The effect of neighborhood structures on examination timetabling with artificial bee colony

Asaju La'aro Bolaji · Ahamad Tajudin Khader · Mohammed Azmi Al-Betar · Mohammed A. Awadallah · J. Joshua Thomas

the date of receipt and acceptance should be inserted later

Abstract Artificial Bee Colony (ABC) algorithm is among the most effective nature-inspired algorithms for solving the combinatorial optimization problems. In this paper, ABC is adopted for university examination timetabling problems (UETP) using a *defacto* dataset established by Carter et al. (1996). ABC has three main operators that drive the search toward the global minima: employed bee, onlooker bee, and scout. For UETP, the employed bee and onlooker bee operators are manipulated to be workable where three neighborhood structures are employed: move, swap and Kempe chain. The effect of these neighborhood structures on the behaviour of ABC for UETP is studied and analyzed in this paper. The experimental design is intentionally made with various convergence cases of different neighborhood structure. The result suggests that the ABC combined with the three neighborhood structures is an effective method for UETP. Comparative evaluation with previous methods is also provided. The results produced by the proposed method are competitive in

Asaju La'aro Bolaji
School of Computer Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia and Department of Computer Science, University of Ilorin, Ilorin, Nigeria.
E-mail: abl10_sa0739@student.usm.my

Ahamad Tajudin Khader
School of Computer Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia.
E-mail: tajudin@cs.usm.my

Mohammed Azmi Al-Betar
School of Computer Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia and Department of Computer Science, Jadara University, Irbid, Jordan.
E-mail: mobetar@cs.usm.my

Mohammed A. Awadallah
School of Computer Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia.
E-mail: mama10_com018@student.usm.my

J. Joshua Thomas
School of Computer Sciences, Universiti Sains Malaysia, 11800, Pulau Pinang, Malaysia.
E-mail: joshopever@yahoo.com

comparison with state of the art methods. Theoretically, this study contributes to the examination timetabling community through an ABC template which is both efficient and flexible for UETP.

Keywords artificial bee colony · nature-inspired algorithm · examination timetabling problem

1 Introduction

Timetabling is an hard scheduling problem faced by many institutions across the globe. Such problem involves assigning a set of events (i.e. courses and exams) to a limited set of resources (i.e. rooms and timeslot), subject to satisfying a set of constraints. The production of a high quality timetabling solution with minimum violations of constraints is one of the major concerns of almost all institutions. Considerable efforts have been exerted by the researchers in the scheduling field to develop an effective techniques to tackle the timetabling problems. Generally, these problems are referred to as NP-hard (Garey and Johnson, 1979) and have been extensively studied in the last three decades (Abramson and Abela, 1991; Burke et al, 1996; Carter and Laporte, 1996; Kostuch, 2005; Burke et al, 2010).

The timetabling problems comes in different forms: educational timetables, nurse scheduling, sport timetables and transportation timetables. The most common example of the educational timetabling problems is the University Course Timetabling Problem (UCTP) and the University Examination Timetabling Problem (UETP) which is the focus of this paper. Both have minor differences in their constraints, where for example, in UCTP, only one course can be assigned to a room at a specific timeslot while for UETP, two or three exams can take place in the same room and timeslot as long as all constraints are met.

The university examination timetabling problem can be defined as the assignment of exams to a limited number of time periods and rooms, subject to a set of hard and soft constraints (Qu et al, 2009b). The main purpose is to generate high-quality timetabling solution that satisfies the hard constraints and reduces the violations of soft constraints as much as possible. A timetable is feasible, if the hard constraints are satisfied and the quality of the timetabling solution is measured by the violations of soft constraints. Examination timetabling problems can be divided into capacitated or uncapacitated with respect to room constraints (Qu et al, 2009b). The uncapacitated examination timetabling problem is the focus of this paper.

Several techniques have been proposed for tackling uncapacitated examination timetabling in the literature, since there is no one single technique that can provide an exact solution (Millar and Kiragu, 1998). One of the most successful methods used to tackle UETP is the meta-heuristic based techniques. These could be divided into local-based search methods as Great Deluge (GD) (Burke et al, 2004; Burke and Newall, 2003; McCollum et al, 2009), Simulated Annealing (Thompson and Dowsland, 1996, 1998), Tabu

Search (Di Gaspero and Schaerf, 2001; White and Xie, 2001; White et al, 2004; Kendall and Hussin, 2005), Variable Neighbourhood Search (VNS) (Ahmadi et al, 2003; Burke et al, 2010)[8] and population-based method such as Ant Colony (Dowsland and Thompson, 2004; Eley, 2006), Evolutionary Algorithms (Paquete and Fonseca, 2001; Côté et al, 2005), Particle Swarm Optimization (Fealko and Adviser-Mukherjee, 2006), Harmony Search Algorithm (Al-Betar and Khader, 2008; Al-Betar et al, 2010a,b, 2011b) and memetic algorithms (Burke et al, 1996; Merlot et al, 2003).

