An Extended Implementation of the Great Deluge Algorithm for Course Timetabling

Paul McMullan

School of Electronics, Electrical Engineering and Computer Science,
Queen's University of Belfast, Northern Ireland
p.p.mcmullan@qub.ac.uk

Abstract. Course Scheduling consists of assigning lecture events to a limited set of specific timeslots and rooms. The objective is to satisfy as many soft constraints as possible, while maintaining a feasible solution timetable. The most successful techniques to date require a compute-intensive examination of the solution neighbourhood to direct searches to an optimum solution. Although they may require fewer neighbourhood moves than more exhaustive techniques to gain comparable results, they can take considerably longer to achieve success. This paper introduces an extended version of the Great Deluge Algorithm for the Course Timetabling problem which, while avoiding the problem of getting trapped in local optima, uses simple Neighbourhood search heuristics to obtain solutions in a relatively short amount of time. The paper presents results based on a standard set of benchmark datasets, beating over half of the currently published best results with in some cases up to 60% of an improvement.

Keywords: Course Scheduling, Great Deluge, Timetabling, Neighbourhood Search.

1 Introduction

The Course Timetabling problem deals with the assignment of course (or lecture events) to a limited set of specific timeslots and rooms, subject to a variety of hard and soft constraints. All hard constraints must be satisfied, obtaining a feasible solution, while as many soft constraints as possible must be satisfied. The feasible solution which satisfies all of the soft constraints as is possible can be considered optimal. The quality of any course timetable solution is based on how close to this goal it approaches. The set of all feasible solutions is vast; given the number of permutations possible for even a modestly sized problem, an exhaustive search for an optimum solution will take an impractical amount of time. Directed searches which attempt to improve the quality of one or more initial solutions can cut down the amount of time needed to reach a solution of acceptable quality. However, although compute-intensive Neighbourhood search heuristics can provide acceptable results, the time taken to obtain them may be impractical for real-world applications [1]. This paper proposes an extension of an improvement algorithm based on the Great Deluge

Y. Shi et al. (Eds.): ICCS 2007, Part I, LNCS 4487, pp. 538 – 545, 2007.

[2] which will employ simple (non-compute-intensive) heuristics while providing better or comparable results to successful techniques outlined in the current literature.

2 The University Course Timetabling Problem

The problem involves the satisfaction of a variety of hard and soft constraints [3]. The following Hard Constraints, which must be satisfied in order to achieve a feasible timetable solution, are presented:

- *No student can be assigned to more than one course at the same time.*
- *The room should satisfy the features required by the course.*
- *The number of students attending the course should be les than or equal to the capacity of the room.*
- *No more than one course is allowed to a timeslot in each room.*

The following soft constraints are used as measure of the overall penalty for a solution, calculated in terms of the total number of students who:

- *Have a course scheduled in the last timeslot of the day.*
- *Have more than 2 consecutive courses.*
- *Have a single course on a day.*

In general, the problem has the following characteristics:

- *A set of N courses, e = {e1,…,eN}*
- *45 timeslots*
- *A set of R rooms*
- *A set of F room features*
- *A set of M students.*

An improvement in the quality of any solution for the course timetabling problem is concerned with the minimisation of soft constraint violations, which directly includes the number of students affected by the violations. The objective cost function is calculated as the sum of the number of violations of the soft constraints.

A full description of the problem has been listed by the EU Metaheuristics Network [4]. A set of eleven specification data sets were subsequently released [3] for the purposes of Benchmarking, and in 2002 an international timetabling competition was run using another twenty sets of data [5] in order to encourage the production of innovative techniques in the automation of course timetable construction.

Recent literature concentrated in automating the process of course timetable creation has described the application of several techniques and heuristics, with varying measures of success. [6] provides an introduction to recent work in the area. Hill-Climbing [7] is the simplest local search algorithm, which is involved in improving a current solution by accepting a candidate solution from the neighbourhood only if it of a better or equivalent fitness to the current. This has the disadvantage of getting trapped quite quickly in local optima, most often with a poor quality solution.

