Journal of Computer Sciences

Original Research Paper

Integrating a Repairing-Based Genetic Algorithm-Neighborhood Search Structure in Solving the Course Timetabling Problem

[1]Chong-Keat Teoh, [2]Habibollah Haron, [3]Antoni Wibowo and [2]Mohd. Salihin Ngadiman

[1]Department of Computer Science and Mathematics,
Faculty of Applied Sciences and Computing (FASC), Tunku Abdul Rahman University College, Malaysia
[2]Department of Computer Science, Faculty of Computing (FC), UniversitiTeknologi Malaysia, Malaysia
[3]BINUS Graduate Program-Master of Information Technology, Bina Nusantara University, Indonesia

Article history
Received: 05-07-2015
Revised: 30-09-2015
Accepted:14-12-2016

Corresponding Author:
Chong-Keat Teoh
Department of Computer Science and Mathematics, Faculty of Applied Sciences and Computing (FASC), Tunku Abdul Rahman University College, Malaysia
Email: teohck@acd.tarc.edu.my

Habibollah Haron
Department of Computer Science, Faculty of Computing (FC), UniversitiTeknologi Malaysia, Malaysia
Emai: habib@utm.my

Abstract: The course timetabling problem is not a trivial task as it is an *NP-hard* and *NP-complete* problem and many solutions have been proposed due to its high complexity search landscape. In essence, the nature of the course timetabling problem is to assign a lecturer-course entity to existing teaching venue and timeslot in an academic institution. In this article, the authors propose a Genetic Algorithm-Neighborhood Search (GANS) to construct a feasible timetable for courses offered by a department in the faculty of a local university in Malaysia. The framework of the solution is as follow: The feasible timetable is first constructed by Genetic Algorithm, which includes are pair operator which attempts to repair infeasible timetables. Upon feasibility, the second phase exploits the initial feasible solution using three neighborhood structures to search for an improved solution and global optimum. The experimental results demonstrate the efficiency and effectiveness of the various neighborhood structures in exploiting the feasible solutions to yield the global optimum.

Keywords: Genetic Algorithm, Neighborhood Search, Repair Operator, University Course Timetabling Problem

Introduction

Problem Background

The notion of manually generating a workable timetable within the context of a higher-learning institution is indefinitely a daunting task and the complexity increases when there are various unique hard constraints that must be satisfied in a feasible solution. The course timetabling problem or commonly known as CTP has garnered favorable response especially in the domain of Operations Research (OR) and Artificial Intelligence (AI) communities. During the last decade, the CTP received great attention partly due to the organization of three competitions, entitled International Timetabling Competitions (ITC) namely the ITC-2002 (Paechter *et al*., 2002), ITC-2007 (Mccollum *et al*., 2010) and ITC-2011 (Post *et al*., 2016). The approaches that are commonly undertaken to solve the problems are metaheuristic algorithms which can be classified to population based algorithms such as Genetic Algorithm (Abdelhalim and El Khayat, 2016), Particle Swarm Optimization (Kennedy and Eberhart, 1995), Ant Colony Optimization (Socha *et al*., 2003) and local search algorithms such as Simulated Annealing (Bellio *et al*., 2016), Tabu Search (Lü and Hao, 2010), Great Deluge (Dueck, 1993) and Variable Neighborhood Search (Hansen and Mladenović, 1997) to name a few. The aforementioned algorithms possess their own sets of strength and weaknesses and in order to obtain a high quality solution, hybrid algorithms are proposed in order for the resultant algorithm to exhibit various strength derived from the initial algorithms such as hybrid cat swarm algorithms (Skoullis *et al*., 2016), hybrid particle swarm optimization (Shiau, 2011), hybrid ant colony systems (Ayob and Jaradat, 2009). This paper presents a hybrid Genetic Algorithm Neighborhood Search which integrates domain-specific exploitative

properties of the Neighborhood Search into Genetic Algorithm to solve the CTP adopted from a real world example from a faculty in Universiti Teknologi Malaysia. A formal description of the dataset is presented in section 2.1 and 2.2 respectively and section 3.0 depicts the architecture of the proposed hybrid genetic algorithm neighborhood search. Section 4.0 presents the results obtained from the experiment and section 5.0 summarizes the contribution of the paper.

