International Journal of the Physical Sciences Vol. 6(17), pp. 4264-4272, 2 September, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.200
ISSN 1992-1950 ©2011 Academic Journals

Full Length Research Paper

Artificial bee colony search algorithm for examination timetabling problems

Malek Alzaqebah* and Salwani Abdullah

Data Mining and Optimisation Research Group (DMO), Centre of Artificial intelligence Technology, Universiti Kebangsaan Malaysia, Selangor, 43600, UKM, Selangor, Malaysia.

Accepted 24 June, 2011

Artificial bee colony is a population-based search algorithm that mimics a natural behavior of real honey bees to find rich food sources to obtain maximum quantity of nectar and share the information of food sources with other bees in the hive. This paper concerns primarily about how to use artificial bee colony to solve examination timetabling problems which it is known as a NP-hard problem. It deals with assigning exams to a limited number of timeslots while satisfying a set of constraints. This algorithm works based on three categories of bees, that is, employed, onlooker and scout bees that communicate with each other in sharing the information of the food sources. Our computational experiments indicate that the proposed approach produces promising results when tested on two set of instances that have been widely used in literature.

Key words: Honeybees algorithms, artificial bee colony, examination timetabling, Metaheuristics.

INTRODUCTION

Scheduling problems are considered NP-hard problems (Lewis, 2008), such as education timetabling, transportation and sport timetabling problems. This paper is concerned with university examination timetabling problems.

Building automated examination timetabling systems attract the attention of a lot of universities in order to utilize the available resources and save times while constructing timetables. Due to the complexity of examination timetabling problems, solving these problems using intelligent ways lead to find good enough solutions that are difficult to be generated manually. As a result, many conferences such as PATAT, MISTA and ICAPS are devoted in order to motivate researchers to find better approaches to solve these kinds of problems. A general timetabling problem consists of assigning a number of events (exams, courses, meetings, etc.) into a limited number of timeslots (periods of time) and location, whilst satisfying a given set of constraints. Hard constraints must be respected in order to produce a feasible timetable, whilst the violation of soft constraints is needed to be minimized as much as possible (Burke et al., 1996). Interested readers can find more details about some approaches which have been employed for solving examination timetabling problem (Burke et al., 1996; Carter et al., 1996; Carter, 1986; Lewis, 2008; Qu et al., 2009).

Various approaches based on honey-bees swarm have been used to solve optimisation problems (Kang et al., 2009; Karaboga and Basturk, 2007). Honey-bees algorithms are classified into three different groups (Baykasoglu et al., 2007) that is, foraging behavior, marriage behavior and queen behavior. As a marriage behavior, the honey-bees mating optimisation algorithm has been applied to solve examination timetabling problems (Sabar et al., 2009).

Artificial bee colony algorithm (ABC) is a population based algorithm that employed the natural metaphors based on the foraging behavior of honey-bees swarm. To our knowledge, the ABC algorithm has not been tested on the examination timetabling problem. This motivates us to investigate the performance of the foraging behavior of the ABC algorithm when tested on two sets of examination timetabling instances.

*Corresponding author. E-mail: malek_zaqeba@ftsm.ukm.my or salwani@ftsm.ukm.my

The rest of the paper is organised as follows. Subsequently, the study formally presents the examination timetabling problem and formulation, after which it outlines the original Artificial Bee Colony algorithm, proposed approach, experimental results and discussion. Finally, some brief concluding comments are given.

PROBLEM DESCRIPTION AND FORMULATION

In this paper, the problem description is separated into two parts as follows:

(1) Problem 1: This problem is proposed by Carter et al. (1996), which consists of 13 benchmark datasets that are taken from a variety of educational institutions, which is known as an uncapacitated examination timetabling problem where a room capacity is not considered.
(2) Problem 2: International timetabling competition (ITC, 2007) dataset which consists of three tracks. In this paper, we consider the first track that represents an exam timetabling model which includes a number of real world constraints.

