Hybrid Variable Neighborhood HyperHeuristics for Exam Timetabling Problems

Rong Qu Edmund K. Burke

Automated Scheduling, Optimisation and Planning (ASAP) Group
School of Computer Science and Information Technology
University of Nottingham, Nottingham, NG8 1BB, U.K.
{rxq,ekb}@cs.nott.ac.uk

1 Introduction

Exam timetabling is one of the most important administrative activities in universities (e.g. [4], [8], and [19]). Such problems have been the subject of significant research effort across Artificial Intelligence research and Operational Research since the 1960s. A general exam timetabling problem consists of assigning a set of exams into a limited number of timeslots while satisfying a set of constraints: *Hard constraints* that cannot be violated in any circumstances; and *Soft constraints* which should be satisfied as much as is possible.

The area of meta-heuristics [12], [17] has demonstrated success in developing state-of-the-art timetabling approaches. However these approaches are usually fine-tuned on particular problems and thus cannot be easily applied to other problems. Hyperheuristics [3] have been employed recently with some success and are motivated by the aim of improving the generality of search methodologies to facilitate the automatic generation of solution to a larger range of scheduling and timetabling problems than is possible today. They can be defined as *heuristics to choose heuristics* [3] where *low level* heuristics (rather than actual solutions) are the focus of search by the *high level* heuristics. The high level heuristics investigated for timetabling and scheduling include Case-Based Reasoning (e.g. [2], [6]), choice function (e.g. [15]) and meta-heuristics (e.g. [5], [7], [10], [11], [14], [16], [18] and [20]). Low level methodologies include constructive heuristics (e.g. [1], [2], [5], [6], [16], [18] and [20]) and improvement techniques by moving strategies (e.g. [6], [14] and [15]).

Variable Neighborhood Search (see [13]) (VNS) tries to escape from local optima by switching between different neighborhood structures. In this work it is employed as the high level heuristic within the graph based hyperheuristic developed in our previous work [5]. The aim is to investigate how different neighborhoods will affect the behavior of the high level search and contribute to its ability of efficiently explore the search space.

Hyperheuristics are different from Variable Neighborhood Search. Firstly, what is searched is different. Hyperheuristics search upon heuristics rather than the actual solutions. Heuristics searched may be moving strategies, constraint satisfaction strategies, graph heuristics or

metaheuristics. VNS searches actual solutions by systematically switching among different neighborhood structures. The aim is to helping the search to escape from local optima. Secondly, the concentration of the search is different. In VNS, the search concentrates locally on solutions within a limited regions. Hyperheuristics are more global by concerning adaptively the combinations of heuristics for solving the overall problem.

2 The Graph Based Hyperheuristic Framework

Our previous work investigated a graph based hyperheuristic on both the course and exam timetabling problems [5]. High level heuristics search for lists of the low level graph heuristics, each of which is used to construct a complete solution. Low level heuristics are constructive graph heuristics (Color Degree, Largest Degree, Largest Enrollment, Largest Weight Degree, Saturation Degree and a Random Ordering method) that order the events to be scheduled into the timetable by how difficult they are (according to the different heuristics).

To construct a solution, the heuristics in a list are used one by one, from the first, to order the events not yet scheduled at that step. The first set of events ordered by the graph heuristic is scheduled into the least-penalty timeslots that are available. Then the next heuristic in the heuristic list is used to order the events left again and the first set of events is scheduled into the timetable. This procedure is repeated until a complete solution is constructed by the whole heuristic list.

For the high level heuristic, the penalty of the solution generated by each heuristic list is used in the objective function. The objective of the high level heuristic is to search for the heuristic list that generates the timetable with the least penalty. Promising results compared with the state-of-the-art approaches on both benchmark course and exam timetabling problems indicate the generality and efficiency of this simple graph based hyperheuristic approach. See [5] for more details.

3 Hybrid Variable Neighborhood Hyperheuristics

3.1 Research Background

In our previous work [16], we studied the reason why this simple general approach works well on both of the exam and course timetabling problems . By making a move (of changing just two heuristics) in the heuristic list, a large part of the solution generated by it will be quite different from that of the previous heuristic list. This is because by changing a heuristic in a heuristic list, the events ordered and scheduled at that step would be (and probably will be completely) different. Thus the events left after this step will also be different, making the complete timetable (probably) very different to that generated by the previous heuristic list. Notice that by making an analogously simple move in a local search based method upon the actual solutions, only two events in the timetable will be different. This provides an illustration as to why (in a certain sense) the same effort within the high level searching is capable of exploring a much larger part of the solution space than a similar amount of effort used by a local search based methods that are applied directly to solutions.

For each heuristic list, let us denote the number of possible neighborhoods by h^l, where h is the number of low level heuristics employed and l is the length of the heuristic list. From the experience of analysing these issues we noted that the search space seems to be more likely to contain large areas of heuristic lists which generate the same quality solutions (perhaps because the search space is so large). We found that the landscape of the high level heuristic is more likely to be smoother than the search space of a more standard approach.

