International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-1, Issue-5, November 2011

The Bees Algorithm for Examination Timetabling Problems

Malek Alzaqebah, Salwani Abdullah

Abstract— **Bees Algorithm (BA) is a population-based algorithm inspired by the honey bees forage for food. The algorithm presents a neighbourhood search associated with a random search which can be used for optimisation problems. In the basic version of BA, when a bee finds a food source, it returns to the hive and shares the information with other bees. Later, the bees will decide how many of them should fly towards the food source, depending on its quality (the quality represents the fitness value). In this paper, we have proposed to investigate the using of BA for examination timetabling problems, in addition a modification on the algorithm has been applied by replacing the fitness value by probability value. Experimental results indicate that the proposed approach produces promising results in solving examination timetabling problems and also show that the modified bees' algorithm outperforms the basic bees algorithm when tested on the same problems.**

***Index Terms*: Bees Algorithm, Examination Timetabling Problems.**

I. Introduction

Examination Timetabling Problem (ETTP) is an optimization problem faced by many academic institutions [13]. ETTP is the process of allocating a number of examinations into a predefined number of timeslots, by satisfying a set of constraints, where the hard constraints cannot be violated and soft constraints must be minimised as much as possible [23]. Numbers of approaches have been previously employed on examination timetabling problems [19].

The Bees Algorithm has been introduced in 2005 by Pham et al.[1]. Pham et al., in 2006 [2] proved that Bee Algorithm generally outperforms other techniques (i.e. Genetic Algorithms such as Ant Colony Optimization and Artificial bee colony) in terms of speed of optimization and accuracy of results [2]. In our research, we have concerned to improve the efficiency of the Bees Algorithm by replacing the fitness value to rank the solutions in the population by using a probability value in order to maintain population diversity.

The details of the basic bees' algorithm and the proposed approach are discussed in Section II, that is followed by some explanations about examination timetabling problems and its formulation (Section III). Our experimental results and comparison are presented in Section IV. The conclusion and future work are discussed in Section IIV.

Manuscript received July 09, 2011.

Malek Alzaqebah, Data Mining and Optimization Research Group (DMO) Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia,0060176169469, (e-mail: malek_zaqeba@ftsm.ukm.my).
Salwani Abdullah, Data Mining and Optimization Research Group (DMO) Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia, 0060193042640, (e-mail: salwani@ftsm.ukm.my).

II. BEES ALGORITHM

A. *The Basic Bees Algorithm*

This section summarises the main steps of the basic Bees Algorithm as stated in [2]. As mentioned before, the Bees Algorithm is one of optimisation algorithms which inspired by the natural foraging behaviour of honey bees. Figure I shows the simplest form of the pseudo code of the algorithm. The algorithm starts with initial population (scout bees) which is generated randomly, and then the BA search process is started until the stopping criterion is met.

```
Initialize population with random solutions.
Evaluate fitness of the population.
While (stopping criterion not met)
//Forming new population.
Select sites for neighbourhood search.
Recruit bees for selected sites (more bees
for best e sites) and evaluate fitness.
   Select the fittest bee from each patch.
   Assign remaining bees to search randomly
   and evaluate their fitness.
End While.
```

Figure I. Original Bees algorithm

In first step, the scout bees that have the highest fitness are chosen as "selected bees" and sites (solutions) visited by them are chosen for neighbourhood search. Then, the algorithm performs the search in the neighbourhood of the selected sites, assigning more bees to search near to the best sites. The bees can be chosen directly according to the fitness associated with the sites they are visiting. The remaining bees in the population are assigned randomly around the search space scouting for new solutions. At the end of each iteration, the colony will have two parts for its new population. The first part will contain the representatives from each selected sites (selected solutions), and the second part will contain other scout bees assigned to conduct random searches.

