ISSN 2277-3061

University Course Timetabling using Multi-population Genetic Algorithm Guided with Local Search and Fuzzy Logic

Sedigheh Asiyaban, Zohreh Mousavinasab
Department of Computer, Omidiyeh Branch, Islamic Azad University, Omidiyeh, Iran
s.asiyaban@iauo.ac.ir
zohreh.mousavinasab@iau.ac.ir

ABSTRACT

Problem of courses timetabling is a time consuming and demanding issues in any education environment that they are involved in every semester. The main aim of timetabling problem is the allocation of a number of courses to a limited set of resources such as classrooms, time slots, professors and students so that some predefined hard and soft constraints are satisfied. Furthermore, the available resources are used to the best.

In fact course timetabling is one of optimization problems. It has been proved computational complexity of this problem is NP, so there is no optimal solution for that. Therefore, approximation and heuristic techniques are used to find near optimal solutions. Genetic algorithm for its multidirectional feature has been one of the most widely used approaches in recent years. Hence, in this paper an improved genetics algorithm for timetabling problem has been proposed. In proposed algorithm, the fitness of solutions to satisfy soft constraints due to ambiguous nature of those has been specified using fuzzy logic. Also, local search methods have been applied to avoid the genetic algorithm to be trapped in a local optimum. As well as, the multi-population property is intended to reduce the time to reach the optimum solution. Evaluation results show that the proposed solutions are able to produce promising results for the university courses timetabling.

Indexing terms/Keywords

Genetic algorithm; Local search; Fuzzy logic; Multi-population.

1. INTRODUCTION

The course timetabling problem is one of the complex and time consuming issues in universities that they are involved at the beginning of each semester. Manually preparing an optimal large-scale scheduling table for a university requires high effort and spending many hours of work by experts. However, the generated schedule may not be optimal. Therefore, automation of university course timetabling problem is very important, because it saves a lot of manpower for educational institutions and provides optimal solutions that while satisfying some constraints can enhance the efficiency, quality of education and services [1-3].

It has been proved that course timetabling problem is classified as NP-complete problem [1]. This means the amount of time and work needed to solve this kind of problems increase exponentially with the problem size and so they are more difficult and time-consuming. Thus, approximation and heuristic optimization techniques are used to solve these problems and obtain practical or near optimal solutions rather than exact solutions. Further studies in this field are based on heuristic approaches [3].

Graph coloring heuristics are from the first methods that had been used for solving scheduling and have been widely studied [4]. In these heuristics timetabling problem is model by defining the courses as vertices and conflicts (the courses which cannot be scheduled at a time) by edges. Also, each color represents a time slot. Difficulty of an event is defined as number of conflicts with other events. Therefore, in this model, Degree of a vertex illustrates the difficulty of scheduling of a course that the vertex referred to it. The goal is to construct a timetable in which the highest number of conflicting events are scheduled in feasible time slots and respect soft constraints as possible [4-8]. These algorithms work efficiently for small-scale scheduling problems, but they are not effective in large scale [1, 4].

Heuristic algorithms are partitioned into two general classes [9]: local search (local area) based algorithms and population (global search or global area) based algorithm. The most important local search algorithms include iterative search, tabu search and simulated annealing [10]. This class of algorithms is focused on exploitation rather than exploration. Population-based algorithms start with some initial solutions and refine in order to obtain optimum solution in the whole search space. Therefore, they are so-called global search algorithms. The most significant algorithms of this class are: genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization [11-14].

In recent years, genetic algorithms have been the most common approach to the problem of timetabling. Therefore, a new genetic algorithm is proposed in this study. Three basic component of a GA that effect the efficiency of the algorithm for a particular problem include fitness function, mutation operator, and crossover operator. As mentioned, the fitness of a solution for university course timetabling is dependent on some hard and soft constraints, and a solution is feasible if it can respect all hard constraints. Thus, the fitness of a solution is a zero-one problem from the view of guarantee of the hard constraints. That is, in general, the solutions that violate hard constraints should be excluded. However, since soft constraints don’t have zero-one property, determination of the fitness of a solution is very vague and uncertain in terms of satisfying soft constraints. To overcome this ambiguity, several modeling techniques have been proposed which have been considered ambiguity. These techniques offer more realistic analysis compared with fixed constraints. The most prominent of these techniques is the use of fuzzy logic [15] to model these constraints [16]. Consequently, in proposed GA, fitness of a feasible solution in terms of respecting soft constraints is specified using fuzzy logic (by a set of fuzzy rules).