Model Formulation

The model and dataset addressed in this article is adopted from a faculty department in a local university in Malaysia. This section elucidates the background and properties of the dataset, the various entities involved in the problem formulation, the constraints which govern the problem and a formal mathematical formulation of the problem model.

Properties of the Dataset

Within the context of Malaysia public university's academic system (at the point in which this article is written), the intake of fresh undergraduates takes place once a year i.e., at any time, students are always in their odd (1st, 3rd, 5th) semesters or (2nd, 4th, 6th) semesters. In this article, we attempt to schedule the courses for students who are in their first, second and third year (1st semester, 3rd semester and 5th semester respectively) for five departments in a faculty in a local university in Malaysia. To present the data in a more concise manner, the dataset is categorized into three problem instances. The first instance consists of courses and the number of lectures for year 1 students, the second instance consists of year 1 and year 2 students and the third dataset consists of year 1, year 2 and year 3 students. Refer to Fig. 1.

Subject (s)

The total number of subjects offered to all 1st-3rd year faculty students. It should be noted that the subject contains the information of the instructor-in-charge as well.

Work_Day (w)

In general, the administration of the university operates for five (5) working days, from Monday to Friday.

Session (sn)

Session refers to the duration for a lecture which is fifty (50) min. In general, the university allocates nine (9) usable sessions for use every day. However, only eight (8) sessions are allocated on Friday due to Muslim prayers. In summary, there is a total of forty-four usable sessions that can be used on a weekly basis.

Venue (v)

According to the dataset obtained, there is a total of seventeen (17) venues which can be used either for lecture or lab (programming) activities.

Based on the aforementioned entities, a complete feasible assignment can be described as $E = \{s, w, sn, v\}$ where E refers to the complete assignment of events which consists of a set of subject, day, session and venue.

Instance	Student year
1	Year 1 (1st semester)
2	Year 1, Year 2 (1st semester, 3rd semester)
3	Year 1, Year 2, Year 3 (1st semester, 3rd semester, 5th semester)

Fig. 1. Categories of the problem instances of the university course timetabling problem

Table 1. Stipulated constraints and its descriptions

Constraint	Description
C_{H1}	All subjects must be scheduled to distinct sessions. A violation occurs if a subject is not scheduled or two subjects are scheduled simultaneously.
C_{H2}	Subjects cannot be scheduled to a same venue simultaneously. A violation occurs if two subjects are scheduled simultaneously and additional violation constitutes additional violation score.
C_{H3}	Each session is exactly one period long. A violation occurs if a subject is scheduled over a session.
C_{H4}	An hour lunch break must be scheduled during 1.00-1.50 pm from Monday until Thursday and no event should be scheduled during this period. A violation occurs if a subject is scheduled during this period.
C_{H5}	A 2-hour break must be scheduled for lunch and religious purpose during 1.00-1.50 pm on Friday and no event should be scheduled during this period. A violation occurs if a subject is scheduled during this period.
C_{S1}	Subjects should not be scheduled at the last period of the day and should not take place in the evening. A violation occurs if a subject is scheduled during this period.
C_{S2}	Timetable for venue should be as compact as possible. A violation occurs if there is a gap in the venue schedule.
C_{S3}	Venues should be fully occupied whenever possible and its requirement should be taken into account

Table 1 describes the constraints of the course timetabling problem and comprises five (5) hard constraints which must be satisfied (to produce a feasible timetable) and three (3) additional soft constraints which increases the quality of the timetable without violating the fulfilled hard constraints previously.

