# Comparison on the Selection Strategies in the Artificial Bee Colony Algorithm for Examination Timetabling Problems

**Malek Alzaqebah, Salwani Abdullah**

***Abstract*—*This paper presents an investigation of selection strategies upon the Artificial Bee Colony (ABC) algorithm in examination timetabling problems. ABC is a global stochastic optimisation algorithm that is based on the behavior of honey bee swarms. Onlooker bees in ABC algorithm choose food source based on the proportional selection strategy. In this paper, three selection strategies are introduced (i.e. disruptive, tournament and rank selection strategies), in order to improve the diversity of the population and avoid the premature convergence in the evolutionary process. Experimental results show that the modified ABC with the three selection strategies outperforms the ABC algorithm alone. Among the selection strategies, the disruptive selection strategy shows the better performance when tested on standard benchmark examination timetabling problem.***

***Index Terms: Artificial Bee Colony Algorithm, Examination Timetabling problems, Selection Strategies.***

## I. INTRODUCTION

Several approaches have been proposed for solving optimization problems. In recent years, the research trend focuses more on heuristic methods rather than the traditional methods to solve the optimisation problems. Swarm intelligence for example focuses on the behaviour of insects to develop some meta-heuristics which can mimic the insect's problem-solving. Artificial bee colony (ABC) algorithm is a part of swarm intelligence algorithms that mimics the natural behavior of real honey bees on searching for food sources. It was proposed by Karaboga [12] for numerical optimisation problem [12]. [11] Proposed an extended version of ABC algorithm for solving constrained optimisation problems.

In this paper, we treat the Examination Timetabling Problems (ETTP), which can be defined as a classic combinatorial optimisation problem. ETTP deals in assigning a number of exams into a limited number of timeslots and locations, whilst reducing the violations of a predefined set of constraints. Usually there are two types of constraints considered in ETTP i.e. hard and soft constraints. Hard constraints cannot be violated in any circumstances and violation of soft constraints is minimised as much as possible. Interested readers can find more details about ETTP research and comprehensive survey papers in [7], [8], [1], [14], [18] and [19]. Example of one of the bee family algorithms i.e. honey-bees mating optimisation algorithm that has been applied to solve ETTP can be found in [16].

The paper is organised as follow: Section II formally presents the ETTP and formulation. Section III describes the original ABC algorithm. The selection strategies that have been applied in ABC are presented in Section IV. The proposed approach is presented in Section V. Our experimental comparison are presented, discussed and evaluated in Section VI. This is followed by some brief concluding comments in Section VII.

## II. PROBLEM DESCRIPTION AND FORMULATION

ETTP can be defined as NP-hard problem that due to the difficulty of satisfying the pre-defined number of constraints. In this paper, the problem description is adapted from the explanation presented in [8]. ETTP consist of a number of inputs as follow:

- $N$ 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 $k$ $(k \in \{1,\ldots,N\})$.

An objective function is formulated which tries to space out students' exams throughout the exam period (Eq. (1)) which is considered as the soft constraint. It can be formulated as the minimisation of:

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

Where

$$F_1(i) = \sum_{j=i+1}^{N} c_{ij} \cdot prox(t_i, t_j) \qquad (3)$$

And

Manuscript Received October 18, 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).

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

Subject to:

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

Where

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

Eq. (2) presents the cost for an exam i which is given by the proximity value multiplied by the number of students in conflict. Eq. (3) represents a proximity value between two exams [10]. Eq. (4) represents a hard constraint (clash-free requirement) so that no student can sit two exams at the same time.

