Journal of Computer Science and Cybernetics, V.32, N.4 (2016), 285–301
DOI no. 10.15625/1813-9663/32/4/7962

A GENETIC ALGORITHM BASED METHOD FOR UNIVERSITY COURSE TIMETABLING PROBLEMS AND APPLICATION IN HANOI OPEN UNIVERSITY

DUONG THANG LONG

Faculty of Information Technology, Hanoi Open University, Vietnam
duongthanglong@gmail.com

Abstract. Timetabling problems is one of very significant problems in many fields of applications. As mentioned in [3], this problem is NP-complete with numerous factors and constraints. In general, it is treated as a multi-objective optimization problem. Currently, the work of scheduling is very difficult in universities, especially in credits training. Almost it is impossible to control all cases of the problem by human. Therefore, we cannot manually give an effective solution for this problem. There are quite many methods to resolve this problem in literature, they are mostly searching methods based on genetic algorithms and their results are proved effective in practice. In this paper, we propose a method based on genetic algorithms for university course timetabling problems with some modifications and apply it to real-world datasets in Hanoi Open University.

Keywords. University course timetabling, genetic algorithms.

1. INTRODUCTION

The university timetabling is a typical scheduling problem. However, it is more complex factors than the form of conventional scheduling problems, such as teachers, students, time-slots, classrooms, credit classes or courses, and especially among the major constraints of these elements. More generally, the timetabling problem including many relevant factors should be considered, such as examinations, practice, lecture halls, etc. [3].

As mentioned in [3], Even and Itai showed that the scheduling problem is a NP-complete problem. Typically, scheduling problems are conducted in the traditional way, by intuitive and direct calculation of human. Currently, due to the diversity and the many changes of the binding between the elements in the problem, which are limited resources and the complexity of the factors, the scheduling problem often takes a lot of time and labor. So the use of computers to perform scheduling, not only to show the research interests of the authors, but also allows achieving superior results despite more constraints. Obviously, this leads to saving a lot of time and effort.

Scheduling problems and methods of solving the problem have been researched from 1960. In [7], pointed out that, Hertz has proposed using Tabu search method to solve scheduling problems including two stages (TATI / TAG) and further pointed out that this is the appropriate way to solve school schedule and calendar with large-scale implementation. Genetic Algorithm (GA) was applied to solve the scheduling problem in universities in order to overcome the limitations of traditional methods. Optimization algorithm ants (ACO) are mentioned by Nothegger et al. for using to solve this hybridisationproblem. Meanwhile, Tassopoulos and Beligiannis's method of swarm optimization establishes a timetable for the various schools in Greece. Al.Betar et al. proposed a method of hybrid (HHS) to solve scheduling problems for the university. HHS integrated algorithm optimization and climbing hills swarm to balance the elements of exploration and search.

The authors, in [6], indicated that the genetic algorithm (GA) is a suitable method that can be used to identify, solve an optimization problem, especially the scheduling problem. The authors focused on the scheduling problem of the university, it can be divided into school schedule and exam scheduling. Accordingly, the method based on improved GA to solve this problem with changing the filter selection, hybridization and mutation to achieve high efficiency. The authors also used an alternative strategy, in order to avoid falling into local optimum.

M. Abbaszadeh et al. in [1] using GA to solve scheduling problems in the universities, they have changed the structure of performing gene sequence, allowing genetic mutations to achieve the transferring 15% of better individuals to the next generation. Moreover, to avoid falling into local optimum, they considered the impact their parameters mutation, removed repetitive genes and replaced by better gene sequences. The result provides high efficiency and maximizes accuracy for binding.

The authors of [8] proposed a genetic algorithm with the binding elements of the problem using fuzzy measure (fuzzy) in order to adapt to the practical constraints, such as the requirements of the faculty for time, teaching expertise.

However, these authors mainly focus on a type of the problem which is all student fixed in each class. Since, we have to assign each class to some courses so the students of the class belong to those courses. This kind of the problem only can be applied to traditional training which does not allow flexibility of student learning. In opposite, in credit training we can allow student choosing courses, time-slots, even teachers for learning but the above methods cannot solve this kind of the problem. So, in this paper, we propose methods for solving scheduling problems for credit course timetabling problem using genetic algorithms with some constraint parameters using fuzziness of hedge algebra and added temperature factor to influence the genetic operations to bring high efficiency. The article consists of Part 1 as an introduction to the use of genetic algorithms for solving scheduling problems. Part 2 discussing in detail the genetic algorithms and scheduling problems in the education of university credit courses. Part 3 proposes a method for scheduling with parameters binding with electronic dimming processing based genetic algorithms mentioned in section 2. Section 4 is to build the program and test software at Hanoi Open University, reviews results. Finally is the conclusion.

