MISTA 2009

An Investigation of a Genetic Algorithm and Sequential Local Search Approach for Curriculum-based Course Timetabling Problems

Salwani Abdullah • Hamza Turabieh • Barry McCollum • Edmund K Burke

1 Introduction

Curriculum-based course timetabling deals with the weekly assignment of a set of lectures for university courses to specific timeslots and rooms, where conflicts between courses are set according to curricula published by the university and not on the basis of enrolment data. The curriculum-based course timetabling is considered as the third track in the 2nd International timetabling competition (ITC2007). The main reason for the wide interest in this formulation is because is capable of representing real world problems that often arise in higher educational institution. In this paper, we consider the same curriculum-based course timetabling problem as described in Gaspero et al. [1]. The solution of the problem is an assignment of a period (day and timeslot) and a room to all lectures of each course subject to a set of hard and soft constraints. There are four hard constraints considered as in Gaspero et al. [1]:

- Lectures: All lectures of a course must be scheduled, and assigned to distinct periods.
- Conflicts: All lectures of courses in the same curriculum or taught by the same teacher must be scheduled in different periods.
- Availability: If the teacher of the course is not available to teach that course at a given period, then no lecture of the course can be scheduled at that period.
- Room Occupation: Two lectures cannot be assigned in the same room in the same period.

The soft constraints considered are also taken from Gaspero et al. [1]:

- Room Capacity: The number of students that attend the course for each lecture, must be less than or equal to the number of seats of all the rooms that host its lectures.

Salwani Abdullah
Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia.
E-mail: salwani@ftsm.ukm.my

Hamza Turabieh
Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia.
E-mail: hamza@ftsm.ukm.my

Barry McCollum
Department of Computer Science, Queen's University Belfast, Belfast BT7 1NN United Kingdom,
E-mail: b.mccollum@qub.ac.uk

Edmund K Burke
Automated Scheduling, Optimization and Planning Research Group, School of Computer Science & Information Technology, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, United Kingdom,
E-mail: ekb@cs.nott.ac.uk

- Min Working Days: The lectures of each course must be spread into the given minimum number of days.
- Isolated Lectures: Lectures belonging to a curriculum should be adjacent to each other (i.e., in consecutive periods).

2 The Algorithm

The algorithm is approach discussed here is divided into two stages: initial construction and improvement. From the literature it can be observed that most of the initial construction solutions in relation to university timetabling problems are generated using graph coloring heuristics. This is directly related to how the problem in general is represented [7, 9]. Improvement is subsequently often achieved through the use of a metaheuristic approach. The following described the approach taken here.

2.1 Construction Algorithm

A construction algorithm proposed by Chiarandini *et al*. [2] and Landa-Silva and Obit [3] is used to generate large populations of random initial solutions. The constructive algorithm consists of three phases i.e. largest degree heuristic, neighbourhood search and tabu search as presented in pseudo code form in Fig. 1. We called this construction algorithm as a hybrid constructive algorithm. This approach was chosen in particular because it was able to produce feasible solutions for all datasets due to the combination of the strength from three phases involved (see [3]).

```
set population size, Popsize
for  i =0 to i < Popsize
     for j=0  to j < Number of lectures
          Phase 1:Apply largest degree heuristic
     end for
     do  while (timetable infeasible)
          Phase 2:Apply neighborhood  search
          Phase 3:Apply tabu search
     end do
end for
return population of feasible timetables
```

Fig. 1 Pseudo code for construction algorithm

2.2 Improvement Algorithm

The improvement algorithm involves the implementation of genetic algorithm (GA) and sequential local search. GA was introduced by John Holland in 1975 [4], which is a family of computational models that employs processes found in the natural biological evolution. GA is an adaptive methods used to solve search and optimization problems. GAs encode a potential solution of a specific problem in a simple chromosome like data structure and apply recombination operators to these structures so as to preserve critical information. It is used to search large, nonlinear solution space where expert knowledge is lacking or difficult to encode. Moreover it requires no gradient information, evolves from one population to another and produces multiple optima rather than single local one. These characteristics make GA a well-suited tool for optimization problems as for example in [5] and [7].