A new nature-inspired algorithm called Artificial Bee Colony (ABC) has been recently proposed by Karaboga who was inspired by imitating the intelligent behaviour of honey bee (Karaboga, 2005). It has been successfully applied to a wide variety of optimisation problems as shown in the survey paper (Karaboga and Akay, 2009b).

The ABC as a stochastic search algorithm firstly begins with an initial population stored in Food Source Memory (FSM). At every iteration, new food sources (solutions) are generated from the neighbouring of the existing population using three operators: Employed bee, Onlooker bee and Scout bee. The new food sources are then evaluated against an objective function and replaced the old population, if their fitnesses are better. This process is repeated until the termination criteria is reached.

Defining efficient neighborhood structures that appropriate to the nature of the combinatorial optimization problem is a big challenge that influences the performance of the algorithm (Aladag & Hocaoglu, 2007). Three neighbourhood structures are incorporated into the employed and onlooker operators namely, move, swap and kempe chain. In this study, an extensive analysis of the effect of neighbourhood structure on the behaviour of ABC for UETP is conducted. Then, the effectiveness of ABC with each and combined neighbourhood operators is evaluated on 13 standard benchmark datasets reflecting real-world examination timetabling instances which were introduced by Carter and Laporte (1996). The ABC with three neighbourhood structures achieved comparably competitive results.

The rest of the paper is organized as follows: Section 2 gives descriptions and formulations of the examination timetabling problem while section 3 presents the fundamentals of ABC. Section 4 describes ABC approach for examination timetabling and the neighbourhood structures is presented in section 5. Section 6 provides an explanation to the experimental results while the last section is devoted for conclusion and some future works.

2 Problem descriptions and formulations

Tackling the exam timetabling problem involves scheduling a set of exams, each taken by a set of students, to a set of time periods (timeslots) subject to hard and soft constraints. The main objective is to obtain a timetable that satisfies the hard constraint (H1) with the minimum penalty of the soft constraint violation (S1). The hard and soft constraints are as follows:

Table 1 The symbols used in the description of UETP

Symbols	Definition
N	The total number of exams.
M	The total number of students.
P	The total number of time periods.
E	Set of exams
S	Set of students
T	Set of time periods
$\boldsymbol{x}$	A timetable solution is given by $(x_1, x_2, \ldots, x_M)$.
x_i	The timeslot of exam i.
$a_{i,j}$	Proximity coefficient matrix element : whether the timetable $\boldsymbol{x}$ is penalized based on the distance between time period of exam i, and time period of exam j $a_{i,j} = \begin{cases} 2^{5-\lvert x_i - x_j \rvert} & \text{if } 1 \leq \lvert x_i - x_j \rvert \leq 5 \\ 0 & \text{Otherwise.} \end{cases}$
$u_{i,j}$	Student-exam matrix element: if student s_i is taking exam j $u_{i,j} = \begin{cases} 1 \text{ if student } i \text{ is sitting for exam } j \\ 0 \text{ Otherwise.} \end{cases}$
$c_{i,j}$	Conflict matrix element: total number of students sharing exam i and exam j. $c_{i,j} = \sum_{k=1}^{N} u_{k,i} \times u_{k,j} \quad \forall i, j \in E$

– H1: no student can sit for two exams simultaneously.
– S1: the exams taken by the same student should be spread out evenly across a timetable.

A detailed description of the problem is summarized by Qu et al (2009b). A timetabling solution is represented by a vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_M)$ of exams, where the value of x_i is the timeslot for exam i. The proximity cost function is used in the evaluation of the timetable by Qu et al (2009b) and refers to the ratio of the penalty assigned to the total number of soft constraint violations and the total number of students. The formulation for the proximity cost function is given in equation (1), while the notation of the variable used is shown in Table 1.

$$\min f(x) = \frac{1}{N} \times \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} c_{i,j} \times a_{i,j} \tag{1}$$

H1: No student can sit for two exams simultaneously

$$x_i \neq x_j \quad \forall x_i, x_j \in X \wedge c_{i,j} \geq 1$$

It is important to note that the value of the proximity cost function *f(x)* is referred to as the fitness cost of a feasible timetable (Carter and Laporte, 1996).