## III. ARTIFICIAL BEE COLONY ALGORITHM (ABC)

### A. The Basic Artificial Bee Colony (ABC) Algorithm

Artificial Bee Colony Algorithm (ABC) was introduced by Karaboga [12] as a global optimisation algorithm that simulates the foraging behavior of honey bees. In ABC, the artificial agents are defined and classified into three types i.e. the *employed bees*, the *onlooker bees*, and the *scout bees*. Each of them plays a different role in the process. The employed bees stay on a food source and provide the neighborhood of the source in its memory. The onlooker bees get the information of food sources from the employed bees in the hive, and select one of the food sources to gather the nectar, and the scout bees is responsible for finding new food sources. ABC system combines local and global search methods, where the local search method is carried out by employed bees and onlooker bees. While the global search method is managed by *onlooker bees* and *scout bees*. The possible solutions in the ABC algorithm represent *food sources* (flowers), and the fitness of the solution is corresponded to the *nectar amount* of the associated *food source*, [11].

```
Initial food sources are produced for all
employed bees
REPEAT
→Each employed bee flies to a food
  source in her memory and determines a
  neighbor 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 neighbor around that,
  onlooker evaluates its nectar amount.
→Abandoned food sources are determined
  and are replaced with the new food
  sources discovered by scout bees.
→The best food source found so far is
  registered.
UNTIL (requirements are met)
```

**Figure I. Original artificial bee colony algorithm**

Figure I shows the basic ABC algorithm as in [12]. In ABC, the number of the employed bees or onlooker bees is equal to the number of food source (SN). At the first step, initial populations (food source positions) are generated based on a constructive heuristic algorithm. An employed bee produces an adjustment on the source position in her memory and discovers a new food source position. If the nectar amount of the new food source is higher than the previous one, then the bee memorises the new food source position, otherwise the bee keeps the old position in her memory. After the employed bees complete the search process, they share the information about the position of the sources with the onlooker bees at the dance area. Each onlooker bee evaluates the nectar information that is collected from all employed bees. Based on the nectar amounts of sources she chooses a food source to produce an adjustment on the source position in her memory, and checks its nectar amount. Scout bees determine the abandoned sources and produce new sources randomly in order to replace the abandoned ones.

### B. Onlooker Bees Selection Process

Onlooker bees select the solution by a stochastic selection scheme, which consists of three steps:

1. Calculates the fitness value by using the fitness function as follow:

$$fit_i=\frac{1}{1+f_i} \quad (5)$$

Where $f_i$ is fitness function and $fit_i$ is the fitness function after a transformation.

2. Calculate the probability value by using the following expression:

$$p_i=\frac{fit_i}{\sum_{i=1}^{SN} fit_i} \quad (6)$$

Where *SN* is the number of food sources, $f_i$ is the fitness function of the $i$th food source.

3. Finally, chose a candidate solution based on the selection probability by "roulette wheel selection" method.

As stated in [3], they mention that, there are two problems of using basic ABC selection strategy as below: "(i) A "super-individual" being too often selected the whole population tends to converge towards his position. The diversity of the population is then too reduced to allow the algorithm to progress; (ii) with the progression of the algorithm, the differences between fitness are reduced. The best ones then get quite the same selection probability as the others and the algorithm stops progressing." Thus, this selection strategy is hard to keep the diversity and to avoid the premature convergence. In order to alleviate these problems, this paper employed three different selection strategies to improve the performance of the ABC algorithm.

## IV. SELECTION STRATEGIES

In this work, we incorporate three selection strategies with ABC algorithm and tested on ETTP. The description of the selection strategy used is described as below:

- **Tournament selection:** The tournament selection is a selection process where a number of individuals ($N_{tour}$) from the population are chosen at random, and then the comparison is made depending on the fitness in order to take the best individual. Parameter $N_{tour}$ is called a tournament size. Normally, tournaments are held only between two individuals (binary tournament), but a generalisation is possible to an arbitrary group. Tournament selection gives more chances for the individuals with high fitness to survive, [5]. In this work, we select two individuals from the population and compare their fitness values, then assign one score (coded as *a*) to a better individual. Repeat such process for all the individuals in the population as shown in Figure II, where $f_i$ is the fitness value of $i$ = 0....$n$, where $n$ is the population size (adapted from, Bao and Zeng, (2009)).

```
for i=1:n
    $a_i \leftarrow 0$;
    for j=1:n
        if $f_i \le f_j$
            $a_i = a_i + 1$;
        end if
    end for
endfor
```

**Figure II. Tournament selection pseudo code**

After calculating the value of (a) for all the individuals, the selection probability for each individuals is calculated using Eq. (7).

