Minimum Student exam conflict in Exam Time Tabling using Symbiotic Organisms Search

Rina D. Zarro[1], Mardin A. Anwer[2], Zana Farhad[3]

[1, 2 &3]*Software and Informatics Engineering Salahaddin University, Erbil, Kurdistan, Iraq*
[1]*rina.zarro@yahoo.com,* [2]*mardinsherwany77@gmail.com,*[3]*zana.softeng@gmail.com*

doi: 10.23918/iec2017.11

ABSTRACT

Time tabling problem has been widely investigated by many researches in the past two decades. However, these problems and their difficulties may vary according to its size, structure or constraints. In this paper, the exam time table for software engineering department is solved using Symbiotic Organisms Search (SOS) algorithm. The algorithm tries to minimize the number of presented conflicts in student exams for those who may possibly have more than one exam in the same day. The problem arises from the current regulations where student can pass to the next year with one failed subject or those transferred students from other universities with more than one subject required to be covered from earlier stages. This paper tries to replace the current manual process with an automatic table generating system. The solution is divided into two phases. First stage is to formulate the objective function and its constraints. In the second phase, Symbiotic Organisms search is incorporated to find the best exam time table with minimum conflict. The SOS algorithm is modified to work on fixed integer permutation representation by introducing a novel way of finding the next new solutions from the existing solutions. For testing the proposed method, software engineering second attempt exam case study is used to test the proposed solution. The algorithm found to perform faster than genetic algorithm and particle swarm algorithms.

Keywords: Exam Time Table, Symbiotic Organism Search Algorithm, Optimization Technique.

1. INTRODUCTION

Time tabling represents one of the most well-known problem in computer science. The importance of this problem comes from the fact that time tables are used or incorporated in every aspect of daily work. It varies from time scheduling of students or lectures in universities and schools, to employees or workers working in companies and factories. It is used to facilitate the job or task time frames according to available resources and persons [1]. This make it very crucial step in arranging these time frames to increase productivity of the current tasks, as well as, manage available time efficiently.

In general, for universities, time table is used to facilitate the use of available resources according to certain situation. This may include, scheduling weekly lectures according to the available halls. In addition, there are more complicated problems where the students number represent a factor for the selected hall [2].

Exam time tabling represents one of the difficult tasks for exam committees, especially when constraint factors are introduced. The problem of conflicts in these tables, represents an extra burden introduced to the students. The conflicts are a

common situation when student may come under the pressure to take more than one exam in the same day. This is a very common situation for course based studies where the student may have one or more subjects not satisfied from the previous year. Therefore, it is very important to create a table which reduces these conflicts as much as possible. This paper presents an exam time tabling system based on Symbiotic Organism Search algorithm where students having exams conflicts are minimized.

2. LITERATURE REIEVEW

Timetable problem is a type of system schedule problem. Scheduling timetable for educational institutions contains time frames including teachers, students, and classroom. An optimal schedule would be where no students, teacher or classroom is used more than once at any given time frame. The problem of the timetable can be described as finding a schedule for classroom, teachers, and student's combination where at the same time frame will have a minimal number of conflicts or overlapping. Therefore, automatic timetables are an essential requirement to manage these events. A good survey on automatic timetabling can be found in [3], and a good literature survey on timetabling techniques is given by Abdullah [2] in his thesis. He discussed various techniques and algorithms used in timetabling generations.

Time tabling problems, generally, are NP hard where a solution in polynomial time is not possible. Many techniques where investigated for solving these problems which may depend on exact solution. However, the most significant techniques are those of metahuristic optimization or naturally inspired algorithms such as genetic algorithm [5] or particle swarm [6]. On exam time table there where similar approaches to those used in generating weekly lecture scheduling. Mahato and Kumar [7] suggested the use of genetic algorithm to solve a multi-objective exam time table generation. The objectives included student number, classes available and minimal exam days which are to be satisfied. Moriera [8] used the same algorithm for constructing an exam time table with a change in the representation strategy. The proposed representation consists of a matrix with each column contains a value withdrawn from the period set, indicating the time at which the exam was allocated. Ahmed and Shaari [5] proposed an exam time tabling based on particle swarm optimization. The proposed method utilizes the particle swarm optimization to be able to solve this type of problems which uses fixed integer. The proposed strategy was then applied to two universities to test its validity.