Extensions to the simple Hill-climbing algorithm involve allowing the acceptance of worse solutions in order to at some point get better ones, or directing Neighbourhood moves according to some criteria dictated by initial or iteration-dependent parameters. Simulated annealing [8] accepts worse solutions with a probability: $P = e^{-\Delta P/T}$, where $\Delta P=f(s^*)-f(s)$ and the parameter T represents the temperature, analogous to the temperature in the process of annealing. The temperature is reduced according to a cooling rate which allows a wider exploration of the solution space at the beginning, avoiding getting trapped in a local optimum. This has been applied to a number of application areas [9]. Tabu-Search [10] modifies the neighbourhood structure of each solution as searching progresses, using a tabu list to exclude solutions or attributes of solutions in order to escape local optima.

Variable Neighbourhood Search (VNS), introduced by Mladevonic and Hansen [11] uses more than one neighbourhood structure which change systematically during the search. This increases the probability of escaping local optima, in that neighbourhoods which are different from the current solution can be considered. The use of composite neighbourhood structures is described in [12], and involves choosing solutions based on a Monte Carlo acceptance criteria [13] from the most effective of a choice of multiple neighbourhood structures.

Evolutionary techniques such as Genetic Algorithms have been employed in this problem area [14] to improve multiple "populations" of course timetable solutions based on selection of the fittest, using selection, crossover and mutation operators. Further experimentation involves the use of hybrid evolutionary approaches and memetic algorithms [14], [15]. These have also been combined with Meta- and Hyper-heuristic techniques [16], [17] which employ high-level heuristics to dictate the choice of heuristics when searching the solution space. Socha et al investigate the use of ant colony optimisation methodologies to the course timetabling problem [18], [19], using the idea of pheromone trails to reinforce successful orderings or moves over those less successful in improving solutions. Other techniques involve the use of Multi-Objective or Multi-Criteria approaches to reaching an optimal solution [15] whereby the quality of a solution is considered as a vector of individual objectives rather than as a single composite sum, weighted according to the importance of each. Fuzzy methodologies have been successfully applied to examination scheduling [20] and more recently to course timetabling problems [21].

The Great Deluge (also known as Degraded Ceiling) was introduced by Dueck [2] as an alternative to Simulated Annealing. This uses a Boundary condition rather than a probability measure with which to accept worse solutions. The Boundary is initially set slightly higher than the initial solution cost, and reduced gradually through the improvement process. New solutions are only accepted if they improve on the current cost evaluation or are within the boundary value. This has been applied successfully to construction and improvement techniques in timetabling [22], [23]. [15] describes a modification of the Great Deluge algorithm for Multi-criteria decision making.

3 The Extended Great Deluge Algorithm

The standard Great Deluge algorithm has been extended to allow a reheat, similar to that employed with simulated annealing in timetabling [15]. The aim of this approach

is to both improve the speed at which an optimal solution can be found and at the same time utilise the benefits of this technique in avoiding the trap of local optima. In order to reduce the amount of time taken, relatively simple neighbourhood moves are employed.

Generally, the Great Deluge or Simulated Annealing processes will terminate when a lack of improvement has been observed for a specified amount of time, as the most optimal solution will have been reached. Rather than terminating, the Extended GD will employ the reheat to widen the boundary condition to allow worse moves to be applied to the current solution. Cooling will continue and the boundary will be reduced at a rate according to the remaining length of the run. The algorithm for the Extended Great Deluge is presented in Figure 1.

```
Set the initial solution s using a construction heuristic;
Calculate initial cost function f(s)
Set Initial Boundary Level B0 = f(s)
Set initial decay Rate ΔB based on Cooling Parameter
While stopping criteria not met do
Apply neighbourhood Heuristic S* on S
      Calculate f(s*)
        If f(s*) <= f(s) or (f(s*) <= B Then
          Accept s = s*
        Lower Boundary B = B – ΔB
        If no improvement in given time T Then
          Reset Boundary Level B0 = f(s)
          Set new decay rate ΔB based on Secondary
              Cooling Parameter
```

Fig. 1. Extended Great Deluge Algorithm

The initial solution construction is handled with an Adaptive (Squeaky-Wheel) ordering heuristic [22] technique. This utilises a weighted order list of the events to be scheduled based on the individual penalties incurred during each iteration of construction. The adaptive heuristic does not attempt to improve the solution itself, but simply continues until a feasible solution is found.

The first parameter used within the Extended Great Deluge is the initial decay rate, which will dictate how fast the Boundary is reduced and ultimately the condition for accepting worse moves is narrowed. The approach outlined in this paper uses a Decay Rate proportional to 50% of the entire run. This will force the algorithm to attempt to reach the optimal solution by, at the very most, half-way through the process. Generally, a continuous lack of improvement will occur before this is reached, at which time the re-heat mechanism is activated.