2. PROBLEM FORMULATION

In general, scheduling problem in universities (course timetabling problem - CTP) includes finding the appropriate allocation of time within a limited time period for all events (such as courses, semester exams) and assigns them to a number of resources (teachers, students and classrooms) to ensure that the constraints are met [1, 2, 3, 4, 6]. However, in credit training, depending on the characteristics of each university, the scheduling problem will be deployed with certain differences. In this paper, we use resources including teachers and classrooms. Remaining student factor will engage in the objective function calculations, which mentioned later. This would be appropriate to the case that the timetable of courses will be implemented before students enroll.

According to the characteristics of each university and the organization of credit training, for convenience in the design of algorithms, in this paper we assume that the steps for holding the problem as follows: at the beginning of each semester, based on the ability of registering to subjects, we propose all courses from subjects, then we schedule or make a timetable for the courses and announced to students in order to register. Each course consists of its name, subjects, type of course-room (either small room or hall room or practice room). Scheduling problem now becomes the work assigned to teachers, time-slots, classrooms for each course proposed that satisfies the given requirements and constraints.

Typically, the constraints of the problem needed to be satisfied are divided into two categories [1, 4, 9]: hard constraints and soft constraints. Hard constraints must certainly be met and fulfilled. It can be the following requirements:

(H1) Not allowed to allocate the resources (teachers, classrooms) for various events (courses) in the same time.

(H2) The classroom assigned to an event (course) must be within the appropriate resources for the event. Accordingly, an event was held in a room must have appropriate infrastructure to be able to organize events, such as computer labs, hall classrooms, etc.

(H3) An event (course) can only be assigned to a teacher if he or she has sufficient knowledge and capability to respond to the teaching expertise of that course.

(H4) Lecturers are assigned to teach only in the present time-slots to work at the school, and will be determined before making timetable.

On the other hand, the soft constraints are desired to be satisfied as highly as possible and to meet as many constraints as possible, but it is not sufficient for a solution of problems. Some soft constraints are usually considered in the study and implementation of the following:

(S1) Assign teachers to the courses so that teaching ability of teachers is as high as possible.

(S2) Priority assigned timeslot to the courses with the teachers that the teacher's desires is met.

(S3) Ensure balance number of course for each teacher, i.e., the minimum and maximum number of courses should be taken.

(S4) It should be given priority courses with prerequisite of a subject to the same timeslot. This aims to increase the ability to enroll for more students on the timetable.

(S5) The ability of students enrolling on the timetable is as high as possible. This special meeting of the given type of credit training, as above, which the courses are firstly assigned time-slots, teachers and classrooms, then students will register to this timetable of courses, rather than a traditional timetable with students are fixed in each course.

For credits training, we do not need to design curriculums with the fixed mandatory subjects of a term, instead, we always design a diagram of pre-requisite subjects for curriculums. Then, students want to choose subjects for learning in a term, he/she must be passed all pre-requisite subjects belonging to the chosen subjects. So the constraints S4, S5 have significant meanings. The S4 constraint shows that when a timetable with more subjects and theirs pre-requisite is at the same timeslot, students have more chance to choose subjects for learning. The S5 constraint indicates that the more chosen subjects, the less time to complete learning at school. In addition, when we give a timetable with more chance for choosing subjects by learners, we may have more students at school in a term, then the school financing may increase, teachers have more teaching time, maximum facilities are used,...

In addition, depending on the particular credit training and the requirements of each university, the soft constraints can be changed, choose additional factors for properly reality. Now we describe the input data for the scheduling problem and formalized as an optimization problem model. In this model, we use the following symbols:

n_R	$-$Number of classrooms, $\{R_1, R_2, ..., R_{n_R}\}$ denotes the set of classrooms
n_C	$-$Number of events (courses), $\{C_1, C_2, ..., C_{n_C}\}$ denotes the set of courses
n_L	$-$Number of lecturers, $\{L_1, L_2, ..., L_{n_L}\}$ denotes the set of lecturers.
n_S	$-$Number of students, $\{S_1, S_2, ..., S_{n_S}\}$ denotes the set of students
n_T	$-$Number of time slots, $\{T_1, T_2, ..., T_{n_T}\}$ denotes the set of time slots.