In this paper, an exam time tabling is presented which is based on symbiotic organism search optimization. First, the paper defines a new strategy for SOS algorithm to be able to work on fixed integer problem. Then, an objective function plus set of constraints is formulated as an objective function.

3. EXAM TIME TABLE PROBLEM

The main problem in exam time tabling is to avoid as much conflict in exam times, so that, students with required subjects can have at most one exam per day. However, this is not the only conflict in the system, there are other factors which makes the task of finding an optimal table very hard.

To formulate our problem, consider a N level studies having a set of M subjects and D days for completing all the exams. For this case, the matrix for the organism must consists of N X D and this must be filled by the M subject. The subject set M is

defined as a subset of number of subjects per level (study stage) and defined as Mi, such that:

$$M = \{M_1, M_2, M_3, \cdots, M_N\}, \quad (1)$$

With,

$$M_i = 1, 2, 3, \cdots, S_i, \quad (2)$$

Where, Si represents the number of subjects per level i . Hence, the candidate solution can be represented as a fixed N X D matrix as shown in Fig. 1. The figure illustrates the candidate solution (organism) with 4 levels of study, 5 days of exam and 3, 4, 4 and 5 subjects per level. The representation is a fixed integer representation with the empty spaces can be filled with zero to indicate that no exam at this day for this level.

1	0	2	0	3	2	1	0	4	3	1	4	3	2	0	1	5	2	3	4
0	0	2	1	3	2	4	0	1	3	4	0	3	2	1	3	5	1	4	2
1	2	0	0	3	1	0	2	4	3	0	2	3	4	1	1	5	2	3	4

FIGURE 1. An example of organism solution representation

The above illustration is used to define the solution space where the system tries to identify the best solution. The best solution is obtained when the number of conflicts in exams per student is minimal. The objective function can be formulated as:

$$obj = \min_{st \in \mathbb{Z}}(st) \quad (1)$$

Where st represents the total number of students having exam conflicts in all days and it is found as the sum product of all exam conflicts per day (std) as follows,

$$st = \sum_{d=1}^{D} st_d \quad (2)$$

The Eq. (1) above, does not consider other factors related to the student examination. For example, a student may have more than two exams per day. Such case will not be reflected in the final solution, meaning, if the final solution has 2 conflicts only, there is a chance these two conflicts may exist for the same student. Therefore, to prevent such situation, a constraint on the number of exam per student is presented to avoid such anomaly. It is given by,

$$e_{d,st} \leq 2 \quad (3)$$

Where $e_{d,st}$ is the number of exams per day (d) per student (st).

4. SYMBIOTIC ORGANISM SEARCH ALGORITHM

Symbiotic Organism Search, known as SOS, is a bio-inspired optimization algorithm based on species behavior towards each other in the echo system. simulates the symbiotic interactions within a paired organism relationship that are used to search

for the fittest organism [8]. The algorithm goes through three phases which are Mutualism, Commensalism and Parasitism.