B. *The Proposed Bees Algorithm*

Figure II presents the pseudo-code of our approach. The algorithm starts with feasible initial solutions generated using a graph colouring heuristic to form an initial population. The size of the population is equal to the number of the scout bees (*ns*). Each scout bee evaluates the solution using the fitness function. The algorithm then starts the search process. The solutions in the population are ranked according to the probability value as in formula (1). Note that in the basic bees algorithm, the solutions are ranked based on the fitness value.

$$P_i = \frac{fi}{\sum_{i=1}^{SN} fi} \quad (1)$$

where SN = number of food sources, f_i = fitness function of the i^{th} food source.

```
Initialization:
Initialize the initial population and
Calculate the fitness values;
Set best solution, Solbest;
Set maximum number of iteration, NumOfIte;
Set the population size, PopSize;
Set ne: number of elite solutions;
Set nre: recruited bees for elite solutions;
Set nb: number of best solutions;
Set nrb: recruited bees for remaining best
solutions;
Set stlim: limit of stagnation cycles for the
abandonment solutions;
iteration ←  0;
Improvement:
do while (iteration < NumOfIte)
  for i=1: popsize
    Scout bees evaluate the solutions;
    Calculate the probability value, P_i;
    Rank the solutions based on P_i
     (For basic BA use the fitness value to rank
     the solutions)
  end for
nbSet ← Select the top nb solutions from the
population;
neSet ← Select the top ne solutions from the
nbSet;
  for j=1: nrb
     for h=1: nb
        Sol*  ← neSet_h;
        Apply random neighbourhood on Sol*;
        Update the population by the Improved
        solutions;
     end for
  end for
  for j=1: nre
    for h=1: ne
     Sol*  ← neSet_h;
        Apply random neighbourhood on Sol*;
        Update the population by the Improved
        solutions;
    end for
  end for
  recruit the remaining bees for random search
      Solbest ← best solution found so far;
   iteration++;
end do
```

Figure II. The pseudo code for the Bees algorithm

Later, the highest ranked solutions (*nb* solutions) will be selected for a local exploration by other bees (foragers) that are directed to the neighbourhood of the selected solutions by the scout bees. For each selected solution, the number of foragers will be allocated deterministically as follows: each scout bee that returns from one of the *nb* best solutions performs the ‘waggle dance’, meaning that it recruits *nrb* mates for local exploration. The scout bees that visit the first *ne* elite solutions among the best *nb* sites recruit *nre* foragers for a neighbourhood search. The scout bees that visit the remaining (*nb–ne*) solutions recruit *nrb* < *nre* foragers for a neighbourhood search. The neighbourhood search is thus able to give more tries for the elite solutions to be improved during the search process, in which the elite solutions are considered as the most promising solutions in the search space.

Figure III shows the block diagram for the proposed algorithm on examination timetabling problems. It illustrates the process of our proposed algorithm. The algorithm starts by generating the conflict matrix, which shows the students in conflict between the exams. Based on this conflict matrix we generate the initial solution using a constructive heuristic algorithm (largest degree in this case). Next, the improvement process (bees algorithm) is executed (as discussed earlier).