Normally, the university timetable should be set weekly time-slots, such as morning Monday, afternoon Tuesday, morning Thursday,... of the week. Here we assume 6 school days of a week from Monday to Saturday, with two sessions in every day which is morning or afternoon, so we have 12 training sessions a week. However, we can divide each day into many period of time and the time-slots can be more than this.

However, we can easily satisfy (H3), (S1) and (S3) constraints directly by assigning lecturers to every course based on experts and specialized users. Because the H3 and S1 constraints can be taken into account, each teacher is manually assigned to some courses by users, since we also easily satisfy H3 and S1 constraints. Similarly, the S3 constraint can be easily satisfied by user based on number of courses assigned to each teacher. This would have the potential to speed up the convergence of methods, because of reducing the search space. The H1 constraint will be taken to meet during scheduled. The remaining constraints can be represented by the matrix, which are called matrix constraints, as following:

(H2) Matrix $C \times R = \{CR_{i,j} | i = \overline{1, n_C},\ j = \overline{1, n_R}\ \}$ defines each course can only be assigned to which classrooms. The value of this matrix is $\{0, 1\}$, 1 denotes that it can be assigned and 0 is not.

(H4, S2) Matrix $L \times T = \{LT_{k,l} | k = \overline{1, n_L},\ l = \overline{1, n_T}\}$ defines each teacher may be present and teach at the school in the time-slots. We integrate H4 into S2 constraint, which is a priority in the timeslot as high as possible. Therefore, this matrix will receive value in the form of language. For example, NO, NORMAL, GOOD, VERY GOOD,.. The NO value denotes a teacher will not be present during at that timeslot.

(S4) Matrix $C \times C = \{CC_{i_1,i_2} | i_1 = \overline{1, n_C},\ i_2 = \overline{1, n_C}\}$ describe prerequisite subjects between courses, it will receive the binary value 0 if there is no prerequisite subjects or 1 if there is prerequisite subjects of them. The courses with prerequisite subjects should be in priority for the same timeslot in order to increase registered ability of students.

(S5) Matrix $S \times C = \{SC_{m,i} | m = \overline{1, n_S},\ i = \overline{1, n_C}\}$ describes constraints on the ability of students registered to the courses, receiving binary values 1 or 0, respectively, may be registered or not.

Now we represent the timetable as a table of the corresponding column is the courses, the corresponding rows are lecturers, times lots, classrooms are as follows:

In which, as mentioned, we assigned lecturers to every courses to certainly satisfy requirements of the H3 constraint, this is also for balancing the number of courses for every teacher by users. However, a course can only continue running if the number of registered

Table 1. Representation of the timetable

Courses	C_1	C_2	..	C_i		C_{n_C}
Lecturers	L_{k_1}		...	L_{k_i}		
Timeslots	T_{l_1}		...	T_{l_i}		
Classroom	R_{j_1}		...	R_{j_i}		

students is enough large. This is a dynamic factor, so universities should take somehow to ensure that as maximum courses are expected to be deployed as possible.

Thus said the problem could be condensed in a shorten form, which distributes set of time-slots $\{T_1, T_2, ..., T_{n_T}\}$ and set of classrooms $\{R_1, R_2, ..., R_{n_R}\}$ on a table so that the full satisfaction of the hard constraints H1, H2, H4 and it can be reached the soft constraints S2, S4, S5 as high as possible. According to the representation of a timetable in Table 1, the hard constraints H1, H2, H4 can be defined as follows:

(H1) No exist two columns (C_{i1}, C_{i2}) which pair values in two rows of teachers and either classrooms or time-slots is identical, ie:

$$\forall(C_{i_1} \neq C_{i_2}) : (L_{k_{i1}}, T_{l_{i1}}) \neq (L_{k_{i2}}, T_{l_{i2}}) \wedge (R_{j_{i1}}, T_{l_{i1}}) \neq (R_{j_{i2}}, T_{l_{i2}}).$$