Based on the above observations in our previous work, we are motivated by how the graph based hyperheuristic will perform upon different neighborhood structures by employing a Hybrid Variable Neighborhood Search and we compare its behavior with that of others investigated in [16].

3.2 The Hybrid Approach

The overall high level heuristic is an iterated approach, in which the Variable Neighborhood Search is restarted after a certain number of iterations is carried out. In order to make a reasonable comparison, the search stops after the same number of evaluations as that employed in [5]. The same low level heuristics are used. For an accurate comparison, no local search is carried out after each move of the searching methods (i.e. Steepest Descent method, Iterated Local Search in [16] and the Hybrid Variable Neighborhood Search studied here), which was implemented for Tabu Search in [5]. This will also demonstrate the generality of this graph based hyperheuristic, as no domain knowledge is involved in the framework except the evaluation function used to evaluate the penalty of the solutions, which can be easily switched upon different problems.

3.3 Neighborhood Structures

We designed two sets of simple neighborhood structures to be employed by the high level Hybrid Variable Neighborhood Search. They can be described as follows:

- Neighborhood set 1 (VNS1): randomly change two, three, four or five heuristics in the heuristic list (single flipped).

- Neighborhood set 2 (VNS2): randomly change two, three, four or five consecutive heuristics to another heuristic in the heuristic list (block flipped).

These neighborhoods are easy to implement and they represent general structures which are widely employed in many Variable Neighborhood Search approaches. The reason why we do not use *swap heuristics* is that swapping heuristics will have the same effect as that of changing the corresponding two heuristics in the heuristic list (which is included in VNS1). It thus makes no sense to swap the heuristics as the moves are made upon heuristics rather than the actual solutions.

3.4 Experimental Results and Analysis

We test the same benchmark exam timetabling problems that were used for the other approaches and that were originally presented in [9]. The hard constraint of the problems is that no *conflict* exams (with common students) should be scheduled at the same time. Soft constraints are concerned with spreading conflicting exams over the timeslots so that students have enough time between exams. The objective function is set as the sum of costs (caused by scheduling conflicting exams too close to each other) per student. Over the years, these problems have been widely studied and have been used as a testbench for a range of fine-tuned state-of-the-art approaches in timetabling research.

The two Hybrid Variable Neighborhood hyperheuristics presented above are employed to solve the benchmark exam timetabling problems [9]. For each problem, 5 runs with distinct seeds are carried out and the best and average results are presented in Table 1. For comparison, we also list the best and average results obtained by employing Tabu Search, Steepest Descent and Iterated Local Search in [16] in Table 1. As the same number of evaluations are made for the high level heuristics, the same/similar computational time will be taken. We do not list the computational time because in many real world timetabling scenarios this does not represent a significant issue.

Table 1: TS (tabu search), SDM (steepest descent method), ILS (iterated local search) and Hybrid Variable Neighborhood hyperheuristics (VNS1: single flipped; VNS2: block flipped) on exam timetabling problems

	TS		SDM		ILS		VNS1		VNS2	
	best	avg	best	avg	best	avg	best	avg	best	avg
car91	5.41	6.13	5.44	6.18	**5.3**	*6.01*	5.4	6.1	5.43	6.12
car92	4.84	5.28	4.87	5.3	4.77	5.18	**4.7**	*5.1*	4.78	5.15
ear83	38.19	45.26	**35.54**	*36.8*	38.39	39.58	37.29	38.63	37.94	38.95
hec92	**12.01**	14.6	12.59	12.74	**12.01**	*12.33*	12.23	12.72	12.67	12.76
kfu93	15.76	19.52	15.25	15.63	**15.09**	15.35	15.11	*15.24*	15.31	15.7
lse91	13.15	14.27	13.01	13.51	12.72	13.1	**12.71**	*13.06*	12.85	13.22
sta83	141.1	159.5	140.3	143.7	**139.2**	*141.6*	139.3	143.3	141.6	144.8
tre92	8.85	9.56	9.01	9.37	8.74	9.0	**8.67**	*8.88*	8.74	8.94
ute92	32.01	36.0	31.77	32.6	30.32	31.3	30.23	31.07	**29.68**	*30.75*
uta93	3.54	4.32	3.61	4.5	**3.32**	*4.01*	3.56	4.05	3.54	4.1
yor83	**40.13**	47.41	42.77	43.6	40.24	*43.15*	43.0	43.93	43.43	44.6

The results in Table 1 indicate that Tabu Search (with a local search after each move to further improve the solution quality) and Steepest Descent perform quite similarly over the set of problems. This is an unusual observation as Tabu Search is superior to Steepest Descent on timetabling problems. Iterated Local Search and Hybrid Variable Neighborhood Search with the single flipped neighborhood work quite similarly and, in general, better than the other three approaches (with the best results presented in bold in Table 1 for 5 and 3 of the 11 problems, respectively, and best average results, presented in italics, in Table 1 for 5 and 4 of the 11 problems, respectively). Please note that the costs are per student so a small difference

in the results indicates a large difference in solution quality.

