Evolving Hyper-Heuristics for a Highly Constrained Examination Timetabling Problem

Nelishia Pillay

School of Computer Science, University of KwaZulu-Natal, South Africa

+27 2605644

+27 2605648

pillayn32@ukzn.ac.za

Abstract: A lot of research has been conducted on hyper-heuristics for examination timetabling. However, most of this work has been focused on an uncapacitated version of the problem. This study reports on evolving hyper-heuristics for a highly constrained version of the problem, namely, the set of problems from the second International Timetabling Competition (ITC '07). Previous work has shown that using an evolutionary algorithm (EA) based hyper-heuristic with more than one chromosome representation is more effective than the standard EA using a single representation. This study evaluates an EA hyper-heuristic, using three different chromosome representations, in solving the capacitated examination timetabling problem. The results produced by the hyper-heuristic were found to be comparable to other methodologies applied to the same problem set.

Keywords: hyper-heuristics, examination timetabling, evolutionary algorithms

1. Introduction

The main aim behind hyper-heuristics is to generalize well in a particular domain rather than producing the best result for one or more problems in that domain (Burke et al. 2003 ; Ross et al. 2005). Hyper-heuristics select or combine either perturbative or constructive low-level heuristics. The study presented in this paper focuses on the combination of constructive heuristics. There have been numerous studies investigating the use of constructive hyper-heuristics in the examination timetabling domain. An overview of the most relevant studies follows.

Qu et al. (2005) apply variable neighborhood search to a space of combinations of two or more constructive low-level heuristics. Burke et al. (2005; 2007) employ a tabu search to explore the space of heuristics combinations. Qu et al. (2009b) analyze the heuristic combinations, found by a tabu search hyper-heuristic, that produce feasible timetables in order to identify patterns of low-level heuristics that lead to good quality solutions. Qu et al. (2009a) compare the performance of different local search strategies in exploring the heuristic space. Iterated local search produced the best results. The study also revealed that searching the solution space whilst constructing the timetable using the heuristic combination output by exploring the heuristic space, produces better quality timetables. A Greedy Adaptive Search Procedure (GRASP) is used by Burke et al. (2009) to search a space of heuristic combinations of two constructive low-level heuristics. The quality of the feasible timetable constructed using the heuristic combination returned by GRASP is further improved using steepest descent. Asmuni et al. (2005; 2007; 2009) combine two or three constructive low-level heuristics using a fuzzy logic function. This function estimates the difficulty of scheduling an examination. Examinations are sorted according to their difficulty and scheduled in sequence. Pillay et al. (2007) implement a genetic programming system to search a space of constructive heuristic combinations. The length of the combinations in the initial population is randomly chosen to be between two and a preset maximum. Tournament selection is used to choose parents, to which the crossover and mutation operators are applied to create the next generation. The studies described thus far have combined heuristics linearly and applied them sequentially. Pillay et al. (2009) achieve good results with combining constructive low-level heuristics hierarchically using logical operators and applying them simultaneously. Four heuristic combinations are created and tested. This work is extended further by Pillay (2009) by employing genetic programming to search a space of such heuristic combinations.

All these studies have used the Carter benchmark set of timetabling problems (Carter et al. 1996) to test the hyper-heuristics. This set of benchmarks is comprised of 13 real-world problems. The hard constraint for this set of problems is that no students must be scheduled to sit two examinations at the same time and the soft constraint aims to spread the examinations for each student. A more recent set of examination timetabling problems has been made available by the organizers of the second International Timetabling Competition (ITC ’07). This set of eight problems is highly constrained and is representative of the current real-world examination timetabling problem. At the time of writing this paper, studies into applying hyper-heuristics to such a highly-constrained, multi-objective examination timetabling problem as that represented by the ITC ’07 problem set had not as yet been conducted or published.