Problem 1

The problem description that is utilised in this paper is adapted from the description presented in Burke et al. (2004). Examination timetabling problems consist of these inputs as stated subsequently:

(1) N is the number of exams
(2) E_i is an exam, $i \in \{1,\ldots,N\}$
(3) T is the given number of available timeslots
(4) M is the number of students
(5) $C = (c_{ij})_{NxN}$ is the conflict matrix where each element denoted by c_{ij}, $i,j \in \{1,\ldots,N\}$ is the number of students taking exams i and j.
(6) t_k ($1 \leq t_k \leq T$) specifies the assigned timeslot for exam k ($k \in \{1,\ldots,N\}$)

We formulate an objective function which tries to space out students' exams throughout the exam period (that is, considered as a soft constraint) (Expression (1)) that can then be formulated as the minimisation of:

$$\frac{\sum_{i=1}^{N-1} F_1(i)}{M} \tag{1}$$

Where $$F_1(i) = \sum_{j=i+1}^{N} c_{ij} \cdot proximity \quad (t_i, t_j) \tag{2}$$

and $$proximity\ (t_i, t_j) = \begin{cases} 2^5 / 2^{|t_i - t_j|} & if \quad 1 \leq |t_i - t_j| \leq 5 \\ 0 & otherwise \end{cases} \tag{3}$$

subject to: $$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} \cdot \lambda(t_i, t_j) = 0$$

where $$\lambda(t_i, t_j) = \begin{cases} 1 & if \quad t_i = t_j \\ 0 & otherwise \end{cases} \tag{4}$$

Equation (2) presents the cost for an exam i which is given by the proximity value multiplied by the number of students in conflict. Equation (3) represents a proximity value between two exams (Carter et al., 1996). Equation (4) represents a clash-free requirement so that no student is asked to sit two exams at the same time. The clash-free requirement is considered to be a hard constraint.

Problem 2

The benchmark instances have been taken from the first track of the second International timetabling competition (ITC, 2007) (McCollum et al., 2010). Eight cases have been introduced. A set of hard and soft constraints are drawn from real world problems and are listed in Tables 1 and 2.

A feasible timetable is one in which all examinations have been assigned to a period and room, and there is no violation of the hard constraints. The objective function is to minimise the violation of the soft constraints as given in Expression (5) (McCollum et al., 2010):

$$\min \sum_{s \in S} (w^{2R} C_s^{2R} + w^{2D} C_s^{2D} + w^{PS} C_s^{PS}) + w^{NMD} C_s^{2NMD} + w^{FL} C^{FL} + w^{p} C^{P} + w^{R} C^{R} \tag{5}$$

Each dataset has its own weight as shown in Table 3 (McCollum et al., 2010).

ARTIFICIAL BEE COLONY ALGORITHM (ABC)

Artificial bee colony algorithm (ABC) was introduced by Karaboga (2005) as a global optimization algorithm that simulates the foraging behavior of honey bees. The algorithm classifies bees into three groups as employed bees, onlooker bees and scouts bees. In this algorithm, employed bees fly around the search space to choose a food source and come back to a hive to share the food source information with onlooker bees. Based on this information, onlooker bees probabilistically choose their food sources. While, the employed bees whose food source has been abandoned become scout bees, and start to search a new food source randomly without any information. If the nectar amount of a new source is higher than the previous one in their memory, they memorize the new position and forget the previous one. ABC system combines local and global search methods, where the local search methods are carried out by employed and onlooker bees. While the global search methods are managed by onlooker bees and scout bees. The combination of the local and global search method is with an aim to attempt the balance between exploration and exploitation process. Figure 1 shows the pseudo code for the artificial bee colony algorithm proposed by Karaboga (2005).

As shown in Figure 1, the position of the food source represents a possible solution and the nectar amount of the food source corresponds to the quality (fitness value) of the associated solution. The number of the employed bees is equal to the number of solutions in the population. At the first step, initial populations (food source positions) are generated based on a constructive heuristic algorithm. After the initialisation, the population is subjected to repeat the cycles of the search process of the employed, onlooker, and scout bees, respectively. An employed bee produces an adjustment on the food source position in her memory and discovers a new food source position. Provided that the nectar amount of the new one is higher than that of the previous source, the bee memorizes the new source position and forgets the old one. Otherwise she keeps the position of the one in her memory. After all

Table 1. Hard constraints.

Hard constraints	Explanation
H1	There cannot be any students sitting for more than one exam at the same time.
H2	The total number of students assigned to each room cannot exceed the room capacity.
H3	The length of exams assigned to each timeslot should not violate the timeslot length.
H4	Some sequences of exams have to be respected. e.g. Exam_A must be schedule after Exam_B.
H5	Room related hard constraints must be satisfied e.g. Exam_A must be scheduled in Room 80.