The 'wait' parameter which dictates when to activate the re-heat mechanism due to lack of improvement can be specified in terms of percentage or number of total moves in the process. Through experimentation with a number of data set instances a general value for this parameter can be established. After reheat the Boundary ceiling is once again set to be greater than the current best evaluation by a similar percentage to that applied in the initial boundary setting. The subsequent decay is set to a 'quicker' rate than with the initial decay, in order to increase the speed of the exploration of neighbouring solutions for improvement. The general setting chosen for the algorithm

outlined is set to 25% of the remaining time, with the improvement wait time remaining unchanged. The neighbourhood structures employed in the process are deliberately kept simple, to ensure the process is not protracted with time-consuming explorations of the neighbouring solution space. The two heuristics used are Move (random event is moved to a random timeslot) and Swap (two random events swap timeslots), while ensuring that a feasible solution is maintained. There are approximately 2 Moves made for each Swap.

4 Experiments and Results

The algorithm was tested on a range of standard benchmark data sets [3]. These are divided into a number of categories accordingly on the amount of data required to be taken into account as part of the scheduling problem. This gives some indication of the comparative complexity of each, and hence the amount of hard and soft constraints which must be considered. For instance, Small data sets require 100 Courses to be scheduled, whereas Medium and Large have 400 Courses.

Table 1. Results for Multiple-run tests of Algorithm

Data Set	*EGD*		*CNS*	*VNS with Tabu*	*Local Search*	*Ant*	*Tabu HH*	*Grph HH*
	Best	*Avg*						
Small1	0 (x 4)	0.8	**0**	**0**	8	1	1	6
Small2	0 (x 2)	2	**0**	**0**	11	3	2	7
Small3	0 (x 4)	1.3	**0**	**0**	8	1	**0**	3
Small4	0 (x 5)	1	**0**	**0**	7	1	1	3
Small5	0 (x 8)	0.2	**0**	**0**	5	**0**	**0**	4
Medium1	80	**101.4**	242	317	199	195	146	372
Medium2	105	**116.9**	161	313	202.5	184	173	419
Medium3	139	**162.1**	265	357	77.5% Inf	248	267	359
Medium4	88	**108.8**	181	247	177.5	164.5	169	348
Medium5	88	**119.7**	151	292	100% Inf	219.5	303	171
Large	730	**834.1**	100 % Inf	100 % Inf	100% Inf	851.5	80% Inf 1166	1068

The algorithm was implemented on a Pentium PC Machine using Visual Studio .NET on a Windows XP Operating System. For each benchmark Data Set the algorithm was run for 200,000 evaluations with 10 test-runs to obtain an average value. Table 1 shows the comparison of our approach with others available in the literature. They include a local search method and ant algorithm [18], [19], Tabu-search hyper-heuristic and Graph hyper-heuristic [10], [20], Variable Neighbourhood Search with Tabu list and Composite Neighbourhood Search [12].

The Extended Great Deluge has beaten all current results for the *Medium* data sets, with approximately 60% improvement given the average cost evaluation obtained over the 10 test runs. The *Large* data set has a marginal improvement on the previous best. Results for the *Small* data sets do not improve on the previously published results, but do compare favourably. It should be noted that in all cases for *Small*, evaluations of zero were obtained, with some achieved more frequently than others.

Between each of the *Small* data sets, if we compare the frequency of achieving zero evaluation, it can be noted that our algorithm achieves zero more often on those which could be considered less difficult (based on the results obtained by the other techniques included in the table) and vice versa. The time taken to achieve the best solution for the data sets ranged between 15 and 60 seconds.