4.1 MUTUALISM PHASE

An example of mutualism, which benefits both organism participants, is the relationship between bees and flowers. Bees fly amongst flowers, gathering nectar to turn into honey – an activity that benefits bees. This activity also benefits flowers because bees distribute pollen in the process, which facilitates pollination. In SOS, Xi is an organism matched to the ith member of the ecosystem. Another organism Xj is then selected randomly from the ecosystem to interact with Xi. Both organisms engage in a mutualistic relationship with the goal of increasing mutual survival advantage in the ecosystem. New candidate solutions for Xi and Xj are calculated based on the mutualistic symbiosis between organism Xi and Xj, which is modeled in Eqs. (4) and (5).

$$X_{inew} = X_i + rand(0,1) * (X_{best} - Mutual\_Vector * BF_1) \quad (4)$$

$$X_{jnew} = X_j + rand(0,1) * (X_{best} - Mutual\_Vector * BF_2) \quad (5)$$

$$Mutual\_Vector = \frac{X_i + X_j}{2} \quad (6)$$

Here, benefit factors (BF1 and BF2) are determined randomly as either 1 or 2. The result of the two vectors are then compared to their corresponding selected vectors and may replace them if they yield a better solution.

4.2 COMMENSALISM PHASE

An example of commensalism is the relationship between remora fish and sharks. The remora attaches itself to the shark and eats food leftovers, thus receiving a benefit. The shark is unaffected by remora fish activities and receives minimal, if any, benefit from the relationship.

Similar to the mutualism phase, an organism, Xj, is selected randomly from the ecosystem to interact with Xi. In this circumstance, organism Xi attempts to benefit from the interaction. However, organism Xj itself neither benefits nor suffers from the relationship. The new candidate solution of Xi is calculated according to the commensal symbiosis between organism Xi and Xj, which is modeled in Eq. (6). Following the rules, organism Xi is updated only if its new fitness is better than its pre-interaction fitness.

$$X_{inew} = X_i + rand(0,1) * (X_{best} - X_j) \quad (6)$$

The part of equation, $(X_{best} - X_j)$, is reflecting as the beneficial advantage provided by X_j to help X_i increasing its survival advantage in ecosystem to the highest degree in current organism (represented by X_{best}).

4.3 PARASITISM PHASE

An example of parasitism is the plasmodium parasite, which uses its relationship with the anopheles' mosquito to pass between human hosts. While the parasite thrives and reproduces inside the human body, its human host suffers malaria and may die as a result.

In SOS, organism Xi is given a role similar to the anopheles mosquito through the creation of an artificial parasite called ''*Parasite_Vector*''. *Parasite_Vector* is created in the search space by duplicating organism *Xi*, then modifying the randomly selected dimensions using a random number. Organism Xj is selected randomly from the ecosystem and serves as a host to the parasite vector. Parasite_Vector tries to replace Xj in the ecosystem. Both organisms are then evaluated to measure their fitness. If Parasite_Vector has a better fitness value, it will kill organism Xj and assume its position in the ecosystem. If the fitness value of Xj is better, Xj will have immunity from the parasite and the Parasite_Vector will no longer be able to live in that ecosystem.

4.4 FIXED INTEGER (PERMUTATION) SYMBIOTIC ORGANISM SEARCH

The above formulation of the three phases works very efficiently on real number problems. However, for the problem in hand, there should be some modification to the equations above to restrict the solutions to real number only, as well as, fix the coded exam numbers in the table. In order to demonstrate this change in equations, the above equations will be used to identify the change in the solution position rather than solution values in the matrix. Hence, equations (4) and (5) are expressed,

$$X_{i,new} = \begin{cases} X_i & segm(x) \geq 0.5 \\ X_{ch} & segm(x) < 0.5 \end{cases} \quad (7)$$

Where,

$$X_{ch} = \begin{cases} X_j & 0.25 \leq segm(x) < 0.5 \\ X_{best} & segm(x) < 0.25 \end{cases} \quad (8)$$

With

$$segm(x) = \frac{1}{1+e^{-x}} \quad (9)$$

$$x = rand(0,1) * (X_{best} - Mutual\_Vector * BF_1) \quad (10)$$

$$x = rand(0,1) * (X_{best} - Mutual\_Vector * BF_2) \quad (11)$$

And x here corresponds to either Eq. (4), (5) This equation changes the position of the solutions on two level, one the overall exam date structure solution and second on single level exam solution. This is illustrated on Fig. 2 and 3 below and the final equation which corresponds to value x above is found as

$$x = rand(0,1) * (X_{best} - X_j) \quad (12)$$

$$X_{i,new} = \begin{cases} X_i & segm(x) = 0 \\ X_{ch} & segm(x) \neq 0 \end{cases} \quad (13)$$

