Evolutionary Non-linear Great Deluge for University Course Timetabling

Dario Landa-Silva and Joe Henry Obit

Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science, The University of Nottingham, UK
{jds,jzh}@cs.nott.ac.uk

Abstract. This paper presents a hybrid evolutionary algorithm to tackle university course timetabling problems. The proposed approach is an extension of a non-linear great deluge algorithm in which evolutionary operators are incorporated. First, we generate a population of feasible solutions using a tailored process that incorporates heuristics for graph colouring and assignment problems. That initialisation process is capable of producing feasible solutions even for the large and most constrained problem instances. Then, the population of feasible timetables is subject to a steady-state evolutionary process that combines mutation and stochastic local search. We conduct experiments to evaluate the performance of the proposed hybrid algorithm and in particular, the contribution of the evolutionary operators. Our results show that the hybrid between non-linear great deluge and evolutionary operators produces very good results on the instances of the university course timetabling problem tackled here.

Keywords: hybrid evolutionary algorithm, non-linear great deluge, course timetabling.

1 Introduction

Finding good quality solutions for timetabling problems is a very challenging task due to the combinatorial and highly constrained nature of these problems [10]. In recent years, several researchers have tackled the course timetabling problem, particulary the set of 11 instances proposed by Socha et al. [14]. Among the algorithms proposed there are: a MAX-MIN ant system by Socha et al. [14]; a tabu search hyper-heuristic strategy by Burke et al. [7]; an evolutionary algorithm, ant colony optimisation, iterated local search, simulated annealing and tabu search by Rossi-Doria et al. [13]; fuzzy multiple heuristic ordering by Asmuni et al. [5]; variable neighbourhood search by Abdullah et al. [1]; iterative improvement with composite neighbourhoods by Abdullah et al. [2,4]; a graph-based hyper-heuristic by Burke et al. [9] and a hybrid evolutionary algorithm by Abdullah et al. [3].

This paper proposes a two-stage hybrid meta-heuristic approach to tackle course timetabling problems. The first stage constructs feasible timetables while the second stage is an improvement process that also operates within the feasible

E. Corchado et al. (Eds.): HAIS 2009, LNAI 5572, pp. 269–276, 2009.

region of the search space. The second stage is a combination of non-linear great deluge [12] with evolutionary operators to improve the quality of timetables by reducing the violation of soft constraints.

The rest of this paper is organised as follows. In Section 2, the subject problem and test instances are described. Then, Section 3 gives details of the proposed hybrid evolutionary algorithm. Results and experiments are presented and discussed in Section 4 while conclusions are given in Section 5. The key contributions of this paper are: an initialisation heuristic that generates feasible timetables everytime and a simple yet effective hybrid evolutionary algorithm that is very competitive with much more elaborate algorithms already presented in the literature.

2 University Course Timetabling

In general, university course timetabling is the process of allocating, subject to predefined constraints, a set of limited timeslots and rooms to courses, in such a way as to achieve as close as possible a set of desirable objectives. In timetabling problems, constraints are commonly divided into *hard* and *soft*. A timetable is said to be feasible if no hard constraints are violated while soft constraints may be violated but we try to minimise such violation in order to increase the quality of the timetable. In this work, we tackle the course timetabling problem defined by Socha et al. [14] where there are: n events $E = \{e_1, e_2, \ldots, e_n\}$, k timeslots $T = \{t_1, t_2, \ldots, t_k\}$, m rooms $R = \{r_1, r_2, \ldots, r_m\}$ and a set S of students. Each room has a limited capacity and a set F of features that might be required by events. Each student must attend a number of events within E. The problem is to assign the n events to the k timeslots and m rooms in such a way that all hard constraints are satisfied and the violation of soft constraints is minimised.

Hard Constraints. There are four in this problem: (1) a student cannot attend two events simultaneously, (2) only one event can be assigned per timeslot in each room, (3) the room capacity must not be exceeded at any time, (4) the room assigned to an event must have the features required by the event.

