Local Search Procedure for Examination Scheduling

Enikuomehin A.O*, Zyad A., Egbudim D.M*. , Zubair A.F**, Ameen*****
*Lagos State University, Ojo, Nigeria
**Nottingham Trent University, Nottingham, United Kingdom
***University of Ilorin, Ilorin, Nigeria.

Abstract

Multi-Constraint Scheduling problems continue to attract attention as a form of optimization problem thereby allowing the use of real time algorithm to test for best convergence. This paper approaches this challenge using the swamp optimization algorithm for scheduling multi-constraint university time tabling problem. Appropriate decision models were formulated and implemented using the enhanced single face swamp algorithm. Results show an impressing convergence and usable solution, such that the algorithm is capable of optimizing a non-linear and multidimensional problem which usually reaches good solutions efficiently while requiring minimal parameterization. The paper recommends that this approach can further be enhanced for problems with large multidimensional constraint.

Keyword: Multi-constraint Optimization problem,, Time tabling

1.0 INTRODUCTION

Timetable scheduling problems are problems of time-based planning and combinatorial optimization that tend to be solved with a cooperation of search and heuristics, which usually lead to satisfactory but suboptimal solutions. There are many kinds of timetable scheduling problems in the daily life, such as examination, lecture, and transportation timetables. In timetable scheduling problems, events must be slotted to certain times which satisfy several of constraints. It is difficult to design knowledge based and OR-based algorithms to solve such problems. On the other hand, constraint satisfaction techniques have been used to solve hard constraints; however, it is more difficult to handle soft constraints such as preferences. Particle Swamp Optimization (PSO) algorithm is used to schedule exam timetables and the main issues we are going to address are as follows:

- The quality of the examination timetable.
- The times spent in producing the timetable.

University course and exams timetabling is a very well-known constrained problem. The problem becomes more difficult when we have to face a dynamic timetabling problem where new courses and exams can appear during the semester. In this paper, we propose a particle swarm approach to simultaneously solve both problems as well as a local search procedure to handle the dynamism. We show that our approach can find high quality solutions within minutes, where a traditional forward checking requires hours to obtain comparable ones.

At the beginning, to construct a course timetabling each department of the University communicates its courses requirements for the semester, giving the time-slots required. Our problem is, thus, an allocation problem where a classroom must be assigned to each course satisfying the associated facility and capacity constraints. Given the dimensionality of our course timetabling problem we cannot ensure optimality in a reasonable time. Here, we consider optimality as either a planning that produces a minimum or no number of clashes in courses each week or a planning that generates the most equitable use of slot available for

each course or exam as to reduce or eliminate the problem faced whereby two courses which is said to be taken by one undergraduate will fall or clash together on the time table. For the first phase, that corresponds to the course timetabling problem we have designed a particle swarm optimization algorithm (PSO). There is a need to reduce the non-static exam time tabling in the university system This will assist in getting a better accurate feasible solution that can optimize time slotting, and also to get a very good solution efficiently and effectively to fulfill hard and soft constraints.

1.1 PARTICLE SWARM OPTIMIZATION ALGORITHM (PSO)

Particle Swarm Optimization is an algorithm capable of optimizing a non-linear and multidimensional problem which usually reaches good solutions efficiently while requiring minimal parameterization. The algorithm and its concept of "Particle Swarm Optimization"(PSO) were introduced in [1]. However, its origins go further backwards since the basic principle of optimization by swarm is inspired in previous attempts at reproducing observed behaviours of animals in their natural habitat, such as bird flocking or fish schooling, and thus ultimately its origins are nature itself. These roots in natural processes of swarms lead to the categorization of the algorithm as one of Swarm Intelligence and Artificial Life. The basic concept of the algorithm is to create a swarm of particles which move in the space around them (the **problem space**) searching for their goal, the place which best suits their needs given by a **fitness function**. A nature analogy with birds is the following: a bird flock flies in its environment looking for the best place to rest (the best place can be a combination of characteristics like space for all the flock, food access, water access or any other relevant characteristic).

Based on this simple concept there are two main ideas behind its optimization properties:

- A single **particle** (which can be seen as a potential solution to the problem) can determine ‘how good’ its current position is. It benefits not only from its problem space exploration knowledge but also from the knowledge obtained and shared by the other particles.
- A stochastic factor in each particle's velocity makes them move through unknown problem space regions. This property combined with a good initial distribution of the swarm enable an extensive exploration of the problem space and gives a very high chance of finding the best solutions efficiently.

