International Conference on Information and Communication Technology and Its Applications (ICTA 2016)
Federal University of Technology, Minna, Nigeria
November 28 – 30, 2016

Concurrent Student Seminar Scheduling Using Genetic Algorithm

Ojoajogu Ajanya and Hamza O. Salami
Department of Computer Science, Federal University of Technology, Minna, Nigeria
ogajanya@yahoo.com, ho.salami@futminna.edu.ng

***Abstract*—Scheduling involves assigning variables to specific domains according to resources and wishes. Usually, many students are ready to present seminars at the same time. In order to maximize time, student seminars can hold concurrently in multiple venues. Manual scheduling of student seminars is tedious since it must be painstakingly planned to minimize clashing of panelists, who must be present when students present. This paper presents a genetic algorithm (GA) for automatically scheduling parallel student seminars to minimize clashes and inconvenience to panelists. Experimental results show that GA-based seminar scheduling is promising.**

Keywords-genetic algorithm; seminar scheduling; parallel seminars

I. Introduction

Order is a critical factor that guarantees proper organization of events; where there is disorderliness, there is bound to be confusion and ultimately, loss of productivity. In the academic community seminars are examples of events which must be properly ordered. Student seminars are oral presentations which are part of the requirements for students' academic programmes. Seminars are usually coordinated by a designated member of staff who takes into consideration the schedules of the panelists who should be physically present at the seminars. A student's panelists comprise his/her project supervisors, co-supervisors and examiners/assessors.

In cases where many students are ready to present seminars around the same period, two approaches can be adopted. On one hand, students can present seminars one-at-a-time. Even though this approach eliminates the clashing of panelists, it creates the problem of having all panelists available throughout the seminar presentations. On the other hand, seminars can be scheduled to run concurrently. The latter approach minimizes the overall seminar presentation time, but requires that students are carefully scheduled such that the panelists who should be on ground during the presentations are not required to be in more that one seminar location at any given time.

A timetable is "a table of events arranged according to the time when they take place" [1]. Timetables are very important for the university administration; they give students and teachers a schedule indicating the right time and the right place to be, the availability of the rooms, the time to be spent by the teachers for the period, the availability of the teachers and the students. Timetable problem is a real-life combinational problem concerned with scheduling of a certain number of events within a specific time frame. Therefore, seminar scheduling is an example of the timetabling problem.

Seminar scheduling entails arranging time slots for seminar sessions while considering some constraints like number of participants, capacity of the venue, number of presentations by facilitators, and so on. Seminar scheduling problem is a problem related to assigning variables to specific domains according to resources and wishes. As population of participants gets larger and larger while resources like venues and supervisors remain constant or diminish, manually performing scheduling under these constraints becomes an incredibly hard problem. Also, each defined constraint restricts the area and the problem becomes harder, thus it takes a long time to get a solution by human-based techniques.

The main contribution of this paper is in the use of Genetic Algorithm (GA) to plan concurrent student seminars in order to avoid/minimize conflicts in the seminar schedules. GA is a search and optimization technique based on natural genetics and natural selection [2]. The rest of this paper is organized as follows: A review of related works is presented in Section II. The seminar scheduling problem is explained in Section III. In Section IV, a detailed description of a GA for automatic seminar scheduling is presented. Experimental results appear in Section V, while the paper is concluded in Section VI.

II. Related Works

Although little has been written about seminar scheduling, it is closely related to other problems that have received significant research attention like the timetabling problem and examination scheduling [3].

A genetic algorithm-based intelligent system was proposed in [4] for course scheduling in higher education. The GA helped to optimize the preparation of class schedules. The test result obtained 99 conflicting classes from 635 existing classes. The average non-conflict scheduling accuracy of the system is 84.4%.

In [5], GA was used to develop an optimization-based prototype for nurse assignment that makes daily decisions on assigning nurses to patients. A prototype for assigning nurses to patients with the purpose of minimizing excess workload on the nurses was developed.