Soft Constraints. There are three in this problem: (1) students should not have exactly one event timetabled on a day; (2) students should not have to attend more that two consecutive events on a day; (3) students should not have to attend an event in the last timeslot of the day.

The benchmark data sets proposed by Socha et al. [14] are split according to their size into 5 small, 5 medium and 1 large. For the small instances, $n = 100$, $m = 5$, $|S| = 80$, $|F| = 5$. For the medium instances, $n = 400$, $m = 10$, $|S| = 200$, $|F| = 5$. For the large instances, $n = 400$, $m = 10$, $|S| = 400$, $|F| = 10$. For all instances, $k = 45$ (9 hours in each of 5 days). It should be noted that although a timetable with zero penalty exists for each of these problem instances (the data sets were generated starting from such a timetable [14]), so far no heuristic method has found the ideal timetable for the medium and large instances. Hence, these data sets are still very challenging for most heuristic search algorithms.

3 Evolutionary Non-linear Great Deluge Algorithm

3.1 The Hybrid Strategy

We now describe the overall hybrid strategy, an extension of our previous algorithm which maintains a single-solution during the search [12]. Here, we extend that algorithm to a population-based evolutionary approach by incorporating tournament selection, a mutation operator and a replacement strategy.

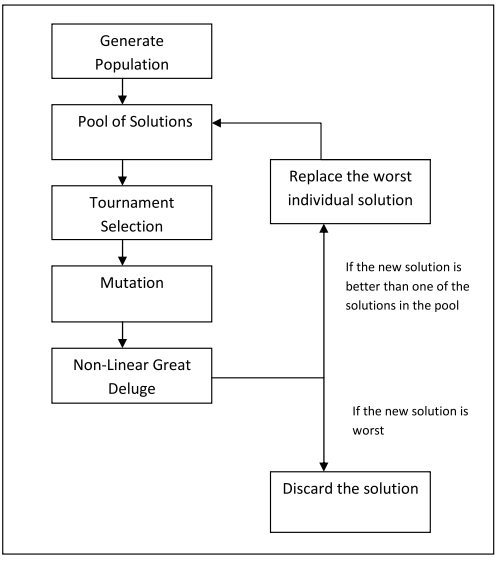

Fig. 1. Evolutionary Non-linear Great Deluge Algorithm

Figure 1 shows the components of this hybrid algorithm. It begins by generating an initial population which becomes the pool of solutions. Then, in each generation the algorithm works as follows. First, tournament selection takes place where 5 individuals are chosen at random and the one with the best fitness is selected (x^t). Then, a mutation operator is applied to x^t while maintaining feasibility obtaining solution x^m. This is followed by applying the non-linear great deluge algorithm to x^m to obtain an improved solution x^i. Then, the worst solution in the pool of solutions, x^w (ties broken at random) is identified and if x^i is better than x^w then x^i replaces x^w. This hybrid algorithm is executed for a pre-determined amount of computation time according to the size of the problem instance. Note that this is a steady-state evolutionary approach that uses non-linear great deluge for intensification and a mutation operator for diversification. The following subsections describe each of the algorithm components is more detail.

3.2 Initialisation Heuristic

The pseudo-code for the initialisation heuristic is shown in Algorithm 1. Two well-known graph colouring heuristics are incorporated. Largest degree (LD) refers to the event with the largest number of conflicting events. Saturation degree (SD) of an event refers to the number of available timeslots to timetable that event without conflicts in the current partial solution.

Algorithm 1. Initialisation Heuristic for Course Timetabling

```
Input: set of events in the unscheduled events list E
Sort unscheduled events list using LD
while (unscheduled events list is not empty) do
    Choose event E_j with the LD
    Calculate SD for event E_j
    if (SD equals zero) then
        Select a timeslot t at random
        From those events already scheduled in timeslot t (if any), move those that conflict
        with event E_j (if any) to the rescheduled events list
        Place E_j into timeslot t
        for (each event E_i in the rescheduled events list with SD different to zero) do
            Find a timeslot t at random to place E_i
            Recalculate SD for all events in the rescheduled events list
        Move all events that remain in the rescheduled events list (those with SD equal to
        zero) to the unscheduled events list
    else
        Find a timeslot t at random to place E_j
    if (unscheduled events list is not empty and time_U has elapsed) then
        One by one, place events from the unscheduled events list into any random selected
        timeslot
while (solution not feasible) do
    Select move move_1 or move move_2 at random and then perform Local Search
    if (solution not feasible and loop >10) then
        Identify an event E_i that violates hard constraints
        Apply Tabu Search for ts_max iterations using move move_1 to reschedule E_i (loop
        is reset to zero at the end of each tabu search)
    loop++;
Output: A feasible solution (timetable)
```