1.2 STATEMENT OF PROBLEM

At the Lagos State University, the time tabling related problems can be addressed as either static or dynamic, which includes situations whereby two courses which is said to be taken by one student will fall or clash together on the time table slot which leads to a dynamic problem. A problem solution must satisfy the following hard constraints:

Static Problem solution set:

- Lectures having students in common cannot take place at the same time-slot, it must take place in a classroom suitable for them in terms of facilities and capacity. Two lectures of the same course scheduled in consecutive time-slots must be assigned to the same classroom also two lectures cannot take place at the same time-slot in the same classroom.

- Each lecture must be assigned to a classroom in its corresponding time-slot.

Dynamic problem solution set:

- There is an available classroom that satisfies the constraints, so it can be assigned to this new activity.
- When a classroom is not available, some re-assignments are required in order to update the original course planning.

Earlier Approaches

Timetable scheduling that previously were done manually, by using the power and the human mind, began to change towards the use of computers. This changes began when [2] started introducing on timetabling, the research is still using the technique de Werra graphs and networks. Although the research could eventually solve the problems of exam timetabling problem and the course timetabling problem in a transaction with a small scale but has a problem in large scale transactions [3]. Due to the continued development of timetabling transactions increased to a large-scale, studies on timetabling has also continued to increase. Timetabling research has been developed starting with the use of human and computer interaction to the methods of Artificial Intelligence and Computational Intelligence [4][5] .noted that several studies have been conducted in the field of timetabling by using the method in the field of operations research, artificial intelligence and computational intelligence.

These methods can be divided into four categories

2.0.1 Sequential Methods

Sequential Methods treating the timetabling problem as a graph problem. The researchers set up events by using domain-specific heuristics and put events in sequence within a valid time slot such that there are no constraints are violated at every time slot. Lai et al. (2008) noted that Werra (1996; 1997) shows how to reduce the course timetabling problem into a graph colouring problem. By applying the colouring technology, the problem can be formulated as a network of arcs and nodes, then the solution can be obtained by finding a set of colour where no two adjacent nodes or edges have the same colour.

2.0.2 Cluster Methods

Problem is divided into several sets of events. Each set is defined so that it meets all hard constraints. Then set the timeslot is set into the concrete to meet the soft constraints. Lai et al. (2008) also noted that some studies using Cluster Methods conducted [6] to formulate the timetabling problem as an assignment problem and resolve the problem by reducing the quadratic assignment problem. Later studies using cluster methods by [7] using the Lagrangian Relaxation method, in this approach, a number of variables is reduced by using grouping. A number of constraints is reduced by replacing it with a relaxed constraints, then a feasible solution obtained by using heuristic methods [8][9][10] [11] Algorithms for propagating resource constraints in AI planning and scheduling: Existing approaches and new results.

2.0.3 Constraint Based Methods

Timetabling problem is modelled as a set of variables (events) that resource (lecturer or rooms) must be assigned in order to meet a number of constraints. noted the use a Prolog-based system which uses backtracking to find a feasible timetable with how to classify constraints and decomposes with respect to message passing and ordering constraints to maximize parallelism and minimize backtracking.

2.0.4 Meta-heuristic Methods

Meta-heuristic Methods implemented in the timetabling problem with starting some initial solution, then use a search method to avoid local optima. Meta-heuristic also includes some heuristic approach inspired from nature and apply process-like nature to get a solution or a population of solutions. using Memetic Algorithm is used to enhance the ability of genetic algorithm by combining the local neighbourhood search, which is used to explore the neighbourhood solutions are obtained from the genetic algorithm to search and navigate local optima for each solution. Several studies using a heuristic approach to nature-like the timetable scheduling is a method of Tabu Search also later used by Hertz (1991) that uses the grouping for the optimization problem and in 1992 he used this method for more complex problems lectures(Lai et al., 2008). In addition, another study using a nature-like Harmony Search algorithm introduced by Geem and Kim (2001). Harmony Search algorithm is an algorithm that imitate the natural phenomenon of musical performance, in which musicians jointly tune the pitches of their instruments to find a euphonious harmony [12] In these studies, Harmony Search for university course timetabling problem and managed to find a near optimal solution.