Genetic algorithm uses mutation and crossover operators, that they are random, to generate new solution. In university course timetabling, if these random operators are applied, the probability of produce a infeasible solution increases. Therefore, in proposed GA, after applying mutation and crossover operators to generate new solutions, a local search algorithm (iterative search in this study) is used. As a result, on the one hand, we take advantage of multidirectional search of GA and benefit from the ability to avoid being trapped in local optimum of local search algorithm, on the other hand. Also, in order to reduce the execution time to reach the optimum solution, the multi-population version of proposed GA has been addressed.

The rest of the paper is organized as follows. In Section 2 the description of course timetabling problem is described and the notations used throughout the paper are given. Section 4 describes the proposed genetic algorithm in detail. In Section 5, we compare experimental results from the proposed GA with those conventional genetic algorithm and iterative search algorithm. Finally, in Section 6, we conclude the paper with directions for future research.

2. DESCRIPTION OF COURSE TIMETABLING PROBLEM

In this paper, course timetabling problem is defined based on educational rules of university system of Iran with some simplification. The main entities are:

Days and time slots: Specified number (5) of working days per week is considered. Every day is partitioned into a fixed number of time slots (8) with same length (one hour) which is the same for all days. Thus, the total number of time slots is equal to the product of the number of days (5) and the number of time slots in a day (8), i.e. $5 \times 8 = 40$. Similar to some studies [1, 2], the time slots are indexed from 1 to 45. Every index determines a day and a time slot in that day. For instance, the index 17 indicates the third working day and the first time slot in that day (hour from 8 to 9). So, a working day and related time slot both are characterized by one index. The set of time slots are displayed as $TS = \{t_1, t_2, \dots, t_{40}\}$.

Courses: Every course consists of a fixed number of lectures per week. These lectures should be scheduled at specific time intervals. Courses are related to different subjects and some of them require special laboratory facilities. In this study it is assumed that each course includes one lecture per week. Indeed, some course has more than one lecture; it models as a number of courses with two constraints: assignment to same professor and no time interference. The set $\mathrm{C} = \{c_1, c_2, \dots, c_{N_C}\}$ represents N_C cources. Each course c_i is defined by its duration, subject, number of students, and special laboratory facilities.

Professors: Each professor has a given schedule to attend at university. Also, she/he is expert in special subjects. The set $\mathrm{P} = \{p_1, p_2, \dots, p_{N_P}\}$ demonstrates N_P professors. Each professor p_i is specified with his/her attendance schedule and subject specializations. The set of subjects are defined by $\mathrm{S} = \{s_1, s_2, \dots, s_{N_S}\}$. N_S represent the number of different subjects.

Classrooms: The set of classrooms is shown by $\mathrm{R} = \{r_1, r_2, \dots, r_{N_R}\}$. N_R is the number of classrooms. Each class r_i is characterized by its laboratory facilities and capacity. Laboratory facilities are described by the set $\mathrm{F} = \{f_1, f_2, \dots, f_{N_f}\}$ that N_f shows the number of distinct facilities.

Curriculums: A curriculum is a group of courses that should be taken by a group of students who have the same entry date and field of study. Thus, fundamental requirement is that the courses in a curriculum don't have any time interference. The set $\mathrm{Cu} = \{cu_1, cu_2, \dots, cu_{N_{cu}}\}$ defines N_{cu} curriculums. Each cu_i is described by its group of courses.

According to mentioned entities, university course timetabling is the allocation of time slot (TS), professor and classroom from the related sets to the set of courses (C) so that all hard constraint are satisfied and soft constraint are met as possible.