Standard GA focuses on the global side of the searching area, while local search method mainly relies on local area. The more intensive the local search, the stronger the need of specialized information about the function to be minimized or maximized (depends on the problem to be solved). As a matter of fact, hybridization between GA and other local search methods has already been suggested and tested in an important number of works and they show the effectiveness of this combination (see [10], [11], [12], [13]). Due to this strength, our

improvement algorithm combines GA and sequential local search. Fig. 2 shows a schematic overview of the approach. The algorithm used in this paper in differ from [8] where there was no crossover operation employed and the algorithm was tested on 11 datasets as in Socha et al. [6]. At the beginning of the search, a set of populations is formed from the generated initial solutions. In each generation, multiple individuals are stochastically selected from the current population based on their fitness, recombined and mutated to form a new population, which becomes current in the next iteration of the algorithm. The repair function is applied to bring infeasible solutions to feasible once that works in three steps as below:

Step 1 : Find free timeslots for each room
Step 2 : Find free timeslots for each event
Step 3 : Find feasible timeslots for rooms and events (i.e. an intersection between Steps 1 and 2).

Sequential Local Search: Before moving to the next generation, we applied a neighbourhood search algorithm to improve the timetable. This process makes the convergence of the genetic algorithm faster. We then reinsert the solution of the local search back to the genetic algorithm to be considered in the next generation.

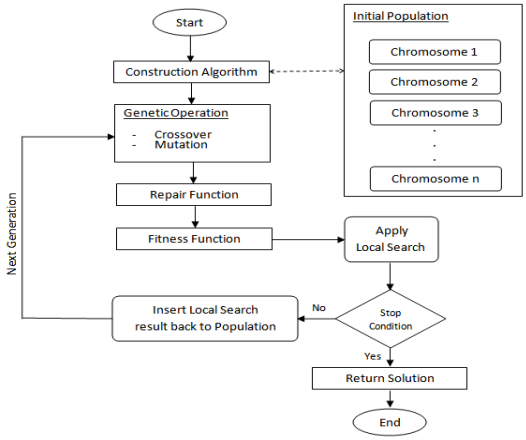

Fig. 2 A schematic overview of the approach

3 Experiments and Results

The experiments for the curriculum-based course timetabling problem discussed in this paper were tested on the twenty one real-world instances provided by the University of Udine. The details of all instances can be found in http://tabu.diegm.uniud.it/ctt/index.php.

Table 1 shows the parameters used in the genetic algorithm after preliminary experiments and, are comparable with the papers in the literature [7]. The proposed algorithm was programmed using Matlab and the simulations were performed on the Intel Pentium 4 2.33 GHz computer.

Table 1 Parameter Setting

Parameter	Generation Number	Population size	Crossover Rate	Mutation Rate	Selection Mechanism	Crossover Type
Value	10000	100	0.8	0.04	Tournament selection	Single point

Table 2 shows the results obtained and the comparison with best known solutions; the best results are presented in bold. The number of generations on each dataset is set to 10000 (with different random seeds). The best results out of 10 runs obtained are presented. Note that

our method produces the best known result in the literature on three of the twenty one problems[1]. The processing time (considered as a stopping condition) for each run is set to 600 seconds based on the time allocate in the ITC2007 [1]. It can be seen that our approach is able to produce timetable with lowest penalty on comp01 and comp02 (tieing with best known results) and also able to find global solution on comp11. At most of the cases, our results are comparable to the performance of tabu search and mathematical programming approaches.
We believe that by introducing the intensification and diversifications strategies will help the algorithm to balance the converging process and then able to further improve the quality of the solutions.