According to [6], scheduling classes is a time consuming job for administrators. Many constraints are defined for classrooms, faculty members, and courses, whereby, a course may require a classroom with some minimum number of seats and with some audiovisual equipment. Also, a faculty member may prefer not to teach two or more courses in a row, or may prefer teaching before certain time. In view of these, the researcher employed GA for finding a "good" schedule that results in an efficient use of each classroom, in relation to time, space, and constraints.

Analytical Hierarchy Process (AHP) and GA have been combined to create a time table schedule that matches most of the teachers' preferences [7]. The approach consists of the integration of a satisfaction function to the genetic algorithm. The parameters of the satisfaction function are the teachers' loads and a set of scores calculated using the analytical hierarchy process. The key point of AHP in calculating the scores is the pairwise comparison of a set of teachers' criteria. The new approach was consequently a combination of AHP and GA, and it gives rise to a new methodology to solve the time table problem that is called AHP/GA.

Researchers in [8] presented an experimental investigation into solving the Assignment model using GA and Simulated Annealing. Various parameters affecting the algorithms are studied and their influence on convergence to the final optimum solution was shown. While solving this problem through GA, a unique encoding scheme was used together with Partially Matched Crossover (PMX).

In their work, [9] combined the attributes of fuzzy logic and Genetic Algorithm to create a Fuzzy Genetic Heuristic (FGH) Algorithm which they used to solve the university course timetable problem. They posit that fuzzy logic models are easy to comprehend because they use linguistic terms and structured rules but do not come with search algorithms. Unlike GA, fuzzy models adopt techniques from other areas such as statistics and linear system identification. Thus they harnessed GA's search ability by merging both paradigms and created FGH algorithm, which describes Fuzzy Set model using GA search attribute. FGH uses an indirect representation featuring event allocation priorities and invokes a timetable builder routine for constructing complete timetable.

The work reported in [3] focused on the preference-based conference scheduling (PBCS) problem, in which preferences of conference attendees are taken into consideration. PBCS was formulated as an integer programming problem. Since the formulation is NP complete, simulated annealing was used to obtain good solutions to a PBCS for a real life conference that involved scheduling 213 sessions over 10 time-blocks for 520 attendees who had preferences.

III. SEMINAR SCHEDULING PROBLEM

The seminar scheduling problem (SSP) is concerned with arranging concurrent student seminars in such a way that no panelists are required to be in more than one seminar at the same time, and the movement of panelists to different venues is minimized. A student's panelists are his/her research supervisors and examiners/assessors. Examiners must be present during a student's seminar because they ask questions and based on the student's response, determine if the student can proceed to the next stage of study or not. It is important for supervisors to be present during their supervisees' seminars because they might be required to clarify issues related to their supervisees' research.

Let $S = \{s_1, s_2, s_3 \ldots s_{NS}\}$ be a set of *NS* students and let $L = \{ l_1, l_2, l_3 \ldots l_{NL}\}$ be a set of *NL* lecturers. The panelist matrix *P* is a *NS* x *NL* matrix. The entry P_{ij} in the matrix is 1 if the j^{th} lecturer l_j is a panelist for the i^{th} student s_i. Otherwise, the entry is zero. A sample of the panelist matrix is shown in Fig. 1. From the first row of the matrix, l_2 and l_3 are panelists for s_1. Furthermore, from column one of the matrix, l_1 is a panelist for only s_3. Let the number of periods (i.e., time slots) and venues be denoted by *NP* and *NV*, respectively. Assume that the *NS* students are to be scheduled concurrently in *NV* venues within *NP* periods. Let a session be defined as a combination of a venue and a period. Fig. 2 shows how sessions are numbered from periods and venues. Note that the number of sessions, $NSS = NV * NP$.

SSP involves assigning students to different sessions such that certain hard and soft constraints are satisfied. Hard constraints are constraints that need to be satisfied for a solution to be feasible. Soft constraints are constraints that must not necessarily be fulfilled, that is, they are allowed to be violated.