(H2) No exists columns that pair values of course and corresponding room in $C \times R$ matrix equal to zero, ie:

$$\forall(C_i, R_{j_i}) : CR_{i,j_i} \neq 0.$$

(H4) No exists columns which pair values of lecturers and corresponding time-slots in $L \times T$ matrix equal to zero, ie:

$$\forall(L_{k_i}, T_{l_i}) : LT_{k_i,l_i} \neq 0.$$

We apply hedge algebras based fuzzy parameters to S2 soft constraint for calculating satisfying of the problem, whereby each teacher will be defined a priority set by the terms of hedge algebras [5]. For example, lecturer L_k select the preferred time-slots by using 3 terms of hedge algebras which including {low, normal, high}. The representative function of priority time-slots for lecturers treating as triangles with the top vertex is the center $(\upsilon(x))$ of x, two remaining vertices is selected from the center of the two beside time-slots in the sequences. The figure 1 illustrates a teacher chooses morning Tuesday at a normal, morning Wednesday at a high, and low for afternoon Wednesday, the remaining time-slots are not chosen, i.e., the teacher is not at university for that time.

We denote that the priorities time-slots of teachers to $p(t)$, the function for representing of scheduled time-slots of teachers is $r(t)$, then the evaluated function of time-slots satisfying for teachers, which can be used by min operator as following:

$$s(t) = \min\{r(t), p(t)\}.$$

The Figure 2 illustrates triangular function $p(t)$ (Figure 2.a) which is chosen by teachers, the function $r(t)$ represents scheduled time-slots (Figure 2.b), and $s(t)$ is the intersection of the two above functions (Figure 2.c).

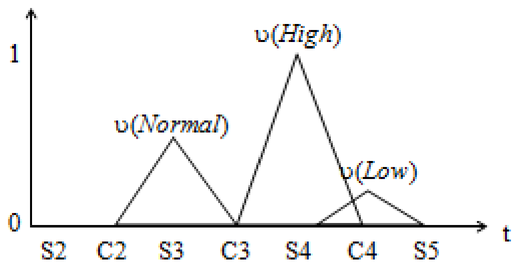

Figure 1. Selection of priority-time-slots of a teacher

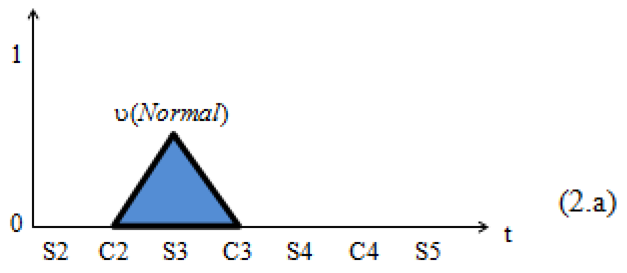

Figure 2.a. time-slots chosen by teachers

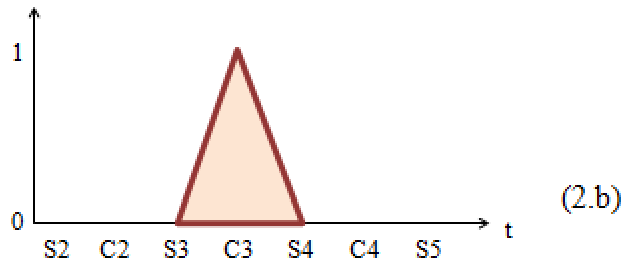

Figure 2.b. time-slots scheduled for

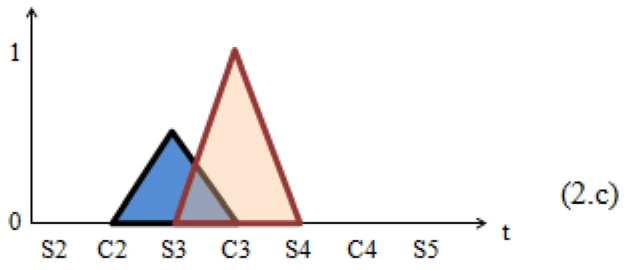

Figure 2.c. time-slots satisfying evaluated of timetable

Figure 2. Evaluation function for time-satisfaction level of teachers

We now can assess the degree of time-satisfaction of teachers for all time, denote by L_k, as following ([7]):