The particle swarm optimization is a powerful, yet simple population based optimization strategy, particularly well-suited for finding extrema in continuous non-linear functions. The approach is derived in part from the interesting way flocks of birds and swarms in nature search for food.

PROPOSED MATHEMATICAL MODEL

Parameters

1. Set of lectures $L = \{l_1, l_2, \ldots, l_L\}$; where $|L| = L$
2. Set of classrooms $R = \{r_1, r_2, \ldots r_R\}$; where $|R| = R$
3. Set of time-slots $T = \{t_1, t_2, \ldots, t_T\}$; where $|T| = T$
4. p_l: number of participants of lecture i, where $p_l \geq 0$, $\forall l \in L$
5. c_r : capacity of classroom r, where $c_r \geq 0$, $\forall r \in R$
6. t_{ai}: time-slot required by lecture l, where $1 \leq t_{ai} \leq T$, $\forall l \in L$
7. • $f_i = \begin{cases} 1 & \text{If lecture } l \text{ requires facilities} \\ 0 & Otherwise \end{cases}$
8. • $e_r = \begin{cases} 1 & \text{If classroom } r \text{ requires facilities} \\ 0 & Otherwise \end{cases}$
9. • $g_{l,m} = \begin{cases} 1 & \text{If lecture } l \text{ and } m \text{ are consecutive} \\ 0 & Otherwise \end{cases}$

Decision variable

• $X_{i,r,t_{ai}} = \begin{cases} 1 & If\ classroom\ r\ is\ assigned\ to\ lecture\ l\ in\ the\ required\ time-slot\ t_{ai} \\ 0 & Otherwise \end{cases}$

Objectives

For this problem we define two different evaluation functions according to the University requirements:

- *O1*: Minimize the total free seats. It measures the amount of seats unused during a week

$$\sum_{l=1}^{L} \cdot \sum_{r=1}^{R} X_{l,r,t_{ai}} \cdot (c_r - p_l) \qquad (6)$$

- O2: Minimize the number of times the classrooms are used during a week

$$\frac{\sum_{r=1}^{R} | \sum_{l=1}^{L} X_{l,r,t_{ai}} - \bar{X}}{R}$$

Where $\bar{X} = \frac{\sum_{l=1}^{L} \cdot \sum_{r=1}^{R} X_{l,r,t_{ai}} - \bar{X}}{R}$

Constraints

• Each classroom posses the required capacity of the assigned lectures

$$X_{l,r,t_{ai}} \cdot p_l \leq c_r \; ; \qquad \forall l = 1 : L \; , \forall r = 1 : R \quad (1)$$

• Each classroom posses the required facilities of the assigned lectures

$$X_{l,r,t_{ai}} \cdot f_r r \leq e_r \; ; \qquad \forall l = 1 : L, \forall r = 1 : R$$

(2)

• Two consecutive lectures must be assigned to the same classroom

$$X_{l,r,t_{ai}} + X_{m,r,t_{am}} \geq 2 \cdot g_{lm} ; \qquad \forall l, m = 1 : L, l \neq m, \forall r = 1 : R$$

(3)

• Each classroom can be assigned only to one lecture each time-slot

$$\sum_{l=1}^{L} X_{l,r,t_{ai}} \leq 1; \qquad \forall r = 1: R \qquad (4)$$

• Each lecture is assigned to one room in its time-slot

$$\sum_{r=1}^{R} X_{l,r,t_{ai}} = 1; \qquad \forall r = 1:L$$

(5)

3.3 IMPLEMNETING THE PARTICLE SWARM OPTIMIZATION ALGORITHM

PSO algorithm is an adaptive method that can be used to solve optimization problem. Conducting search uses a population of particles. Each particle corresponds to individual in evolutionary algorithm. A flock or swarm of particles is randomly generated initially, each particle's position representing a possible solution point in the problem space. Each particle has an updating position vector X^i and updating velocity vector V^i by moving through the problem space. Kennedy and Eberhart proposed the formula of updating position vector

X^i :

$$X_{k+1}^i = X_k^i + V_{k+1}^i$$

And the formulae for updating velocity vector V^i:

$$V_{k+1}^i = W_k V_k^i + C_1 r_1 (p_k^i - x_k^i) + C_2 r_2 (p_k^g - x_k^i)$$

Where $C_1$and C_2 are positive constant and $r_1$and $r_2$are uniformly distributed random number in [0,1]. The velocity vector V_i is range of [-V_{max} ,V_{max}].