$$P_i = \frac{a_i}{\sum_{i=0}^{n} a_i} \tag{7}$$

- **Rank selection:** In the rank selection, the fitness value is calculated using Eq. (5). Individuals are sorted in descending order based on the fitness value. The index $k$ is given to each individual from the best to the worst, i.e. for the best fitness $k$ =1, and for the worst fitness $k = n$, where $n$ is the population size and $N$ is the maximum number of iterations. Finally, the selection probability is calculated using Eq. (8), [17]:

$$P_k = \frac{1}{n} + a(t)\frac{n+1-2k}{n(n+1)}, \quad k = (1,2,.....n) \tag{8}$$

$$where \qquad a(t) = 0.2 + \frac{3t}{4N}, \quad t = (1,2,.....N)$$

- **Disruptive selection:** Disruptive selection gives more chance for higher and lower individuals to be selected by changing the definition of the fitness function as in Eq. (9), [13].

$$fit_i = | f_i - \overline{f_t} | \qquad P_i \frac{fit_i}{\sum_{i=0}^{n} fit_i} \tag{9}$$

Where $f_i$ is the fitness function, $\overline{f_t}$ is the average value of the fitness function $f_i$ of the individuals in the population.

## V. THE ALGORITHM: ARTIFICIAL BEE COLONY SEARCH ALGORITHM

### A. Construction Heuristic

In this work, we used the graph colouring approach (i.e. 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 see [8].

### B. Improvement Algorithm: Artificial Bee Colony Search Algorithm

Figure III illustrates the pseudo code of the proposed approach. The algorithm starts with feasible initial solutions which are generated by a largest degree heuristic in the constructive phase. The position of a food source represents a possible solution and the *nectar amount* of a 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.