Model Formulation

In order to evaluate the performance of the timetable solution, the violation of each constraints is calculated based on the mathematical formulation given in Equation 1. Since producing a feasible timetable solution is of the utmost importance, a numerical weight denoted by α of value ten (10) is added to increase its significance. A feasible timetable solution will not record any C_{H1}-C_{H5} violation score (0). On the other hand, increasing the quality of the timetable is handled by the satisfaction of C_{S1}-C_{S2} soft constraints multiplied with a numerical weight denoted by β with value of one (1):

$$\min f(x) = \left[\alpha * \sum_{i=1}^{s}\sum_{x=1}^{5} g\left(C_{H_x}\right)\right] + \left[\beta * \sum_{i=1}^{s}\sum_{y=1}^{3} g\left(C_{S_y}\right)\right] \quad (1)$$

- s.t.
- $\forall (s_i, v_i, w_i, sn_i) \Lambda (s_j, v_j, w_j, sn_j)$
- C_{H1} :$(w_i = w_j) \Lambda (sn_i = sn_j) \Lambda (v_i = v_j)$
- C_{H2} : $(w_i = w_j) \Lambda (sn_i = sn_j) \Lambda (v_i = v_j)$
- C_{H3}: $\varnothing$*
- C_{H4} :$[(w_i = 1) \vee (w_i = 2) \vee (w_i = 3) \vee (w_i = 4)] \Lambda (sn_i = 6)$
- C_{H5} :$(w_i = 5) \Lambda [(sn_i = 6) \vee (sn_i = 7)]$
- C_{S1}:$[(w_i = 1) \vee (w_i = 2) \vee (w_i = 3) \vee (w_i = 4) \vee (w_i = 5)] \Lambda (sn_i = 9)$
- C_{S2}: $[v_i, sn_{imod}\, sn_{total} = 1 \Lambda v_i, sn_{i+1} = 0] \vee [v, sn_{imod}\, sn_{total} = 0 \Lambda v_i, sn_{i-1} = 0] \vee [v, sn_{imod}\, sn_{total} = sn_i \Lambda v_i, sn_{i+1} = 0 \Lambda v_i, sn_{i-1} = 0]$
- C_{S3}:$\varnothing$ *
- *$\varnothing$ denotes that the constraints are represented by the encoding of the candidate solution*

Where:
α = 10
β = 1
C_{Hx} = The x^{th} hard constraint
C_{Sy} = The y^{th} soft constraint
s_i = The i^{th} subject
v_i = The i^{th} venue
w_i = The i^{th} working day
sn_i = The i^{th} session

The chromosome of the solution is designed such that constraint C_{H3} and C_{S3} are satisfied at all times, thus recording the value of $\varnothing$ in the formulation.

Hybrid Genetic Algorithm Neighborhood Search

Genetic Algorithm (GA) is a popular multi-directional population-based metaheuristic algorithm that simulates the principles of natural selection. The algorithm has received widespread popularity through the works of Holland (1975) and is also reported to be very successful at solving real-life complex engineering problems. In the domain of timetabling, a hybrid variant of GA has been applied successfully by Kohshori and Abadeh (2012) in generating a timetable. In this article, the experiment setup encodes the chromosome such that it satisfies constraint C_{H3} and C_{S3} naturally in order to reduce the load on the algorithm. The proposed *GANS* adopts a repair operator featured in the works of Pongcharoen *et al*. (2008) and the parameters of the proposed *GANS* are tabulated according to Table 2 and Fig. 2 illustrates the encoding of the chromosome used in *GANS*.

	t_1	t_2	t_3	..	..	..	..	..	t_{45}
r_1	c_1	-	-	-	-	-	-	-	-
r_2	-	c_3	-	c_{11}	-	c_{23}	-	-	c_{76}
r_3	-	-	-	-	-	-	-	c_{73}	-
r_4	-	c_5	-	-	c_{24}	-	c_{18}	-	-
r_5	-	-	-	-	-	-	-	-	-
r_6	-	-	-	-	-	-	-	-	-
r_7	-	-	c_7	-	c_{37}	-	-	c_8	-
r_8	-	-	-	-	-	-	-	-	-
r_9	c_8	-	-	-	-	-	-	-	-
r_{10}	-	-	-	-	-	c_2	-	-	-
r_{11}	-	-	-	c_{65}	-	-	-	-	-
r_{12}	-	-	-	-	-	-	-	c_6	-
r_{13}	-	c_{40}	-	-	-	-	-	-	-
r_{14}	-	-	-	-	c_{77}	-	c_3	-	-
r_{15}	-	-	-	-	-	-	-	-	c_{22}
r_{16}	c_{67}	-	-	-	-	-	-	-	-
r_{17}	c_{12}	-	-	c_{55}	-	-	c_4	-	-