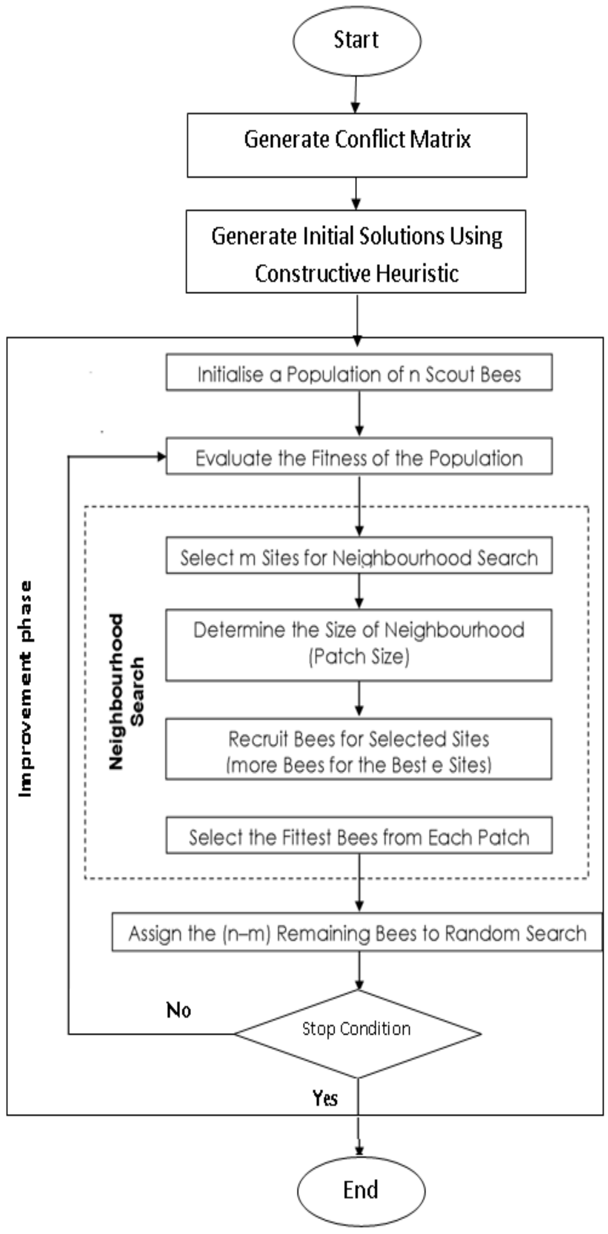

Start
Generate Conflict Matrix
Generate Initial Solutions Using Constructive Heuristic
Improvement phase
Initialise a Population of n Scout Bees
Evaluate the Fitness of the Population
Neighbourhood Search
Select m Sites for Neighbourhood Search
Determine the Size of Neighbourhood (Patch Size)
Recruit Bees for Selected Sites (more Bees for the Best e Sites)
Select the Fittest Bees from Each Patch
Assign the (n–m) Remaining Bees to Random Search
No
Stop Condition
Yes
End

Figure III. The Block Diagram for the Proposed Algorithm

III. PROBLEM DEFINITION

In this Paper, the problem description is separated into two problems:

- Problem I: (Toronto Benchmark) This problem is an incapacitated examination timetabling problem where a room capacity requirement is not considered while constructing a timetable. This problem has been introduced by Carter [13], along with a set of 13 real-world instances from a variety of educational institutions that have been accepted as a benchmark datasets for over a decade.

- Problem II: (International Timetabling Competition (ITC2007)) This represents an exam timetabling model that incorporates a significant number of real-world constraints. This formulation was introduced as part of the 2nd International Timetabling Competition (ITC2007).

A. Problem I (The Toronto Benchmark)

The description of this problem is adapted from Burke et al. [3]. Examination timetabling problems consist of the inputs as stated below:

- N is the number of exams.
- E_i is an exam, $i \in \{1\ldots N\}$.
- T is the given number of available timeslots.
- M is the number of students.
- $C = (c_{ij})_{NxN}$ 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.
- t_k $(1 \leq t_k \leq T)$ specifies the assigned timeslot for exam k $(k \in \{1,\ldots,N\})$.

We have formulated an objective function which tries to space out students' exams throughout the exam period (Expression (2)) that can then be formulated as the minimization of:

$$\frac{\sum_{i=1}^{N-1} F_1(i)}{M} \tag{2}$$

Where

$$F_1(i) = \sum_{j=i+1}^{N} c_{ij}.proximity(t_i,t_j)\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

And

$$proximity(t_i,t_j) = \begin{cases} 2^5/2^{|t_i-t_j|} & if \quad 1 \leq |t_i - t_j| \leq 5 \\ 0 & otherwise \end{cases} \tag{4}$$

Subject to:

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

Where

$$\lambda(t_i,t_j) = \begin{cases} 1 & if \quad t_i = t_j \\ 0 & otherwise \end{cases}\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

Equation (3) presents the cost for an exam which is given by the proximity value multiplied by the number of students in conflict. Equation (4) represents a proximity value between two exams [13]. Equation (5) 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.

B. Problem II (The International Timetabling Competition (ITC2007))

The benchmark instances considered for problem are taken from the third track of the second International Timetabling Competition (ITC 2007) [22]. (http://www.cs.qub.ac.uk/itc2007/index.htm). Eight cases have been introduced. A set of hard and soft constraints are drawn from real world problems and are listed as bellow.

Hard Constraints

- There cannot be any students sitting for more than one exam at the same time.
- The total number of students assigned to each room cannot exceed the room capacity.
- The length of exams assigned to each timeslot should not violate the timeslot length.
- Some sequences of exams have to be respected. e.g. Exam_X must be schedule after Exam_Y.
- Room related hard constraints must be satisfied e.g. Exam_X must be scheduled in Room 20.