The constraints that are enforced strictly due to educational, management, etc reasons of a university are referred hard constraints. Thus, they must be respect in a proposed feasible solution. Lack of compliance with these constraints makes the solution invalid. Assumed hard constraints are:

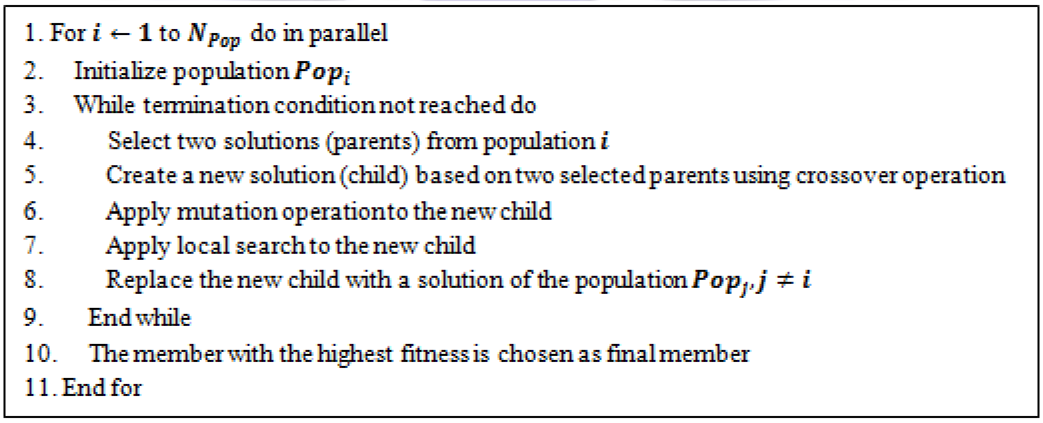
1. The courses related to a curriculum have no time interference.
2. Number of courses devoted to a given class in a specified time slot is the maximum one.
3. Number of courses devoted to a given professor in a specified time slot is the maximum one.
4. No course is assigned to a given professor out of her/her attendance schedule at university.
5. Each course is allocated to a classroom with enough laboratory facilities and capacity.

The constraints that are desired to respect, but the absence of them don't make the solution invalid are called soft constraints. Satisfying these constraints increase the quality of course timetabling and productivity of university facilities. In general, quality of a feasible solution can be evaluated based on how much soft constraints are met. Supposed soft constraints are:

1. Each course is assigned to a professor who is an expert in its field.
2. The minimum and maximum number of hours of attendance will be respected for each student per day.
3. The obtained course timetabling for each curriculum is in consecutive days.
4. Courses are held in the morning.

3. PROPOSED GENETIC ALGORITHM

The proposed algorithm is a combination of the concepts of genetic algorithms, fuzzy logic, local search and multi-population evolution. In fact, the proposed method is a modified genetic algorithm in which a new stage (local search using iterative search) is added to iteration loop of conventional GA. Figure 1 illustrate an overview of proposed method. Each step is described in more detail in following sub-sections.



```
1. For $i \leftarrow 1$ to $N_{Pop}$ do in parallel
2.    Initialize population $Pop_i$
3.    While termination condition not reached do
4.        Select two solutions (parents) from population $i$
5.        Create a new solution (child) based on two selected parents using crossover operation
6.        Apply mutation operation to the new child
7.        Apply local search to the new child
8.        Replace the new child with a solution of the population $Pop_j, j \neq i$
9.     End while
10.    The member with the highest fitness is chosen as final member
11. End for
```

Fig 1: Overview of proposed genetic algorithm

3.1 Solution Encoding

Genetic algorithm requires the coding of problem variables for evolutionary process. Encoding is a method for representing the solution of a problem that must be optimized. Each solution based on selected encoding is referred as a chromosome in GA, and each building element of a chromosome is called a gene. Encoding highly influences the success of GAs in finding optimum solution and simplicity of their implementation.

In the proposed GA, the chromosome is represented as a cell array of size N_C (the number of courses), and each cell (gene) defines the characteristics of a course, namely, professor, time slot and classroom, as figure 2 illustrates. With this encoding, retrieval and evaluation of generated course timetabling is easily possible.

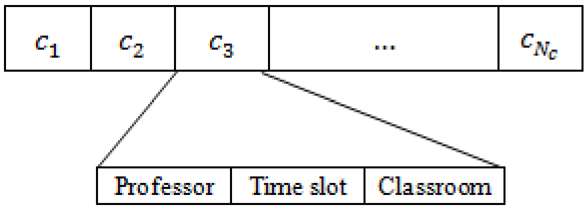


Fig 2: Encoding of a chromosome

3.2 Multi-Population Evolution

In order to reduce execution time, several versions of the algorithm with a small population size rather than a large-sized version are executed in parallel (line 1 in figure 1). Also, to preserve diversity of small population as a large population, the new generated children from different population are exchanged with each other (line 8 in figure 1).