$$F_k^2 = \frac{\int_0^\infty s(t)dt}{\int_0^\infty r(t)dt}.$$

The S4 soft constraint can easily count how many pairs of courses, where one of them is prerequisite of another according to $C \times C$ matrix and the courses is at the same scheduled time, called x is the number of counts, then this would be highly bound rates are as follows:

$$F^4 = 1 - \frac{1}{\sqrt{x^2+1}}.$$

The S5 soft constraint is evaluated based on the capability of registering students according to the register matrix $S \times C$ and the timetable which is scheduled. For each student S_q, we build a plan for registering on the timetable, this may be applied the optimal method to build, called y_q is the number of registered courses by student S_q. Notice that each subject can be established many courses, so students cannot enroll more than one course of each subjects. To ensure compliance with regulations and consistent practice, the most appropriate value y_q is about 4 to 7 in one semester and should not exceed 10 subjects, whereas evaluation for this form of binding the following trapezoidal:

$$F_q^5 = \begin{cases} \dfrac{y_q - 1}{3} & , 1 \le y_q \le 4 \\ 1 & , 4 \le y_q \le 7 \\ \dfrac{10 - y_q}{3} & , 7 \le y_q \le 10 \\ 0 & , \textit{else}. \end{cases}$$

Model of the problem can be stated as a formalization of multi objective optimization problems as follows:

$$\left(\sum_{k=1}^{n_L} F_k^2\right) \to \max, \quad F^4 \to \max, \quad \left(\sum_{q=1}^{n_S} F_q^5\right) \to \max$$

subject to

(H1.a) $\sum\limits_{i=1}^{n_C} z_{i,l,j} \le 1, \quad l = \overline{1, n_T},\ j = \overline{1, n_R}.$

(H1.b) $\sum\limits_{i=1}^{n_C} w_{i,k,l} \le 1, \quad k = \overline{1, n_L},\ l = \overline{1, n_T}.$

(H2) $\sum\limits_{l=1}^{n_T} \sum\limits_{j=1}^{n_R} CR_{i,j} \cdot z_{i,l,j} = 1, \quad i = \overline{1, n_C}.$

(H4) $\sum\limits_{k=1}^{n_L} \sum\limits_{l=1}^{n_T} \omega\left(LT_{k,l}\right) \cdot w_{i,k,l} = 1, \quad i = \overline{1, n_C}.$

which $z_{i,l,j}$ is a binary variable defined the course C_i is assigned to time-slot T_l and classrooms R_j if its value is 1 or otherwise $\omega(.)$ is a function to determine a value of LT matrix is NO

($\omega = 0$) or not ($\omega = 1$), $w_{i,k,l}$ is a binary variable determining the course C_i is assigned to teacher L_k and time-slot T_l if its value is 1 or otherwise.

In the next section we will design optimization search algorithm to the scheduling problem based on genetic algorithms.

3. GENETIC ALGORITHM BASED METHOD FOR CREDIT COURSE TIMETABLING PROBLEMS

In this section, an algorithm based on genetic algorithm (GA) for scheduling problem as mentioned in section 2 is introduced. The method based on GA optimization parameters has combined to temperature evolution control, i.e., it will participate in the genetic operations to increase the convergence of the algorithm, as well as restrictions fall into local optimum [5]. GA is known as a repeating search procedure and widely used in solving optimization problems, this method is based on the basis of biological evolution. Some candidate solutions of the problem will be maintained in iterations of evolution. Genetic operators such as mutation and crossover are applied to evolve the solution in hopes of finding a good solution and at the same time have a high chance to maintain into the next generation. First, this approach needs to encode solutions of the problem into an appropriate gene sequence which can apply appropriate genetic operators to. There are two different approaches to this encoding [8], which is directly encoding and indirectly one. For direct encoding, all the properties of the solution as courses, teachers, time-slots, classrooms, etc., are directly encoded in the gene sequence of the gene (chromosome). However, direct encoding method may risk making an invalid solution, i.e., the hard constraint violations since apply genetic operations to generate offspring from parent. Another way is the indirect encoding method, with each gene sequence will need a procedure to build the suitable timetable from there. For example, a genetic sequence (chromosome) usually represents a sequence of events that are put on the timetable based on a given procedure.