In the first while loop, the initialisation heuristic attempts to place all events into timeslots while avoiding conflicts. In order to do that, the heuristic uses the saturation degree criterion and a list of rescheduled events to temporarily place conflicting events. The heuristic tries to do this for a given $time_U$ but once that time has elapsed, all remaining unscheduled events are placed into random timeslots. That is, if by the end of the first while loop the solution is not yet feasible, at least the penalty due to hard constraint violations is already very low. In the second while loop, the heuristic uses simple local search and tabu search to achieve feasibility. Two neighbourhood moves $move_1$ and $move_2$ (described below) are used. The local search attempts to improve the solution but it also works as a disturbing operator, hence the reason for the maximum of 10 trials before switching to tabu search. The tabu search uses the move $move_1$ only and is carried out for a fixed number of iterations ts_{max}. In our experiments, this initialisation heuristic always finds a feasible solution for all the problem instances considered.

3.3 Mutation Operator

With a probability equal to 0.5, the mutation operator is applied to the solution selected from the tournament (x^t). The mutation operator selects at random 1 out of 3 types of neighbourhood moves in order to change the solution while maintaining feasibility. These moves are described below.

- $move_1$. Selects one event at random and assigns it to a feasible timeslot and room. Note that in the tabu search step of Algorithm 1, $move_1$ selects only events that violate hard constraints.
- $move_2$. Selects two events at random and swaps their timeslots and rooms while ensuring feasibility is maintained.
- $move_3$. Selects three events at random, then it exchanges the position of the events at random and ensuring feasibility is maintained.

3.4 Non-linear Great Deluge

The standard great deluge algorithm was developed by Dueck [11]. The basic idea behind this approach is to explore neighbour solutions accepting those that are not worse than the current one. Otherwise, the candidate solution is accepted only if its penalty is not above a pre-defined water level which decreases linearly as the search progresses. The water level determines the speed of the search. The higher the decay rate the faster the water level goes down and the faster the algorithm terminates. Burke et al. [6] proposed to initialise the water level equal to the initial cost function. The decay rate at each iteration is constant and they interpreted the parameter as a function of the expected search time and the expected solution quality. To calculate the decay rate B, they first estimate the desired result (solution quality) $f(S^{'})$ and then calculate $B = B_0 - f(S^{'})/\text{Number of moves}$. In our previous work [12], we proposed a great deluge approach in which the decay rate of the water level is non-linear and is determined by the following expression:

$$B = B \times (\exp^{-\delta(rnd[min,max])}) + \beta \tag{1}$$

The parameters in Eq. (1) control the speed and the shape of the water level decay rate. Therefore, the higher the values of min and max the faster the water level decreases. In return, the improvement is quickly achieved but it will suffer from this greediness by trapping itself in local optima. To counterbalance this greediness, floating the water level (relaxation) is necessary. Then, in addition to using a non-linear decay rate for the water level B, we also allow B to go up when its value is about to converge with the penalty cost of the candidate solution. We increase the water level B by a random number within the interval $[B_{min}, B_{max}]$. For all instances types, the interval used was $[2, 4]$. Full details of this strategy to control the water level decay rate in the modified great deluge can be seen in [12].

Then, the non-linear great deluge component in Figure 1 is our previous single-solution approach which produced very good results on the same instances of the

course timetabling problem considered here (see [12]). In this paper we show that by incorporating some components from evolutionary algorithms, the resulting hybrid approach produces some better results than those currently reported in the literature.