At each iteration step, a function F^i is calculated by position vector

X^ievaluating each particle's quality. The vector P^irepresents ever the best position of each particle and P^g represents the best position obtained so far in the population. Changing velocity this way enables the particle to search around its individual best position P^i, and updating global best position P^g , until searing for the global best position in the limited iteration.

The PSO algorithm consists of just three steps, which are repeated until some stopping condition is met:

1. Evaluate the fitness of each particle.
2. Update individual and global best fitnesses and positions.
3. Update velocity and position of each particle.

The first two steps are fairly trivial. Fitness evaluation is conducted by supplying the candidate solution to the objective function. Individual and global best fitnesses and positions are updated by comparing the newly evaluated fitnesses against the previous individual and global best fitnesses, and replacing the best fitnesses and positions as necessary.The velocity and position update step is responsible for the optimization ability of the PSO algorithm.

The velocity of each particle in the swarm is updated using the above equation..

FLOWCHART REPRESENTATION OF THE ALGORITHM

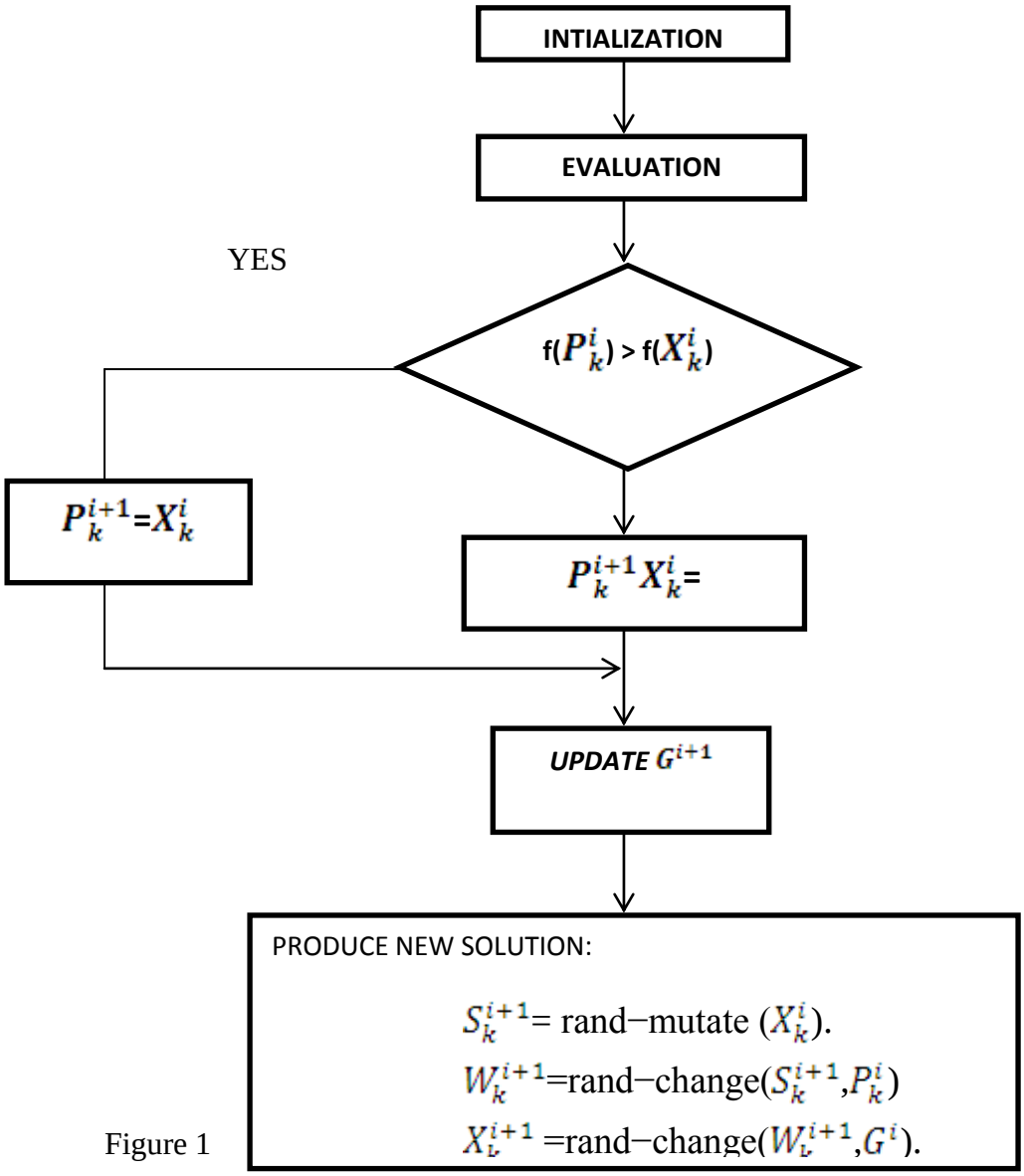

Figure 1

3.4 PSO VARIATIONS

Apart from the canonical PSO algorithm described in previous section, many variations of the PSO algorithm exist. For instance, the inertia weight coefficient was originally not a part of the PSO algorithm[13] but was a later modification that became generally accepted. Additionally, some variations of the PSO do not include a single global best aspect of the algorithm, and instead use multiple global best that are shared by separate subpopulations of the particles.

3.5ANALYSIS OF THE PROPOSED MODEL (PSO FOR TIMETABLE SCHEDULING)

The proposed method attempts every particle would be self-changed two slots of particle and gives the chance to have the global best and the local best solutions. It increases the chance of particles to find a better solution and to leap the local optimal solutions. The self-mutated PSO method can be shown in Figure 1 and described as follows:

First of all, perform with PSO operation similar, the proposed self-mutated PSO method also random generation of an initial population that we want to use to solve the problem. For example, there are M particles to be used; the particles can be represented as:

$$X= \{x_1, x_2, \cdots, x_m\} \quad (7)$$

In beginning, we randomly produce a group of 20 candidate solutions as particles. Each particle is equivalent to a candidate solution of a problem. The performance of 20 candidate solutions is first evaluated and ordered. The best previous position of the kth particle should be put in P_k^iat the ith iteration. The best position amongst all the particles from the first iteration to the ith iteration should be put in G^i. Then new timetables of next generation are produced by following several steps:

