Construction of Examination Timetables Based on Ordering Heuristics

Syariza Abdul Rahman, Andrzej Bargiela, Edmund K. Burke and Ender Özcan
University of Nottingham, School of Computer Science,
Jubilee Campus,
Nottingham NG8 1BB, UK
{sax, abb, ekb, exo}@cs.nott.ac.uk

Barry McCollum
University of Queen Belfast, School of Electronics, Electrical Engineering and Computer Science, University Road, Belfast, BT7 1NN, Northern Ireland, UK
b.mccollum@qub.ac.uk

***Abstract*—In this paper, we combine graph coloring heuristics, namely largest degree and saturation degree with the concept of a heuristic modifier under the framework of squeaky wheel optimization for solving a set of examination timetabling problems. Both heuristics interact intelligently and adaptively determine the ordering of examinations to be processed during each of iteration. A variety of approaches based on different heuristics and their combinations are investigated. Experimental results on a set of benchmark problems show that the proposed approaches can produce high quality solutions comparable to the other constructive methods. For one problem instance, the best results based on constructive heuristics provided in the literature are improved by one of the proposed methods. We conclude that our approach is simple, yet effective.**

Keywords-Examination timetabling; constructive heuristic; priority

I. INTRODUCTION

Examination timetabling is an NP hard real world problem [1]. The complexity of the problem arises due to several reasons e.g. the introduction of flexible course structures, increasing student enrolments etc… Further research work is required to enhance the quality of the timetable in such a manner as to satisfy both the institutional and personal preferences. Research in the area of Artificial Intelligence [2] and Operation Research [3] have been implemented using various approaches in order to solve this difficult timetabling problem with the aim to find a more generic and effective approaches. Generally, the goal in timetabling is to find a solution that optimizes some desired objective function based on a set of given constraints. There are two types of constraints i.e. hard constraints and soft constraints. In creating a feasible solution, the hard constraints must not be disobeyed or 'broken' in any circumstances. The soft constraints, on the other hand, can be broken and the extents of these breaches determine the quality of the obtained solutions.

A number of research papers formulate examination timetabling as a graph coloring problem [3] where the vertices represent the examination, the edges represent the conflict (students taking both corresponding examinations at one time) occurring between two examinations and the colors for the vertices represent the time slots for the examination. The incorporation of graph theory is known as one of the earliest approach applied in examination timetabling [4]. In general, using this approach a timetable is constructed by using some sequential strategies which attempts to place the examinations into time-slots with the aim of providing a feasible solution. The placement of the examinations is usually related to the difficulty of examinations to be scheduled, where the most difficult examination scheduled the first. Reference [5] list the most commonly used sequential strategies for examination timetabling, i.e., largest degree, largest weighted degree, color degree and saturation degree.

Much research in the area of timetabling has utilized meta-heuristic approaches with great success. These methods begin with one or more initial solutions and employ search strategies for the purpose of improvement [6]. Various search strategies are designed to escape from local minima e.g. tabu search, simulated annealing, genetic algorithm and ant colony optimization. Hybrid, meta-heuristics approaches have shown to be particularly effective. An overview of meta-heuristic approaches can be found in [7], [6] and [3].

Other methods, based on local search technique have been introduced recently. These methods try to escape from local optima by navigating the search space and exploring the neighborhood structure that is different from that deployed by meta-heuristics. Several studies have been implemented using this idea i.e. large neighborhood search [8], variable neighborhood search [9] and iterated local search [10]. One disadvantage of the approaches described is that there is often a reliance on parameter tuning in the production of solutions in particular circumstances. This has motivated the introduction of hyper-heuristics [11], memetic algorithms using hyper-heuristics to choose from multiple hill climbers [12], case-based reasoning [13], fuzzy approaches [14] and constraint-based reasoning [15] within the timetabling arena.

The early approach of sequential heuristics during the construction phase continues to have great success [16]. These sequential heuristics are proven to be very effective when a backtracking procedure is employed [17]. The backtracking procedure is useful in order to ensure that solutions are feasible. Often some already placed examinations are unscheduled in order to place more difficult examinations.

Various heuristics are implemented as part of backtracking procedures [3]. The current examination is then fixed to the available time-slot while the recent unscheduled examination will be scheduled in other available time-slot. It is found that this procedure can reduce the length of the examination session by half compared to sequential techniques without backtracking. In particular, it has been shown that saturation degree is a dynamic ordering and it can produce a good sequence in ordering the examinations.