4 Experiments and Results

We now evaluate the performance of the proposed hybrid algorithm. For each type of dataset, a fixed computation time ($time_{max}$) in seconds was used as the stopping condition: 2600 for small problems, 7200 for medium problems and 10000 for the large problem. This fixed computation time is for the whole process including the construction of the initial population. In the initialisation heuristic (Algorithm 1), we set $time_U$ to 0.5, 5 and 25 seconds for small, medium and large instances respectively and $ts_{max} = 500$ iterations for all problem instances. We executed the proposed hybrid algorithm 10 times for each problem instance.

Table 1 shows the average results obtained by the previous Non-linear Great Deluge, the Evolutionary Non-linear Great Deluge proposed here, and the results from other algorithms reported in the literature. For each dataset, the best results are indicated in bold. The main goal of this comparison is to assess whether extending the non-linear great deluge to a hybrid evolutionary approach helps to produce better solutions for the course timetabling problem.

We can see in Table 1 that the hybrid evolutionary algorithm described here (ENGD) clearly outperforms our previous single-solution algorithm (NGD). It

Table 1. Comparison of results obtained by the approach proposed in this paper against the best known results from the literature on the course timetabling problem. ENGD is the Evolutionary Non-Linear Great Deluge proposed here, ENGD(-m) is ENGD without Mutation, NGD is the Non-Linear Great Deluge in [12], MMAS is the MAX-MIN Ant System in [14], CFHH is the Choice Function Hyper-heuristic in [7], VNS-T is the Hybrid of VNS with Tabu Search in [2], HEA is the Hybrid Evolutionary Algorithm in [3]. In the first column, S1-S5 are small problem instances, M1-M5 are medium problem instances, while L1 is the large problem instance.

	NGD	ENGD(-m)	ENGD	Best Known
S1	3	3	**0**	**0** (VNS-T)
S2	4	2	1	**0** (VNS-T)
S3	6	2	**0**	**0** (CFHH)
S4	6	3	**0**	**0** (VNS-T)
S5	**0**	**0**	**0**	**0** (MMAS)
M1	140	157	**126**	146 (CFHH)
M2	130	178	**123**	147 (HEA)
M3	189	240	**185**	246 (HEA)
M4	112	152	116	164.5 (MMAS)
M5	141	142	**129**	130 (HEA)
L1	876	995	821	**529** (HEA)

is also evident that the tailored mutation operator makes a significant contribution to the good performance of ENGD as the results obtained by ENGD(-m) are considerably worse. The proposed hybrid evolutionary approach matches the best known solution quality for almost all small problem instances except S2 and improves the best known results for most medium instances except M4. Only on the case of the large problem instance, we see that our algorithm does not match the best known result. It is also important to stress that the proposed initialisation heuristic is an important component of this hybrid algorithm because finding a feasible solution is crucial when tackling course timetabling problems.

Overall, this experimental evidence shows that by combining some key evolutionary components and an effective stochastic local search procedure, we have been able to produce a hybrid evolutionary approach that is still quite simple but much more effective (than the single-solution stochastic local search) in generating best known solutions for a well-known set of difficult course timetabling problem instances. The proposed algorithm seems particularly effective on small and medium problem instances.

5 Conclusions

Solving timetabling problems remains a challenge to many heuristic algorithms. In this paper, we tackled a well-known set of benchmark instances of the university course timetabling problem. Previous to this work, several algorithms ranging from relatively simple iterative neighbourhood search procedures [2] to more elaborate hyper-heuristic approaches [9] have been applied to this problem. We extended our previous approach, a single-solution non-linear great deluge algorithm, towards an evolutionary variant by incorporating some key operators like a population of solutions, tournament selection, a mutation operator and a steady-state replacement strategy. The results from our experiments provide evidence that our hybrid evolutionary algorithm is capable of producing best known solutions for a number of the test instances used here. The tailored mutation operator which uses 3 neighbourhood moves seems to make a substantial contribution to the good performance of the proposed algorithm. Obtaining the best timetables (with penalty equal to zero) for the medium and large instances is still a challenge. Our future work contemplates the investigation of cooperative strategies and information sharing mechanisms to tackle these university course timetabling problems.