The Carter dataset used in this study consists of 13 datasets, which reflect the real-world examination timetabling problems. For the purpose of our study,

Table 2 Table 2: Characteristics of Uncapacitated Exam Dataset

Dataset	Time-Periods	Exams	Students	Density
CAR-S-91-I	35	682	16,925	0.13
CAR-F-92-I	32	543	18,419	0.14
EAR-F-83-I	24	190	1125	0.27
HEC-S-92-I	18	81	2823	0.42
KFU-S-93	20	461	5349	0.06
LSE-F-91	18	381	2726	0.06
RYE-S-93	23	481	11,483	0.07
STA-F-83-I	13	139	611	0.14
TRE-S-92	23	261	4360	0.18
UTA-S-92-I	35	622	21,266	0.13
UTE-S-92	10	184	2750	0.08
YOR-F-83-I	21	181	941	0.29

12 datasets circulated in the literature were used. The characteristics of Carter datasets, varying in size and complexity, are shown in Table 2. The last column illustrates the density of the conflict matrix, which is the ratio between the number of elements of values $c_{i,j} > 0$ and the total number of elements in the conflict matrix (Qu et al, 2009b).

3 Artificial Bee Colony Algorithm

Artificial Bee Colony (ABC) is a swarm metaheuristic algorithm which was originally introduced in 2005 by Karaboga for tackling numerical optimization problems (Karaboga, 2005). This algorithm is considered a stochastic optimization algorithm based on the model proposed by Teodorović and DellOrco (2005) for the foraging manners of honey bee in their colonies. The ABC consists of three vital components: employed, unemployed foraging bees, and food sources. The first two components i.e., employed and unemployed forager search for rich food sources, which is the third component. The two principal modes of behaviour which are necessary for self-organization and collective intelligence are also described by the model. In practice, such mode includes the recruitment of foragers to the rich food sources resulting in positive feedback and abandonment of poor food sources by foragers causing negative feedback.

In the colony of ABC there are three groups of bees: employed, onlooker and scout bees. Associated with particular food source is employed bee whose behaviour is studied by the onlookers to select the desired food source while the scout bee searches for new food sources randomly once it is exhausted. Both onlookers and scouts are considered as unemployed foragers. The position of a food source in ABC corresponds to the possible solution of the problem to be optimized and the nectar amount of a food source represents the fitness (quality) of the associated solution. The number of employed bees is equal to the number of food sources (solutions), since each employed bee is associated with one and only one food source (Karaboga, 2005).

The key phases of the algorithm as proposed by Karaboga and Akay (2009a) are as follows:

- Generate the initial population of the food sources randomly.
- REPEAT
  - Send the employed bees onto the food sources and calculate the fitness cost.
  - Evaluate the probability values for the food sources
  - Send the onlooker bees onto the food sources depending on probability and calculate the fitness cost.
  - Abandon the exploitation process, if the sources are exhausted by the bees.
  - Send the scouts into the search area for discovering new food sources, randomly
  - Memorize the best food source found so far.
- UNTIL (requirements are met).

4 Artificial Bee Colony for UETP

The implementation of ABC for the UETP includes the followings six steps:

1. **Step 1: Initialization of the ABC and UETP parameters:**
   The parameters of UETP are normally extracted from the problem instances. These parameters include the set of exams, the set of timeslots, the set of rooms, etc. The main decision variable of UETP is the exams. Each exam can be assigned to a feasible timeslot in the timetable solution. A set of all feasible timeslots can be considered as the available range of such exams. In fact, the feasible timeslot of each exam changes during the search in the neighbourhood of ABC. The proximity cost function described in (1) is used to evaluate each solution. Here, the parameters of the ABC used for UETP are initialized. That is, the Solution Number (SN) which is similar to the population size in genetic algorithms; Maximum Cycle Number (MCN) which is similar to the number of iterations and Limit which the determine when a solution will be abandoned. These parameters will be explained in more detail in the next steps.
2. **Step 2: Initialize the Food Source Memory (FSM):**
   The Food Source Memory (FSM) is an augmented matrix of size SN comprising a vector in each row representing a timetable solution as in (2). Note that the vectors in FSM are generated with a method that combines the saturation degree (SD) and backtracking algorithms as used previously (Al-Betar et al, 2010b; Bolaji et al, 2011). Here, SD starts with an empty timetable, where the exam with the least number of valid timeslots in the scheduled list is assigned first. The next selected exam to be scheduled is based on the number of available timeslots; where some exams may not be assigned because of non-availability of the timeslots, then the backtracking algorithm (BA) is applied to re-assign unscheduled exams. The process, SD and BA, is repeated several times until all exams are assigned to feasible timeslots. Using this techniques, the feasibility of all the solutions is guaranteed and sorted in ascending order according to their objective function

values.