Table 2. Soft constraints.

Soft constraints	Mathematical symbol	Explanation
S1	C_S^{2R}	Two exams in a row: Minimise the number of consecutive exams in a row for a student.
S2	C_S^{2D}	Two exams in a day: Student should not be assigned to sit more than two exams in a day. Of course, this constraint only becomes important when there are more than two examination periods in the same day.
S3	C_S^{PS}	Periods spread: All students should have a fair distribution of exams over their timetable.
S4	C_S^{2NMD}	Mixed durations: The numbers of exams with different durations that are scheduled into the same room has to be minimised as much as possible.
S5	C^{FL}	Larger examinations appearing later in the timetable: Minimise the number of examinations of large class size that appear later in the examination timetable (to facilitate the assessment process).
S6	C^{P}	Period penalty: Some periods have an associated penalty, minimise the number of exams scheduled in penalised periods.
S7	C^{R}	Room penalty: Some rooms have an associated penalty, minimise the number of exams scheduled in penalised rooms.

Table 3. The associate weight of ITC2007 collection of examination datasets.

Data sets	W^{2D}	W^{2R}	W^{PS}	W^{NMD}	W^{FL}	W^{P}	W^{R}
Exam_1	5	7	5	10	100	30	5
Exam_2	5	15	1	25	250	30	5
Exam_3	10	15	4	20	200	20	10
Exam_4	5	9	2	10	50	10	5
Exam_5	15	40	5	0	250	30	10
Exam_6	5	20	20	25	25	30	15
Exam_7	5	25	10	15	250	30	10
Exam_8	0	150	15	25	250	30	5

the employed bees complete the search process, they share the information about the position of the food sources with the onlooker bees at the dance area. Each onlooker assesses the nectar information that is taken from all employed bees and then chooses a food source depending on the nectar amounts of sources. As in the case of the employed bee, she produces a modification on the food source position in her memory, and checks its nectar amount. The abandoned food sources are determined, and new food sources are

Initial food sources are produced for all employed bees
REPEAT
Each employed bee flies to a food source in her memory and determines a neighbour source, then evaluates its nectar amount and dances in the hive.
Each onlooker watches the dance of employed bees and chooses one of their sources depending on the dances, and then goes to that source. After choosing a neighbour around that, she evaluates its nectar amount.
Abandoned food sources are determined and replaced with the new food sources discovered by scouts.
The best food source found so far is registered.
UNTIL (requirements are met)

Figure 1. Original artificial bee colony search algorithm.

randomly produced to replace the abandoned ones by scout bees.

The algorithm: Artificial bee colony search algorithm

Construction heuristic

In this paper, we employ the graph colouring approach (that is, largest degree heuristic) to generate the initial solution, where examinations with the largest number of conflicts are scheduled first. For more details about graph colouring applications to timetabling can be seen in Burke et al. (1996).

Improvement algorithm

Figure 2 illustrates the pseudo-code that represents our approach. The algorithm starts with feasible initial solutions which are generated by a largest degree heuristic, in the constructive phase.

The employed bees work on random solutions and apply a random neighborhood structure on each solution. The solutions are arranged based on the penalty cost function, then the probability for each solution is determined. Onlooker bees work on the highest probability solution, and enhance it by applying a random neighborhood structure in order to reduce the violation of the soft constraints. Finally, scout bees determine the abandoned food sources and replace them with a new food source by performing several moves.

Initialisation:
Initialize the initial population and Evaluate fitness;
Calculate the initial cost function value, f(Sol);
Set best solution, Solbest ← Sol;
Set maximum number of iteration, NumOfIte;
Set the population size;
//where population size = OnlookerBee = EmployeedBee;
iteration ← 0;
Improvement:
do while (iteration < NumOfIte)
for i=1: EmployeedBee
Select a random solution and apply random neighborhood structure;
Sort the solutions in ascending order based on the Penalty cost;
Determine the probability for each solution, based on the following formula :

$$p_i = \frac{\sum \left(1/fit_i \right)^{-1}}{fit_i}$$

end for

for i=1: OnlookerBee
Sol* ← select the solution who has the higher probability;
Sol** ← Apply a random Nbs on Sol*;
if (Sol** < Solbest)
Solbest=Sol**;
end if
end for
Scoutbee determines the abandoned food source and replace it with the new food source.
iteration++

end do

Figure 2. The pseudo code for the artificial bee colony search algorithm.

