0.05
}
```

Fig. 2 The pseudo-code of the random graph heuristic sequence generator using a tie breaker for Saturation Degree (tb:tie-breaker)

In our previous work and work undertaken in [3, 7, 9], it was shown that using Saturation Degree on its own usually outperforms other graph heuristics and results in a better outcome. Therefore, we investigated applying Saturation Degree without breaking ties as the primary heuristic in a sequence generator in one set of experiments. In addition, we applied Saturation Degree while breaking the ties using some other heuristic orderings such as LWD, LD and LE in another set of experiments. Finally, the effect of hybridising Saturation Degree with different percentages of another graph heuristic (i.e. LWD, LD, LUWD, LUD, CD or LE) in both experiments was analysed.

4.1 Analysis of Saturation Degree Tie Breakers

Several heuristic orderings can be used to break ties that occur in the Saturation Degree list during solution construction. An illustrative example of breaking ties in a Saturation Degree list is shown in Figure 3. We assume that the exams are ordered according to their LWD and Saturation Degree as shown. We also assume that the saturation degree is the same for the first 3 exams (i.e. e1, e2 and e3) in the list. In this case, LWD is used to break the ties. The tied exams are re-ordered according to their position in the LWD list. Therefore, e2 is scheduled first and removed from both lists. The Saturation Degree list is then re-ordered again in the next iteration of solution construction.

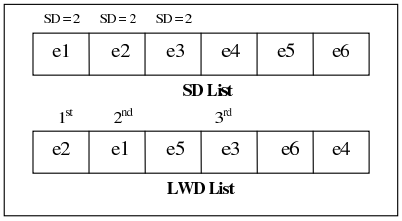

Fig. 3 An illustrative example of breaking Saturation Degree ties

We applied several tie breakers to four instances of the Toronto benchmark exam timetabling problems [9]. This was run for 20 times on each instance with different random seeds. Table 4 presents a comparison between the results obtained using Saturation Degree without breaking ties and Saturation Degree using different tie breakers.

It is observed that a Saturation Degree ordering without any tiebreakers does not produce a feasible solution when applied to HEC92 I and YOR83 I. After breaking the ties in Saturation Degree, a feasible solution could be obtained for all instances. The results also show that LWD is overall the best tie breaker for Saturation Degree as the exams are ordered according to their largest weighted degree when they have the same saturation degree.

A t-test is also carried out to give an indication if the results using the three different tie breakers are significantly different. Tables 5, 6 and 7 summarise the p-values of the t-tests carried out between the different tie breakers results. It can be seen that the results between the different tiebreakers are significantly different in all the cases except for hec92 I and sta83 I where LD and CD have no significant difference.

	hec92 I	yor83 I	sta83 I	tre92
SD without breaking ties	-	-	178.24	9.68
SD tb LWD Best	**13.40**	**43.84**	**166.88**	**9.16**
SD tb LD Best	13.42	44.44	170.20	9.59
SD tb CD Best	13.67	46.78	168.21	9.19

Table 4 Results using SD without tie breakers and with several different tie breakers. A (-) indicates that a feasible solution could not be obtained. The notation X tb Y denotes Y is used to break ties in X

	hec92 I	yor83 I	sta83 I	tre92
p-value	0.03	1.2E-04	0.01	6.71E-06
t Stat	2.14	6.25	2.59	7.05

Table 5 t-test on the results from breaking the ties using LWD and LD

	hec92 I	yor83 I	sta83 I	tre92
p-value	6.6E-03	6.42E-11	0.03	6.4E-06
t Stat	3.30	41.37	1.99	7.08

Table 6 t-test on the results from breaking the ties using LWD and CD

	hec92 I	yor83 I	sta83 I	tre92
p-value	0.47	6.12E-08	0.27	0.02
t Stat	0.07	17.38	0.61	2.41

Table 7 t-test on the results from breaking the ties using LD and CD

4.2 Hybridising Heuristic Sequences after Breaking the SD Ties

To analyse the effect of hybridisations, we decided to hybridise the SD ordering using a LWD tie breaker with different graph heuristics and to apply the sequences to the same four instances of the Toronto benchmark exam timetabling problems. Table 8 presents the results of applying these different hybridisations as well as a comparison against a hybridisation without using any tie breakers. As shown in Table 8, hybridising the tie breaking SD ordering with other graph heuristics produced better results.