Where,

$$X_{ch} = \begin{cases} X_j & 0.5 \leq segm(x) \\ X_{best} & segm(x) < 0.5 \end{cases} \quad (14)$$

The above set of equations are based on integer based Genetic algorithm strategy for integer numbers.

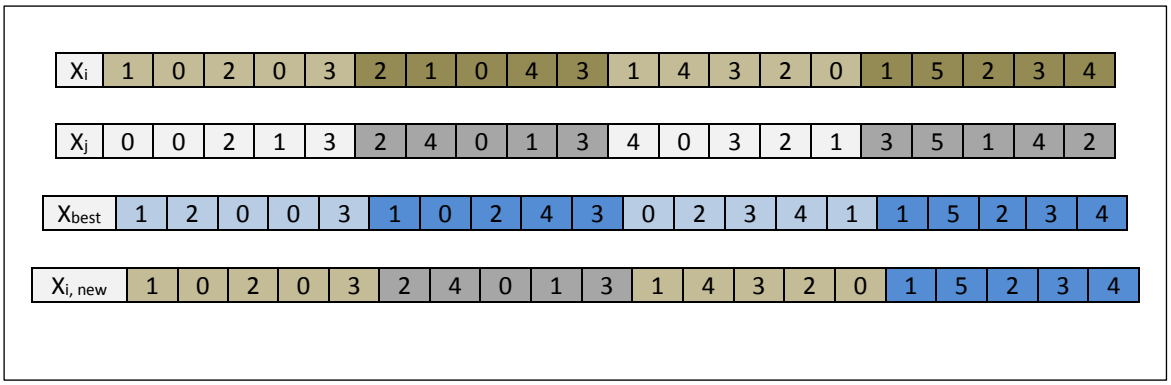

FIGURE 2. An example of new solution representation based on Eq. (9)

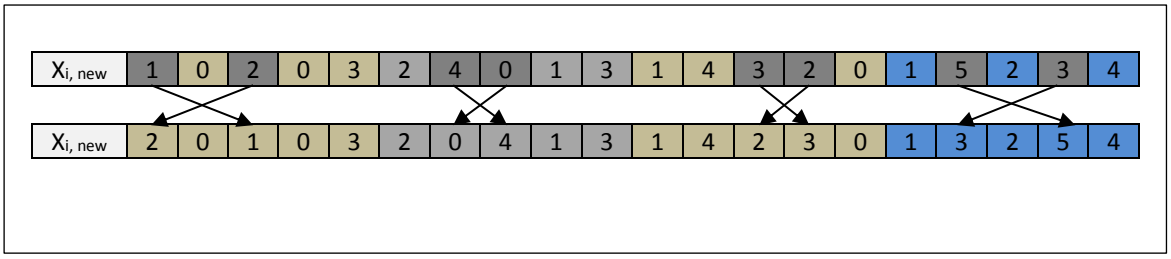

FIGURE 3. An example of new solution representation with random exchange

5. RESULT AND DISCUSSION

For testing the proposed method, a case study from Salahaddin university/college of engineering, software engineering department exam sample is considered. The data sets consist of multiple tables of students with complicated conflicts which has an exact solution of zero conflicts. Table 1 shows the student setup with these conflicts shown in brackets.

As shown from Table 1, the total number of exams for conflicted student at second level are 29. The number in the bracket is the number of subjects which are 8. There are 10 students that must take exams in 4 subjects with total exams between these students at second level are 11. So, there is 8 subjects in the second year may cause a conflict with 4 subjects in the first level. Second row shows that, there are 7 subjects in third level has conflicts with 5 subject in second and 2 subjects in first level. Finally, for final level there are 5 subject having 3 conflicts with 3 subjects of second level, 1 subject in second level and 1 subject in first level.