3.3 Initialization of Population

Genetic algorithm starts by generating an initial population of solutions (chromosomes). These initial solutions can be constructed in randomized or heuristics manner. Furthermore, each solution should be represented according to the specified encoding in sub-section 3-1.

In this study, the initial population is generated so that the random properties of solutions are preserved, as well as all hard constraints are met. To implement this goal, some matrices, usually binary, based on problem inputs (i.e. courses, professors, classrooms, time slots, curriculum, and etc.) are produced. A detail of construction of these matrices and initial population with mentioned features is presented in [17].

3.4 Fitness Function

Fitness of a solution is dependent upon meeting the hard and soft constraints. Given that the initial population and the genetic operators are defined so that they satisfy all hard constraints of existing solutions in the population, fitness of a solution is related only to respect the soft constraints. Consequently, the fitness function is expresses exclusively based on respecting soft constraints.

In addition, since soft constraints are vague and there are some if-then relationships between them, fuzzy logic is used to define fitness function. Thus, a membership function is defined for each of soft constraint separately. And the fitness function is defined as follows:

$$F_{fitness}\ (Sol) = \sum_{i=1}^{4} w_i \mu_i \qquad (1)$$

Which μ_i and w_i respectively are value of membership function and weight of soft constraints i that described in section 2. The values of membership functions are defined in the range [0, 1] and higher value indicates a better fitness. Details of definition of membership functions are explained in [17].

3.5 Selection

In GAs, in order to produce a new generation using genetic operators, some chromosomes from the current generation are selected. In general, the higher fitness solution has the more selection chance.

In proposed algorithm, tournament selection is used. In this method, some chromosomes are selected by roulette wheel and then, the best chromosome is selected. In each generation, two chromosomes are chosen by tournament selection (line 4 in figure 1).

3.6 Crossover Operator

Crossover operator is responsible for generating a new solution by using a pair of solutions. Uniform crossover operator is used in proposed method (line 5 in figure 1). In uniform approach, genes of parental chromosomes are transferred to the new solution based on a stochastic scheme. For this purpose, a random binary string of size N_C (chromosome length) is generated. Zero (one) means the gene is selected from the first (second) parent. Due to the random nature of operator, it is possible that some hard constraints in the new solution are violated. Thus, the new solution will be reviewed in compliance with hard constraints.

3.7 Mutation Operator

Mutation operator randomly selects a gene from the new solution and changes its value with probability of $\mathrm{P_{mut}}$. Applying this operator is necessary in order to avoid the local optimum. in this study, the allocated time slot to each gene is altered randomly with probability $\mathrm{P_{mut}}$ (line 6 in figure 1). However, to avoid violating the hard constraints, the proper time slot is chosen randomly and intelligently.

3.8 Local Search

In proposed algorithm, local search is based on iterative search algorithm (line 7 in figure 1). This algorithm operates based on a set of neighborhood structures. Neighborhood structures are used in this paper are as follows:

$\mathbf{H_1}$: One professor is selected at random; then, time slots of two courses (in new solution) assigned to her/him are exchanged with each other.

$\mathbf{H_2}$: One course is picked arbitrary from new solution; then professor of that is changed. Also, if it is necessary, classroom and time slot of it is altered.

$\mathbf{H_3}$: A course from new solution is selected randomly. Then another course from new solution with same subject is chosen and characteristics of them interchanged.

All of these neighborhood structures are applied so that the hard constraints are met. In each iteration, iterative search algorithm work as follows:

- A list of size K of mentioned neighborhood structures are produced randomly.
- K neighborhood structures are applied to the main solution (the solution from line 6 in figure 1) and produce K new solutions.
- Fitness of all new solution is calculated using equation (1) in sub-section 3-4.
- Solution with the best fitness from K new solution is selected.
- If the best new solution is better than the main solution, then the best new solution is substituted with the main solution.
- If the best new solution is not better than the main solution, to avoid trapping in local optimum, the best new solution with a very small probability is substituted with the main solution.