Reference [18] introduced fuzzy methodologies for examination timetabling by combining two heuristics to order the examinations based on the difficulty of scheduling them. Three heuristics were used in the experiment i.e. largest degree, saturation degree and largest enrolment with three combinations of two heuristics. A fuzzy approach is used to represent the knowledge from the heuristics (named as input variables), evaluate them and construct an examination weight as an input variable. The examinations are then ordered based on the examination weight values and are scheduled in the timetable without violating any of the hard constraints. The 'bumped back' strategy is employed only if the examination cannot be scheduled in the timetable. This approach has shown to produce a competitive result when tested on the Toronto data sets even though it doesn't employ any improvement method in the algorithm. The work shows that tuning procedure is needed for different combination of heuristics in order to obtain good result.

In the recent application of examination timetabling, interest in adaptive approaches has prompted development of more general techniques that would allow finding the best initial solution without necessitating a backtracking strategy. In [19], the adaptive heuristic orderings technique can adapt to any given problem by adding a heuristic modifier to the basic heuristic technique (e.g. largest degree first). It works by promoting difficult examinations to be schedule first at each of iteration based on its order. Different consideration of hard constraints and soft constraints are taken into account in order to test the application of heuristic modifier. This technique has introduced a good initialization strategy for examination timetabling problems. The results have shown that adaptive approach could improve the quality of the obtained solution compared to basic heuristic approach alone and it is faster and easier to implement. They have proved that this method is capable of turning poor initial ordering into a good one and at the same time supplying more independence on the choice of heuristic ordering. In other recent study, [20] has also implemented adaptive approach to examination timetabling by hybridizing the graph heuristics.

In this study, we investigate the use of adaptive strategies that order (prioritize) the examinations to be scheduled within a constructive approach. These approaches differ from the previously proposed approaches where we have incorporated a strategy to choose examination differently from its original ordering. Additionally, we have incorporated a stochastic component into the process of assigning a selected examination to a time-slot. In Section 2, we present the proposed intelligent heuristics algorithm which is inspired from the Squeaky wheel optimization [21]. Section 3 describes the experimental data and discusses the results. Finally, the conclusion is provided in Section 4. Adaptive Heuristics Ordering the Examinations Based on Priorities

The adaptive approaches we propose are based on the concept of squeaky wheel optimization (SWO) [21]. A squeaky wheel optimization is an iterative greedy approach that cycles around three successive processes: Constructer, Analyzer and Prioritizer. A candidate solution to a problem at hand is assumed to consist of elements. Hence, a solution is constructed element by element using an initial priority ordering of the elements at each step. For example, in the context of examination timetabling, an examination is an element. Using graph coloring heuristics, an initial ordering can be obtained for the examinations. Once the constructor makes an assignment to an element, it goes under an analysis process to see whether such an assignment generates a problem or not. For example, an available time-slot might not be found for a given examination. If a problem occurs, a strategy is used to increase the priority of the element so that it would be ahead of the other elements with lower priorities in the next iteration. In a way, modifying the priorities might change the previous ordering of elements causing construction of a new candidate solution in the next iteration. The iterations continue until certain criteria are met. Finally, the best solution found so far is returned.

- *Constructor*. The constructor generates a solution iteratively by going over each unscheduled examination one by one in the provided order. This order is based on a graph coloring heuristic. At each step, a given unscheduled examination is assigned to a time-slot with the least penalty. Eventually, it is possible that some of the examinations will be still unscheduled at the end.
- *Analyzer*. A certain value is added to the difficulty for the unscheduled examination in order to show that the examination is more difficult to schedule than expected. This value is allowed to increase at each of iteration, if the examination cannot be scheduled.
- *Prioritizer*. The new order of examinations is obtained based on the difficulty values updated by the Analyzer.

Our approaches adapt the examination orderings based on two heuristics in order to schedule them. Each examination has a priority determined by the chosen graph coloring heuristic. Such a value can be considered as a default difficulty level of scheduling for a given examination. If an assignment cannot be found for a certain examination, then it can be considered to be more difficult to schedule then expected. This unscheduled examination is given more priority in the next iteration. Its difficulty level is modified using a heuristic value added on top of the value provided by the graph coloring heuristic. This adaptive approach requires no backtracking strategy, if there are unscheduled examinations.