TABLE 1.
Software Engineering exam time table case study

Level	No of students	Level 1	Level2	Level3	Level 4
Level2	10	11 (4)	25(8)		
Level3	9	4(2)	10(5)	24(7)	
Level4	10	1(1)	2(1)	9(3)	13(5)

The exact solution represents the best solution with zero conflicts which was obtained through the algorithm. Table 2 compares the obtained solution from the proposed algorithm with GA and PSO according to number of iteration. The setup of the algorithm was based on 15 solutions per pool with GA crossover and mutation are set to 0.7 and 0.15 respectively [9]. The PSO setup is according to the approach used in [5]. The SOS method achieved the exact solution with slightly less number of iterations. This fast convergence was obtained because the mechanism of SOS algorithm works in three phases where the solution can be modified to obtain a better solution. This mechanism is compared to GA which has two phases (crossover and mutation) and PSO which has one (particle position updating). Moreover, the nature of the problem, where permutation representation is presented, make GA more suitable that PSO as the crossover operation can restrict the solution with the content of the chromosome, while SOS can maintain the existence of the permutation representation with the suggested algorithm which double the effect of GA crossover through mutualism and commensalism phases.

TABLE 2.
Compares between SOS, GA and PSO for exam conflicts

Algorithm	**Iteration per conflicts**				
	10	15	20	25	30
GA	5	1	0	0	0
PSO	10	4	2	0	0
SOS	2	0	0	0	0

6. CONCLUSION

This paper presents a Symbiotic Search Algorithm used to solve a permutation represented exam time table. The proposed algorithm uses the original set of formulas of mutualism, commensalism and parasitism phases to exchange solutions between existing solutions in the search space. To test the proposed algorithm, a case study with an existing exam conflicts was used. The algorithm was compared against GA and PSO existing setups. The proposed algorithm performed a faster convergence compared to GA and PSO. In the end, the use of such kind of algorithm can prove to be crucial in helping exam committees reduce the time consumed by the current manual procedures used to reduce the number of conflicts.

REFERENCES

[1] H. Babaei, J. Karimpour, A. Hadidi, A survey of approaches for university course timetabling problem, Computers & Industrial Engineering, Volume 86, August 2015, pp 43-59.

[2] S. Abdullah, Heuristic Approaches For University Timetabling Problems, PhD thesis, University of Nottingham, 2006.

[3] Schaerf, A., A Survey of Automated Timetabling, Artificial Intelligence Review, Springer, 1999, pp 87-127.

[4] A. Colorni, M. Dorigo, V. Maniezzo, Genetic Algorithms: A New Approach to the Timetable Problem, Combinatorial Optimization, the series NATO ASI Series, Springer, Volume 82, 1990, pp 235-239.

[5] A. Ahmed, and F. Shaari, "Solving University/Polytechnics Exam Timetable Problem using Particle Swarm Optimization", Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication, 2016, No. 46, pp 1-4.

[6] M. Mahto , L. Kumar, Exam Time Table Scheduling using Genetic Algorithm, International Journal of Enhanced Research in Management & Computer Applications, Vol. 4, Issue 8, 2015, pp 31-35.

[7] J. J. Moreira, A System of Automatic Construction of Exam Timetable Using Genetic Algorithms, Tékhne, No.9, 2008, pp.319-336.

[8] M. Y. Cheng, D. Prayogo, Symbiotic Organisms Search: A new metaheuristic optimization algorithm, Computers & Structures, Elsevier, Volume 139, 2014, pp. 98–112.

[9] T. Wong, P. Côté, P. Gely, Final Exam Timetabling: A Practical Approach, Proceedings of the 2002 IEEE Canadian Conference on Electrical & Computer Engineering, 2002, pp 726-731.