4. EXPERIMENTAL RESULTS

In this section, an empirical investigation has been offered, examining the effectiveness of proposed GA rather than conventional GA and local search algorithm. The input parameters used in our experiments were generated as follows:

- Number of courses (N_C) are selected from the range $[20, 300]$.
- Duration of each course are defined randomly as a integer number in rang [1,3].
- Number of subject is 7, and subject of each course and expertise of each professor are determined randomly as an integer number in range [1, 7].
- Number of laboratory facilities is 4, and required facility for each course and available facility of each classroom are defined randomly as a subset of integer number in the range [1,4].
- Number of available classroom is generated according to a uniform distribution as an integer number from the range [$\left(\frac{N_C}{40}\right) * 2,\ \left(\frac{N_C}{20}\right) * 3$].
- Number of professor is defined according to a uniform distribution as an integer number from the range [$\left(\frac{N_C}{16}\right) * 2,$ $\left(\frac{N_C}{8}\right) * 3$].
- Attendance schedule of each professor is generated as a subset of TS of size 16.
- Number of curriculums is described as a integer in range [$\left(\frac{N_C}{12}\right),\ \left(\frac{N_C}{20}\right)$].

The determined intervals for number of classrooms, professors and curriculums are in accordance with Iran education rules.

Chosen values for parameters of algorithm are defined in table 1. These values are selected experimentally and using the previous experiences in solving the university course timetabling.

Table 1. Setup of parameters of algorithms

Parameter	Value
Number of neighborhood structure (K) in local search	10
Size of a population in proposed GA	100
Number of iteration of local search in proposed GA	N_C
Number of iteration of local search algorithm lonely	15000
Number of iteration proposed GA	600
Number of iteration conventional GA	3000
Probability of mutation operator (P_{mut})	0.45
Number of populations	2
w_i for $i = 1, 2, 3, 4$	1

We investigated the performance of the following algorithms:

Conventional genetic algorithm: This is the algorithm that addressed in figure 1 without applying local search algorithm (line 7).

Local search: The local search that defined in subsection of 3-8, is used lonely.

Proposed genetic algorithm: this is the contribution of this study that described in section 3 in details.

Figure 3 shows the fitness of final solution with the different methods when number of courses, N_C, varies from 20 to 300 with step 20. For each N_C value, 100 input parameters are generated randomly as mentioned above, and average results are used. From the simulation results, we can see that, however conventional GA has a performance same as proposed GA, but, when number of courses increases, has the lowest ability to find optimum solution. Also, it can be observed that proposed GA has always higher fitness than local search algorithm. This is because of taking advantage of multidirectional search of GA and utilizing from the ability of local search algorithm to avoid being trapped in local optimum.

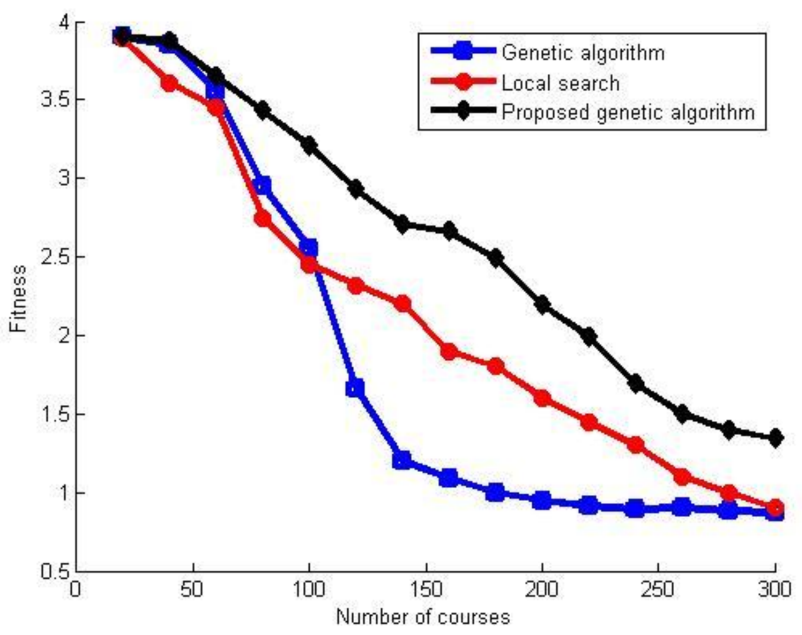


Fig 3: Fitness evaluation of different algorithms when number of courses (N_C) is varying. Algorithms are running according to defined parameters in table 1.