The main contribution of the study presented in this paper is the evaluation of the performance of an evolutionary algorithm hyper-heuristic on the set of highly-constrained capacitated examination timetabling problems. The study presented in this paper employs an evolutionary algorithm (EA) to search the heuristic space of linear combinations of constructive low-level heuristics. In previous work three different representations, namely, fixed length, variable length, and *n*-times representation were evaluated for the Carter benchmark problems. A separate EA run using each of the representations as well as an EA combining all three representations were implemented. The study revealed that the EA combining all three representations performed better than the EAs using each of the representations separately. Thus, the EA in this study combines the three chromosome representations. Note that the aim of the study is not to compare the three representations but test the effect of an EA combining the three representations on a more highly constrained, capacitated version of the examination timetabling problem.

The following section provides an overview of the examination timetable problem as defined for the second International Timetabling Competition. Section 3 presents the EA-based hyper-heuristic. The experimental setup for testing the EA-HH is described in section 4.

The performance of the EA-HH on the eight problems is discussed in section 5. The outcome of this study and future extensions of this work are summarized in section 6.

2. The Examination Timetabling Problem for "ITC'07"

The examination timetabling problem requires the allocation of examinations to timeslots so that the hard constraints of the problem are satisfied and the soft constraint cost is minimized. A timetable is said to be feasible if it meets all the hard constraints of the problem. The hard and soft constraints differ drastically from one examination timetabling problem to the next. The ITC '07 problem set has the following hard constraints:

- All examinations must be scheduled.
- There are no clashes, i.e. a student is not scheduled to sit two examinations during the same period.
- The duration of the period that each examination is assigned to is not less than the duration required for the examination.
- The number of students writing an examination does not exceed the capacity of the room the examination is assigned to.
- Period related hard constraints must be met. There are three such constraints: some examinations must occur after other examinations; certain examinations must be written during the same period while others must not be scheduled in the same period.
- Room related hard constraints must be satisfied. In some cases an examination must be assigned exclusively to a room.

The soft constraints for the ITC '07 problem set are summarized below:

- Two in a row – The number of examinations taken back to back by students is minimized.
- Two in a day – The number of examinations written in the same day by students is minimized.
- Period spread – The number of examinations written within a specified period, e.g. 5 days, is minimized.
- Mixed durations – Examinations are of different durations. The number of examinations with different durations in the same room for a period is minimized.
- Larger examinations scheduled earlier in the examination timetable – The number of examinations with a "larger" number of students scheduled in the latter part of the timetable is minimized.
- Room penalties – Certain rooms have a penalty associated with using them. The number of times rooms with penalties are utilized is minimized.
- Period penalties – Certain periods also have a penalty associated with their use. The number of times these periods are used is minimized.

A more detailed description of these soft constraints can be found in (McCollum, 2007). The winner of the competition has taken a multi-phased approach to the problem (Muller, 2008). An iterative forward search, using conflict-based statistics to prevent cycling, is firstly applied to find a feasible timetable. The second phase employs hill-climbing to further improve the quality of the feasible timetable. If hill-climbing can no longer improve the solution, a variation of the Great Deluge algorithm is applied for further improvement.

Gogos et al. (2008), who were placed second, use a combination of the Greedy Randomized Adaptive Search Procedure (GRASP), simulated annealing and mathematical programming to solve the examination timetabling problem.

A variation of GRASP incorporating tabu search is firstly used to find a feasible solution. This solution is then improved using simulated annealing. In the last phase integer programming with branch and bound is used to further improve the quality of the timetable.

Atsuta et al. (McCollum et al. 2009b) were placed third in the competition and implement a constraint satisfaction problem solver which uses tabu search and local iterated search in solving the examination timetabling problem.

De Smet (2007) combines the use of the drools-solver and tabu search to solve the problem. This approach was placed fourth in the competition.

Pillay (2007) takes a developmental approach (DA) to the examination timetabling problem. The DA mimics the processes of cell biology. Each organism developed represents a timetable with each cell representative of an examination period. The creation of an organism begins with a single cell which is developed into a fully grown organism by means of cell division, cell interaction and cell migration. The fully grown organism then goes through a process of maturation in which cell migration is used to further improve the quality of the timetable. The DA was placed fifth in the competition.