Soft Constraints

- Two exams in a row: Minimize the number of consecutive exams in a row for a student. (coded as C_S^{2R})
- 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. (coded as C_S^{2D})
- Periods spread: all students should have a fair distribution of exams over their timetable. (coded as C_S^{PS})
- Mixed durations: The numbers of exams with different durations that are scheduled into the same room has to be minimized as much as possible (coded as C_S^{2NMD}).
- Larger examinations appearing later in the timetable: Minimize the number of examinations of large class size that appear later in the examination timetable (to facilitate the assessment process) (coded as C^{FL}).
- Period Penalty: some periods have an associated penalty; minimize the number of exams scheduled in penalized periods (coded as C^P).

- Room Penalty: some rooms have an associated penalty; minimize the number of exams scheduled in penalized rooms (coded as C^R).

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 minimize the violation of the soft constraints as given in expression (5) [22].

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

IV. SIMULATION RESULTS AND COMPARISON

We have compared the performance of our proposed modification with the basic BA in order to show the effects of employing the probability value on the basic BA algorithm. Table 4 shows the parameter settings which have been used in this work (according to Pham & Castellani [18], and some preliminary experiments).

TABLE I. PARAMETERS SETTING.

Parameter	Value
Iteration	500
population size	50
ne: number of elite sites	2
nre: recruited bees for elite sites	30
nb: number of best sites	4
nrb: recruited bees for remaining best sites	10
stlim: limit of stagnation cycles for site abandonment	10

The following neighbourhoods have been employed in this paper, in order to enhance the performance of searching algorithms.

Nbs1: Select 2 exams at random and swap timeslots.

Nbs2: Select a single exam at random and move to a new random feasible timeslots.

Nbs3: Select 4 exams randomly and swap the timeslots between them feasibly.

Nbs4: Select 2 exams at random and move to a new random feasible timeslots.

The following section illustrates the best results produced by BA and the Probable BA (coded as PBA). Both algorithms were run for 10 times.

The Toronto Benchmark Experimental Results

Table II shows the comparison of the BA and PBA with the best known results (shown in bold). The comparison between the basic BA and PBA shows, that the PBA perform better than the basic BA (shown in italic). Overall comparison with the best known results shows that even though we are unable to beat any of the best known results in the literature, but we are still able to produce good enough solutions, where the difference between PBA and the best known results in the literature falls in the range of 0.43% to 25%.

TABLE II. RESULTS COMPARISON ON INCAPACITATED PROBLEMS

Instance	Basic BA	PBA	Best known	Authors for best known
car91	5.79	*5.67*	**4.50**	[21]
car92	*4.76*	4.78	**3.98**	[21]
ear83 I	38.93	*37.67*	**29.3**	[11]
hec92 I	11.64	*11.33*	**9.2**	[11]
kfu93	15.70	*15.23*	**13.0**	[9]
lse92	*12.66*	12.81	**9.6**	[11]
sta83 I	158.05	*157.59*	**156.9**	[9]
tre92	9.05	*9.00*	**7.9**	[9]
uta92 I	3.92	*3.88*	**3.14**	[21]
ute92	28.05	*27.58*	**24.8**	[9]
yor83 I	40.01	*39.28*	**34.9**	[9]