Neighborhood structure

In this paper, ten neighborhood structures have been employed in order to enhance the performance of searching algorithms. These neighborhood structures are presented as follows (Abdullah et al., 2007):

Nbs 1: Select two exams at random and swap timeslots.
Nbs 2: Choose a single exam at random and move to a new random feasible timeslots.
Nbs 3: Select two timeslots at random and simply swap all the exams in one timeslot with all the exams in the other timeslot.
Nbs 4: Select 3 exams randomly and swap the timeslots between them feasibly.
Nbs 5: Select 4 exams randomly and swap the timeslots between them feasibly.

Table 4. Uncapacitated examination timetabling datasets.

Datasets	Number of timeslots	Number of examinations	Number of students	Conflict density
car92	32	543	18419	0.14
car91	35	682	16925	0.13
ear83 I	24	190	1125	0.27
hec92 I	18	81	2823	0.42
kfu93	20	461	5349	0.06
lse91	18	381	2726	0.06
pur93 I	42	2419	30032	0.03
rye92	23	486	11483	0.07
sta83 I	13	139	611	0.14
tre92	23	261	4360	0.18
uta92 I	35	622	21267	0.13
ute92	10	184	2750	0.08
yor83 I	21	181	941	0.29

Table 5. ITC2007 Examination datasets.

Dataset	D1	D2	D3	D4	D5	D6	D7	CD
Exam_1	7891	7833	607	54	7	12	0	5.05
Exam_2	12743	12484	870	40	49	12	2	1.17
Exam_3	16439	16365	934	36	48	170	15	2.62
Exam_4	5045	4421	273	21	1	40	0	15.0
Exam_5	9253	8719	1018	42	3	27	0	0.87
Exam_6	7909	7909	242	16	8	23	0	6.16
Exam_7	14676	13795	1096	80	15	28	0	1.93
Exam_8	7718	7718	598	80	8	20	1	4.55

Nbs 6: Take two timeslots at random, say t_i and t_j (where $j>i$) where timeslots are ordered $t_1,t_2,t_3,\ldots,t_p$. Take all exams that in t_i and allocate them to t_j, then allocate those that were in t_{j-1} to t_{j-2} and so on until we allocate those that were t_{j+1} to t_i and terminate the process.
Nbs 7: Move the highest penalty exams from a random 10% selection of the exams to a random feasible timeslots.
Nbs 8: Carry out the same process as in Nbs 7 but with 20% of the exams.
Nbs 9: Move the highest penalty exams from a random 10% selection of the exams to a new feasible timeslots which can generate the lowest penalty cost.
Nbs 10: Carry out the same process as in Nbs 9 but with 20% of the exams.

Note: For Problem 2, the neighborhood structures employed are Nbs 1, Nbs 2 and Nbs 3 with an aim to reduce the time taken in generating the candidate solutions (note that for the ITC2007 datasets, the algorithm is run for 600 s as imposed during the examination timetabling competition).

Benchmark dataset specification

Problem 1: Uncapacitated datasets

Table 4 shows the dataset's specification of the uncapacitated examination timetabling problems, which is available at (http://www.cs.nott.ac.uk/~rxq/data.htm).

Problem 2: Competition datasets (ITC2007)

For competition datasets (ITC2007), the total number of datasets are twelve, however, only eight datasets are available at: (http://www.cs.qub.ac.uk/itc2007/Login/SecretPage.php). The competition is described in detail (McCollum et al., 2010). Table 5, shows the ITC2007 examination datasets which have been used in this paper.

Note:

D1: Number of students.
D2: Number of actual students in the datasets.
D3: Number of exams.
D4: Number of timeslots
D5: Number of rooms.
D6: Period hard constraints.
D7: Room hard constraints.
CD: Conflict density.

SIMULATION RESULTS

ABC algorithm has been developed using Java sun micro

Table 6. Parameters setting.

Parameter	Value
Iteration	500
population size	50
Scout Bee	1

Table 7. Results comparison on uncapacitated problems.