Fig. 2. Encoding of the GANS chromosome

```
Input: Population Size, Problem Size, P_crossover, P mutation
Output: S_best

Population ← Initialize_Population(Population Size, Problem Size);
Evaluate_Population(Population);
S_best← Get_Best_Solution(Population);
while ¬StopCondition() do
      Parents ← Select_Parents(Population, Population Size);
      Children ← Ø;
      foreachParent1, Parent2∈Parents do
            Child1, Child2← Crossover(Parent1, Parent2, P_crossover);
            Children1← Mutate(Child1, P mutation);
            Children2← Mutate(Child2, P mutation);
      end
      Evaluate_Population(Children);
      S_best← Get_Best_Solution(Children);
      Population ← Replace(Population, Children);
end
return S_best;
while ¬StopCondition() do
      S_best← Neighborhood_Search(S_best, Neighborhood_Search)
end
```

Fig. 3. Pseudocode for the proposed GANS algorithm

Table 2. Parameters employed in the proposed GANS

Parameter	Value/description
Population Size, $Population_{Size}$	10
Selection Scheme	tournament selection
Crossover probability, $P_{crossover}$	0.5
Mutation probability, $P_{mutation}$	0.1
Neighborhood Structure, $NS_{structure}$	3

Table 3. Description of the various neighborhood structures

Neighborhood Structure, $NS_{structure}$	Description
NS_1-Last Timeslot Move	This move selects a course scheduled at the last timeslot of the day and attempts to move it to an earlier timeslot.
NS_2-Room Compactness Move	This move selects a course which has empty gaps in between and attempts to move it to an adjacent timeslot of an existing course
NS_3-Room Move	This move selects a room and attempts to reassign a new room to the course.

The concept of Variable Neighborhood Search which is also referred to as Neighborhood Search (NS) was first introduced by Hansen and Mladenović (1997) where the algorithm explores the vicinity (neighborhood) of a promising solution in hope of finding a better solution. Contrasting to population-based metaheuristic algorithm where a multiple solutions are manipulated at a time, the NS algorithm only exploits one solution at a time. The ability to explore promising region is determined by the definition of neighborhood structures and in general, well-defined neighborhood structures tend to lead to the discovery of high-quality solutions. It has been reported that the algorithm performs significantly well at solving constraint satisfaction problems (Hoos and Stützle, 2005). The key ingredient here is to ensure well-defined neighborhood structure (Gaspero and Schaerf, 2006) as poorly defined neighborhood structure will hinder the progress of the algorithm (Papadimitriou and Steiglitz, 1982). In the domain of CTP, a successful implementation of a well-defined neighborhood structure is described in the works of Muller (2009), who incidentally is also the Track 3 winner in the ITC-2007 timetabling competition. In his works, he introduced six domain-specific neighborhood structures that possess high inter-connectivity to exploit the incumbent solution. Figure 3 describes the pseudocode of the proposed *GANS*.

In the proposed *GANS*, upon feasibility of the candidate solution, the algorithm begins to exploit the incumbent best solution, *S_best* for a feasible solution x^* by means of neighborhood search such that $f(x^*) \leq f(x)$ for all x in the feasible search space region. 3 domain-specific neighborhood structures, $NS_{structure}$ are defined in aiding the algorithm to obtain a better solution, x^* which are defined as in Table 3. The proposed *GANS* is coded entirely using MATLAB® as the developing tool.

Results and Discussion

This section describes the results obtained from the proposed *GANS*. The performance measurements which are taken into account consists of the hard and soft fitness score, g_{hard} and g_{soft}, iteration time (t) recorded in seconds, the number of iteration, *iteration*, the mean, x□ and standard deviation, σ for each problem instance. The algorithm terminates when the global optimum is returned or when the algorithm has elapsed 300 seconds (whichever comes first). Table 4 tabulates the results for all three problem instances.