The employed bees work on random solutions and apply a random neighborhood structure on each solution. 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 the employed bees complete the search process, they share the position information of the sources with the onlooker bees on the dance area. Onlooker bees work on the selected solution based on the selection strategy explained above, 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.

```
Initialization:
Initialise the initial population and
evaluate the fitness;
Calculate the initial fitness 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
       Sol*  ← Select a random solution
       Sol** ← Apply a random
                  neighbourhood structure
                  on          Sol*;
       if (Sol** < Solbest)
            Solbest=Sol**;

     end for
     for i=1: OnlookerBee
        Calculate the selection
          probability $P_i$, based on the
          corresponding selection
          strategy (minimise of [4]:
          Eq.(6), Eq.(7), Eq.(8) or
          Eq.(9), respectively)
          Sol*  ← select the solution
                  depending on $P_i$;
          Sol** ←  Apply a random
          neighbourhood structure 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 III. The pseudo code for the artificial bee colony search algorithm**

### C. Neighborhood Structure

In this paper, ten neighborhood structures are employed in order to enhance the performance of searching algorithms. These neighborhood structures are presented as [1]:

**Nbs1** Select two exams at random and swap timeslots.

**Nbs2** Choose a single exam at random and move to a new random feasible timeslot.

**Nbs3** Select two timeslots at random and simply swap all the exams in one timeslot with all the exams in the other timeslot.

**Nbs4** Select 3 exams randomly and swap the timeslots between them feasibly.

**Nbs5** Select 4 exams randomly and swap the timeslots between them feasibly.

**Nbs6** 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.

**Nbs7** Move the highest penalty exams from a random 10% selection of the exams to a random feasible timeslot.

**Nbs8** Carry out the same process as in Nbs7 but with 20% of the exams.

**Nbs9** 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.

**Nbs10** Carry out the same process as in Nbs9 but with 20% of the exams.

## VI. EXPERIMENTAL COMPARISON

Table I shows the parameters setting, which have been used in this work.

TABLE I. PARAMETERS SETTING.

| Parameter | Value |
|---|---|
| Iteration | 500 |
| population size | 50 |
| scout bee | 1 |

TABLE II. RESULTS COMPARISON.

| Dataset | ABC | | DABC | | RABC | | TABC | | Best known |
|---|---|---|---|---|---|---|---|---|---|
| | *Best* | *Avg.* | *Best* | *Avg.* | *Best* | *Avg.* | *Best* | *Avg.* | |
| **car91** | 5.86 | 6.03 | 5.42 | 5.83 | 5.88 | 5.98 | **5.38** | 5.78 | **4.50** |
| **car92** | 4.92 | 5.21 | **4.84** | 4.9 | 4.98 | 5.01 | 4.98 | 5.01 | **3.98** |
| **ear83I** | 38.34 | 38.71 | **37.54** | 37.73 | 37.67 | 37.9 | 37.88 | 38.2 | **29.3** |
| **hec92I** | 11.51 | 11.93 | **11.21** | 11.52 | 11.28 | 11.71 | 11.5 | 11.64 | **9.2** |
| **kfu93** | 16.04 | 16.56 | **15.13** | 15.83 | 16.06 | 16.4 | 15.78 | 16.1 | **13.0** |
| **lse91** | 12.42 | 12.95 | **12.06** | 12.62 | 12.81 | 13.01 | 12.41 | 12.91 | **9.6** |
| **sta83I** | 158.12 | 158.63 | **157.52** | 157.76 | 157.78 | 158.03 | 157.69 | 157.81 | **3.7** |
| **tre92** | 9.58 | 10.12 | **9.23** | 9.79 | 9.46 | 9.98 | 9.49 | 10.02 | **6.8** |
| **uta92I** | 3.99 | 4.61 | **3.94** | 4.02 | 3.98 | 4.22 | 3.95 | 4.1 | **156.9** |
| **ute92** | 27.80 | 28.51 | **27.57** | 27.90 | 27.63 | 27.98 | 27.6 | 27.74 | **7.9** |