$$\mathbf{FSM} = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{SN} & x_2^{SN} & \cdots & x_N^{SN} \end{bmatrix} \begin{bmatrix} f(\boldsymbol{x}^1) \\ f(\boldsymbol{x}^2) \\ \vdots \\ f(\boldsymbol{x}^{SN}) \end{bmatrix} \tag{2}$$

.

3. **Step 3: Send the employed bees to the food sources:**
   Here, the employed bee operator selects a timetabling solution from the population one by one and applies the three neighbourhood structures to generate new solutions. The fitness of each offspring solution is calculated. If it is better than that of parent solution, then the offspring replaces the parent in FSM. This process is iteratively repeated until all solutions have been explored (see algorithm: 1 for details). where $\boldsymbol{x}^i$ is the solution and

Algorithm 1 Employed Bee Phase

```
for i = 1 ··· SN do
    rand ∈ {1, 2, 3}
    if rand = 1 then
        x^{i(new)} = Move(x^i)
    else
        if rand = 2 then
            x^{i(new)} = Swap(x^i)
        else
            if rand = 3 then
                x^{i(new)} = Kempe(x^i)
            end if
        end if
    end if
    if x^{i(new)} is better than x^i then
        x^i = x^{i(new)}
    end if
    next i
end for
```

$\boldsymbol{x}^{i(new)}$ is the new neighbouring solution.

4. **Step 4: Send the onlooker bees:**
   The onlooker bee has the same food sources (timetabling solutions) of the employed bees. It initially calculates the selection probability of each food source generated by the employed bee in the previous step. The fittest food sources are selected by the onlooker using roulette wheel selection mechanism. The process of selection in the onlooker phase works as follows:
   - Assign to each employee bee a selection probability as follows:

$$p_j = \frac{f(\boldsymbol{x}^j)}{\sum_{k=1}^{SN} f(\boldsymbol{x}^k)}$$

Note that the $\sum_{i=1}^{SN} p_i$ is unity.

  - The onlooker sample the fitness of the food source of each employed bee and selects the one with highest probability. It then adjust the selected food source to its neighbourhood using the same strategy as the employed bee. The fitness of the new solution is calculated and if it is better it replaces the current one.
5. **Step 5: Send the Scout to search for possible new food sources:** This is known to be the colony explorer. It works once a solution is abandoned, i.e. if a solution in the FSM has not improved for certain number of iterations. ABC generates a new solution randomly and substitutes the abandoned one. Memorize the fitness of the best food source x_{best} found so far in FSM.
6. **Step 6: Stopping Criteria:**
   Steps 3 to 5 are repeated until a stop criterion is met. This is originally determined using MCN value.

5 Neighbourhood Structure (NS)

In this section, the neighbourhood structures (Move, swap and kempe chain) used in the employed and onlooker phases given in section 4 shall be described in details. Neighbourhood structure (NS) is a commonly used technique in solving timetabling problems. NS can be used to obtain a new set of neighbor solutions by applying a small perturbation to a given solution and each neighbourhood solution is reached immediately from a given solution by a move (Glover and Laguna, 1998). The neighbourhood structure begins with an initial solution and progressively explores the neighborhood of the solution for improvement. Thus, the current solution is iteratively replaced by one of its neighbors (often improving) until a specific stopping condition is met (Lü et al, 2011). The ABC operators such as employed and onlooker bees, use three different neighbourhood structures to explore the solution space thoroughly in order to enhance the quality of the solution and thus reduce the redundancy or ineffectiveness of using a particular type alone. The three neighbourhood structures are: move, swap, and kempe chain. They have been used by other researchers and proven to be very efficient for exam timetabling problems (Al-Betar et al, 2010a; Thompson and Dowsland, 1998; Burke et al, 2010).