The experiment is repeated for five times and in all three problem instances, a global optimum (fitness score of zero) is returned within a reasonably short amount of time that is 11.34 sec or 71.60 sec on the average for problem instance 1; 24.77 sec or 246.10 sec on the average for problem instance 2; 108.76 sec or 267.09 sec on the average for problem instance 3. Additionally, the *GANS* exhibits consistent performance as the mean values for g_{soft} recorded for each instance are relatively low. It is worth to note, that the neighborhood search mechanism is only invoked for the soft constraint evaluation only when the solution is feasible (g_{hard} equals zero). This clearly demonstrates the efficacy of the proposed *GANS*. The progression of the g_{soft} fitness scores for each instance are also illustrated in Fig. 4-6 respectively.

Table 4. Results for three problem instances solved using the proposed *GANS*

	Hard constraint			Soft constraint		
Problem Instance	g_{hard}	t(s)	Iteration	g_{soft}	t(s)	Iteration
1	0.00	3.80	3.00	0.00	13.66	4837.00
	0.00	3.50	3.00	0.00	11.34	3825.00
	0.00	3.56	3.00	0.00	19.01	6992.00
	0.00	3.53	3.00	3.00	300.72	115445.00
	0.00	3.51	3.00	0.00	13.25	4656.00
x□	0.00	3.58	3.00	0.60	71.60	27151.00
σ	0.00	0.13	0.00	1.34	128.12	49371.70
2	0.00	7.50	3.00	3.00	301.47	72573.00
	0.00	7.46	3.00	3.00	301.45	71883.00
	0.00	7.53	3.00	2.00	301.41	72805.00
	0.00	7.54	3.00	1.00	301.38	76577.00
	0.00	7.50	3.00	0.00	24.77	6439.00
x□	0.00	7.51	3.00	1.80	246.10	60055.40
σ	0.00	0.03	0.00	1.30	123.73	30028.39
3	0.00	46.87	5.00	7.00	306.69	40838.00
	0.00	45.90	5.00	5.00	306.53	46199.00
	0.00	46.03	5.00	0.00	108.76	15867.00
	0.00	45.97	5.00	13.00	306.85	45888.00
	0.00	46.67	5.00	9.00	306.64	45630.00
x□	0.00	46.29	5.00	6.80	267.09	38884.40
σ	0.00	0.45	0.00	4.82	88.51	13054.45