Table 2 Results

Instance	Initial solution	GA and Sequential Local Search			Best Known Solution[1]	
		Best	Average	Time (s)	Result / Penalty	Method Used
comp01	1869	**4**	6.8	513.12	**4**	Tabu. Search
comp02	6776	**20**	27.6	497.21	**20**	Tabu Search
comp03	6041	41	48.3	347.5	38	Tabu Search
comp04	4429	20	21.3	592.19	18	Tabu Search
comp05	7513	235	237.8	438.3	219	Tabu Search
comp06	4310	24	24	561.9	16	Mathematical Programming
comp07	3119	12	13.2	342.97	3	Mathematical Programming
comp08	3007	22	22.8	418.4	20	Mathematical Programming
comp09	4537	71	75.2	382.66	54	Tabu Search
comp10	2479	13	14.8	212.74	2	Mathematical Programming
comp11	1212	**0**	5.8	320.18	**0**	Tabu Search
comp12	3155	261	265.1	125.09	239	Tabu Search
comp13	4828	67	69.3	476.59	32	Tabu Search
comp14	3254	36	36	572.0	27	Tabu Search
comp15	5717	29	34.2	471.64	28	Tabu Search
comp16	4888	30	30	521.41	16	Tabu Search
comp17	3808	35	37.4	418.23	34	Tabu Search
comp18	1495	39	45.2	572.97	34	Tabu Search
comp19	4609	41	47.1	205.18	32	Tabu Search
comp20	5852	19	19	536.2	11	Tabu Search
comp21	4459	88	109.2	312.46	52	Tabu Search

Interested readers can find more details on instances and solutions that were contributed from the research community at http://tabu.diegm.uniud.it/ctt/index.php

4 Conclusion

This paper has focused on investigating a genetic algorithm and sequential local search for the curriculum-based course timetabling problem. Preliminary comparisons indicate that this algorithm is able to produce one global solution and two best known results. This is due to the hybridization of GA and sequential local search that able to embed an exploration and exploitation through the implementation of genetic operators within GA and sequential local search, respectively. Future research will be aimed at enhancing the performance of the local search and applying more genetic operators and tested on other real-world instances.

References

1. L Di Gaspero, A Schaerf and B McCollum, The Second International Timetabling Competition (ICT-2007) Curriculum-based Course Timetabling (Track 3), http://pst.istc.cnr.it/ssc-at-icaps-07/papers/digaspero-et-al-SSC07.pdf (2007).

[1] http://tabu.diegm.uniud.it/ctt/index.php

2. M Chiarandini, M Birattari, K Socha and O Rossi-Doria, An effective hybrid algorithm for university course timetabling. *Journal of Scheduling* 9(5), 403-432 (2006).
3. D Landa-Silva and JH Obit, Great Deluge with Nonlinear Decay Rate for Solving Course Timetabling Problems. Proceedings of the 2008 IEEE Conference on Intelligent Systems (IS 2008), IEEE Press, 8.11-8.18 (2008).
4. J Holland, Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press (1975).
5. EK Burke, D Elliman and RF Weare. A hybrid genetic algorithm for highly constrained timetable ing problems. In Proceeding of ICGA, 605–610 (1995).
6. K Socha, J Knowles and M Samples, 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 Volume 2463, 1-13 (2002).
7. A Schaerf (1999), A survey of automated timetabling. *Artificial Intelligence Review* 13(2), 87-127 (2004).
8. S Abdullah and H Turabieh, Generating university course timetable using genetic algorithms and local search. The Third 2008 International Conference on Convergence and Hybrid Information Technology ICCIT, vol. I, 254-260 (2008).
9. E. Burke, J. Kingston and D. de Werra, Applications to Timetabling, Handbook of Graph Theory, (J. Gross and J. Yellen eds.), pp. 445-474, (Chapman Hall/CRC Press, 2003).
10. T. Goel and K. Deb. Hybrid methods for multi-objective evolutionary algorithms. In Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02), 188–192, (2002).
11. Uckun, S., Bagchi, S., Kawamura, K. and Y. Miyabe, 1993. Managing Genetic Search in Job Shop Scheduling". In IEEE Expert Intelligent Systems and Their Application, 15 – 24, (1993)
12. Wilke, P., Grobner, M., Oster, N.: A Hybrid Genetic Algorithm for School Timetabling. In: McKay, B., Slaney, J. (eds.) AI 2002: Advances in Artificial Intelligence, Springer-Verlag, New York, 455–464. (2002)
13. M. Rahoual and R. Saad, "Solving Timetabling Problems by Hybridizing Genetic Algorithms and Tabu Search", Proc. 6th Int. Conf. on the Practice and Theory of Automated Timetabling, Brno, Czech Republic, 467-472.,(2006)