3.1. Gene sequence encoding for a solution of the problem

The encoding method may also use one of three methods, which is binary encoding with each gene is represented binary value parameter of the schedule, or integer encoding or real encoding. Authors, in [5], show that there are no specific directions for using the type of encoding scheme in the specified problem rather, it depends upon the applicability and the requirements of the problem, but the real encoding is more used than binary encoding or integer encoding for function optimization problems. In this paper, we use a method of real encoding and direct form. To overcome the case of a solution encoding genes fall into the hard constraint violations, we use a penalty in fitness function of solutions, which will be described later.

We encode all parameters of the timetable given in table 1 to a chromosome. So a chromosome has $n_C \times 3$ length of gens. In the case of easier, without loss of generality, we can assign teachers to every course in the timetable by experts or users, and it now need to be scheduled (assigned time-slots, classrooms) for all courses. We have just designed a solution gene sequence corresponding to table 1. The length of chromosome is now $n_C \times 2$, as following:

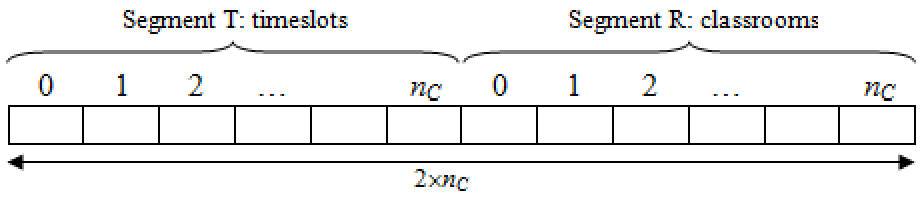

Figure 3. Chromosome of gene encoding of solution

The value of each gene (g_i) of this encoding is limited to $[0, 1]$, thereby to determine the value of real domain corresponding of time-slots as well as classrooms as a function as follow:

$$f^T : [0,1] \to [1, n_T], \quad f^T(g_i) = \lceil g_i \cdot n_T \rceil$$

$$f^R : [0,1] \to [1, n_R], \quad f^R(g_i) = \lceil g_i \cdot n_R \rceil$$

in which, symbols $\lceil \cdot \rceil$ to get the nearest upper integer of values.

3.2. Fitness function (for each individuals corresponding gene sequence)

The fitness function of a solution depends on the binding hard constraints and soft constraints. As mentioned, the direct encoding method, the hard constraints will be evaluated as a coefficients penalty in fitness function to avoid violations hard constraints of individuals. Soft constraints are in the partly qualitative form, they are vague values and difficult to be quantified precisely and there are some cause and effect relationships between them, which we can describe as fuzzy rules. Then, we use fuzzy logic to argue, assessment of soft constraints and define a proper function for each soft constraint.

According to the analyzed model of the problem, in the second part, we have to maximize three functions F_k^2, F^4, F_q^5 corresponding to soft constraints S2, S4 and S5 as high as possible. However, in GA, it is usual converted to minimize these functions. We give a function to minimize as goal of problem as following:

$$F^S = w_2 \cdot \left(\frac{1}{N_L} \sum_{k=1}^{n_L} (1 - F_k^2) \right) + w_3 \cdot (1 - F^4) + w_4 \cdot \left(\frac{1}{N_S} \sum_{q=1}^{n_S} (1 - F_q^5) \right)$$

which, the values w_1, w_2, w_3 to weight the components of the objective function. To set the penalty coefficient in the objective function for evaluating the degree of constraint violation hard constraints, we use the following function:

$$F^H = \frac{1}{1 + c(H_1) + c(H_2) + c(H_4)}$$

$c(.)$ is a function to count the number of violations of hard constraints H1, H2, H4 and it is calculated as follows:

$$c(H_1) = \left\| \{ (z_{i_1,l,j} \times z_{i_2,l,j}) = 1 : i_1 \neq i_2 \} \right\| + \left\| \{ (w_{i_1,k,l} \times w_{i_2,k,l}) = 1 : i_1 \neq i_2 \} \right\|,$$

$$c(H_2) = \left\| \{ (CR_{i,j} = 0, z_{i,l,j} = 1) \} \right\|,$$

$$c(H_4) = \|\{(LT_{k,l} = 0, w_{i,k,l} = 1)\}\|,$$

Finally, we build a fitness function of individual corresponding to the gene sequence weighted by sum of weighting of F^H and F^S

$$\begin{aligned} fitness = & w_1 \cdot \left(1 - \frac{1}{1 + c(H_1) + c(H_2) + c(H_4)}\right) + w_2 \cdot \left(\tfrac{1}{N_L} \sum_{k=1}^{n_L} (1 - F_k^2)\right) \\ & + w_3 \cdot (1 - F^4) + w_4 \cdot \left(\tfrac{1}{N_S} \sum_{q=1}^{n_S} (1 - F_q^5)\right). \end{aligned}$$

This fitness is a smooth function, so it can give well evolution in GA. So, the hard constraints can be met and fulfilled by setting the weight of F^H much greater than the other (F^S), then soft constraints can be satisfied after that.