| **yor83I** | 41.44 | 41.87 | **40.94** | 41.23 | 41.83 | 42.1 | 41.28 | 41.64 | **3.14** |

Table II provides the results comparison between three modified ABC algorithms (i.e. ABC with different selection strategies) and the basic ABC.

Three different modified ABC algorithms are called ABC algorithm based on disruptive selection (DABC), ABC algorithm based on rank selection (RABC), and ABC algorithm based on tournament selection (TABC). We run the experiments for 5 times for each dataset. Note that, the dataset's specification of the examination timetabling problems that were proposed by [10]. As shown in Table II the best results are presented in bold. The above comparison shows that, the ABC algorithm with three selection strategies perform better than the basic ABC algorithm alone. However, in most of the tested datasets, DABC outperforms other algorithms in comparison. The comparison between ABC algorithm with three selection strategies and the best known results shows that even we are unable to beat any of the best known results in the literature, but we are still able to produce promising solutions.

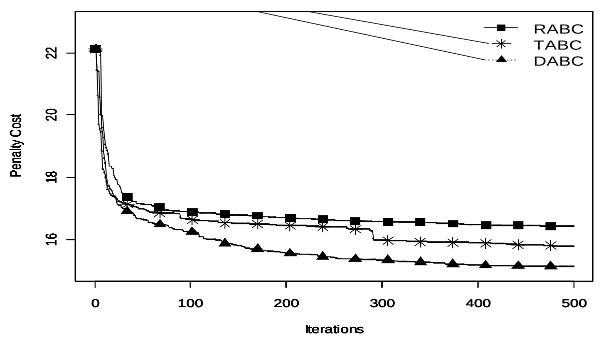
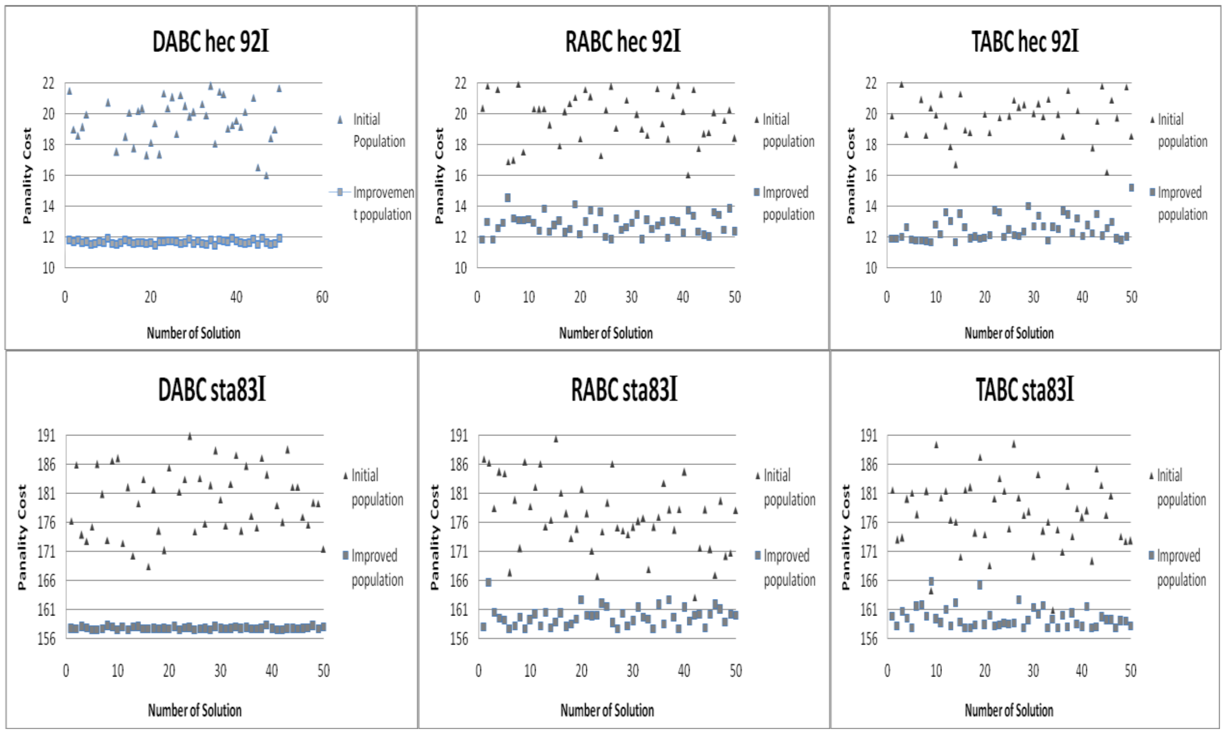


**Figure IV. Convergence of kfu93 dataset**

Figure IV shows that the behaviour of ABC algorithm based on three selection strategies over kfu93 dataset. The x-axis represents the number of iteration, while the y-axis represents the penalty cost. This graph shows how DABC, TABC and RABC explore the search space. 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 [15]. The higher conflict density signifies that more exams are conflicting with each other. The conflict density value for kfu93 is 0.06. As shown in Figure IV, the behaviour of the three selection strategies 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, the graph shows that DABC can explore the search space better than RABC and TABC. This is due to the nature of the selection strategy, where the tournament selection randomly selects a number of solutions ($N_{tour}$) and compares them based on a probability. The solution with a highest fitness value will be chosen. In a rank selection, the solutions are ranked based on the fitness values, so this function is biased to work with the solution at higher rank (i.e. good fitness), while the disruptive selection concentrates on both the worse and the high fitness, and tries to keep the diversity of population by improving the worse fitness solutions in concurrent with the high fitness solutions.

Figure V shows the convergence of three datasets i.e. hec92I, sta83I and tre92. The x-axis represents the number of solution, while the y-axis represents the penalty cost.



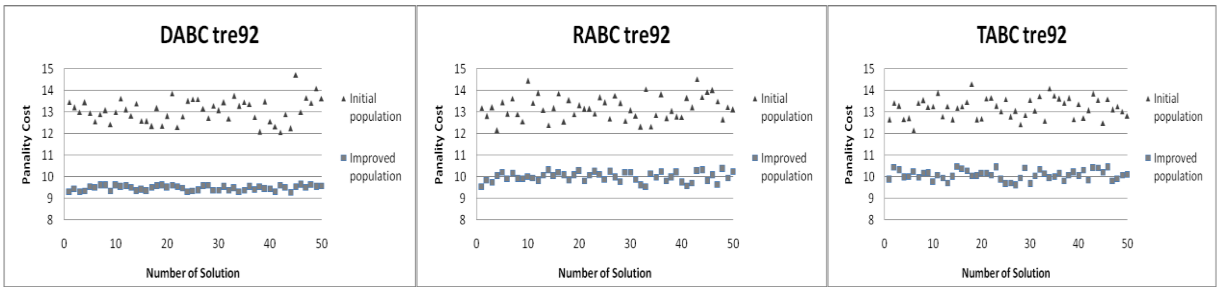


**Figure V. Convergence of three datasets i.e. hec92I, sta83I and tre92**

These graphs show how the DABC, TABC and RABC spray the population at initial stage (represented by the triangle symbol), and then after 500 iterations the improved solutions are represented by the square symbol. From these figures, we can conclude that the DABC gives a chance for all the solutions in the population to be improved and converged together. This can be seen that the plotted square symbols are concentrated (not scattered) to each other, that represents the closeness of the quality of the solutions in the population.

## VI. CONCLUSION AND FUTURE WORK

The aim of this paper is to compare the performance of the ABC algorithm when uses different selection strategies. Through the results obtained, it is concluded that ABC algorithm with a disruptive selection strategy is able to produce better results when compared to other selection strategies tested in this work. We believe the performance of the ABC algorithm can be enhanced by applying a suitable mechanism to choose the neighborhood structure based on the current solution in hand. We also believe that the hybridisation of the ABC algorithm based on a disruptive selection with a local search will further improve the solution obtained so far. This is subject to the future work.