We have considered only hard constraint to this problem i.e. to avoid any conflicts among examinations. A common objective function for examination timetabling is the proximity cost penalty function which describes the average penalty per student. It was introduced by [22] in 1996 in conjunction with the first acquaintance of benchmark data sets for examination timetabling problem. This objective function is used in this study to measure the quality of the obtained solution. Formally, this cost function represents the spread of students in examination schedule and it has been formulated as the minimization of:

$$\frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} c_{ij} w_{|t_j - t_i|}}{M} \qquad (1)$$

where N is the number of examinations, c_{ij} is the number of students entitled for both examination i and j, t_i is the assigned time-slot for examination i, $w_{|tj - ti|}$ is the weight whenever a student who is entitled for two examinations are scheduled $|t_j - t_i|$ apart and M is the number of students. The penalty weight, $w_{|tj - ti|}$ is calculated as $2^{5 - |tj - ti|}$ where, $|t_j - t_i| \varepsilon \{1, 2, 3, 4, 5\}$.

This study extends the previous work provided in [19]. We used the idea of *difficulty* and *heuristic modifier* within the Analyzer.

$$difficulty_i(t) = heuristic_i(t) + heurmod_i(t) \qquad (2)$$

The $heurmod_i(t)$ for examinations i at iteration t gives priority to examination if only there exist unscheduled examinations and is added to chosen $heuristic_i(t)$ at each of iteration. The $difficulty_i(t)$ is a discrete variable and will estimate the priority of the examination after completing the iteration.

A. Graph Coloring Heuristics

In this study, we used two types of heuristics ordering:

- *Largest Degree*. The ordering is based on the largest number of conflicting examinations and the $heuristic_i(t)$ holds the number of conflicting examinations for examination i. The $difficulty_i(t)$ will be increased at each of iteration t if the examinations are unscheduled. At this stage, the $heuristic_i(t)$ remains unchanged and the $heurmod_i(t)$ will increased during the iteration. Priority is given to the highest value of difficulty.
- *Saturation Degree*. The ordering is based on the number of time-slots in conflict where the examinations with the fewest conflicts will be scheduled first. Specifically, once we have done an assignment, if the next examinations are conflicting with the current examination and the assigned examinations, so the number of slots for the next examinations will be reduced by one at each of assignment. We have initialized the $difficulty_i(t)$ for this heuristic with 1. This value will keep increasing if examination cannot be scheduled during the iteration until the maximum number of time-slot. The higher priority of choosing the examination is given to the higher value of difficulty.

B. Graph Coloring Heuristics

Different heuristic modifiers are used in order to stress the priority to the difficult examinations. Equation (3), (4), (5) and (6) show the description of each characteristic, where c is a constant and give different value to the difficulty.

- *Custom* (*C*). This is a conventional heuristic. We made the heuristic as an adaptive approach and vary the choice of examinations. If there are several examinations to choose with the same heuristic value, we will choose the examination randomly.

$$heurmod_i(t) = heurmod_i\,(t\text{-}1),\ heurmod_i(0)= 0 \qquad (3)$$

- *Additive* (*AD*). The modifier is incrementing by 1 at each of iteration if unscheduled examination occurred. This approach does not make much improvement to the difficulty if the heuristic value is small and it takes longer time or need more iteration to show that the examination is very difficult. In other way, this approach has a modest effect on the problem.

$$heurmod_i(t) = heurmod_i(t\text{-}1) + 1,\ heurmod_i(0) = 0 \qquad (4)$$

- *Multiplicative* (*MP*). We have multiplied the modifier by 2 to show the higher priority for the problematic examinations.

$$heurmod_i(t) = heurmod_i(t\text{-}1) + c,\ heurmod_i(0) = 0,\ c = 2 \qquad (5)$$

- *Exponential* (*EX*). This modifier will upgrade significantly the priority if the examination is difficult since the priority is increased by $2n$, where n is the total number of times the examinations cannot be scheduled.

$$heurmod_i(t) = c\cdot heurmod_i(t\text{-}1),\ heurmod_i(0) = 1,\ c = 2 \qquad (6)$$

C. Shuffling the Ordering of Examinations

In order to choose the examination to be scheduled, we have ordered the examinations based on the difficulty (priority). Previous results indicate that the measures used for the difficulty are approximate measures. Making use of such measures in our approaches, the ordering that we generate might not be indicating the exact ordering that should be. Instead of using the ordering of examinations directly, they can be shuffled and an unscheduled examination can be chosen based on a shuffling strategy.