- **Neighbourhood Move (NM)**: moves selected exam to a feasible period and room randomly i.e. replace the time period x'_i of exam i by another feasible timeslot.
- **Neighbourhood Swap (NS)**: swap two selected exams at random i.e. select exam i and event j randomly, swap their time periods (x'_i, x'_j).
- **Neighbourhood Kempe Chain (NK)**: Firstly, select the timeslot x'_i of exam i and randomly select another q' timeslot. Secondly, all exams that have the same timeslot x'_i that are in conflict with one or more exams timetabled in q_i are entered to chain δ where $\delta = \{j|x'_j = x'_i \wedge t_{i,q'} = 0 \wedge \forall j \in E\}$. Thirdly, all exams that have the same timeslot q' that are conflicting with one or more exams timetabled in x'_i are entered to a chain

δ' where $\delta' = \{k|x'_k = q' \wedge t_{k,x'_i} = 0 \wedge \forall k \in E\}$ and Lastly, simply assign the exams in δ to q' and the exams in δ' to x'_i.

6 Experimental Results and Analysis

The method is coded in Microsoft Visual C++ 6.0 on Windows 7 platform on Intel 2 GHz Core 2 Quad processor with 2 GB of RAM. The ABC required a maximum of 7 hours to obtain the recorded result, although the computational time is not provided in the literature. The parameters used for ABC is as follows: MCN=10,000; SN=10; limit=100.

In this section, the effectiveness of neighborhood structure on the performance of ABC-based UETP is experimentally studied using the Carter dataset. Seven convergence cases are run where each representing a version of ABC combined single, double or triple combinations of the neighborhood structures within the employee and onlooker bees operators as shown in Table 3. For example,case 1 is ABC version with a single move combined with employee and onlooker bees. Apparently, all possible combinations of the neighborhood structures are studied separately.

Each convergence case is ran ten times. The best result amongst the ten runs of each case is recorded in Table 4 for each Carter dataset. Numbers in Table 4 refer to the penalty value of the soft constraint violations. (lowest is best). The best solution achieved by any version of ABC is highlighted in bold.

Generally, the ABC combined with the three neighborhood structures (i.e., case 7) has a better performance than all other cases that combine single or double neighborhood structures. Furthermore, case 5 that combines MOVE and KEMPE is able to compete with case 7. This shows the efficiency of combining these neighborhood structures with ABC for UETP. Apparently, the performance of case 3 is better than case 1 and 2, in terms of the solution quality in almost all the instances and with little difference between the 2 cases (case 1, 2, 3 combined single neighborhood structure). However, with different combinations of these neighborhoods, the efficiency of ABC is clearly improved with further reduction in the proximity cost function as shown from cases 4 to 6. A plausible observation can be set as the combination of two or more neighborhood structures within ABC-based UETP enhances the search capability and therefore an impressive result is obtained. It is worth mentioning that each neighborhood structure is able to navigate the UETP search space in a way different from the others. As such, the selection of an efficient neighborhood structure is inevitably required to achieve superior results.

Table 5 and 6 showed the penalty value achieved by the proposed method in comparison with those provided by some state of the art techniques and best known results as given by (Qu et al, 2009b). This includes a total of 10 comparative methods comprising metaheuristic-based methods, Heuristic and Hyper-Heuristic Methods. The key for the comparative methods is as follows:

M1: Graph-Based Hyper-Heuristic (Burke et al, 2007).
M2: Graph-Based Hyper-Heuristic (Qu et al, 2009a).

Table 3 Cases studied the effect of neighborhood structure on ABC-based UETP

CASE	MOVE	SWAP	KEMPE
Case 1	✓	*X*	*X*
Case 2	*X*	✓	*X*
Case 3	*X*	*X*	✓
Case 4	✓	✓	*X*
Case 5	✓	*X*	✓
Case 6	*X*	✓	✓
Case 7	✓	✓	✓

M3: Graph-Based Hyper-Heuristic (Qu and Burke, 2008).
M4: Graph-Based Hyper-Heuristic (Pillay and Banzhaf, 2009).
M5: Fuzzy Multiple Heuristic Orderings (Asmuni et al, 2009).
M6: Harmony Search Algorithm (Al-Betar et al, 2010b).
M7: Ant Algorithms (Eley, 2006).
M8: Multi-Objective Evolutionary Algorithm (Côté et al, 2005).
M9: An integrated hybrid approach by (Turabieh and Abdullah, 2011).
M10: A hybrid Variable Neighbourhood Search with Genetic Algorithm (Burke et al, 2010).