## REFERENCES

1. 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. Springer, ISBN:978-0-387-71919-1.
2. 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.
3. L. Bao and J. Zeng, (2009). Comparison and Analysis of the Selection Mechanism in the Artificial Bee Colony Algorithm. HIS (1) 411-416
4. A. Baykasoglu, L. Ozbakır and P. Tapkan, (2007). Artificial Bee Colony Algorithm and Its Application to Generalized Assignment Problem, Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, I-Tech Education and Publishing.
5. T. Blickle and L. Thiele, (1995). A Mathematical Analysis of Tournament Selection, Proc. of the Sixth International Conference on Genetic Algorithms, San Francisco, CA, pp. 2-8.
6. 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.
7. E.K. Burke, D.G. Elliman, P.H. Ford and R.F. Weare, (1996). Examination timetabling in British universities - A survey. In E.K. Burke and P. Ross. (eds). Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 1153, 76-92.
8. E E.K. Burke and J.P. Newall. Solving examination timetabling problems through adaptation of heuristic orderings. Annals of Operations Research, 129, (2004) 107-134.
9. M. Caramia, P. Dell'Olmo and G.F. Italiano, (2001). New algorithms for examination timetabling. Algorithms Engineering 4th International Workshop, Proceedings WAE, Saarbrücken, Germany, Springer Lecture Notes in Computer Science, vol. 1982. (2001) 230-241.
10. M.W. Carter, G. Laporte and S.Y. Lee, (1996). Examination Timetabling: Algorithmic Strategies and Applications. Journal of the Operational Research Society 47, 373-383.
11. D. Karaboga, and B. Basturk, (2007). Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems, LNCS: Advances in Soft Computing: Foundations of Fuzzy Logic and Soft Computing, Vol: 4529/2007, Springer- Verlag, IFSA 789-798.
12. D., Karaboga, (2005). An Idea Based On Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department.
13. T. Kuo, and S. Y. Huang, (1997). Using Disruptive Selection to Maintain Diversity in Genetic Algorithms, Applied Intelligence, vol.7, no.3, 257-267.
14. 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.
15. 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.
16. N. R. Sabar, M. Ayob and G. Kendall, (2009). Solving examination timetabling problems using honey-bee mating optimization (ETP-HBMO). In: Proceedings of the 4th Multidisciplinary International Scheduling Conference: Theory andAp- plications (MISTA 2009), 10–12 Aug 2009, Dublin, Ireland. 399–408.
17. A.G. Song, and J.R. Luo, (1999). A Ranking Based Adaptive Evolutionary Operator Genetic Algorithm, Acta Electronica Sinica, vol.27 no.1, 85-88.
18. H. Turabieh, and S. Abdullah, (2011). A Hybrid Fish Swarm Optimisation Algorithm for Solving the Examination Timetabling Problems. Learning and Intelligent Optimisation Workshop (LION 5), Rome, LNCS, Springer-Verlag Berlin.
19. H. Turabieh and S. Abdullah, (2011). An Integrated Hybrid Approach to the Examination Timetabling Problem. OMEGA - The International Journal of Management Science, doi.org/10.1016/j.omega.2010.12.005.
20. F. Von and Karl, (1974). Decoding the Language of the Bee, Science, Volume 185, Issue 4152, 663-668.
21. Y. Yang and S. Petrovic, (2005). "A novel similarity measure for heuristic selection in examination timetabling," Lecture Notes in Comput. Sci., vol 3616, pp. 334-353. [Practice and Theory of Automated Timetabling V, 2004].