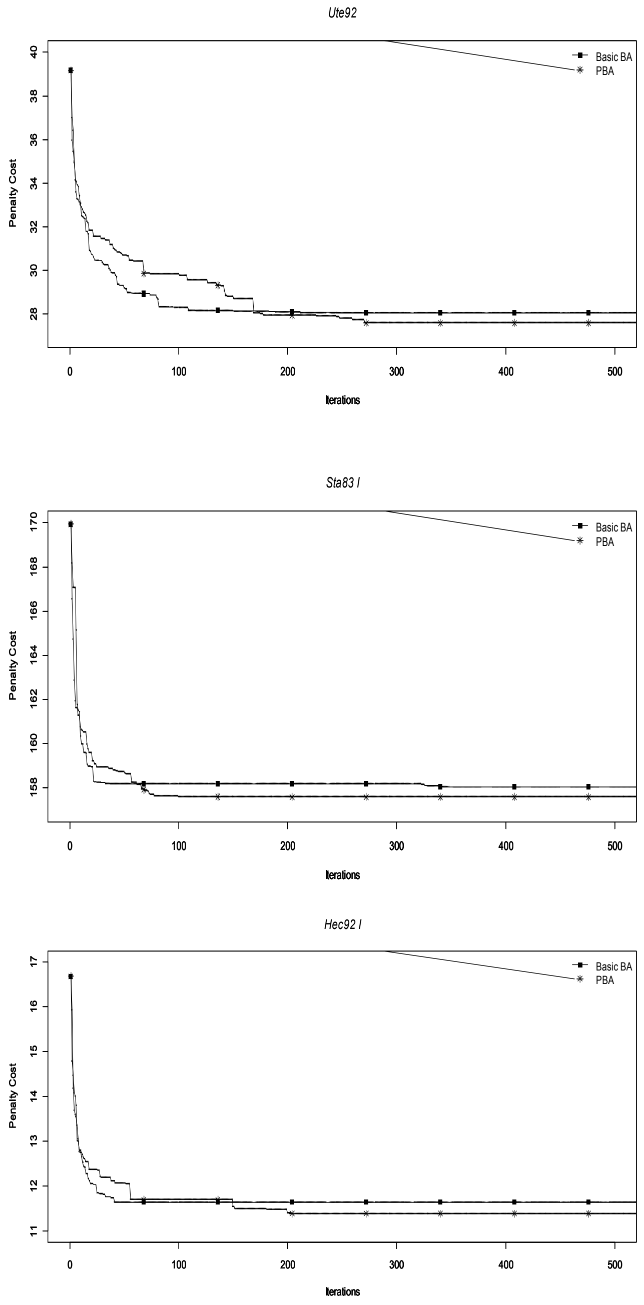


Figure IV. Convergence graphs for ute92, sta83 I and hec92 I datasets.

Figure IV shows the behaviour of BA and PBA during the search process. We have drawn the graph for ute92, sta83 I and hec92 I to show the effect of using the probability value rather than the actual penalty value when choosing the best and the elite solutions for BA search.

This graph shows the effects of the modification which are implemented on the basic BA algorithm, the convergence of the lines in these graphs show how BA and PBA explore the search space. However, as the number of iterations increases, the slope of the curves indicates a smaller decrease in the penalty cost. The behaviour of the BA and PBA algorithms works similar at the beginning of the iterations where the improvement of the solutions can easily be obtained. Later it becomes steady and hard to be improved. The comparison between basic BA and PBA shows that the PBA performs better than the basic BA that due to the using of the probability values to rank the solutions which give the most promising solutions a chance to be improved and more over this method can maintain the population diversity during the evolution process.

The International Timetabling Competition (ITC2007) dataset Experimental Results

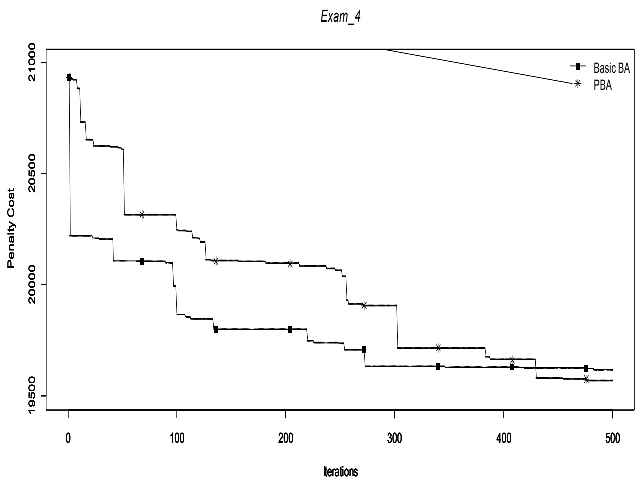
The basic BA and PBA algorithms are also tested on The International Timetabling Competition (ITC2007) datasets. The results are shown in Table III, which provides the comparison between the basic BA and PBA and with some available results in the literature. PBA shows better results than the basic BA, this can be indicated from the differences between PBA and the basic BA results, in a range of 0.28% to 3.98%. This again show that the choosing the elite solutions based on the probability value (rather than fitness value) helps to improve the quality of the solutions.