As a novel strategy, we have partitioned all ordered examinations into fixed size of blocks and shuffled all examinations within each block randomly before making an assignment. This strategy uses a *block size* parameter. As an example, if the block size is fixed as 2, then all examinations are sorted first with respect to their difficulty of scheduling using the chosen measure(s) and each 2 consecutive examinations are either swapped or remain in the same position based on a coin flip. The examinations are scheduled based on this new ordering. The technique has also been tested with a block size of 0, indicating that the measure(s) used directly determines the difficulty of scheduling an examination for comparison purposes. The experiments are performed using different block sizes in order to observe the affect of this parameter on the performance of the approach. This approach will be referred to as *Block*.

For the saturation degree graph colouring heuristic, it is not possible to rearrange the examinations using the block approach due to its dynamic nature. In a previous study [23], it is suggested that a random examination can be chosen from a fixed number of top examinations. Notice that, this strategy can be used with both saturation and largest degree heuristics. An examination is chosen randomly from a given number of top examinations, referred to as *top window size*. After the selected examination is scheduled, the difficulties are updated and no rearrangement takes place. For example, if the top window size is 2, one examination to be scheduled is chosen randomly from the first 2 unscheduled examinations based on their difficulties. The difference between this strategy from the previous one is that in this strategy, there is a chance that (even though this is a slim chance) the most difficult examination might be scheduled at last. This strategy will be referred to as *Top Window*. We have experimented with Block and Top Window with different sizes in {none, 2, 3, 4, 5, 6, 7, 8, 9, 10}.

D. Time-slot Choice

After an unscheduled examination is chosen, it is assigned to the most suitable time-slot. This assignment decision is based on the least penalty value obtained considering all time-slots. In the previous studies, it seems that the first time-slot that generates the least penalty is chosen for assignment. It is possible that there might be several time-slots that generate the same least penalty value; hence, we have included an element of randomness in making this choice. In such a situation, we introduce the possibility of an examination to be assigned to a different time-slot during another iteration even though the order of examinations does not change.

II. Experiments

Pentium IV 1.86 GHz. Windows machines having 1.97 Gb memories were used during the experiments. All runs were repeated fifty times to generate solution for each combination of graph coloring heuristic, heuristic modifier, shuffling strategy and the relevant parameter due to the stochastic nature of the proposed approaches. A run terminated whenever the maximum number of iterations was reached. Two different values, {2000, 4000} were used for the maximum number of iterations during the experiments.

A. Experimental Data

The characteristics of the experimental data sets are summarized in Table I. It was introduced by [22] from various universities with different characteristics and various density of examinations conflict. These benchmarks are very well known in the timetabling community. Unfortunately, there are different versions of these data sets. We adapt the notation used in [7] to specify the data sets used during our experiments.

B. Experimental Results

The experimental results are provided in Table II for the largest degree and the saturation degree graph colouring heuristics using different combination of algorithmic choices, respectively. The tables report the best penalty values obtained out of 50 runs for two graph colouring heuristics (bestLD and bestSD) for each combination and for each problem instance. The best result for each problem instances is highlighted in bold font.

TABLE I. THE CHARACTERISTICS OF THE EXPERIMENTAL DATA SET

Problem	Number of time-slots	Number of examinations	Number of Students	Conflict Density
car92 I	32	543	18 419	0.14
car91 I	35	682	16 925	0.13
ears83 I	24	190	1 125	0.27
hec92 I	18	81	2 823	0.42
kfu93	20	461	5 349	0.06
lse91	18	381	2 726	0.06
rye92	23	486	11 483	0.08
sta83 I	13	139	611	0.14
tre92	23	261	4 360	0.18
uta92 I	35	622	21 266	0.13
ute92	10	184	2 750	0.08
yor83 I	21	181	941	0.29