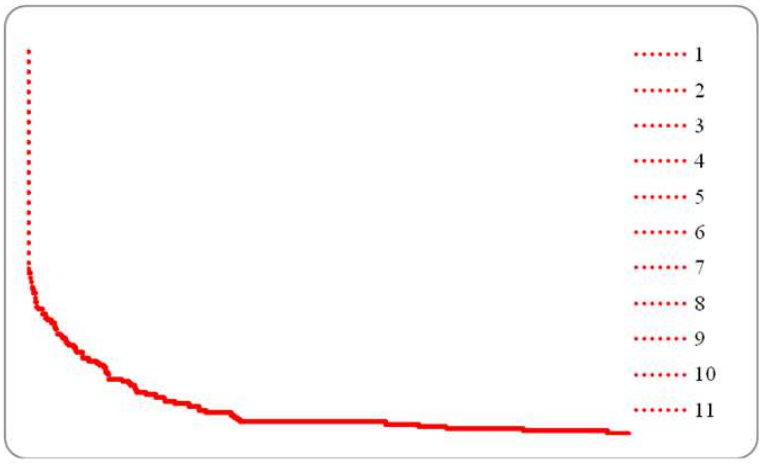

Fig. 4. Progression of soft constraint fitness score, g_{soft} for Instance 1

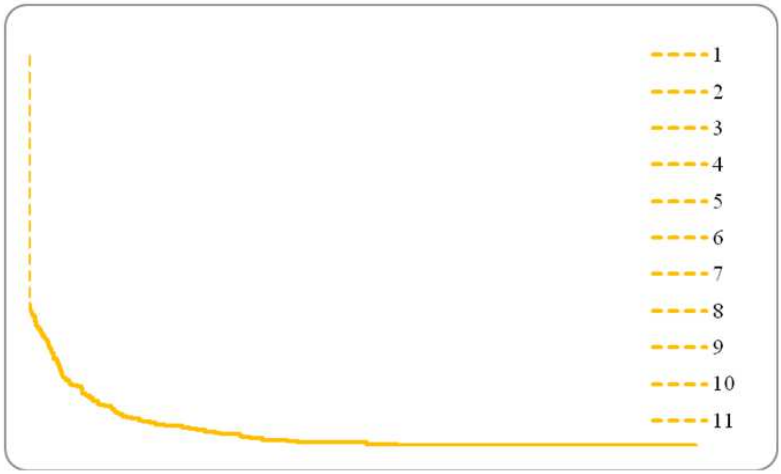

Fig. 5. Progression of soft constraint fitness score, g_{soft} for Instance 2

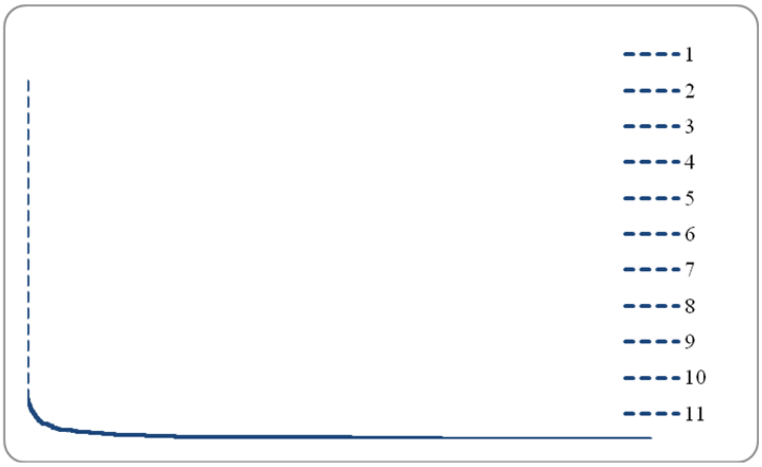

Fig. 6. Progression of soft constraint fitness score, g_{soft} for Instance 3

Conclusion

This article presents a hybridized Genetic Algorithm-Neighborhood Search (*GANS*) and integrates a repair operator to solve a real-world university course timetabling problem. The algorithm first obtains a feasible solution with the assistance of a repair operator and attempts to improve on the best solution using three various neighborhood structures. From the experiment, the proposed algorithm demonstrated promising results in terms of computational time and fitness score and additionally is able to obtain the global optimum for all the tested instances. To propose some future works, the authors intend to extend the algorithm to solve other timetabling problems such as the curriculum-based course timetabling problem and examination timetabling problem. Additional domain-specific neighborhood structures that may reduce the number of iterations and shorten the computational time will also be looked into as the exploitation ability exhibited by the Neighborhood Search algorithm is very promising.

Acknowledgement

The authors express their thanks to UniversitiTeknologi Malaysia and the Ministry of Higher Education (MOHE) Malaysia for supporting this research under themyBrain scholarship and Research University Grant (RUG), number Q.J130000.2628.09J33. The authors would also like to extend their gratitude to the Research Management Center (RMC)-UTM for supporting this research project.

Author's Contributions

Chong-KeatTeoh: Designed the algorithm performed data analysis and manuscript write-up.

Habibollah Haron: Who provided invaluable input in organizing the research materials.

Antoni Wibowo: Who provided indispensable technical and programming input.

Mohd. Salihin Ngadiman: Who gave constructive and critical feedback to the research works.

Ethics

The authors have deliberated thoroughly throughout the course of the publishing of this manuscript and conclude that all information and results reflected in this manuscript are true to its account and are not plagiarized from other sources. The dataset employed for the testing of the algorithm does not divulge any sensitive information but yet retains its degree of complexity.