4 Conclusion

The overall observation from the results is that, in general, Iterated Local Search performs better than Tabu Search and Steepest Descent and both Variable Neighborhood Search methods. However, all of the methods produce at least on best result in these experiments. We can observe, from these experiments, that the method employed for high level search is not crucial within the graph based hyperheuristic approach for exam timetabling problems.

References

[1] Asmuni H., Burke E. and Garibaldi J. (2004): "Fuzzy Multiple Ordering Criteria for Examination Timetabling". In: *the 5th International Conference on the Practice and Theory of Automated Timetabling* 51-65. Pittsburg, USA. Aug, 2004.

[2] Burke, E., Dror, M., Petrovic, S. and Qu, R. (2005): "Hybrid Graph Heuristics within a Hyper-heuristic Approach to Exam Timetabling Problems". In: Golden, B.L., Raghavan, S. and Wasil, E.A. (eds.). (2005): *The Next Wave in Computing, Optimization, and Decision Technologies*. Conference Volume of the 9th informs Computing Society Conference. 79-91. published by Springer. Jan 2005.

[3] Burke, E., Hart, E., Kendall, G., Newall, J., Ross, P. and Schulenburg, S. (2003): "Hyper-heuristics: an Emerging Direction in Modern Search Technology". In: Glover F. and Kochenberger G. (eds.): *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston, USA, 457-474.

[4] Burke E., Jackson S., Kingston H. and Weare F. (1997).: "Automated Timetabling: the State of the Art". In: *The Computer Journal* **40** (9), 565-571.

[5] Burke, E., Meisels, A., Petrovic, S. and Qu, R. (2005): "A Graph-Based Hyper Heuristic for Timetabling Problems". Accepted by *EJOR*, 2005.

[6] Burke E.K., Petrovic S. and Qu R. (2005): "Case Based Heuristic Selection for Timetabling Problems". To appear in *Journal of Scheduling*

[7] Burke E.K., Kendall G. and Soubeiga E. (2003): "A Tabu Search Hyperheuristic for Timetabling and Rostering". *Journal of Heuristics* **9** (6), 451-470.

[8] Carter M. and Laporte, G. (1996): "Recent Developments in Practical Exam Timetabling". In: Burke, E. and Ross, P. (eds.): *Selected Papers from the 1st International Conference on the Practice and Theory of Automated Timetabling (PATAT95).* LNCS 1153, Springer-Verlag. 3-21.

[9] Carter, M., Laporte, G., Lee, S. (1996): "Examination Timetabling: Algorithmic Strategies and Applications". *JORS* **47** 373-383.

[10] Dowsland K., Soubeiga E. and Burke E.K. (2005): "A Simulated Annealing Hyper-heuristic for Determining Shipper Sizes". Accepted by *EJOR*.

[11] Gaw A., Rattadilok P. and Kwan R.S. (2004): "Distributed Choice Function Hyperheuristics for Timetabling and Scheduling". In: Burke E.K. and Trick M. (eds.) Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT 04), Pittsburgh, USA, 495-497.

[12] Glover F. and Kochenberger G. (eds.): *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston, USA.

[13] Hansen P. and Mladenovic N. (2003): "Variable Neighborhood Search". In: Glover F. and Kochenberger G. (eds.): *Hanbook of Metaheuristics*. Kluwer Academic Publishers, Boston, USA, 145-184.

[14] Han L., Kendall G. (2003).: Investigation of a Tabu Assisted Hyper-Heuristic Genetic Algorithm. In: *Congress on Evolutionary Computation (CEC 2003)*, Canberra, Australia, 2230-2237.

[15] Kendall, G., Cowling, P., Soubeiga, E. (2002): Choice Function and Random Hyper-Heuristics. In: *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL 2002)*, 667-671.

[16] Qu R. and Burke E.K. (2005).: Analysing High Level Heuristics within a Graph-Based Hyper Heuristic for Exam Timetabling Problems. Technical Report [NOTTCS-TR-2005-3], School of CSiT, University of Nottingham, U.K.

[17] Reeves C.R. (1996): "Modern Heuristic Techniques". In: R-Smith, V.J., Osman, I.H., Reeves, C.R. and Smith, G.D. (eds.): *Modern Heuristic Search Methods*. 1-25.

[18] Ross P., Marin Blasques J.G., Schulenburg S. and Hart E. (2003): Learning a Procedure that Can Solve Hard Bin-Packing Problems: A New GA-Based Approach to Hyper-heuristics. In: Cant-Paz E. et al (eds.): *Genetic and Evolutionary Computation*, LNCS 2724, 1295-1306.

[19] Schaerf A. (1999): A Survey of Automated Timetabling. In: *Artificial Intelligence Review*. **13** (2): 87-127.

[20] H. Terashima-Marin, R. Ross and M. Valenzuela-Rendon (1999): Evolution of Constraint Satisfaction Strategies in Examination Timetabling. In: *Genetic Algorithms and Classifier Systems 1999*. 635-642.