References

1. Abdullah, S., Burke, E.K., McCollum, B.: An Investigation of Variable Neighbourhood Search for University Course Timetabling. In: Proceedings of MISTA 2005: The 2nd Multidisciplinary Conference on Scheduling: Theory and Applications, pp. 413–427 (2005)
2. Abdullah, S., Burke, E.K., McCollum, B.: Using a Randomised Iterative Improvement Algorithm with Composite Neighbourhood Structures for University Course Timetabling. In: Proceedings of MIC 2005: The 6th Meta-heuristic International Conference, Vienna, Austria, August 22-26 (2005)

3. Abdullah, S., Burke, E.K., McCollum, B.: A Hybrid Evolutionary Approach to the University Course Timetabling Problem. In: Proceedings of CEC 2007: The 2007 IEEE Congress on Evolutionary Computation, pp. 1764–1768 (2007)
4. Abdullah, S., Burke, E.K., McCollum, B.: Using a Randomised Iterative Improvement Algorithm with Composite Neighborhood Structures for University Course Timetabling. In: Metaheuristics - Progress in Complex Systems Optimization, pp. 153–172. Springer, Heidelberg (2007)
5. Asmuni, H., Burke, E.K., Garibaldi, J.: Fuzzy Multiple Heuristic Ordering for Course Timetabling. In: Proceedings of the 5th United Kingdom Workshop on Computational Intelligence (UKCI 2005), pp. 302–309 (2005)
6. Burke, E.K., Bykov, Y., Newall, J., Petrovic, S.: A Time-predefined Approach to Course Timetabling. Yugoslav Journal of Operations Research (YUJOR) 13(2), 139–151 (2003)
7. Burke, E.K., Kendall, G., Soubeiga, E.: A Tabu-search Hyperheuristic for Timetabling and Rostering. Journal of Heuristics 9, 451–470 (2003)
8. Burke, E.K., Eckersley, A., McCollum, B., Petrovic, S., Qu, R.: Hybrid Variable Neighbourhood Approaches to University Exam Timetabling. Technical Report NOTTCS-TR-2006-2, University of Nottingham, School of Computer Science (2006)
9. Burke, E.K., McCollum, B., Meisels, A., Petrovic, S., Qu, R.: A Graph Based Hyper-Heuristic for Educational Timetabling Problems. European Journal of Operational Research 176, 177–192 (2007)
10. Cooper, T., Kingston, H.: The Complexity of Timetable Construction Problems. In: Burke, E.K., Ross, P. (eds.) PATAT 1995. LNCS, vol. 1153, pp. 283–295. Springer, Heidelberg (1996)
11. Dueck, G.: New Optimization Heuristic: The Great Deluge Algorithm and the Record-to-record Travel. Journal of Computational Physics 104, 86–92 (1993)
12. Landa-Silva, D., Obit, J.H.: Great Deluge with Nonlinear Decay Rate for Solving Course Timetabling Problems. In: Proceedings of the 2008 IEEE Conference on Intelligent Systems (IS 2008), pp. 8.11–8.18. IEEE Press, Los Alamitos (2008)
13. Rossi-Doria, O., Sampels, M., Birattari, M., Chiarandini, M., Dorigo, M., Gambardella, L., Knowles, J., Manfrin, M., Mastrolilli, M., Paechter, B., Paquete, L., Stuetzle, T.: A Comparion of the Performance of Different Metaheuristics on the Timetabling Problem. In: Burke, E.K., De Causmaecker, P. (eds.) PATAT 2002. LNCS, vol. 2740, pp. 333–352. Springer, Heidelberg (2003)
14. Socha, K., Knowles, J., Sampels, M.: A Max-min Ant System for the University Course Timetabling Problem. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) Ant Algorithms 2002. LNCS, vol. 2463, pp. 1–13. Springer, Heidelberg (2002)