As shown in table 5 and 6, it can be seen that ABC algorithm produce comparable results. The best penalty values (lowest is best) are highlighted in bold, while '+' indicates that the method could not find a feasible timetable. In general, it is able to achieve very close to the best results.

Table 4 Experimental Results

Dataset	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7
CAR-S-91-I	6.79	6.50	6.86	6.48	5.70	5.99	**5.38**
CAR-F-92-I	5.71	5.43	5.64	5.39	4.75	4.66	**4.61**
EAR-F-83-I	46.37	47.60	42.79	44.79	38.83	39.18	**38.58**
HEC-S-92-I	13.54	14.99	11.81	14.02	11.34	15.98	**11.17**
KFU-S-93	18.49	16.88	17.14	17.35	15.04	15.00	**14.89**
LSE-F-91	14.88	14.55	15.38	13.94	12.19	12.17	**11.74**
RYE-S-93	12.18	11.97	13.56	12.12	10.11	9.91	**9.80**
STA-F-83-I	161.35	162.77	158.87	161.62	157.30	157.42	**157.21**
TRE-S-92	11.04	10.93	10.25	10.03	9.26	9.14	**8.96**
UTA-S-92-I	4.52	4.46	4.49	4.44	3.81	3.82	**3.65**
UTE-S-92	30.44	29.52	31.43	27.46	27.88	27.43	**26.89**
YOR-F-83-I	45.61	48.55	43.88	45.21	40.43	39.84	**39.34**

7 Conclusion

In order to tackle the university examination timetabling problems (UETP), Artificial Bee Colony has been presented using a *defacto* dataset established by Carter et al. (1996). Three main operators in ABC are able to guide the search toward the global optima: employee bee, onlooker bee, and scout. For

Table 5 Comparison with previous Methods

Dataset	ABC	M1	M2	M3	M4	M5	Best Known Results
CAR-S-91-I	5.38	5.36	5.11	5.16	4.97	5.29	**4.5**
CAR-F-92-I	4.61	4.93	4.32	4.16	4.84	4.54	**3.9**
EAR-F-83-I	38.58	37.92	35.56	35.86	36.86	37.02	**29.3**
HEC-S-92-I	11.17	12.25	11.62	11.94	11.85	11.78	**9.2**
KFU-S-93	14.89	15.2	15.18	14.79	14.62	15.8	**13.0**
LSE-F-91	11.74	11.33	11.32	11.15	11.14	12.09	**9.6**
RYE-S-93	9.80	+	+	+	9.65	10.38	**6.8**
STA-F-83-I	157.21	158.19	158.88	159	158.33	160.42	**156.9**
TRE-S-92	8.96	8.92	8.52	8.6	8.48	8.67	**7.88**
UTA-S-92-I	3.65	3.88	3.21	3.59	3.4	3.57	**3.14**
UTE-S-92	26.89	28.01	28	28.3	28.88	28.07	**24.4**
YOR-F-83-I	39.34	41.37	40.71	41.81	40.74	39.8	**34.9**

Table 6 Comparison with previous Methods

Dataset	ABC	M6	M7	M8	M9	M10	Best Known Results
CAR-S-91-I	5.38	4.99	5.4	5.2	4.80	4.6	**4.5**
CAR-F-92-I	4.61	4.29	4.2	4.3	4.10	**3.9**	**3.9**
EAR-F-83-I	38.58	34.42	34.2	36.8	34.92	32.8	**29.3**
HEC-S-92-I	11.17	10.40	10.4	11.1	10.73	10.0	**9.2**
KFU-S-93	14.89	13.5	14.3	14.5	**13.0**	**13.0**	**13.0**
LSE-F-91	11.74	10.48	11.3	11.3	10.01	10.0	**9.6**
RYE-S-93	9.80	8.79	8.8	9.8	9.65	+	**6.8**
STA-F-83-I	157.21	157.04	158.03	157.3	158.26	**156.9**	**156.9**
TRE-S-92	8.96	8.16	8.6	8.6	**7.88**	7.9	**7.88**
UTA-S-92-I	3.65	3.43	3.5	3.5	3.20	3.2	**3.14**
UTE-S-92	26.89	25.09	25.3	26.4	26.11	24.8	**24.4**
YOR-F-83-I	39.34	35.86	36.4	39.4	36.22	**34.9**	**34.9**

UETP, the employee bee and onlooker bee operators are redefined to hold three neighborhood structures: move, swap and Kempe chain. The influence of these neighborhood structures on the behaviour of ABC for UETP is studied and analyzed in this paper.