The organizers of the competition take a two-phased approach to the problem (McCollum et al. 2009a; McCollum et al. 2009b). The first phase is a construction phase which uses an adaptive ordering heuristic to create a feasible solution. The feasible solution is improved using an extension of the Great Deluge algorithm.

The results obtained by these methods are presented in section 5.

3. The Evolutionary Algorithm

The EA employs the generational control model (Koza 1992) and the population size remains fixed from one generation to the next. An initial population is created and iteratively improved via the processes of evaluation, selection and recreation. These processes are described in the sections below.

3.1 Initial Population Creation

Each element of the population is a string containing two or more characters representing the following constructive low-level heuristics:

- Largest degree (*l*) – Examinations involved in the largest number of clashes are scheduled first.
- Largest enrolment (*e*) – Examinations with the largest number of students are given priority.
- Largest weighted degree (*w*) – Examinations with the largest number of students involved in clashes are allocated first.
- Saturation degree (*s*) – Examinations with the least number of feasible options available on the timetable developed thus far are given priority.
- Spread heuristic (*h*) – Is an estimate of the spread of examinations over a range of periods for each student. The estimate is defined in terms of the proximity of the examinations for a student to each other, and weighted by the number of students involved. Thus, examinations with a higher value are given priority. Like the saturation degree, this heuristic is not static and its value depends on the current state of the timetable. Thus, it needs to be recalculated whenever an allocation is made to the timetable.

These low-level heuristics are combined using one of the following representations:

- Fixed length heuristic combination (FHC) – The length of the combination is equal to the number of examinations, e.g. *well* if the number of examinations is four. One heuristic is used to schedule each examination.
- Variable length heuristic combination (VHC) – Studies conducted by Cowling et al. (2002) and Han et al. (2003) applying a hyper-heuristic genetic algorithm to the trainer scheduling problem have revealed that a chromosome representation with variable length produces better results than a fixed length representation as the GA is able to evolve a chromosome of the optimal length. A similar representation is used in this study. The length of each combination is randomly chosen to be between two and a specified maximum, e.g. *lessh*. Each heuristic is used to schedule an examination.

  If the length of the combination is less than the number of examinations the combination is wrapped around beginning at the start of the string again. If the combination is longer than the number of examinations, only a substring of the combination is applied. Thus, two combinations of length larger than the number of examinations would essentially be clones of each other. Due to this together with the fact that mutation and crossover may produce clones, the reproduction operator is not used.
- N-times heuristic combination (NHC) – Each combination is composed of integers and characters representing low-level heuristics, e.g. *3h2l3s1w1s*. The integer preceding the heuristic specifies the number of examinations the heuristic will be used to schedule. In the example the first three examinations will be allocated according to the spread heuristic, the next two with the largest degree heuristic and so on. The sum of the integer values in the combination is equal to the number of examinations to be scheduled. The reason for including this representation is that it may result in the algorithm converging quicker to certain areas of the heuristic space. For example, it may take longer to evolve

the combination *leslllllllhh* than it would take to evolve *1l1e1s8l2h*. In this way more of the heuristic space may be explored in a shorter time.

The size of the initial population is a genetic parameter and differs for each problem domain. The population consists of an equal number of combinations of each type of representation. Previous work has shown that in the domain of EA-based hyper-heuristics for examination timetabling different representations are suitable for different problems. Thus, an EA providing more than one chromosome representation in the initial population is more effective. The EA converges to the most suitable representation.

3.2 Evaluation and Selection

Each heuristic combination is assigned a fitness measure. The fitness measure of a combination is a function of the hard and soft constraint cost of the timetable constructed using the combination. During the timetable construction process each examination is allocated to the feasible minimum cost timeslot. If there is more than one option the period is randomly chosen from the possible options. If a feasible period is not available, a period is randomly selected. If there is more than one room available, the room with the best fit is chosen. If there is more than room with the same best fit value, the lowest penalty is used to decide which room to use. The fitness measure is the soft constraint cost multiplied by the hard constraint cost incremented by one. Based on trial runs performed, this fitness function proved to be representative of the fitness of an individual without any processing overheads. The fitness measure is used by the selection method to choose parents of the next generation. The tournament selection method is used in this study.