References

Abdelhalim, E.A. and G.A. El-Khayat, 2016. A utilization-based genetic algorithm for solving the University Timetabling Problem (UGA). Alexandria Eng. J., 55: 1395-1409. DOI: 10.1016/j.aej.2016.02.017\

Ayob, M. and G. Jaradat, 2009. Hybrid ant colony systems for course timetabling problems. Proceedings of the 2nd Conference on Data Mining and Optimization, Oct. 27-28, IEEE Xplore Press, pp: 120-126. DOI: 10.1109/DMO.2009.5341898

Bellio, R., S. Ceschia, L.D. Gaspero, A. Schaerf and T. Urli, 2016. Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem. Comput. Operat. Res., 65: 83-92. DOI: 10.1016/j.cor.2015.07.002

Dueck, G., 1993. New optimization heuristics: The great deluge algorithm and the record-to-record travel. J. Computat. Phys., 104: 86-92. DOI: 10.1006/jcph.1993.1010

Gaspero, L.D. and A. Schaerf, 2006. Neighborhood Portfolio Approach for Local Search applied to Timetabling Problems, J. Math. Modell. Algorithms, 5: 1-18. DOI: 10.1007/s10852-005-9032-z

Hansen, P. and N. Mladenović, 1997. Variable Neighborhood Search. Comput. Operat. Res., 24: 1097-1100. DOI: 10.1016/S0305-0548(97)00031-2

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. 1st Edn., University of Michigan Press, Ann Arbor, ISBN-10: 0472084607, pp: 183.

Hoos, H.H. and T. Stützle, 2005. Stochastic Local Search: Foundations and Applications. 1st Edn., Morgan Kaufmann Publishers, Amsterdam, ISBN-10: 1558608729, pp: 658.

Kennedy, J. and R. Eberhart, 1995. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, Nov. 27-Dec. 1, IEEE Xplore Press, pp: 1942-1948. DOI: 10.1109/ICNN.1995.488968

Kohshori, M. and M. Abadeh, 2012. Hybrid genetic algorithms for university course timetabling. Int. J. Comput. Sci., 9: 446-465.

Lü, Z. and J.K. Hao, 2010. Adaptive Tabu Search for course timetabling. Eur. J. Operat. Res., 200: 235-244. DOI: 10.1016/j.ejor.2008.12.007

Mccollum, B., A. Schaerf, B. Paechter, P. Mcmullan and R. Lewis *et al*., 2010. Setting the research agenda in automated timetabling: The second international timetabling competition. INFORMS J. Comput., 22: 120-130. DOI: 10.1287/ijoc.1090.0320

Muller, T., 2009. ITC-2007 solver description: A hybrid approach. Annals Operat. Res., 172: 429-429. DOI: 10.1007/s10479-009-0644-y

Paechter, B., L.M. Gambardella and O. Rossi-Doria, 2002. The first international timetabling competition.

Papadimitriou, C.H. and K. Steiglitz, 1982. Combinatorial Optimization: Algorithms and Complexity. 1st Edn., Prentice Hall, Englewood Cliffs, ISBN-10: 0131524623, pp: 496.

Pongcharoen, P., W. Promtet, P. Yenradee and C. Hicks, 2008. Stochastic optimisation timetabling tool for university course scheduling. Int. J. Product. Econom., 112: 903-918. DOI: 10.1016/j.ijpe.2007.07.009

Post, G., L.D. Gaspero, J.H. Kingston, B. McCollum and A. Schaerf, 2016. The third international timetabling competition. Annals Operat. Res., 239: 69-75. DOI: 10.1007/s10479-013-1340-5

Shiau, D.F., 2011. A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences. Expert Syst. Applic., 38: 235-248. DOI: 10.1016/j.eswa.2010.06.051

Skoullis, V.I., I.X. Tassopoulos and G.N. Beligiannis, 2016. Solving the high school timetabling problem using a hybrid cat swarm optimization based algorithm. Applied Soft Comput. DOI: 10.1016/j.asoc.2016.10.038

Socha, K., M. Sampels and M. Manfrin, 2003. Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. Proceedings of the International Conference on Applications of Evolutionary Computing, Apr. 14-16, Springer, UK., pp: 334-345. DOI: 10.1007/3-540-36605-9_31