Instance	Our approach	Best known	Authors for best known
car91	5.86	**4.50**	Yang and Petrovic (2004)
car92	4.92	**3.98**	Yang and Petrovic (2004)
ear83 I	38.34	**29.3**	Caramia et al. (2001)
hec92 I	11.51	**9.2**	Caramia et al. (2001)
kfu93	16.04	**13.0**	Burke et al. (2010)
lse91	12.42	**9.6**	Caramia et al. (2001)
pur93 I	7.6	**3.7**	Caramia et al. (2001)
rye92	10.73	**6.8**	Caramia et al. (2001)
sta83 I	158.01	**156.9**	Burke et al. (2010)
tre92	9.58	**7.9**	Burke et al. (2010)
uta92 I	3.99	**3.14**	Yang and Petrovic (2004)
ute92	27.80	**24.8**	Burke et al. (2010)
yor83 I	41.44	**34.9**	Burke et al. (2010)

system. The parameter settings used in this work are shown in Table 6.

Problem 1

Table 7 provides the comparison of our results with the best known results in the literature. We also include other population-based approaches here for the comparison purpose. The purpose here is to compare the performance among the population-based approaches in solving the same problem. The comparison between ABC result and the best known results shows that even if we are unable to beat any of the best known results in the literature, we are still able to produce promising solutions.

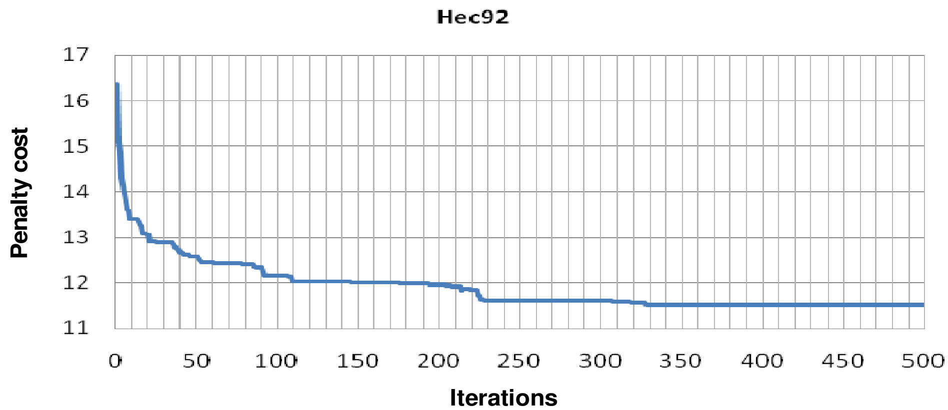
Figure 3 shows the behavior of the algorithm over three datasets that is, hec92I, sta83I and kfu93. The *x*-axis represents the number of iterations, while the *y*-axis represents the penalty cost. These graphs show how our algorithm explores the search space in which we believe that the way the algorithm behaves has a correlation with the complexity of the datasets (represented by the conflict density value). Note that the details of the conflict density values can be found in Qu et al. (2009). The higher conflict density signifies that more exams are conflicting with each other. The conflict density value for hec92I is 0.42, sta83I is 0.14 and kfu93 is 0.06. The behavior of the algorithm works similar at the beginning of the iterations where the improvement of the solution can easily be obtained. Later it becomes steady and hard to be improved. However, for the kfu93 dataset (where the conflict density value is lower compared to hec92I and sta83I datasets), the algorithm is able to slowly improve the quality of the solution until it get stuck in the local optimum when the number of iteration almost reaches the maximum number of iteration used in this experiment.

Problem 2

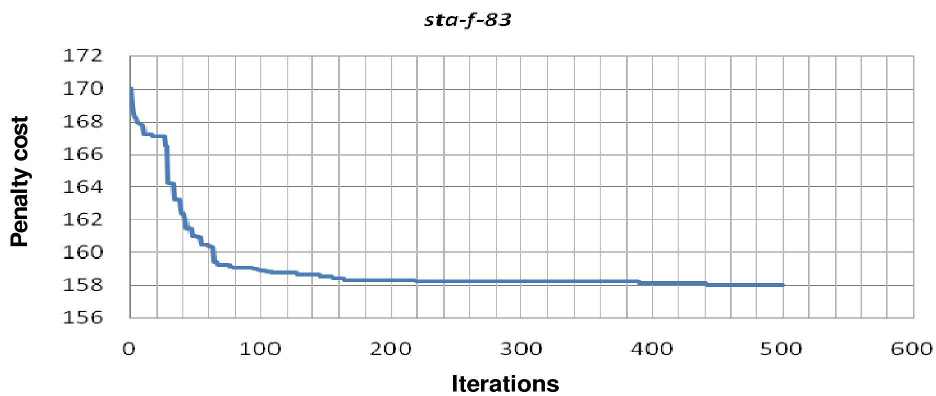
Table 8 shows the comparison of our results with some other available results in the literature. The best results are presented in bold.