- Movement of the particles is processed by the following procedure:
    1. Each particle (X_k^i) must be changed two slots at random by itself.
       S_k^{i+1}= rand−mutate (X_k^i).
    2. 2. Copy a slot for a subject randomly from the local best (P_k^i) to particle (S_k^i).
       W_k^{i+1}= rand−change (S_k^{i+1},P_k^i)
    3. Copy a slot for a subject randomly from the global best (G^i) to W_k^{i+1} at random.
       X_k^{i+1} =rand−change (W_k^{i+1},G^i).

The evolutionary cycle will repeat over and over again until an optimal timetable is found or a certain maximum number of iteration is reached. Therefore, it is easier to escape from local optimum and approach the global or near global optimum in short time.

Algorithm 1.*Feasible solutions construction procedure*

Procedurefeasible_solutions_construction ()

repeat

a random lecture $l_1$is selected

for each feasible classroom available *r* in $l_1$time-slot in increasing size order **do**

if it is possible to assign all l_1's consecutive lectures to r **then**

break;

until all the lectures are assigned to a classroom

- **Dynamic Approach**

For the dynamic problem we propose a local-search approach that works on the solution found by the particle swarm algorithm. For this algorithm we define two main parameters of the search: the maximum number of tries to find a solution and the maximum number of perturbations they can produce in the schedule.

Local-search procedure: This approach performs an incremental hill-climbing procedure. It uses as representation a matrix of size R $\times T_r$, where T_r corresponds to the number of time-slots needed by the requirement:

1. It starts trying to assign the new requirement to an empty classroom. If it is possible, then a solution with

no swaps has been found.

2. It tries to move one lecture assigned to a classroom which satisfies the requirement to an empty classroom.

If it is possible, then a solution with one movement has been found.

3. It tries to swap a lecture assigned to a classroom which satisfies the constraints with another lecture in thetime-slot. If it is possible, this assignment will produce a solution with two perturbations.

4. At each new step, the hill-climbing method increments the maximum number of swaps allowed in the static planning. This incremental feature of the algorithm is focused on reducing the number of updates required to satisfy the constraints. The search process ends when a maximum number of solutions is reached or it is not possible to find more solutions with the maximum number of swaps allowed. The choice of programming language used for this system is based on the overall goal of this study.