To illustrate the time efficiency of proposed GA two experiments have been performed (figure 4 and 5). Figure 4 plots the elapsed time (in terms of second) in different algorithms to reach fitness of 2.7 when number of courses varies. Also, Figure 5 shows the obtained fitness from different methods after on hour. We observe from figure 4 and 5 that the proposed GA outperforms the other algorithms in terms of execution time.

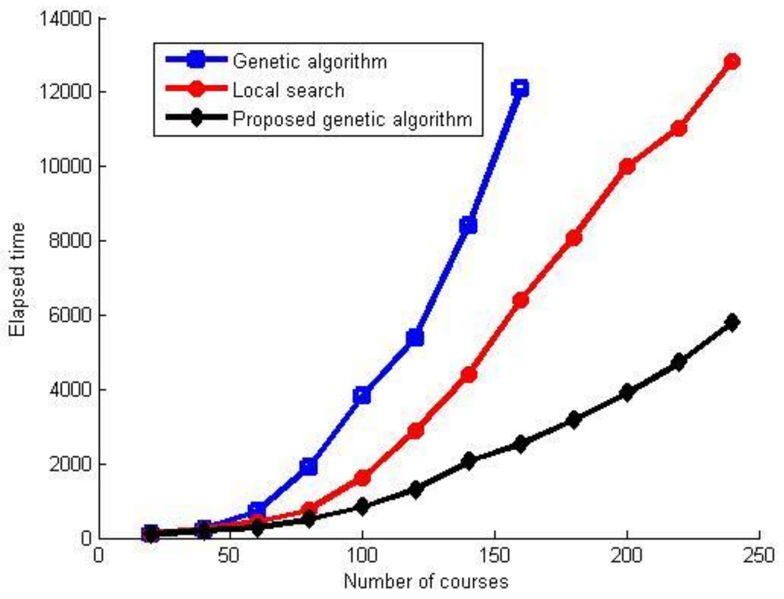


Fig 4: Elapsed time (in terms of second) in different algorithms to reach fitness of 2.7

Because of trapping in local optimum, conventional GA cannot reach the fitness 2.7, when the number of courses increases. Therefore, in figure 4 the plot related to GA has not been drawn from a point to the next. Also, figure 4and 5 shows the speed of proposed GA in improving the fitness has a linear relation with the elapsed time, but local search and GA has a near log relation with passing the time.

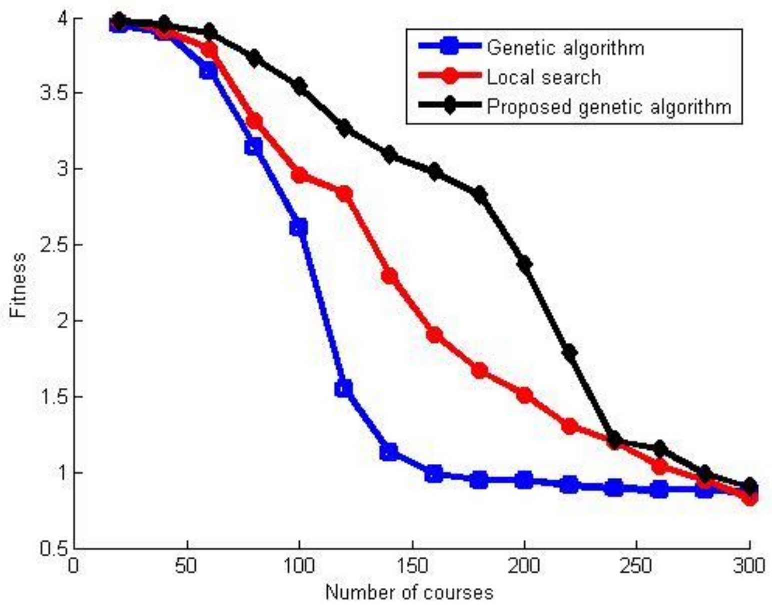


Fig 5: Obtained fitness from different methods after one hour

5. CONCLUSIONS AND FUTURE WORK

University course timetabling is a complicated and time consuming work, which should be performed one or more times in a year in each education environment. It has been proved that this problem is classified as NP-complete. So, focus of research is on heuristic approaches. In this paper we have introduced an algorithm for university course timetabling problem. The main based of the proposed algorithm is genetic algorithm and fuzzy logic is used to determine the fitness of feasible solution in terms of respecting soft constraints. In proposed GA, after the applying random operators (mutation and crossover) to create a new solution, iterative search with some certain neighbor structures are uses to guide the new solution to near optimum solution. Experimental results demonstrate the higher efficiency of proposed GA than conventional GA and iterative search. That is, a more acceptable solution in less time is found because of take advantage of multidirectional search of GA, benefit from the ability to avoid being trapped in local optimum of local search algorithm, and using multi-population version.