A tournament of *t* individuals is randomly chosen. The fittest individual in the tournament is returned as a winner and is used as a parent for the next generation. The value of *t* is a genetic parameter and is problem dependant.

3.3 Recreation

Two genetic operators, namely, mutation and crossover are implemented to create the next generation. The mutation operator randomly changes a low-level heuristic in a copy of the selected parent. The tournament selection method is evoked to choose a parent. For example, if *welsh* is a chosen parent, the offspring could be *weesh*. In this case *l* was chosen to be replaced. The heuristic *e* was randomly chosen to replace *l*. There is no limit set on the size of the offspring.

The crossover operator randomly chooses two crossover points in two parents selected using tournament selection, and swaps the fragments at the crossover points to produce two offspring. For example suppose w*e**h**ll* and ***l**eh* are the chosen parents and three is the crossover point in the first parent and two in the second. The resulting offspring are w*eeh* and *lhll*. Trial runs have indicated that returning the fitter offspring is more effective than returning both offspring. Crossover occurs between parents of the same representation, i.e. both parents must be NHC or VHC. Crossover is also permitted between VHC and FHC parents. Crossover is structure-preserving in the case of NHC ensuring that an integer value precedes each heuristic.

4. Experimental Setup

The EA-HH was tested on the eight problems from the examination timetabling track of ITC ’07. The four hidden data sets are not publicly available. The characteristics of the eight data sets as presented by McCollum et al. (2009a) are listed in Table 1.

Table 1: Characteristics of the Problem Set

Problem	**Conflict Density (%)**	**No. of Exams**	**No. of Students**	**No. of Periods**	**No. of Rooms**
Exam_1	5.05	607	7891	54	7
Exam_2	1.17	870	12743	40	49
Exam_3	2.62	934	16439	36	48
Exam_4	15.0	270	5045	21	1
Exam_5	0.87	1018	9253	42	3
Exam_6	6.16	242	7909	16	8
Exam_7	1.93	1096	14676	80	15
Exam_8	4.55	598	7718	80	8

The genetic parameters used by the EA are tabulated in Table 2. These values were obtained by performing test runs. Due to the stochastic nature of evolutionary algorithms, ten runs were performed for each problem set, each with a different random number generator seed. The EA was implemented in Java and simulations were run on a system with a 1995 Mhz Intel Core 2 Duo processor and 2 gigabytes of memory.

Table 2: Genetic Parameters

Parameter	**Value**
Number of generations	100
Population size	500
Maximum initial length	5
Tournament size	10
Crossover rate	0.3
Mutation rate	0.7

5. Results and Discussion

The EA-HH produced feasible timetables for all eight problem sets. The best soft constraint cost obtained over ten runs for each problem is listed in Table 3. Although the main aim of a hyper-heuristic is to generalize well rather than producing the best result, for completeness Table 3 compares the performance of the EA-HH to other methodologies applied to the same set of problems. These methodologies are described in section 2. Note that while these methodologies perform one or more improvement phases to reduce the soft constraint cost of the timetable, the EA-HH does not perform additional optimization once a feasible solution is found. Furthermore, in this study the time limitation imposed by the competition was not adhered to and the time taken by the EA-HH was not monitored. It is assumed that the EA-HH will have longer runtimes due to the overhead of solving the problem using each heuristic combination in each population.

Table 3: A Comparison of Results

Problem	EA-HH	Muller	Gogos	Atusta et al.	De Smet	Pillay	McCollum et al.
Exam_1	8559	4370	5905	8006	6670	12035	4633
Exam_2	830	400	1008	3470	623	2886	405
Exam_3	11576	10049	13771	17669	-	15917	9064
Exam_4	21901	18141	18674	22559	-	23582	15663
Exam_5	3969	2988	4138	4714	3848	6860	3042
Exam_6	28340	26585	27640	29155	27815	33005	25880
Exam_7	8167	4213	6572	10506	5436	17666	4037
Exam_8	12658	7742	10521	14317	-	15592	7461

Although the EA-HH does not further optimize feasible solutions, its performance is comparable to the other methodologies applied to this problem set. For all problem sets the EA-HH has produced better results than at least one to three other methodologies. Table 4 lists the representation, i.e. FHC, VHC or NHC, of the best heuristic combination evolved for each problem over the ten runs. Note that the representation converged to for each run maybe different and that the population at the end of the run will have a majority of the individuals with the same structure, because the EA has converged to a particular area of the heuristic space, but not all the individuals will necessarily have the same representation.