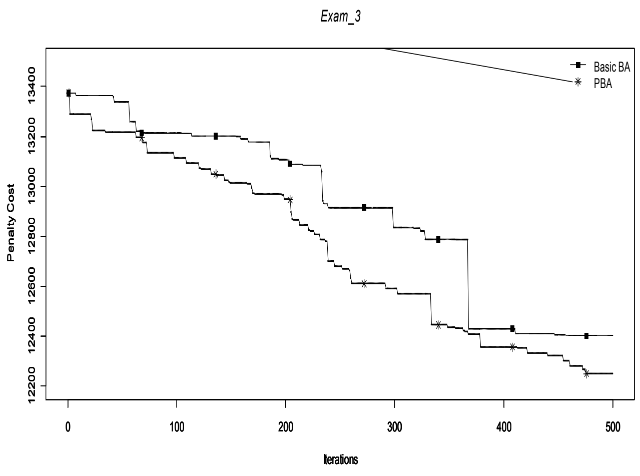
TABLE III. RESULTS COMPARISON ON ITC2007 DATASETS

Datasets	Muller [24]	Atsuta et al. [25]	Pillay [26]	Gogos et al. [27]	Basic BA	PBA
Exam_1	**4370**	8006	12035	4699	6145	*6049*
Exam_2	400	3470	3074	**385**	1417	*1370*
Exam_3	10049	18622	15917	**8500**	12404	*12251*
Exam_4	18141	22559	23582	**14879**	19625	*19569*
Exam_5	2988	4714	6860	**2795**	12783	*11108*
Exam_6	26950	29155	32250	**25410**	27090	*27000*
Exam_7	4213	10473	17666	**3884**	6771	*6501*
Exam_8	7861	14317	16184	**7440**	11655	*11240*

As shown in figure V the behaviour of the two algorithms show that the PBA improves the solution better than the basic BA. This is can easily observed from the convergence line for the three graphs. This is due to the employment of the probability values by the PBA to rank the solutions that give more chances to improve the highest ranked solutions (unlike using the fitness values, which eliminate some of the solutions that might promise some good improvements).

In this paper, we show the convergence for three datasets i.e. Exam_3, Exam_4 and Exam_5, which we choose them based on the conflict density value considered as medium, high and low, respectively as mentioned in McCollum et al. [22]. From the figure we can see that the PBA algorithm has an ability to further improve the quality of the solutions.





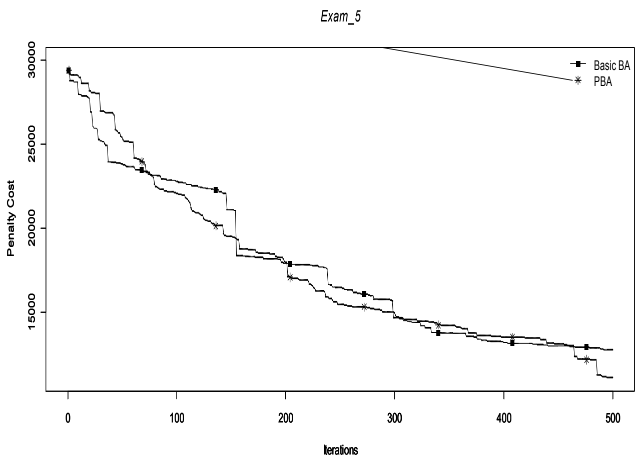


Figure V. Convergence graph for Exam_3, Exam_4 and Exam_5 dataset

V. CONCLUSION AND FUTURE WORK

The general idea of this paper was to propose the use of the basic BA on examination timetabling problems and a simple modification had been applied to the basic BA, called probability BA (PBA). The results revealed that the modification of the algorithm (PBA) effectively enhanced the solutions when tested on the examination timetabling problems. As a future work, a different selection strategy to select the solutions in the population and a combination with a local search will be applied on the basic BA.

REFERENCES

[1] D.T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim and M . Zaidi, (2005) The Bees Algorithm. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK.

[2] D.T. Pham, E. Kog, A. Ghanbarzadeh, S. Otri, S. Rahim, M. Zaidi (2006), The Bees Algorithm – A Novel Tool for Complex Optimisation Problems, IPROMS 2006 Proceeding 2nd International Virtual Conference on Intelligent Production Machines and Systems, Oxford, Elsevier.

[3] S. Abdullah, and E.K. Burke, (2006) A Multi-start large neighbourhood search approach with local search methods for examination timetabling. In The International Conference on Automated Planning and Scheduling (ICAPS 2006) (Eds. Long, D., Smith, S.F., Borrajo, D. & McCluskey, L.), 6-10 June, Cumbria, UK, , 334-337.

[4] S. Abdullah, S. Ahmadi, E.K. Burke and M. Dror (2007) Investigating Ahuja Orlin's large neighbourhood search approach for examination timetabling. OR Spectrum, 29(2), 351-372.