For problems where a feasible solution could not be obtained using SD without breaking ties (i.e. HEC92 I and YOR83 I), breaking the ties using LWD and a hybridisation using LD produced the best results. However, for problems where a feasible solution was obtained without handling ties in SD (i.e. STA83 I and TRE92), breaking the ties using LWD and a hybridisation using CD performed better. A possible reason could be the occurrence of many ties in the SD ordering which prevents it from producing a feasible solution. Therefore, when tie breaking the SD ordering and hybridising it with LD, some exams will be ordered according to their number of conflicts with unscheduled exams. Generating a feasible solution using SD without handling ties is an indication that no ties or a very small number of ties occur. Therefore, applying a tie breaker and hybridising it with CD will order some exams according to the conflicts with the exams already scheduled, producing better results than a LD hybridisation in this case.

A t-test is also carried out to give an indication if the results of using different hybridisations are significantly different. Tables 9, 10 and 11 summarise the p-values of the t-tests. It can be seen that the results between the different hybridisations are significantly different in all the cases except for hec92 I and tre92 when comparing hybridising SD tb LWD with LD and CD where no significant difference can be seen.

	hec92 I	yor83 I	sta83 I	tre92 I
SD + LWD Average	13.02	44.59	170.08	9.25
SD + LWD Best	12.20	42.49	164.65	9.02
best % hybridisation	54	18	7	49
SD tb LWD + LD Average	12.60	43.03	168.21	9.03
SD tb LWD + LD Best	**11.89**	**42.16**	168.21	8.94
best % hybridisation	28	24	3	18
SD tb LWD + CD Average	12.69	43.34	163.32	9.00
SD tb LWD + CD Best	12.25	42.61	**159.50**	**8.83**
best % hybridisation	15	9	78	84

Table 8 Results of hybridising SD with other graph heuristics with and without breaking ties. The notation X tb Y denotes Y is used to break ties in X

	hec92 I	yor83 I	sta83 I	tre92
p-value	2.19E-08	2.45E-08	4.83E-04	2.91E-10
t Stat	7.52	7.47	3.70	9.36

Table 9 t-test on the results from hybridising SD with LWD and SD tb LWD with LD

	hec92 I	yor83 I	sta83 I	tre92
p-value	3.54E-04	9.53E-07	1.55E-13	5.26E-10
t Stat	3.82	6.05	13.13	9.09

Table 10 t-test on the results from hybridising SD with LWD and SD tb LWD with CD

	hec92 I	yor83 I	sta83 I	tre92
p-value	0.14	0.003	1.03E-16	0.1
t Stat	1.11	2.98	17.75	1.33

Table 11 t-test on the results from hybridising SD tb LWD with LD and SD tb LWD with CD

4.3 Relating Conflict Density of Exam Timetabling Problems to SD Tie Breaking

By analysing the properties of the four problems and the results obtained using the different tie breakers and hybridisations, a relationship becomes apparent between the conflict density of exams in a problem and the percentage of hybridisations required (see Figure 4). As shown in Table 8, the problems with conflict density of

less than 25% (i.e. less than half of the exams are in conflict) obtained the best results when a hybridisation of less than 50% is used (i.e. more tie breaking SD is used than CD or LD). As the conflict density exceeds 25%, the percentage of the tie breaking SD used increases and the hybridised graph heuristic appears less in the sequences generating the best results. Having a higher conflict density means a higher probability of ties occurring in the SD ordering during the solution construction. Therefore, using SD and breaking the ties proves to be more effective. In contrast, instances with a lower conflict density will have a lower probability of ties occurring in the SD ordering during construction and using a lower percentage of the tie breaking SD is more effective. The conclusions made here using four problems proved to be enough as they were verified later in the paper after running the approach on the rest of the Toronto benchmark instances and the instances randomly generated. The results are shown in section 5.