We can see that the results obtained by the ABC algorithm are far behind the best known results. However, it is managed to produce feasible solutions for ITC2007 datasets.

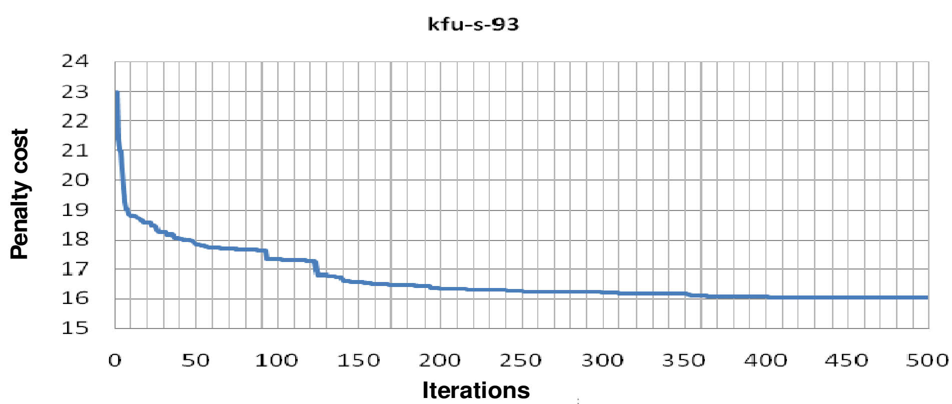
Figure 4 shows the behaviour of the algorithm when applied on Problem 2 where the conflict density for Exam_3 is equal to 2.62. Again, the *x*-axis represents the number of iterations, while the *y*-axis represents the penalty cost. The ABC algorithm is unable to find better results for ITC 2007 datasets compare with the best known



(a)



(b)



(c)

Figure 3. Convergence graph for (a) hec92I, (b) sta83I and (c) kfu93.

results in the literature. This is due to the algorithm that slowly improved the quality of the solution until the termination condition is met (which is set to 600 s). From Figure 4 we can see that the improvement of Exam_3 stopped when the computational time is equal to 600 s, where at that point the number of iteration is equal to 117. We believe that if we prolong the search until the number of iteration is equal to 500 (as in the experiment applied to

Table 8. Results comparison on ITC2007 datasets.

Datasets	Muller (2009)	Atsuta et al. (2007)	Pillay (2007)	Gogos et al. (2010)	Gogos et al. (2009)	Our approach
Exam_1	**4370**	8006	12035	4775	4699	6582
Exam_2	400	3470	3074	**385**	**385**	1517
Exam_3	10049	18622	15917	8996	**8500**	11912
Exam_4	18141	22559	23582	16204	**14879**	19657
Exam_5	2988	4714	6860	2929	**2795**	17659
Exam_6	26950	29155	32250	25740	**25410**	26905
Exam_7	4213	10473	17666	4087	**3884**	6840
Exam_8	7861	14317	16184	7777	**7440**	11464

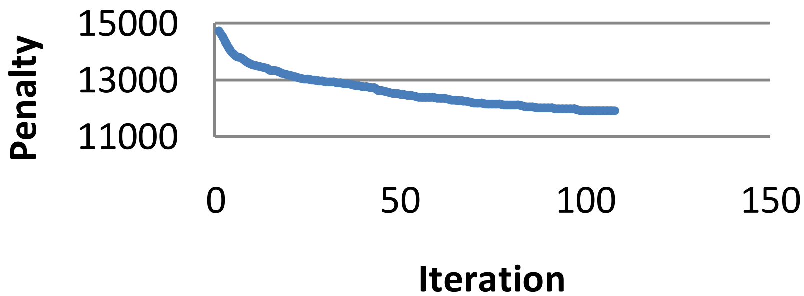


Figure 4. Convergence graph for Exam_3.

uncapacitated problem), the ABC algorithm still has a chance to further improve the quality of the solution.