The experimental design is intentionally made with various convergence cases of different neighborhood structures. The result suggests that the ABC combined with the three neighborhood structures is an effective method for UETP. Comparative evaluation with previous methods is also provided. The results produced by the proposed method are competitive in comparison with the state of the art methods.

The main contribution of this study is to provide the examination timetabling community with an ABC template which combines both efficiency and flexibility for tackling UETP.

In view of the fact that, ABC-based UETP combined with various neighborhood structures has bee proved to be very efficient, future work can improve the ABC-based UETP method by:

– Hybridizing ABC with other gradient descent methods to improve its exploitation.
– Studying the suitable parameters for ABC-based UETP.
– Investigating other efficient neighborhood structures.
– Combining different selection schemes in onlooker bee phase such as, linear rank, exponential rank, tournament selection and many others (Al-Betar et al, 2011a).
– Investigating the performance of the proposed ABC Technique using the third track dataset of the 2nd International Timetabling Competition (ITC-2007) presented by (McCollum et al, 2010)

Acknowledgement
This research was supported by the Universiti Sains Malaysia under IPS-Graduate Fellowship Scheme, 2011 awarded to the first author and CS-USM Postdoctoral Research Fellowship awarded to the third author.

References

Abramson D, Abela J (1991) A parallel genetic algorithm for solving the school timetabling problem. In: in proceedings of the Fifth Australian Computer Science Conference (ACSC-15), Volume 14, pp 1–14
Ahmadi S, Barone R, Cheng P, Cowling P, McCollum B (2003) Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem. Proceedings of Multidisciplinary International Scheduling: Theory and Applications (MISTA 2003), Nottingham, August pp 13–16
Al-Betar M, Khader A (2008) A harmony search algorithm for university course timetabling. Annals of Operations Research DOI: 101007/s10479-010-0769-z, pp 1–29
Al-Betar M, Khader A, Nadi F (2010a) Selection mechanisms in memory consideration for examination timetabling with harmony search. In: Proceedings of the 12th annual conference on Genetic and evolutionary computation, ACM, pp 1203–1210
Al-Betar M, Khader A, Thomas JJ (2010b) A combination of metaheuristic components based on harmony search for the uncapacitated examination timetabling. Practice and Theory of Automated Timetabling (PATAT 2010) Belfast, North Ireland pp 57–80
Al-Betar M, Doush I, Khader A, Awadallah M (2011a) Novel selection schemes for harmony search. Applied Mathematics and Computation 218((10):6095–6117
Al-Betar MA, Khader AT, Zaman M (2011b) University Course Timetabling Using a Hybrid Harmony Search Metaheuristic Algorithm. IEEE Transactions on Systems, Man, and Cybernetics — Part C: Applications and Reviews DOI: 101109/TSMCC20112174356
Asmuni H, Burke E, Garibaldi J, McCollum B, Parkes A (2009) An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables. Computers & Operations Research 36(4):981–1001
Bolaji A, Khader A, Al-Betar M, Awadallah M (2011) An improved artificial bee colony for course timetabling. In: Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011 Sixth International Conference on, IEEE, pp 9–14
Burke E, Newall J (2003) Enhancing timetable solutions with local search methods. In Edmund Burke and Patrick De Causmaecker, editors, Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 2740 Springer-Verlag, Berlin pp 195–206
Burke E, Newall J, Weare R (1996) A memetic algorithm for university exam timetabling. In: First International Conference on the Practice and Theory of Automated Timetabling, Springer, pp 241–250
Burke E, Bykov Y, Newall J, Petrovic S (2004) A time-predefined local search approach to exam timetabling problems. IIE Transactions 36(6):509–528

Burke E, McCollum B, Meisels A, Petrovic S, Qu R (2007) A graph-based hyper-heuristic for educational timetabling problems. European Journal of Operational Research 176(1):177–192

Burke E, Eckersley A, McCollum B, Petrovic S, Qu R (2010) Hybrid variable neighbourhood approaches to university exam timetabling. European Journal of Operational Research 206(1):46–53

Carter M, Laporte G (1996) Recent developments in practical examination timetabling. In Edmund Burke and Patrick De Causmaecker, editors, Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 1153 Springer-Verlag, Berlin pp 3–21

Côté P, Wong T, Sabourin R (2005) A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem. Practice and Theory of Automated Timetabling V pp 294–312

Di Gaspero L, Schaerf A (2001) Tabu search techniques for examination timetabling. In Edmund Burke and Patrick De Causmaecker, editors, Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 2079 Springer-Verlag, Berlin pp 104–117

Dowsland K, Thompson J (2004) Ant colony optimization for the examination scheduling problem. Journal of the Operational Research Society 56(4):426–438

Eley M (2006) Ant algorithms for the exam timetabling problem. In: Proceedings of the 6th international conference on Practice and theory of automated timetabling VI, Springer-Verlag, pp 364–382

Fealko D, Adviser-Mukherjee S (2006) Evaluating particle swarm intelligence techniques for solving university examination timetabling problems. A Dissertation for the degree of Doctor of Philosophy, Graduate School of computer and Information Sciences, Nova Southeastern University.