Table 4. Representations Converged to

Problem	Representation
Exam_1	FHC
Exam_2	VHC
Exam_3	VHC
Exam_4	VHC
Exam_5	FHC
Exam_6	NHC
Exam_7	FHC
Exam_8	NHC

The algorithm converged to a combination with the NHC representation for two of the problem sets, with the FHC representation for three of the problem sets and with the VHC representation for three of the problem sets. An analysis into a possible correlation between the representation converged to and the characteristics of each problem will be conducted as part of future work.

6. Conclusion and Future Work

This paper presents an EA-based hyper-heuristic for a highly constrained examination timetabling problem, namely, that used for the examination timetabling track of the second International Timetabling Competition. The EA combines three different chromosome representations. The EA-HH produced feasible timetables for all eight competition timetabling problems. Furthermore, the quality of the timetables produced by the EA-HH was comparable to and in some cases better than the best timetable produced by other methodologies, even through the EA-HH did not perform additional optimization after a feasible solution was obtained. However, the time needed to find an optimal heuristic combination was not monitored and thus the approach may have the advantage of longer runtimes. It is interesting to note that the representation of the heuristic combination producing the best quality timetable differed for each problem. Future work will investigate whether there is a correlation between the representation of the best heuristic combination and the characteristics of the problem.

7. References

1. Asmuni H, Burke EK, Garibaldi JM (2005) Fuzzy Multiple Ordering Criteria for Examination Timetabling. In Burke EK, Trick M (eds.), selected Papers from the 5th International Conference on the Theory and Practice of Automated Timetabling (PATAT 2004)- The Theory and Practice of Automated Timetabling V, Lecture Notes in Computer Science, 3616, 147–160.
2. Asmuni H, Burke EK, Garibaldi JM, McCollum B (2007) Determining Rules in Fuzzy Multiple Heuristic Orderings for Constructing Examination Timetables. In proceedings of the 3rd Multidisciplinary International Scheduling: Theory and Applications Conference (MISTA 2007), 59-66.
3. Asmuni H, Burke EK, Garibaldi JM, McCollum B, Parkes AJ (2009) An Investigation of Fuzzy Multiple Heuristic Orderings in the Construction of University Examination Timetables. Computers and Operations Research, Elsevier, 36(4), 981-1001.
4. Burke E, Hart E, Kendall G, Newall J, Ross P, Schulenburg S (2003) Hyper-Heuristics: An Emerging Direction in Modern Research. Handbook of Metaheuristics, chapter 16. Kluwer Academic Publishers, 457– 474.
5. Burke EK , Dror M , Petrovic S, Qu R (2005) Hybrid Graph Heuristics with a Hyper-Heuristic Approach to Exam Timetabling Problems. In Golden BL, Raghavan S, Wasil E.A. (eds.), The Next Wave in Computing, Optimization, and Decision Technologies – Conference Volume of the 9th Informs Computing Society Conference, 79 -91.
6. Burke EK, McCollum B, Meisels A, Petrovic S, Qu R (2007) A Graph-Based Hyper-Heuristic for Educational Timetabling Problems. European Journal of Operational Research (EJOR), 176, 177 – 192.
7. Burke EK, Qu R, Soghier A (2009) Adaptive Selection of Heuristics within a GRASP for Examination Timetabling Problems. In proceedings of Multidisciplinary International Conference on Scheduling 2009 (MISTA 2009), 409-422.
8. Carter MW, Laporte G, Lee SY (1996) Examination Timetabling: Algorithmic Strategies and Applications. The Journal of the Operational Research Society, 47(3), 373-383.
9. Cowling P, Kendall G, Han L (2002) An Adaptive Length Chromosome Hyperheuristic Genetic Algorithm for a Trainer Scheduling Problem. In proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL '02), 267-271.
10. DeSmet G (2007) ITC-2007- Examination Track Drools-solver. http://www.cd.qub.ac.uk/winner/bestexamsolutions/Geoffrey_De_smet_examination_description.pdf, last accessed 12 February 2010.