3.3. The genetic operations

In this article, we use the genetic operations are integrated temperature T_k as an additional parameter (where k is the index of current generation), called a temperature genetic algorithm [19]. The probability for the selection and mutation operations are changing through each generation, they are scaled to the temperature parameters of current generation. The initial temperature parameter is calculated based on the number of generations (often quite large to ensure the diversity of the population). After each generation, the temperature parameters will be descending to ensure the convergence and stability of algorithms.

We have to calculate genetic operations for generate new population. As in [5], the selection operation is nonlinear ranking of exponential function, the individuals can be sorted by decreasing order of its fitness values, individual i (ranked i^{th}) will be selected in populations parents as the probability is calculated based on the current temperature parameters of the evolution. In crossover operation, randomly selects one of three evenly distributed as a cutoff point crossover, linear crossover and extended linear crossover. The mutation and reproduction operations are calculated as in [5] to create new generation for evolution. The genetic operations are as follows:

(a) Selection: we use the exponential nonlinear ranking for selection, individuals of a population are ranked in decreasing order of Fitness, the i^{th} ranked individual will be selected for parent with the following probabilities:

$p = ((1\text{-}a).a^{-i})/(aN_{pop}\text{-}1)$,
$a = 1+\gamma(T_k)/N_{pop}$,
$\gamma(\mathrm{T}_k) = 1+(\gamma_{\max} - 1).(ln(T)\text{-}ln(T_k))/(ln(T)\text{-}ln(T_{end}))$,

where, N_{pop} is the number of individuals in population, $T_k = T_0$, α^k is the current temperateness of the present generation k ($k = 1,...,G_{\max}$), this parameter decreases from the initial temperature T to $T_{end} = T_0.\alpha^{G_{\max}}$ ($0 < \alpha < 1$, usually chosen $\alpha = 0.7$), $G_{\max}$ is the number of evolutionary generations. The function $\gamma(T_k)$ increases linearly by the number of generations that have evolved from 1 to $\gamma_{\max}$ (usually chosen $\gamma_{\max} = 0.9$).

(b) Crossover: uniform randomly selecting one of three following crossover operations:
(b.1) Single point cross over: for an individual X, $X|_i$ denotes the first sequence of gens on X taking into account the position i and $_i|X$ is the rest of gene sequence on X (from $(i{+}1)^{th}$ gen to the end). Given two parents X and Y, an uniform randomly selected i^{th} position on

two parents, since two off springs U and V were generated by permutation on two parts of X and Y as followed:

$$U = X|_i \oplus_i |Y,$$
$$V = Y|_i \oplus_i |X,$$

where, $\oplus$ denotes concatenate two gen sequences.

(b.2) Linear arithmetic crossover: randomly selected $a \quad \in$(0,1), two off spring U and V are generated from two parents X and Y as follows:

$$U[j] = a.X[j] + (1-a).Y[j],$$
$$V[j] = (1-a).X[j] + a.Y[j],\ j = 1, ..., L_{idv}$$

where $X[j]$ is the j^{th} gen of X, L_{idv} is the length of gen sequences. For convenience, we write in the form $U = a.X + (1-a).Y$ and $V = (1-a).X + a.Y$.

(b.3) Extended linear arithmetic crossover: uniform randomly selected a position i^{th} then it divides each gene sequence of the parent X and Y into two parts, applying the linear arithmetic crossover (b.2) on each of the subgroups as follows:

$$U|_i = a.X|_i + (1-a).Y|_i \text{ and } {}_i|U = (1-a).{}_i|X + a_i|Y,$$
$$V|_i = (1-a).X|_i + a.Y|_i \text{ and } {}_i|V = a_i|X + (1-a).{}_i|Y,$$
$$U = U|_i \oplus_i |U \text{ and } V = V|_i \oplus_i |V.$$

where, a is uniform randomly selected as in (b.2).