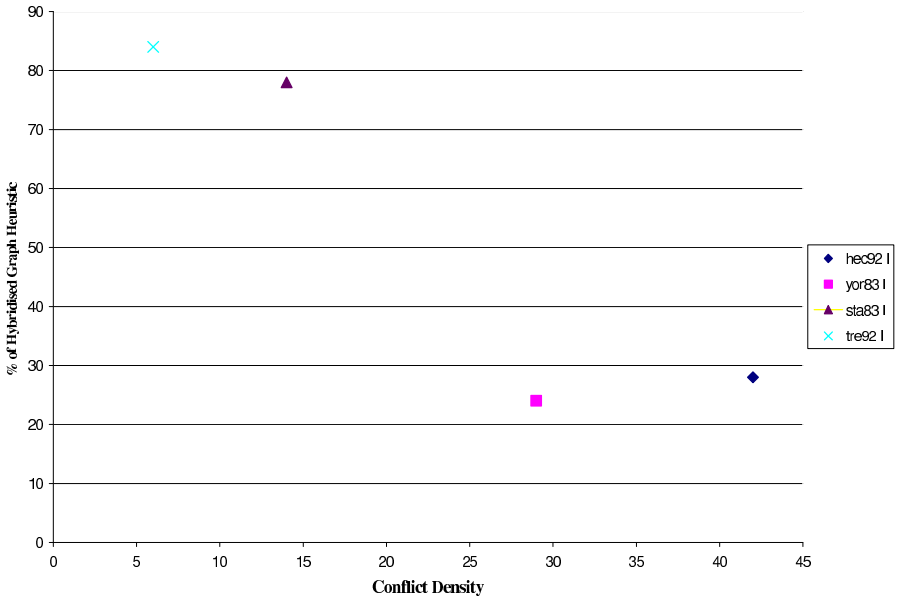

Fig. 4 The relation between conflict density and the percentage of hybridisations obtaining the best solutions

5 Adaptive Tie Breaking and Hybridisation for Benchmark Exam Timetabling Problems

The above observations indicate that there are two important factors to choose the most effective SD tie breaker and the percentage of hybridisation. The two factors are stated as follows:

- Whether a problem is solved using pure SD or not. If it is solved using pure SD then the tie breaking SD should be hybridised with CD. Otherwise it should be hybridised using LD.
- According to the conflict density of the problem, the percentage range of the tie breaking SD to be used could be defined.

Based on the above off-line learning we developed an approach which adapts to the problem in hand, choosing the most appropriate heuristic to be hybridised with the tie breaking SD and search for a solution within the percentage range of the tie breaking SD in a sequence. Figure 5 presents the pseudocode of the Adaptive Tie Breaking (ATB) approach we developed.

```
Schedule all exams using SD only
Set h1 to SD and set the tiebreaker as LWD      // h1 : 1st heuristic in hybridisation
if(feasible solution could be obtained using SD only)
{
        Set h2 to CD     //h2: 2nd heuristic in hybridisation
}
else
{
        Set h2 to LD
}
if(Conflict density > 25 %)
{
        i1 = 0.5;
        i2 = 1;
}
else
{
        i1 = 0;
        i2 = 0.5;
}
for  i1 to i2            // i1 to i2: range of percentages of hybridisation
{
        i = i1;                    // i: percentage of hybridisation
        for  n = 1 to n = number of exams x 10
        {
                Initialise heuristic sequence h = {h1 h1 ……h1}
                h = randomly change (i x size of h) heuristics in h to h2
                Construct a solution c using h
                if a feasible solution (sc) is obtained & sc < s
                {
                        save the best solution, s = sc
                        save h
                }
                else
                {
                        Discard sequence h
                }
        }
        i = i + 0.05;
        Save the smallest s and the corresponding h
}
```

Fig. 5 The psuedocode for the Adaptive Tie Breaking (ATB) approach

According to the conflict density of the problem, the range of percentages of hybridisations is defined. For problems with a conflict density greater than 25%, a

percentage of 50% or greater of tie breaking SD is used. For problems with conflict density less than 25%, a percentage of 50% or less of tie breaking SD is used.

We tested this approach on the Toronto benchmark exam timetabling problems and present the results in Table 12. Furthermore, we tested the approach on the data set we developed to test the generality of the approach and present the results in Table 13. The average computational time across the instances is also presented for 5 runs on a Pentium IV machine with a 1 GB memory. Please note that this run time is acceptable in university timetabling problems because the timetables are usually produced months before the actual schedule is required [16].