The hard constraints for the SSP are:

- Each student shall present exactly one seminar.
- Only one student can present a seminar in a given session.
- All the panelists for a student must be present during the student's seminar.
- No panelist can be in more than one venue at a time.

The only soft constraint considered seeks to minimize inconvenience resulting from lecturers moving from one venue to another at different time periods:

Each panelist should remain in one venue throughout the entire seminar presentations.

Fig. 3 shows two ways of scheduling student seminars within two venues and two periods. Panelists (obtained from Fig. 1) are shown alongside each student. Panelists causing hard constraint violations are shown in red, while those causing soft constraint violations are shown in green. There are two hard constraint violations from the seminar schedule of Fig. 3(a) since l2 and l5 are required to be in multiple venues at the same period; l2 is required to be in venue 1 and venue 2 in period 1, whereas l5 is required to be in both venues in period 2. Furthermore, there are two soft constraint violations from Fig. 3(a); l3 moves from venue 1 to venue 2, while l4 moves from venue 2 to venue 1. There are no hard constraint violations on Fig. 3(b). The only soft constraint violations are as a result of l2 and l5 changing venues.

IV. GENETIC ALGORITHM

Genetic Algorithm (GA) is a search algorithm which mimics the survival of the fittest strategy that occurs in the natural world [7]. It is an iterative search technique developed based on evolutionary genetics which seeks the best of a set of solutions [10]. GA follows five phases namely: generating an initial population, evaluating the fitness of the chromosome, selection, crossover and mutation. The process of selection, crossover and mutation are iterated until the optimal solution is obtained [11]. This algorithm explores the search space and makes use of the generated knowledge to find a better population. A flowchart of the standard GA is presented in Fig. 4. The performance of GA is usually evaluated in terms of convergence rate and the number of generations to reach the optimal solution.

Lecturers

Students	l_1	l_2	l_3	l_4	l_5
s_1	0	1	1	0	0
s_2	0	1	0	1	0
s_3	1	0	1	0	1
s_4	0	0	0	1	1

Figure 1. A Panelist Matrix with Four Students and Five Lecturers

	Period 1	**Period 2**
Venue 1	Session 1	Session 4
Venue 2	Session 2	Session 5
Venue 3	Session 3	Session 6

(a)

	Period 1	**Period 2**	**Period 3**
Venue 1	Session 1	Session 3	Session 5
Venue 2	Session 2	Session 4	Session 4

(b)

Figure 2. Sessions (a) three venues and two periods (b) two venues and three periods

	Period 1	**Period 2**
Venue 1	s_1 (l_2, l_3)	s_4 (l_4, l_5)
Venue 2	s_2 (l_2, l_4)	s_3 (l_1, l_3, l_5)

(a)

	Period 1	**Period 2**
Venue 1	s_1 (l_2, l_3)	s_3 (l_1, l_3, l_5)
Venue 2	s_4 (l_4, l_5)	s_2 (l_2, l_4)

(b)

Figure 3. Constraint violations (a) Hard and soft constraints violations (b) Soft constraints violations

A notable characteristic of GA is that it is a parallel population-based search with stochastic selection, crossover and mutation [12]. Secondly, GA works on the chromosome which is an encoded version of potential solution parameters rather than optimizing the parameters themselves [12]. Thirdly, GA uses fitness values obtained from objective functions without other artificial over engineered black box mathematics [7]. The user typically chooses the best structure of the last population as the final solution. The algorithm is complete when one of the following occurs: a specified tolerance threshold is achieved; a specified number of generations has passed; a specified amount of computational time has passed; or the solution fitness has plateaued [13].

The inputs to the GA for solving the SSP are the number of venues NV, the number of periods NP, and the panelist matrix P. The output of the GA is the seminar schedule similar to those shown in Fig. 3.