The proposed algorithm still has some aspects which can be investigated. In future work, the more soft constraints should be studied to get closer to the real-world needs and obtain more efficient solution. Furthermore, proper fuzzy fitness function should be defined for these new soft constraints. Also, the use of local search algorithm with higher exploitation power, especially adaptive approaches can be considered as future work.

ACKNOWLEDGEMENTS

We are appreciate to thanks Islamic Azad university-Omidieh branch for their fund support of this research.

REFERENCES

[1] Yang S, Jat S. N. 2011. Genetic algorithms with guided and local search strategies for university course timetabling. IEEE Transactions on System, Man, and Cybernetics -PART C: Applications and Reviews, vol. 41, NO. 1, January.

[2] Lewis R. 2006. Metaheuristics for university course timetabling. PhD Thesis,School of Computing, Napier University, Edinburgh.

[3] Aydın M. A. 2008. Solving university course timetabling problem using genetic algorithms. MSc thesis, Bahcesehir University Institute of Sciences Industrial Engineering, İstanbul.

[4] Welsh D. J. A, Powell M. B. 1967. The upper bound for the chromatic number of a graph and its application to timetabling problems. The Computer Journal, vol. 11, pp. 41–47.

[5] Burke E.K, Newall J.P. 2004. Solving examination timetabling problems through adaption of heuristic orderings. Annals of Operations Research, vol. 129, pp. 107– 134.

[6] Asmuni H, Burke E. K, Garibaldi J. M.2005. Fuzzy multiple heuristicordering for course timetabling. Proceeding of the 5th United Kingdom Workshop on Computational Intelligence, London,pp. 302–309.

[7] Khuri S, Walters T, Sugono Y. 2000. A grouping genetic algorithm for coloring the edges of graphs. Proceeding of the ACM/SIGAPP Symposium on Applied Computing, ACM Press, pp.422-427.

[8] Campbell D. 2004. Harmonious and achromatic numbers and related coloring problems in graph theory. MSc Thesis, University of Dundee.

[9] Lewis R. 2008. A survey of metaheuristic based techniques for universitytimetabling problems. OR Spectrum, vol. 30, no. 1, pp. 167–190.

[10] Lu Z, Hao J. K. 2010. Adaptive tabu search for course timetabling. Europe Journal Operational Research, vol. 200, no. 1, pp. 235–244.

[11] Irene S. F. H, Deris S, Hashim S. Z. M. 2009. A study on PSO-baseduniversity course timetabling problem. Proceeding of International Conference Advance Computing Control, pp. 648–651.

[12] Ossam C. 2009. University scheduling using genetic algorithm. MSc thesis, Dalarna University, 2009.

[13] Behdad M, Dehghani T, Zaker Tavallaei M. 2010, A new approach in university course timetabling by using genetic algorithm. Proceeding of the 14th int'l CSI computer conference (CSICC09), Tehran, Iran, pp 205-215.

[14] Zheng Y, Liu L. 2011. A novel quantum-inspired genetic algorithm for a weekly university scheduling optimization. Proceeding of International Conference on Information Science and TechnologyMarch 26·28, pp373-376, Nanjing, Jiangsu, China.

[15] Masri A, Ghaith J. 2009. Hybrid ant colony systems for course timetablingproblems. Proceeding of 2nd International Conference on Data Mining Optim., pp. 120–126.

[16] Klir G, Folger T. 1988. Fuzzy sets, uncertainty and information, Prentice Hall, New Jersey.

[17] Chaudhuri A, De K. 2010. Fuzzy Genetic Heuristic for University Course Timetable Problem. International journal of Advance Soft Computing Application, Vol. 2, No. 1, March.

[18] Asiyaban S, Mousavinasab Z, 2013. University course timetabling using guided genetic algorithm with local search and fuzzy logic, technical Report.