To verify the results obtained, we reversed the decisions taken by our approach in choosing a low-level heuristic and a percentage range of hybridisation, and applied the reversed approach to some of the problems with different characteristics and present a comparison in Table 15.

	ATB Average	% of tb SD Average	ATB best	% of tb SD best	Time (s)
hec92 I	12.61	75	11.89	85	274
yor83 I	43.03	71	42.16	81	1683
ear83 I	38.35	90	38.16	77	1475
sta83 I	163.33	42	159.50	56	357
car92	4.5	49	4.44	67	35812
car91	5.35	57	5.23	62	100304
uta92 I	3.64	52	3.50	53	54893
ute92	29.72	43	28.81	48	1469
lse91	11.89	47	11.73	39	2053
tre92	9.01	43	8.83	45	4293
kfu93	16.39	35	15.38	35	2697
hec92 II	12.52	69	11.84	73	346
yor83 II	51.24	91	50.40	84	1144
ear83 II	38.35	90	38.16	82	1457
sta83 II	36.10	47	35.00	42	1415
uta92 II	3.56	46	3.48	48	66339

Table 12 Results from the the adaptive tie breaking (ATB) approach on the Toronto Benchmark data set, Percentage of tie breaking SD (% of tb SD).Computational time is presented in seconds.

The results obtained indicate the generality of our adaptive approach to all these exam timetabling instances regardless of the problem size. We compare our approach with other approaches where a random exam is chosen in the situation where a tie occurs. The results are presented in Table 14. We also highlight the best results reported in the literature. In addition, the standard deviation from the best reported results obtained is shown (σ in Table 14). Recall that the aim of the paper is to illustrate the effect of breaking ties in heuristic orderings and automatically hybridising and adapting heuristics. We do not expect to outperform other heuristic and meta-heuristic approaches which are tailored specially for specific instances of this exam timetabling benchmark. However, we demonstrate that we can achieve results that are competitive with the best in the literature.

	ATB Average	% of tb SD Average	ATB best	% of tb SD best	Time (s)
SP5	3.71	34	3.55	34	10
SP10	11.91	48	10.54	45	30
SP15	17.15	37	15.56	35	29
SP20	20.33	44	18.69	41	33
SP25	25.18	54	23.22	52	43
SP30	33.89	52	31.56	60	78
SP35	47.42	64	45.19	74	141
SP40	28.58	53	27.28	61	109
SP45	32.76	78	31.08	81	100
LP5	9.27	42	9.12	39	4610
LP10	15.15	45	14.96	40	1803
LP15	13.31	29	13.05	24	2540
LP20	11.39	12	11.30	10	5708
LP25	9.93	89	9.90	86	16396
LP30	7.45	72	7.43	72	29367
LP35	6.81	90	6.76	91	40160
LP40	5.23	69	5.22	73	42306
LP45	4.85	72	4.77	81	59045

Table 13 Results from the the adaptive tie breaking (ATB) approach on the random data set, Percentage of tie breaking SD (% of tb SD).Computational time is presented in seconds.

In comparison with the tabu search based hyper-heuristic in [13], our tie breaking approach performs better in 8 out of 11 cases reported. Furthermore, it outperforms in all the cases in comparison with the pure GHH investigated in [7]. Only the problems presented in Table 14 were compared to other results since the others were not reported in the literature. Computational time was also not compared for the same reason.

To validate our results we present the average, best and standard deviation of the results obtained when using different combinations of the low-level heuristic and the percentage range used in the hybridisations in Table 15. A paired t-test obtained a stat of 2.1, which is close to 2.11 ($p = 0.05$), demonstrating the adaptiveness of our approach during solution construction, indicating that hybridising the tie breaking SD with CD improves the quality of the solutions for problems solved using only SD. Otherwise a LD hybridisation is better. In addition, the comparison indicates that for problems with conflict density greater than 25% more tie breaking SD should be used than any low-level heuristic used in the hybridisation.