Garey M, Johnson D (1979) Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences. WH Freeman and Company, San Francisco, Calif

Glover F, Laguna M (1998) Tabu search, vol 1. Kluwer Academic Pub

Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Techn Rep TR06, Erciyes Univ Press, Erciyes

Karaboga D, Akay B (2009a) A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation 214(1):108–132

Karaboga D, Akay B (2009b) A survey: algorithms simulating bee swarm intelligence. Artificial Intelligence Review 31(1):61–85

Kendall G, Hussin N (2005) An investigation of a tabu-search-based hyper-heuristic for examination timetabling. Multidisciplinary Scheduling: Theory and Applications pp 309–328

Kostuch P (2005) The university course timetabling problem with a three-phase approach. In: Burke E, Trick M (eds) Practice and Theory of Automated Timetabling (PATAT) V, vol 3616 pp 109–125

Lü Z, Hao J, Glover F (2011) Neighborhood analysis: a case study on curriculum-based course timetabling. Journal of Heuristics 17(2):97–118

McCollum B, McMullan P, Parkes A, Burke E, Abdullah S (2009) An extended great deluge approach to the examination timetabling problem. Proceedings of the 4th Multidisciplinary International Scheduling: Theory and Applications 2009 (MISTA 2009) pp 424–434

McCollum B, Schaerf A, Paechter B, McMullan P, Lewis R, Parkes A, Gaspero L, Qu R, Burke E (2010) Setting the research agenda in automated timetabling: The second international timetabling competition. INFORMS Journal on Computing 22(1):120–130

Merlot L, Boland N, Hughes B, Stuckey P (2003) A hybrid algorithm for the examination timetabling problem. In Edmund Burke and Patrick De Causmaecker, editors, The Practice and Theory of Automated Timetabling Lecture Notes in Computer Science 2740 Springer-Verlag, Berlin, pp 207–231

Millar H, Kiragu M (1998) Cyclic and non-cyclic scheduling of 12 h shift nurses by network programming. European journal of operational research 104(3):582–592

Paquete L, Fonseca C (2001) A study of examination timetabling with multiobjective evolutionary algorithms. In: 4th Metaheuristics International Conference (MIC 2001), pp 149–154

Pillay N, Banzhaf W (2009) A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. European Journal of Operational Research 197(2):482–491

Qu R, Burke E (2008) Hybridizations within a graph-based hyper-heuristic framework for university timetabling problems. Journal of the Operational Research Society 60(9):1273–1285

Qu R, Burke E, McCollum B (2009a) Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems. European Journal of Operational Research 198(2):392–404

Qu R, Burke E, McCollum B, Merlot L, Lee S (2009b) A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling 12(1):55–89

Teodorović D, DellOrco M (2005) Bee colony optimization–a cooperative learning approach to complex transportation problems. In: Advanced OR and AI Methods in Transportation. Proceedings of the 10th Meeting of the EURO Working Group on Transportation, Poznan, Poland, Citeseer, pp 51–60

Thompson J, Dowsland K (1996) Variants of simulated annealing for the examination timetabling problem. Annals of Operations Research 63(1):105–128

Thompson J, Dowsland K (1998) A robust simulated annealing based examination timetabling system. Computers & Operations Research 25(7-8):637–648

Turabieh H, Abdullah S (2011) An integrated hybrid approach to the examination timetabling problem. Omega 39(6):589–607

White G, Xie B (2001) Examination timetables and tabu search with longer-term memory. In: Burke E and Erbens W, (eds) The Practice and Theory of Automated Timetabling Lecture Notes in Computer Science, Vol 2079 Springer-Verlag, Berlin, pp 85–103

White G, Xie B, Zonjic S (2004) Using tabu search with longer-term memory and relaxation to create examination timetables. European Journal of Operational Research 153(1):80–91