TABLE II. COMPARISON FOR DIFFERENT HEURISTICS WITH DIFFERENT COMBINATION OF ALGORITHMIC CHOICES

Problem	Combination of Algorithmic Choices {number of iterations, heuristic type, modifier type, Block/Top Window size}	
	bestLD	*bestSD*
car92 I	4.56 {4000, LD, EX, 3}	**4.38** {2000, SD, EX, none}
car91 I	5.36 {4000, LD, EX, 9}	**5.08** {4000, SD, EX, 5}
ears83 I	40.00 {4000, LD, MP, 3}	**38.44** {4000, SD, MP, 2}
hec92 I	11.84 {2000, LD, MP, 6}	**11.61** {2000, SD, C, 5}
kfu93	15.54 {4000, LD, EX, none}	**14.67** {4000, SD, EX, 2}
lse91	11.78 {4000, LD, EX, 3}	**11.69** {2000, SD, MP, 6}
rye92	9.69 {4000, LD, EX, 4}	**9.49** {4000, SD, AD, 5}
sta83 I	157.85 {4000, LD, EX, 9}	**157.72** {4000, SD, C, none}
tre92	8.88 {4000, LD, EX, 2}	**8.78** {4000, SD, C, 9}
uta92 I	3.66 {4000, LD, EX, 2}	**3.55** {4000, SD, EX, 3}
ute92	26.82 {4000, LD, EX, 7}	**26.63** {2000, SD, EX, 7}
yor83 I	41.59 {4000, LD, EX, 6}	**40.45** {4000, SD, C, 5}

A comparison in Table II shows that a saturation degree based approaches provides a better performance as compared to the largest degree based approaches in all problem instances. In considering number for iterations, saturation degree with 4000 iterations has performed better than the 2000 iterations by producing eight best results out of twelve. This is of course by giving more processing time it will give more chance for the algorithm to search and find good solution.

From Table II, the best results for saturation degree are mostly obtained by using the exponential modifier with five best results and from this result it shows that by upgrading the modifier in large amount of values it can significantly give more priority to the difficult examinations and at the same time give a better new ordering for examinations. It is then followed by custom modifier approach with four best results where it does not make use of any heuristic modifier. The difference of custom approach from previous implementations is that, we have still utilised the idea of assigning a random time-slot in case of equal quality possibilities for a given unscheduled examination. The multiplicative and additive modifier has obtained two and one respectively, for saturation degree. The top window size affects the performance of the approach.

In considering largest degree graph colouring heuristic, Table II shows that the exponential heuristic modifier is the best choice in combination with the largest degree graph colouring heuristic for changing the order of examinations based on difficulty. The exponential heuristic modifier provided 10 best results for 12 problems, followed by the multiplicative heuristic modifier with two best results. The additive heuristic modifier has not delivered a good performance since it makes small changes in updating the difficulty value and longer time are needed to show big changes to the examinations ordering. The block size choice affects the performance. Considering the average penalty values, the block size of 6 is the best, but this performance variation is not significant. The conventional heuristic (with no block size) in this experiment has shown comparable result too. It has produced one best result for largest degree using exponential approach.

TABLE III. COMPARISON FOR DIFFERENT APPROACHES FOR (A) CONSTRUCTIVE HEURISTICS AND (B) OTHER IMPROVEMENT APPROACHES

Problem	**[19]**	**[22]**	**[24]**	**[18]**	**bestLD**	**bestSD**
car92 I	***4.32***	6.2	4.53	4.54	4.56	4.38
car91 I	**4.97**	7.1	5.36	5.29	5.36	5.08
ears83 I	**36.16**	36.4	37.92	37.02	40.00	38.44
hec92 I	11.61	**10.8**	12.25	11.78	11.84	11.61
kfu93	15.02	**14.0**	15.2	15.8	15.54	14.67
lse91	10.96	**10.5**	11.33	12.09	11.78	11.69
rye92	-	**7.3**	-	10.38	9.69	9.49
sta83 I	161.9	161.5	158.2	160.4	157.85	**157.72**
tre92	**8.38**	9.6	8.92	8.67	8.88	8.78
uta92 I	**3.36**	3.5	3.88	3.57	3.66	3.55
ute92	27.41	**25.8**	28.01	28.07	26.82	26.63
yor83 I	40.77	41.7	41.37	**39.8**	41.59	40.45

(A)

Problem	**[25]**	**[26]**	**[13]**	**[10]**
car92 I	4.5	5.2	4.6	6.0
car91 I	***3.93***	4.2	4.0	6.6
ears83 I	33.7	34.2	32.8	***29.3***
hec92 I	10.83	10.2	10.0	***9.2***
kfu93	13.82	14.2	***13.0***	13.8
lse91	10.35	11.2	10.0	***9.6***
rye92	8.53	8.8	-	***6.8***
sta83 I	158.35	***157.2***	159.9	158.2
tre92	7.92	8.2	***7.9***	9.4
uta92 I	***3.14***	3.2	3.2	3.5
ute92	25.39	25.2	24.8	***24.4***
yor83 I	36.35	36.2	37.3	***36.2***

(B)

The bold entries indicate the best results for given approaches only, while the bold and italic ones indicate the best results found so far for the given problem instance

As can be seen in Table II, increasing the block or top window size does not seem to improve the performance much. This might be because the arrangement of examinations in bigger chunks reduces the effectiveness of the approach by increasing the chance of a move towards a more random ordering of examinations. As another approach using a graph colouring heuristic can be considered to execute for different parametric choices, i.e., for largest degree and saturation degree using a number of block and top window sizes, respectively. Since, the parametric choices in both cases is a constant factor (2 to 10), it does not affect the overall running time.