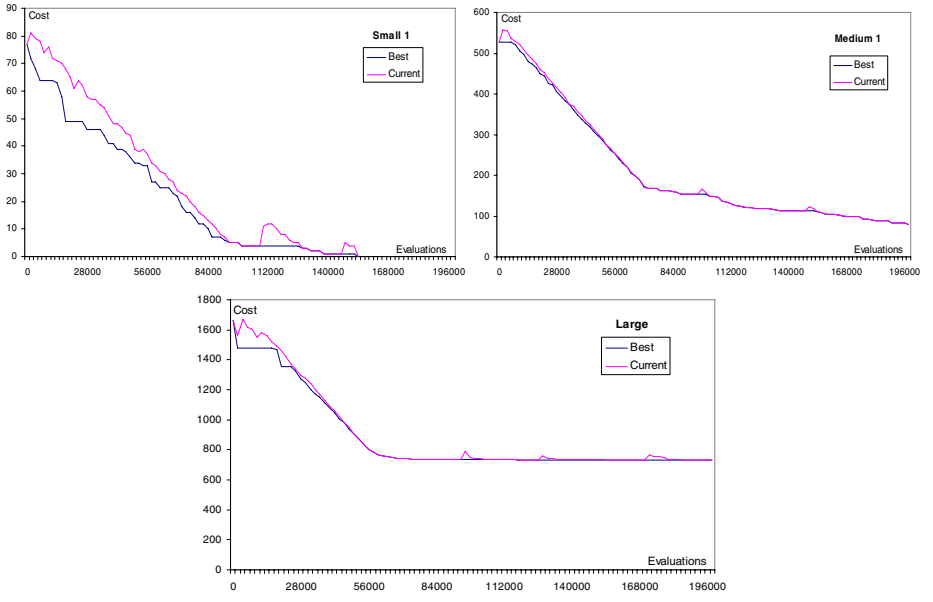

Fig. 2a, b and c. Performance of boundary for Small, Medium and Large datasets

Figures 2a, b and c illustrate the effectiveness of the reheat and rapid cooling mechanism. The initial state at which further improvement is no longer achieved is reached before the process is 50% complete. The reheat clearly indicates that further improvement is achieved. In the case of *Small*, in most cases this allows the technique to explore a number of severe non-improvement moves and achieve a zero evaluation. The technique is also very effective for the *Medium* data sets, with the graphs clearly indicating that as much as a further reduction of 50% in the cost evaluation can be achieved. This is less effective for the *Large* data set, although some improvement as a result of the reheat can be observed. Some initial investigation was also carried out on the Competition data instances [5]. Adhering to the time restrictions dictated by the rules of the competition, it was observed that the Extended Great Deluge performed comparably to results obtained by the top 3 entrants.

The nature of how the Boundary conditions are set within the Great Deluge algorithm can explain to some extent the problems in achieving a zero evaluation consistently for the *Small* data sets. The Boundary is set as a percentage of the current or best evaluation initially or when a reheat takes place. A common percentage value for setting the boundary is 110% of the current best evaluation. When the evaluation tends towards zero, any new boundary value will be quite close or equal to the best

evaluation (in terms of rounding to whole numbers). Therefore the boundary becomes ineffective in allowing worse moves to be accepted towards escaping local optima.

5 Conclusions and Future Work

This paper has introduced a variant of the Great Deluge (Degraded Ceiling) algorithm which attempts to maximise the inherent advantages with this technique in escaping from local optima while also maintaining a relatively simple set of neighbourhood moves, and hence reducing the time required for an iterative search process. As has been shown, this approach has been successful in achieving a general improvement in solution evaluation as compared to currently published results for the course timetabling benchmark data sets, given a common termination criterion. This represents a significant contribution to the area of Course timetabling.

A much wider analysis of the technique is proposed in order to determine whether further improvement can be achieved by modification of the associated parameters and variables used in the process, while retaining generality for all data sets. In order to help ensure generality is maintained, it is intended to include the Competition data sets along with the standard benchmarks for comparison. The variables which may affect the results include Construction Time and Initial Ordering heuristics, Initial and post-reheat decay rates, Non-Improvement 'wait' time, Neighbourhood search heuristics (initially will involve modification of the Move / Swap ratio and pattern), Initial Boundary evaluation factor and boundary / multiple non-improvement heuristics.

It is also clear that the technique will need to resolve the issues identified when processing smaller data sets or indeed larger problems for which solutions approaching an evaluation of zero are found. Current experimentation is involved with activating a more directed set of neighbourhood structures in these circumstances, at the expense of some computation time, towards the goal of achieving zero evaluation. Further results from all experimentation outlined will be published in due course.