Problems	ATB Best	σ	GHH Best (Burke et al., 2007)	Tabu search HH (Kendall, 2005)	Best Reported (Qu et al., 2009a)
hec92 I	11.89	1.90	12.72	11.86	**9.2**
sta83 I	159.50	1.56	158.19	157.38	**157.3**
yor83 I	42.16	4.21	40.13	-	**36.20**
ute92 I	28.81	3.11	31.65	27.60	**24.4**
ear83 I	38.16	6.26	38.19	40.18	**29.3**
tre92	8.83	0.65	8.85	8.39	**7.9**
lse91	11.73	1.50	13.15	-	**9.6**
kfu93	15.38	1.68	15.76	15.84	**13.0**
car92 I	4.44	0.36	4.84	4.67	**3.93**
uta92 I	3.50	0.25	3.88	-	**3.14**
car91 I	5.23	0.52	5.41	5.37	**4.5**

Table 14 Best results obtained by the adaptive tie breaking (ATB) approach compared to other Hyper-Heuristic approaches and the best reported in the literature

Problems	ATB Average	Reverse ATB Average	ATB Best	Reverse ATB Best	ATB σ	Reverse ATB σ
ute92 I	29.72	30.27	28.81	29.94	0.531	0.120
ear83 I	38.35	38.41	38.16	38.27	0.116	0.091
lse91	11.89	13.04	11.73	12.17	0.099	0.511
kfu93	16.39	17.13	15.38	16.90	0.589	0.142
car92 I	4.5	4.64	4.44	4.49	0.042	0.096
uta92 I	3.56	3.82	3.50	3.52	0.042	0.182
car91 I	5.35	5.51	5.23	5.24	0.076	0.165
SP5	3.71	4.24	3.55	3.82	0.099	0.252
SP10	11.91	11.98	10.54	10.78	0.797	0.702
SP15	17.15	17.71	15.56	15.98	0.924	1.008
SP20	20.33	20.52	18.69	18.89	0.953	0.950
SP25	25.18	25.21	24.17	25.18	0.589	0.030
SP30	33.89	34.22	31.56	32.02	1.351	1.279
SP35	47.42	47.85	45.19	46.00	1.293	1.077
SP40	28.58	29.57	27.28	27.61	0.756	1.140
SP45	32.76	32.86	31.08	31.18	0.976	0.979
LP5	9.27	9.35	9.12	9.29	0.093	0.046
LP25	9.93	9.99	9.90	9.96	0.025	0.03
LP45	4.85	5.03	4.77	4.78	0.053	0.154

Table 15 A comparison of the results obtained by the adaptive tie breaking and the reverse of the approach

6 Conclusions

This paper presents an adaptive approach where a Saturation Degree heuristic, using Largest Weighted Degree to break ties, is dynamically hybridised with another low-level heuristic for exam timetabling problems. The hybridisation is performed according to the conflict density of the problem and the ability of the problem to be solved using a pure Saturation Degree (SD) heuristic. Largest Colour Degree First (CD) and Largest Degree First (LD) are used in the hybridisation process. CD is used for hybridisation if the problem is solved using a pure SD heuristic or LD is used otherwise. The amount of heuristic hybridisation is determined according to the conflict density of the problem. If the conflict density of a problem is greater than 25%, using more SD in a hybridisation with LD or CD and breaking the ties was more effective. On the other hand, for problems with conflict density less than 25% using more LD or CD than SD in a hybridisation was more effective. Our approach is simple and performs the same regardless of the problem instance size although large instances take more time to solve. It performs better than a pure graph based hyper-heuristic and obtains reasonably good results when compared to a tabu search hyper-heuristic.

To test the generality of our approach and verify our results, we developed an exam timetabling instance generator and introduced a set of benchmark exam timetabling problems and reported the first results on this data. The generator takes the problem size and conflict density as inputs and generates a random problem with the required characteristics. To encourage scientific comparisons we made the generator and data set available at http://www.asap.cs.nott.ac.uk/resources/data.shtml. The aim is to have a larger variety of exam timetabling problem instances with different characteristics.

Future research directions include observing the effect of more problem characteristics such as the number of students and enrolments on the performance of the approaches developed. Using hybridisations of more than two heuristics and different tie breakers during solution construction could also be investigated.