Table III reports the best results obtained for each data set in the literature using both constructive and improvement approaches. A comparison to previously proposed constructive approaches reveals that our constructive approach using the saturation degree heuristic provides new best result for sta83. For the rest of the problems, the results obtained are still comparable. Except for ears83 I, the adaptive approach using saturation degree performs better than at least one approach for each problem instance. Our approach generates better results as compared to the approach presented in [24] and [18] almost for all problem instances, except for ears83 I. Reference [18] also performs slightly better than our approach for car91 I and yor83 I. One of the best previously approaches is described in [22] which generates a better performance in 5 out of 12 problems. A comparison to the constructive approach proposed in [19] shows that ours generates better results for kfu93, sta83 I and ute92 and a tie for hec92 I. Additionally, their approach can not even generate a feasible solution for rye92, while we obtain a good solution. The best results obtained using improvement approaches can not be improved further in any case but we have obtained one best result for car92 I within the improvement approaches. For only sta83 I, the proposed approaches generate a comparable result.

III. CONCLUSION

In this paper we investigate the use of adaptive strategies that order (prioritize) the examinations to be scheduled within a constructive approach. These approaches differ from the previously proposed approaches where we have incorporated a strategy to choose examination differently from its original ordering. In this study, we have also incorporated a stochastic component into the process of assigning a selected examination to a time-slot. Our adaptive approaches can produce comparable solutions to the other approaches. The difficulty levels generated by combining a graph coloring heuristic and a heuristic modifier are used in ordering the examinations for the timetabling process. We have observed that by increasing the difficulty in certain ways, we can obtain good approximate solutions. As a dynamic graph coloring heuristic, saturation degree has produced most of the best results as compared to the largest degree heuristic. In considering the appropriate heuristic modifier, exponential approach is the best for largest degree and saturation. The block and top window size approach in this study have varied in certain ways since the incorporation of stochastic element in our approach. We have identified that the best block and top window size to use is nine and below. Still, this approach is simple, very affective and requires less computational time, hence it has potential for practical use. In future work, we intend studying the datasets introduced in Track on of the 2nd International Timetabling Competition [27]. This datasets represent real-life data instances with richer problems and several new requirements and limitations that satisfy the real world implementations in examination timetabling.

REFERENCES

[1] D. Schindl, “Some new heredity classes where graph coloring remains NP-hard,” Disc. Math., vol. 295, pp. 197-202, 2005.

[2] V. Tam, J. Ho and A. Kwan, “Applying an improved heuristic based optimiser to solve a set of challenging university timetabling problems: an experience report,” Lecture Notes in Comput. Sci., vol. 3157, pp. 164-172, 2004 [PRICAI 2004: Trends in Artificial Intelligence].

[3] R. Qu, E. K. Burke, B. McCollum, L. G. T. Merlot and S. Y. Lee, “A survey of search methodologies and automated system development for examination timetabling,” J. Sched., in press.

[4] M. W. Carter, “A survey of practical applications of examination timetabling algorithms,” Oper. Res., vol. 34, pp. 193-202, 1986.

[5] E. K. Burke, J. Kingston and D. de Werra, “Applications to timetabling.” in Handbook of Graph Theory, J. Gross and J. Yellen, Eds. Chapman Hall/CRC Press, 2004, pp. 445-474.

[6] S. Petrovic and E. K. Burke, “Chapter 45: University Timetabling.” in Handbook of Scheduling: Algorithms, Models and Performance Analysis, J. Leung, Eds. CRC Press, 2004.

[7] E. K. Burke and G. Kendall, Introductory Tutorials in Optimisation, Decision Support and Search Methodology. New York: Springer, 2005.

[8] S. Abdullah, S. Ahmadi, E. K. Burke and M. Dror, “Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling,” OR Spectrum, vol. 29, pp. 351-372, 2007.