A. *Chromosome Representation*

Each chromosome is a row vector having NS genes. The value in each gene determines the session in which a student presents his/her seminar. For example, the value of the first gene states the session during which s1 presents, the value of the second gene specifies the session during which s2 presents, and so on. Because each student must present in a unique session, the values in the chromosome are distinct integers ranging from 1 to NSS. It is noteworthy that this chromosome representation ensures that the first two hard constraints are always satisfied. Fig. 5 shows how the seminar schedules shown in Fig. 3 are encoded as chromosomes. The chromosome in Fig. 5(a) indicates that s1, s2, s3 and s4 present in the first, second, fourth and third sessions respectively, while Fig. 5(b) shows that s1, s2, s3 and s4 present in the first, fourth, third and second sessions, respectively.

B. *Fitness Function*

As mentioned in Section IV(A), the first two hard constraints have been taken care of from the chromosome representation. The third hard constraint is assumed to hold all the time. Thus once a student appears in a session, all the student's panelists are assumed to be present. The fitness function therefore handles the last hard constraint and the only soft constraint. The last hard constraint states that panelists can only be in one place at a time. A clash occurs when a panelist is required to be in different venues at the same period. Similarly, the soft constraint seeks to eliminate the inconvenience of panelists moving from one venue to the other. The fitness function F for the GA can be described mathematically using (1).

$$F(C)=\sum_{i=1}^{NL}(100*Clashes(C,i)+Movements(C,i)) \qquad (1)$$

Where, NL is the number of lecturers; C is the chromosome; and Clashes(C, i) is a function that computes the number of times l_i is required to be in multiple venues at the same period, when chromosome C is used to generate the seminar schedule. Movements(C, i) is a function that computes the number of times l_i is required to move from one venue to another, when chromosome C is used to generate the seminar schedule.

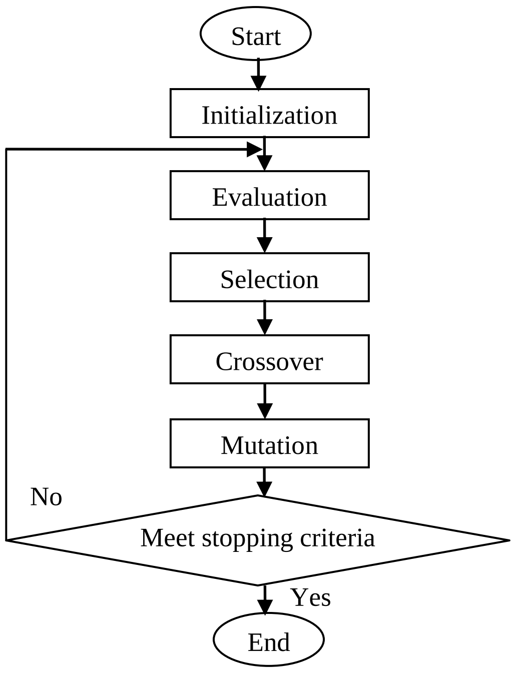

Figure 4. Flowchart of the Genetic algorithm process [14]

Because hard constraints are more severe than soft constraints, the former are penalized 100 times more than the latter. The GA searches for the chromosome with the minimum fitness value (minimum total number of clashes and movements). The fitness values of the seminar schedule presented in Fig. 3(a) is 202 since there are two clashes and two movements of panelists. On the other hand, the schedule in Fig. 3(b) has a fitness value of 2 since all hard constraints were satisfied but panelists changed venues twice.

C. *Selection*

This process is used in the GA to determine which solutions are to be preserved and allowed to reproduce and which ones are to be discarded based on their fitness values. The primary objective of the selection operator is to retain the good solutions and eliminate the bad ones in a population and still keep the population size constant. The type of selection operator to be used is the rank selection, chosen for its ability to allow fittest individual to be selected. The fitness values of the chromosomes were used to sort them in ascending order. This prevents bias towards individuals that are highly fit, thereby reducing speedy convergence.