Finally, the solutions obtained in this paper could be further improved by applying a meta-heuristic approach. Therefore, it would be interesting to study the effect of different improvement meta-heuristics on exam timetabling problems. The objective would be choosing the meta-heuristic that would produce the best improvement and automating this process.

References

[1] Asmuni H, Burke E, Garibaldi J, McCollum B (2005) Fuzzy multiple ordering criteria for examination timetabling. In: Burke E, Trick M (eds) Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol 3616, Springer, pp 334–353

[2] Brelaz D (1979) New methods to colour the vertices of a graph. Communications of the ACM 22:251–256
[3] Burke E, Newall J (2004) Solving examination timetabling problems through adaptation of heuristic orderings. Annals of operations Research 129:107–134
[4] Burke E, Kendall G, Newall J, Hart E, Ross P, Schulenburg S (2003) Hyper-heuristics: An emerging direction in modern search technology. In: Glover F, Kochenberger G (eds) Handbook of Meta-Heuristics, Kluwer, pp 457–474
[5] Burke E, de Werra D, Kingston J (2004) Applications to timetabling. In: Gross J, Yellen J (eds) Handbook of Graph Theory, Chapman Hall/CRC Press, chap 5.6, pp 445–474
[6] Burke E, Petrovic S, Qu R (2006) Case based heuristic selection for timetabling problems. Journal of Scheduling 9(2):115–132
[7] Burke E, McCollum B, Meisels A, Petrovic S, Qu R (2007) A graph-based hyper-heuristic for educational timetabling problems. European Journal of Operational Research 176:177–192
[8] Carter M, Laporte G (1996) Recent developments in practical exam timetabling. In: Burke E, Ross P (eds) Selected Papers from the 1st International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol 1153, pp 3–21
[9] Carter M, Laporte G, Lee S (1996) Examination timetabling: Algorithmic strategies and applications. Journal of Operational Research Society 74:373–383
[10] Cote P, Wong T, Sabouri R (2005) Application of a hybrid multi-objective evolutionary algorithm to the uncapicitated exam proximity problem. In: Burke E, Trick M (eds) Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol 3616, Springer, pp 151–168
[11] Di Gaspero L, Schaerf A (2001) Tabu search techniques for examination timetabling. In: Burke E, Erben W (eds) Selected Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol 2079, Springer, Berlin, pp 104–117
[12] Duong T, Lam K (2004) Combining constraint programming and simulated annealing on university exam timetabling. In: Proceedings of the 2nd International Conference in Computer Sciences, Research, Innovation & Vision for the Future(RIVF2004), pp 205–210
[13] Kendall G, Hussin N (2005) An investigation of a tabu search based hyper-heuristic for examination timetabling. In: Kendall G, Burke E, Petrovic S, Gendreau M (eds) Selected Papers from MISTA 2003, Springer, pp 309–328
[14] McCollum B, Schaerf A, Paechter B, McMullan P, Lewis R, Parkes L AJand Di Gaspero (2010) Setting the research agenda in automated timetabling: The second international timetabling competition. INFORMS Journal on Computing 22(1)
[15] Mumford C (2007) An order based evolutionary approach to dual objective examination timetabling. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched 2007)

[16] Qu R, Burke E, McCollum B (2009) Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems. European Journal of Operational Research 198:392–404

[17] Qu R, Burke E, McCollum B, Merlot L, Lee S (2009) A survey of search methodologies and automated approaches for examination timetabling. Journal of Scheduling 12:55–89

[18] Ross P (2005) Hyper-heuristics. In: Burke E, Kendall G (eds) Search Methodologies: Introductory Tutorials in Optimisation, Decision Support Techniques, Springer, chap 17, pp 529–556

[19] Thompson J, Dowsland K (1998) A robust simulated annealing based examination timetabling system. Computer & Operations Research 25:637–648

[20] White G, Xie B (2001) Examination timetables and tabu search with longer-term memory. In: Burke E, Erben W (eds) Selected Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol 2079, Springer, Berlin, pp 85–103

[21] Wong T, Cote P, Gely P (2002) Final exam timetabling: a practical approach. IEEE Canadian Conference on Electrical and Computer Engineering(CCECE 2002) 2:726–731