References

1. B. McCollum, "University Timetabling: Bridging the Gap between Research and Practice", Proceedings of the 6th international conference on the Practice and Theory of Automated Timetabling, Brno, (2006), 15-35.
2. G. Dueck, "Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing", J. Computational Physics, Vol. 90, (1990), 161-175.
3. B. Paechter, Website: http://www.dcs.napier.ac.uk/~benp/
4. Metaheuristic Network Website: http://www.metaheuristics.org/
5. International Timetabling Competition Website: http://www.idsia.ch/Files/ttcomp2002/
6. E. K. Burke, , M. Trick, (eds), The Practice and Theory of Automated Timetabling V: Revised Selected Papers from the 5th International conference, Pittsburgh 2004, Springer Lecture Notes in Computer Science, Vol. 3616. (2005).

7. J. S. Appleby, D. V. Blake, E. A. Newman, "Techniques for Producing School Timetables on a Computer and their Application to other Scheduling Problems", The Computer Journal, Vol. 3, (1960), 237-245.
8. S. Kirkpatrick, J. C. D. Gellat, M. P. Vecci, "Optimization by Simulated Annealing", Science, Vol. 220, (1983), 671-680.
9. C. Koulmas, S. R. Antony, R. Jaen, "A Survey of Simulated Annealing Applications to Operational Research Problems", Omega International Journal of Management Science, Vol. 22, (1994), 41-56.
10. E.K. Burke, G. Kendall, E. Soubeiga, "A Tabu-Search Hyperheuristic for Timetabling and Rostering", Journal of Heuristics, 9(6): (2003), 451-470.
11. N. Mladenovic, P. Hansen, "Variable Neighbourhood Search", Computers and Operations Research, 24(11), (1997), 1097-1100.
12. S. Adbullah, E. K. Burke, B. McCollum, "Using a Randomised Iterative Improvement Algorithm with Composite Neighbourhood Structures for University Course Timetabling", Comp. Science Interfaces Book Series (Eds. Doerner, K.F., Gendreau, M., Greistorfer, P., Gutjahr, W.J., Hartl, R.F. & Reimann, M.), Springer Operations Research (2006).
13. M. Ayob, G. Kendall, "A Monte Carlo Hyper-Heuristic to Optimise Component Placement Sequencing for Multi-head Placement Machine", Proceedings of the International Conference on Intelligent Technologies, InTech '03, (2003), 132-141.
14. E. K. Burke, J. P. Newall, "A Multi-Stage Evolutionary Algorithm for the Timetable Problem", IEEE Transactions on Evolutionary Computation, Vol 3.1, (1999), 63-74.
15. S. Petrovic, E. K. Burke, "University Timetabling", Handbook of Scheduling: Algorithms, Models and Performance Analysis, CRC Press, Chapter 45, (2004)
16. O. Rossi-Doria, B. Paechter, "An Hyperheuristic Approach to Course Timetabling Problem using an Evolutionary Algorithm", Technical Report, Napier University, Edinburgh, Scotland, (2003)
17. C. Head, S. Shaban, "A heuristic approach to simultaneous course/student timetabling", Computers & Operations Research, 34(4): (2007), 919-933.
18. K. Socha, J. Knowles, M. Sampels, "A Max-Min Ant System for the University Course Timetabling Problem", Proceedings of the 3rd International Workshop on Ant Algorithms, ANTS 2002, Springer Lecture Notes in Computer Science, Vol. 2463, Springer-Verlag, (2002), 1-13.
19. K. Socha, M. Sampels, M. Manfrin, "Ant Algorithms for the University Course Timetabling Problem with regard to the State-of-the-Art", Proceedings of the 3rd International Workshop on Evolutionary Computation in Combinatorial Optimization (EVOCOP 2003), United Kingdom, (2003), 335-345.
20. E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. Qu, "A Graph-based Hyper Heuristic for Timetabling Problems", European Journal of Operational Research, 176: (2007), 177-192.
21. H. Asmuni, E. K. Burke, J. M. Garibaldi, "Fuzzy Multiple Heuristic Ordering for Course Timetabling", Proceedings of 5th UK Workshop on Computational Intelligence (UKCI '05), (2005), 302-309.
22. E. K. Burke, Y. Bykov, J. P. Newall, S. Petrovic, "A Time-Predefined Approach to Course Timetabling", Yugoslav J. of Operational Research (YUJOR), 13(2): (2003) 139-151.
23. E. K. Burke, Y. Bykov, J. P. Newall, S. Petrovic, "A Time-Predefined Local Search Approach to Exam Timetabling Problems", IIE Transactions, 36(6), (2004), 509-528.