D. *Crossover*

The crossover operator is used to create new solutions from the existing solutions available in the mating pool after applying rank selection operator. Crossover exchanges the gene information between the solutions in the mating pool. Partially-mapped crossover was used in this work because it prevents duplication of genes. Crossover operators like single point crossovers and double point crossovers often result in duplication of genes indicating that a student presents in multiple sessions, violating the hard constraint that requires each student to present his/her seminar once.

E. *Mutation*

Mutation is the infrequent introduction of new features into the solution strings of the population pool to maintain diversity in the population. Mutation was achieved by swapping two randomly selected genes of a chromosome based on the probability of mutation. The mutation operator stops the algorithm from getting stuck in local minima. Fig. 6 shows how mutation swaps a pair of genes (sessions) between two students. The first and third genes, which are shaded were swapped during mutation.

F. *Stopping Criteria*

Selection, crossover and mutation were repeated after generation of the initial population until one of the following conditions is satisfied: (i) the best fitness value of zero is obtained, signifying that no constraint is violated, (ii) the maximum number of generations is reached, (iii) or the fitness value does not improve within a preset number of generations.

V. EXPERIMENTAL RESULTS

This section discusses experimental results used to validate the proposed GA. The GA was implemented using the MATLAB® simulation tool.

A. *Evaluation Criteria*

The fitness values and running times were used to evaluate the performance of the developed GA-based seminar scheduler.

B. *Experimental Dataset*

Eleven datasets were used to evaluate the GA-based student seminar scheduler. Table I shows the characteristics of each dataset. The first dataset comprises real data obtained from the Department of Computer Science, Federal University of Technology, Minna when Masters students of the Department presented their final thesis seminars in April 2016. The panelist matrix for that dataset is shown in Table II. For the other ten datasets, the number of venues, number of periods, and the panelist matrices were randomly generated such that each student had between two and four randomly selected panelists. Note that the second, third, fourth and fifth datasets have the same panelist matrix and number of sessions, but different combinations of number of venues and periods.

C. *Experimental Setup*

MATLAB R2010a was used to implement the GA. The experiments were carried out on a computer system having a processor speed of 2.4GHz as well as main memory capacity of 4GB, and running the 64-bit windows 7 operating system. The parameters used to run the GA experiments are as follows: the GA was halted after 1,000 generations or if the best fitness value obtained did not improve within 50 generations. The probability of mutating each gene in the population is 0.02 while the probability of crossover is 0.9. All the experiments were run 30 times since GA is stochastic.

D. Results and Discussion

Fig. 7 shows a seminar schedule generated by GA. The fitness value obtained from this schedule is four, since there are no panelist clashes but panelists had to change venue four times, as indicated by the green-colored panelists.

The second, third, fourth and fifth datasets all have 12 sessions as well as the same panelist matrix. They only differ in the number of venues and periods. Fig. 8 shows the relationship between the average fitness value and number of venues for a fixed number of sessions using the four datasets. It can be seen that as the number of venues increases, the number of constraint violations also increases. In the extreme case of excessive parallelism, there is only one period and multiple venues, so all seminars take place at once, resulting in significant constraint violations. On the other hand, when there is only one venue, there are no constraint violations because is no parallelism at all and each student presents in a distinct period.

Table III shows the fitness values and execution time of GA for all the datasets. There are no results for Dataset 9, because the GA is programmed to compare the number of students with the number of sessions before actual execution starts. If the former is greater than the later, the GA displays an error message and terminates because hard constraint(s) are guaranteed to be violated when the number of students exceeds the number of sessions. Otherwise, the GA proceeds normally. The number of venues and periods for Dataset 9 are two and twelve respectively, resulting in twenty four sessions, while the number of students is twenty seven. Fitness values for majority of the datasets shown in Table III are less than 100, indicating that only soft constraints were violated. This shows that the GA is highly successful in finding good seminar schedules.