[5] S. Abdullah, E.K. Burke and B. McCollum, (2007) Using a Randomised Iterative Improvement Algorithm with Composite Neighbourhood Structures for University Course Timetabling. In Metaheuristics: Progress in complex systems optimization (Operations Research / Computer Science Interfaces Series), Chapter 8. published by Springer, ISBN:978-0-387-71919-1.

[6] S. Abdullah, H. Turabeih and B. McCollum, (2009) A hybridization of electromagnetic like mechanism and great deluge for examination timetabling problems, HM2009: 6th International Workshop on Hybrid Metaheuristics, Udine, 60-72.

[7] L. Bao, J. Zeng, (2009) Comparison and Analysis of the Selection Mechanism in the Artificial Bee Colony Algorithm. HIS (1): 411-416

[8] E.K. Burke, Y. Bykov, J.P. Newall and S. Petrovic, (2004) A time-predefined local search approach to exam timetabling problem. IIE Transactions, 36(6), 509 528.

[9] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic and R. Qu, (2010), Hybrid variable neighbourhood approaches to university exam timetabling, European Journal of Operation Research. 206(1), 46-53.

[10] S. Camazine, J.L. Deneubourg, N. Franks, J. Sneyd, G. Theraulaz and E. *Bonabeau. Self-Organization in Biological Systems*. Princeton University Press, Princeton, 2003 NJ.

[11] M. Caramia, P. Dell'Olmo and G.F. Italiano, (2001) New algorithms for examination timetabling. Algorithms Engineering 4th International Workshop, Proceedings WAE 2001, Saarbrücken, Germany, Springer Lecture Notes in Computer Science, vol. 1982. 230-241.

[12] M.W. Carter (1986) A survey of practical applications of examination timetabling algorithms. Operations Research, 34(2): 193-202.

[13] M.W. Carter, , G. G. Laporte, and S.Y. Lee, (1996) Examination Timetabling: Algorithmic Strategies and Applications. Journal of the Operational Research Society 47, 373-383.

[14] F. Kang, J. Li, and Q. Xu, "*Structural inverse analysis by hybrid simplex artificial bee colony algorithms*," Computers & Structures, vol. 87, no. 13-14, pp. 861–870, 2009

[15] R. Lewis (2008) A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum, 30(1). 167-190.

[16] Pan, Tasgetiren, Q-K., Suganthan, M. F., P. N., Chua, T. J (2010) A Discrete Artificial Bee Colony Algorithm for the Lot-streaming Flow Shop Scheduling Problem, Information Sciences, Elsevier, Netherlands, In Press.

[17] D. T. Pham, A. A. Afify, and E. Koç, (2007). Manufacturing cell formation using the Bees Algorithm. In Proceedings of the Third International Virtual Conference on Intelligent production machines and systems (IPROMS 2007), Whittles, Dunbeath, Scotland.

[18] D.T. Pham, M. Castellani, (2009), The Bees Algorithm – Modelling Foraging Behaviour to Solve Continuous Optimisation Problems. Proc. ImechE, Part C, 223(12): 2919-2938.

[19] R. Qu, E.K. Burke, B. McCollum, and L.T.G. Merlot, (2009) A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling, 12. 55-89.

[20] A. Singh, (2009) An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem, Applied Soft Computing 9 625–631. Timetabling (PATAT 2008)

[21] Y. Yang, and S. Petrovic, (2005). A novel similarity measure for heuristic selection in examination timetabling. In E. K. Burke & M. Trick (Eds.), Lecture notes in computer science: Vol. 3616. Practice and theory of automated timetabling V: selected papers from the 5th international conference (pp. 377–396). Berlin: Springer,

[22] B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A.J. Parkes, L. Gaspero, R. Qu, E.K. Burke, (2010). Setting the research agenda in automated timetabling: the second international timetabling competition. INFORMS. J. Computing, 22: 120-130.

[23] E.K. Burke, D.G. Elliman, P.H. Ford, R.F. Weare, (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. Lecture Notes in Computer Science, 1153. Springer.

[24] T. Muller, (2009) ITC2007 Solver Description: A Hybrid Approach. Annals of Operation Research. 172(1): 429-446.

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

[26] A. Pillay (2007). Developmental approach to the examination timetabling problem.www.cs.qub.ac.uk/itc2007.

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