11. Gogos C, Alefragis P, Housoss E (2008) A Multi-Stage Algorithmic Process for the Solution of the Examination Timetabling Problem. In the proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2008),Montreal,http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Gogos-HC2b.pdf, last accessed 12 February 2010.
12. Han L, Kendall G (2003) Guided Operators for a Hyper-Heuristic for Genetic Algorithms. In proceedings of AI-2003 Advances in Artificial Intelligence-The 16th Australian Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence, 2903, 807-820.
13. Koza JR (1992) Genetic Programming I: On the Programming of Computers by Means of Natural Selection, MIT Press.
14. McCollum B (2007) International Timetabling Competition: Evaluation Function. http://www.cs.qub.ac.uk/itc2007/examtrack/, last accessed 12 February 2010.
15. McCollum B, McMullan P, Paechter B, Lewis R, Schaerf A, Di Gaspero L, Parkes AJ, Qu R, Burke EK (2009a) Setting the Research Agenda in Automated Timetabling: The Second International Timetabling Competition. INFORMS Journal on Computing, doi: 101287/ijoc.1090.0320.
16. McCollum B, McMullan PJ, Parkes AJ, Burke EK, Abdullah S (2009b) An Extended Great Deluge Approach to the Examination Timetabling Problem. In proceedings of the Multidisciplinary International Conference on Scheduling: Theory and Application (MISTA 2009), 424-434.
17. Muller T (2008) ITC2007 Solver Description: A Hybrid Approach. In the proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT'08), http://w1.cirrelt.ca/~patat2008/PATAT_7_PROCEEDINGS/Papers/Muller-HC1c.pdf, last accessed 12 February 2010.
18. Pillay N, Banzhaf W (2007) A Genetic Programming Approach to the Generation of Hyper-Heuristics for the Uncapacitated Examination Timetabling Problem. In Neves et al. (eds.), Progress in Artificial Intelligence, Lecture Notes in Artificial Intelligence, 4874, 223-234.
19. Pillay N (2008) A Developmental Approach to the Examination Timetabling Problem. http://www.cd.qub.ac.uk/winner/bestexamsolutions/pillay.pdf, last accessed 12 February 2010.
20. Pillay N (2009) Evolving Hyper-Heuristics for the Uncapacitated Examination Timetabling Problem. In proceedings of Multidisciplinary International Conference on Scheduling 2009 (MISTA 2009), 409-422.
21. Pillay N, Banzhaf W (2009) A Study of Heuristic Combinations for Hyper-Heuristic Systems for the Uncapacitated Examination Timetabling Problem. European Journal of Operational Research(EJOR), 197, 482-491.
22. Qu R, Burke EK (2005) Hybrid Neighbourhood HyperHeuristics for Exam Timetabling Problems. In Proceedings of the MIC2005: The Sixth Metaheuristics International Conference, Vienna, Austria.

23. Qu, R, Burke EK (2009a) Hybridizations within a Graph Based Hyper-Heuristic Framework for University Timetabling Problems. Journal of Operational Research Society (JORS), 60, 1273-1285.
24. Qu R, Burke EK, McCollum B (2009b) Adaptive Automated Construction of Hybrid Heuristics for Exam Timetabling and Graph Colouring Problems. European Journal of Operational Research (EJOR), 198(2), 392-404.
25. Ross P (2005) Hyper-heuristics. In Burke E.K., Kendall G. (eds): Search Methodologies: Introductory Tutorials in Optimization and Decision Support Methodologies, chapter 17. Kluwer, 529 -556.