VI. CONCLUSION

In this research work, a Genetic Algorithm was formulated for scheduling seminars. The GA seeks to minimize clashes among panelists and inconvenience due to panelists' change of venues. Experimental results show that soft constraint violations regarding movement of panelists can hardly be avoided, whereas hard constraint violations can be avoided depending on the inputs to the GA. In future, we plan to incorporate panelists' preferences for period(s) in the GA. Furthermore, the GA can be improved so that empty sessions when no students are presenting do not appear between sessions when students are presenting. For example, sessions 3, 5 and 11 on Fig. 7 are empty, so they should appear at later periods, so that students and lecturers can finish with the seminar as soon as possible.

(a)

1	**2**	**3**	**4**
1	2	4	3

(b)

1	**2**	**3**	**4**
1	4	3	2

Figure 5. Chromosome representation for seminar schedule (a) Chromosome for Fig. 3(a); (b) Chromosome for Fig. 3(b)

(a)

1	**2**	**3**	**4**
2	4	1	3

(b)

1	**2**	**3**	**4**
1	4	2	3

Figure 6. Mutation (a) Before mutation (b) After mutation

TABLE I. SUMMARY OF THE DATASETS

Dataset	Number of venues	Number of periods	Number of students	Number of Panelists
1	2	7	11	13
2	6	2	6	10
3	4	3	6	10
4	3	4	6	10
5	2	6	6	10
6	3	6	15	15
7	3	8	20	18
8	3	10	25	18
9	2	12	27	20
10	3	15	30	20
11	2	20	40	20

TABLE II. PANELIST MATRIX FOR FIRST DATASET

Students	Lecturers												
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	1	0	0	1	0	0	0	0	0	1	0	0	0
2	0	1	1	0	1	0	0	0	0	0	0	0	0
3	0	1	1	0	0	0	0	0	0	0	0	0	1
4	1	0	1	0	0	0	0	0	0	0	1	0	0
5	0	0	1	0	0	0	0	1	1	0	0	0	0
6	1	0	0	0	0	1	0	0	0	0	0	1	0
7	1	1	0	0	1	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	1	0	0	0	0	0	0
9	1	1	0	0	0	0	0	0	0	0	0	0	0
10	1	1	0	0	1	0	0	0	0	0	0	0	0
11	1	0	0	0	0	0	1	0	0	0	0	0	0

	Period 1	**Period 2**	**Period 3**	**Period 4**	**Period 5**	**Period 6**	**Period 7**
Venue 1	s_1 (l_1, l_4, l_{10})			s_{11} (l_1, l_7)	s_9 (l_1, l_2)		s_8 (l_2, l_7)
Venue 2	s_3 (l_2, l_3, l_{13})	s_{10} (l_1, l_2, l_5)	s_7 (l_1, l_2, l_5)	s_2 (l_2, l_3, l_5)	s_5 (l_3, l_8, l_9)	s_4 (l_1, l_3, l_{11})	s_6 (l_1, l_6, l_{12})