DISCUSSION

The primary aim of this paper is to find good enough feasible solutions for examination timetabling problems by applying the artificial bee colony (ABC) algorithm. Through the results obtained, it is concluded that our preliminary results are comparable with the result obtained by previous empirical researcher. As a future work we would like to enhance the performance the ABC algorithm by applying different selection strategies in selecting onlooker bees from the population. Also a suitable mechanism to choose the neighborhood structure based on the solution state will be explored. This is subject to our future work.

REFERENCES

Abdullah S, Burke EK, McCollum B (2007). Using a Randomised Iterative Improvement Algorithm with Composite Neighbourhood Structures for the University Course Timetabling Problem. In Proceedings of MIC05: The 6th Metaheuristic International Conference, Vienna, Austria, In Metaheuristics - Progress in Complex Systems Optimization, Computer Science Interfaces Book Series, Springer Operations Research, ISBN-13:978-0-387-71919-1, 39: 153-169.

Atsuta M, Nonobe N, Ibaraki T (2007). ITC2007 Track 1: An Approach using general CSP solver. www.cs.qub.ac.uk/itc2007.

Baykasoglu A, Ozbakir L, Tapkan P (2007). Artificial bee colony algorithm and its application to generalized assignment problem. In Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, TSC. Felix and KT. Mano j, Eds. Itech Education and Publishing, Vienna, Austria, 113–143.

Burke EK, Bykov Y, Newall JP, Petrovic S (2004). A time-predefined local search approach to exam timetabling problem. IIE Trans., 36(6): 509-528.

Burke EK, Eckersley A, McCollum B, Petrovic S, Qu R (2010). Hybrid Variable Neighbourhood Approaches to Exam Timetabling. Eur. J. Oper. Res., 206: 46-53.

Burke EK, Elliman DG, Ford PH, Weare RF (1996). Examination Timetabling in British Universities – A Survey. In EK.Burke and P.Ross (eds.), The Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference, pp 76-90. Lect. Notes Comput. Sci., p. 1153. Springer.

Caramia M, Dell'Olmo P, Italiano GF (2001). New algorithms for examination timetabling. Algorithms Engineering 4th International Workshop, Proceedings WAE 2001, Saarbrücken, Germany, Springer Lect. Notes Comput. Sci., 1982: 230-241.

Carter MW (1986). A survey of practical applications of examination timetabling algorithms. J. Oper. Res., 34(2): 193-202.

Carter MW, Laporte G, Lee SY (1996). Examination Timetabling: Algorithmic Strategies and Applications. J. Oper. Res., 47: 373-383.

Gogos C, Alefragis P, Housos E (2010). An improved multi-staged algorithmic process for the solution of the examination timetabling problem, Annals of Operations Research. DOI: 10.1007/s10479-010-0712-3.

Gogos C, Goulas G, Alefragis P, Housos E (2009). Pursuit of Better Results for the Examination Timetabling Problem Using Grid Resources, CI-Sched '09. IEEE Symposium Computat. Intell. Scheduling, pp. 48-53.

Kang F, Li J, Xu Q (2009). Structural inverse analysis by hybrid simplex artificial bee colony algorithms, Comput. Struct., 87: 861-870.
Karaboga D (2005). An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.
Karaboga D, Basturk B (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Glob. Optim., 39: 459-471.
Lewis R (2008). A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum, 30(1): 167-190.
McCollum B, Schaerf A, Paechter B, McMullan P, Lewis R, Parkes AJ, Gaspero L, Qu R, Burke EK (2010). Setting the research agenda in automated timetabling: the second international timetabling competition. INFORMS. J. Comput., 22: 120-130.
Muller T (2009). ITC2007 Solver Description: A Hybrid Approach. Ann. Oper. Res., 172(1): 429-446.
Pillay A (2007). Developmental approach to the examination timetabling problem. www.cs.qub.ac.uk/itc2007.
Qu R, Burke EK, McCollum B, Merlot LTG (2009). A survey of search methodologies and automated system development for examination timetabling. J. Scheduling, 12: 55-89.
Sabar NR, Ayob M, Kendall G (2009). Solving examination timetabling problems using honey-bee mating optimization (ETP-HBMO). In: Proceedings of the 4th Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA 2009), 10–12 Aug 2009, Dublin, Ireland, pp. 399-408.
Yang Y, Petrovic S (2004). A novel similarity measure for heuristic selection in examination timetabling, Lecture Notes Comput. Sci., 3616: 334-353. 2005 Practice and Theory of Automated Timetabling V.