[9] S. Abdullah, S. Ahmadi, E. K. Burke, M. Dror and B. McCollum, “A tabu based large neighbourhood search methodology for the capacitated examination timetabling problem,” J. Oper. Res. Soc., vol. 58, pp. 1494-1502, 2007.

[10] M. Caramia, P. DellOlmo and G. F. Italiano, “New algorithms for examination timetabling,” in Lecture Notes in Comput. Sci., vol. 1982, pp. 230-241, 2001 [Algorithm Engineering 4th Int. Workshop, Proc., 2000].

[11] E. K. Burke, G. Kendall and E. Soubeiga, “A tabu-search hyper-heuristic for timetabling and rostering,” J. Heur., vol. 9, pp. 451-470, 2003.

[12] E. Ersoy, E. Özcan and Ş. Uyar, “Memetic algorithms and hyperhill-climbers,” Proc. of the 3rd Multidisciplinary Int. conf. on scheduling: theory and applications, P. Baptiste, G. Kendall, A. M. Kordon and F. Sourd, Eds. pp. 159-166, August 2007.

[13] E. K. Burke, S. Petrovic and R. Qu, “Case based heuristic selection for timetabling problems,” J. Sched., vol. 9, pp. 115-132, 2006.

[14] H. Asmuni, E. K. Burke, J. M. Garibaldi and B. McCollum, “A novel fuzzy approach to evaluate the quality of examination timetabling,” Lecture Notes in Comput. Sci., vol 3867, pp. 327-346, 2007 [Practice and Theory of Automated Timetabling VI, 2006].

[15] C. Cheng, L. Kang, N. Leung and G. M. White, “Investigations of a constraint logic programming approach to university timetabling,” Lecture Notes in Comput. Sci., vol 1153, pp. 112-129, 1996 [Practice and Theory of Automated Timetabling I, 1995].

[16] E. K. Burke, A. Eckersley, B. McCollum, S. Petrovic and R. Qu, “Hybrid variable neighbourhood approaches to university examination timetabling,” European J. Oper. Res., in press.

[17] M. W. Carter, G. Laporte and J. W. Chinneck, “A general examination scheduling system,” Interfaces, vol. 24, pp. 109-120, 1994.

[18] H. Asmuni, E. K. Burke, J. M. Garibaldi, B. McCollum and A. J. Parkes, “An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables,” Comput. Oper. Res., vol. 36, pp. 981-1001, 2009.

[19] E. K. Burke and J. P. Newall, “Solving examination timetabling problems through adaptation of heuristics orderings,” Ann. Oper. Res., vol. 129, pp. 107-134, 2004.

[20] R. Qu, E. K. Burke and B. McCollum, “Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems,” European J. Oper. Res., in press.

[21] D. E. Joslin and D. P. Clements, ““Squeaky wheel” optimization,” J. Artificial Intelligence Res., vol. 10, pp. 353-373, 1999.

[22] M. W. Carter and G. Laporte, “Recent developments in practical examination timetabling,” Lecture Notes in Comput. Sci., vol 1153, pp. 1-21, 1996 [Practice and Theory of Automated Timetabling I, 1995].

[23] E. K. Burke, J. Newall, R. F. Weare, “A simple heuristically guided search for the timetable problem,” ICSC, pp. 574-579, 1998 [In proceedings of the international ICSC symposium on engineering of intelligent systems (EIS'98)].

[24] E. K. Burke, A. Meisels, S. Petrovis and R. Qu, “A graph-based hyper-heuristic for educational timetabling problem,” European J. Oper. Res., vol. 176, pp. 177-192, 2007.

[25] Y. Yang and S. Petrovic, “A novel similarity measure for heuristic selection in examination timetabling,” Lecture Notes in Comput. Sci., vol 3616, pp. 334-353, 2005 [Practice and Theory of Automated Timetabling V, 2004].

[26] P. Côté, T. Wong and R. Sabouri, “Application of a hybrid multi-objective evolutionary algorithm to the uncapacitated exam proximity problem,” Lecture Notes in Comput. Sci., vol 3616, pp. 151-168, 2005 [Practice and Theory of Automated Timetabling V, 2004].

[27] B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A. Parkes, L. Di Gaspero, R. Qu, and E. K. Burke, “Setting the research agenda in automated timetabling: the second international timetabling competition,” INFORMS J. Comput., in press.