CONCLUSION AND RECOMMENDATION

The Class timetable scheduling problem consists of allocating a number of courses or classes to a limited set of resources such as rooms, time slots, set of lecturers and group of students in such a way to satisfy predefined hard and soft constraints within efficient running time.In this project we have proposed a particle swarm approach, usually used to solve continuous problems, to the dynamic course and exam timetabling problem. Our approach can efficiently handle the creation of new courses or exams after the initial static start-up planning. Our particle swarm approach has shown to solve very quickly the initial static problem, optimizing the use of small classrooms, to let the biggest ones available for exams or faculty course. Therefore, finding a classroom with a big capacity for an exam becomes easier for the local search procedure during the dynamic phase of the timetabling. We have shown that our approach can find very good quality solutions within minutes, where a traditional forward checking method needs hours to obtain comparable quality solutions.

The timetabling problem (TP) is a combinatorial problem that can be viewed as an optimization task. It consists of allocating schedules to several workers or students, which

also require some resources. In order to get a feasible timetable, a set of hard constraints must be satisfied (most of them technical constraints); moreover, a good timetable must satisfy some soft constraints (frequently, comfort-related constraints), and if all soft constraints are met, we can consider the solution as optimal. This NP-hard problem presents several variants, such as the school, employee, and university timetabling problems. A possible future study could investigate what the optimal collision cut-off should be, before two classes can be offered in the same time slot. Such an investigation could be helpful to inform administrators' decision during the process of setting enrolment numbers for classes. In addition, the system could also be modified to tell administrators what students have been affected by a collision between two classes. These way departments could help these students to make different plans.

REFERENCES

1. Eberhart, R. C., & Kennedy, J. (1995, October). A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science* (Vol. 1, pp. 39-43).
2. de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, *19*(2), 151-162.
3. Lin, T. L., Horng, S. J., Kao, T. W., Chen, Y. H., Run, R. S., Chen, R. J., ... & Kuo, I. H. (2010). An efficient job-shop scheduling algorithm based on particle swarm optimization. *Expert Systems with Applications*, *37*(3), 2629-2636.
4. Bhaduri, A. (2009, March). A clonal selection based shuffled frog leaping algorithm. In *Advance Computing Conference, 2009. IACC 2009. IEEE International* (pp. 125-130)
5. Hammer, P. L., Mahadev, N. V. R., & De Werra, D. (1985). The struction of a graph: Application toCN-free graphs. *Combinatorica*, *5*(2), 141-147
6. Ferland, J. A., & Roy, S. (1985). Timetabling problem for university as assignment of activities to resources. *Computers & operations research*,*12*(2), 207-218.
7. Tripathy, S. C., Kalantar, M., & Balasubramanian, R. (1992). Stability simulation and parameter optimization of a hybrid wind-diesel power generation system. *International journal of energy research*, *16*(1), 31-42.,
8. Herroelen, W., & Leus, R. (2005). Project scheduling under uncertainty: Survey and research potentials. *European journal of operational research*,*165*(2), 289-306.Zhang, C. Y., Li, P., Rao, Y., & Guan, Z. (2008)
9. A very fast TS/SA algorithm for the job shop scheduling problem. *Computers & Operations Research*, *35*(1), 282-294
10. Zhuravlev, S., Blagodurov, S., & Fedorova, A. (2010, March). Addressing shared resource contention in multicore processors via scheduling. In *ACM SIGARCH Computer Architecture News* (Vol. 38, No. 1, pp. 129-142). ACM.

11. Baptiste, P., Le Pape, C., & Nuijten, W. (2012). *Constraint-based scheduling: applying constraint programming to scheduling problems* (Vol. 39). Springer Science & Business Media.. Laborie, P. (2014, May).
12. Al-Betar, M. A., Khader, A. T., & Thomas, J. J. (2010, August). A combination of metaheuristic components based on harmony search for the uncapacitated examination timetabling. In *8th International conference on the practice and theory of automated timetabling (PATAT 2010), Belfast, Northern Ireland* (pp. 57-80).
13. Karaboga, Dervis, and Bahriye Basturk. "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm."*Journal of global optimization* 39.3 (2007): 459-471