Figure 7. Generated schedule for first dataset

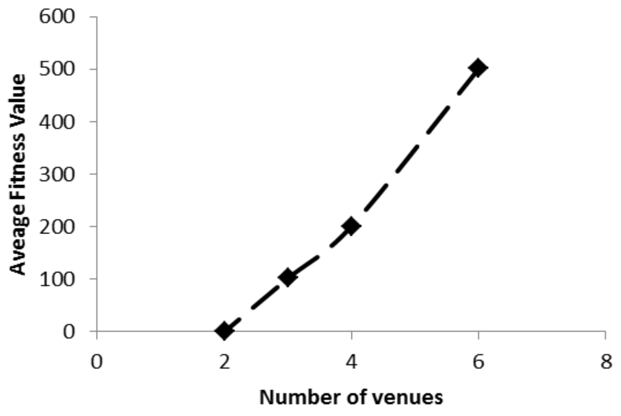

Figure 8. Relationship between average fitness value and number of venues

TABLE III. FITNESS VALUES AND EXECUTION TIMES FOR GA

Dataset	Fitness Value			Execution Time (seconds)		
	Mean	Min	Max	Mean	Min	Max
1	4.20	3.00	6.00	0.90	0.56	1.64
2	501.00	501.00	501.00	0.53	0.39	0.69
3	201.27	201.00	202.00	0.51	0.37	0.74
4	102.00	102.00	102.00	0.40	0.28	0.48
5	0.00	0.00	0.00	0.01	0.00	0.02
6	112.47	109.00	115.00	1.27	0.57	3.52
7	16.83	14.00	20.00	2.21	0.96	3.71
8	24.10	20.00	29.00	2.88	1.30	4.94
9	-	-	-	-	-	-
10	23.50	20.00	28.00	2.17	1.27	6.54
11	18.17	13.00	26.00	3.94	2.17	7.09

REFERENCES

[1] Collins Dictionary, “Timetable,” retrieved 16th October, 2015 from https://www.collinsdictionary.com/dictionary/english/timetable.

[2] D. A. Singh, E. Leavline, R. Priyanka, and P. Priya, “Dimensionality reduction using genetic algorithm for improving accuracy in medical diagnosis,” I.J. Intelligent Systems and Applications, vol. 8 no. 1, pp. 67-73, 2016.

[3] S. E. Sampson, “Practical implications of preference-based conference ccheduling.,” Production and Operations Management, vol. 13 no. 3, pp. 205-215, 2004.

[4] A. Mardiyono, “An intelligent system for course scheduling in higher educations,” International Journal of Information Technology and Business Management, vol. 32 no. 1, pp. 29-34, 2014.

[5] P. Punnakitikashem, J. M. Rosenberger, D. B. Behan, R. L. Baker, and K. Goss, “An optimization-based prototype for nurse assignment,” Proc. 7th Asian Pacific industrial engineering and management systems conference, 2006, pp. 17- 20.

[6] K. Ellingsen, and M. Penaloza, “A genetic algorithm approach for finding a good course schedule,” South Dakota School of Mines and technology 2003, unpublished.

[7] I. Sbiety, M. Dbouk, and H. Kobeissi, “Combining the analytical hierachy process and the genetic algorithm to solve the timetable problem,” International Journal of Software Engineering and Applications (IJSEA) vol. 5 no. 4, pp. 39-50, 2014.

[8] A. Sahu, and R. Tapadar, “ Solving the assignment problem using genetic algorithm and simulated annealing,” IMECS, pp. 762-765, 2006.

[9] A. Chaudhuri, and K. De, “Fuzzy genetic heuristic for university course timetable problem, “ International Journal of Advances in Soft Computing and its Applications, vol. 2 no. 1, pp.100-123, 2010.

[10] T. Engin, “Genetıc algorithm optimization method in the timetable schedules of public transportation,” International Journal Of Engineering Sciences & Research Technology, vol. 3 no. 5, pp. 555-562, 2014.

[11] N. Kumar, and R. K. Karambir, “A comparative analysis of PMX, CX and OX crossover operators for solving traveling salesman problem,” International Journal of Latest Research in Science and Technology, vol. 1 no. 2, pp 98 – 101, 2012.

[12] M. Tabassum, and K. Mathew, “A genetic algorithm analysis towards optimization solutions,” International Journal of Digital Information and Wireless Communications, vol. 4 no. 1, pp. 124-142, 2014.

[13] O. Al Jadaan, L. Rajamani, and C. R. Rao, “Improved selection operator for GA,” Journal of Theoretical & Applied Information Technology,” vol. 4 no. 4, pp. 269-277, 2008.

[14] W.-J Shyr, “Parameters determination for optimum design by evolutionary algorithm,” INTECH Open Access Publisher, 2010.