(c) Mutation: suppose a gene whose value denoted by $X[i]$ is inlimited range $[L_c,\ L_d]$ is chosen for mutation, then the new value of mutated gen is calculated by

$$X[i]' = X[i] + z.(X[i] - L_c) \text{ if } u < 0.5,$$
$$= X[i] + z.(L_d - X[i]) \text{ if } u \geq 0.5,$$

where u randomly selected in $[0, 1]$, and

$$z = \text{sign}(u - 0.5).T_k((1 + 1/T_k)^{|2u-1|} - 1),\ T_k \text{ is the temperature of the } k^{th} \text{ generation.}$$

(d) Replacement: is the method of parent substitution by offspring, each offspring will compete with the best parent. Call $g_{p1},\ g_{p2},\ g_c$ the fitness values of the parents and a child, $g^* = \max\{g_{p1},\ g_{p2}\}$, then the child is accepted with probability $p = \min\{1,\ e^{-(gc-g*)/Tk}\}$. In the case of unacceptable child, the parent corresponding to g^* is accepted for the next generation.

The next section we develop a computer program for the proposed approach, tested on a data set of examples and real data sets in the Faculty of Information Technology - Hanoi Open University.

4. EXPERIMENTS RESULTS

The computer program is built in C/C++, compiled on Windows 8 platform. The configuration of computer to run experiments on PCs with processor speed of 3.3GHz, 4G RAMS.

4.1. Experimenting with sample problems

Data of the sample problem is given as follows:

Table 2. The parameters of the test sample data

Parameters	*Meaning*
$n_C = 5$	Three courses $\{C_1, C_2, C_3\}$
$n_L = 2$	Two lectures $\{L_1, L_2\}$
$n_R = 2$	Two classrooms $\{R_1, R_2\}$
$n_T = 3$	Three time-slots $\{T_1, T_2, T_3\}$
$n_S = 20$	Twenty students for registering

As proposed, the teachers are predefined to teach courses such as Table 3. This will remove the checking constraint H1 and reduce the search space of GA optimization, leading to speed up the convergence of the algorithm.

Table 3. Determining teachers to the courses

Courses	C_1	C_2	C_3
Teachers	L_1	L_2	L_1

In this experiment, we leave out the $C \times C$ matrix, corresponding weighting components of fitness function is $w_3 = 0$, as the criteria of courses with prerequisite conditions that they may have same time-slots in the time table as much as possible. Since this constraint is also expressed in considering the goal of increasing student registration at S5 soft constraint. The remaining constraints are reflected in following matrices:

Table 4. Constraints between courses and classrooms ($C \times R$)

	C_1	C_2	C_3
R_1	1		1
R_2		1	

Table 5. Constraints between teachers and time-slots ($C \times T$)

	T_1	T_2	T_3
L_1	low	high	normal
L_2	normal	high	

Table 6. Constraints between students and courses ($S \times C$)

Students	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
C_1		1		1		1		1		1			1	1			1	1		1
C_2	1	1						1	1	1	1			1			1	1		1
C_3			1					1				1	1			1	1	1	1	1

The experiment running with the proposed method, we manually choose GA parameters based on experiments as the following table:

Table 7. Parameters for running experiments

Parameters	Values
α $-$descending temperature of scaling	0.7
$\gamma_{\max}$ - maximum temperature factor	9
N_{pop} - population size	50
$G_{\max}$ - number of generations	10
p_c - crossover probability	0.9
p_m - mutation probability	0.1
w_1	0.9
w_2	0.
w_3	0.
w_4	0.1

In this experiment, we do not use the soft constraints between teachers and time-slots, the soft constraints between courses and course by prerequisite conditions, corresponding w_2 and w_3 respectively. The results of experiment running with the given data is in the following timetable, in which all hard constraints are satisfied, the soft constraints are evaluated according to fitness function value reached 0.886667. Note that Table 5 shows constraints between teachers and time-slots ($L \times T$) that there is only one timetable which met these constraints (see below table).

Table 8. The timetable result of the experiment

Courses	C_1	C_2	C_3
Teachers	L_1	L_2	L_1
Time-slots	T_1	T_2	T_3
Classrooms	R_1	R_2	R_1

As above result, the possibility of registration of students can achieve is 29 out of its original capacity of 29 (reached 100%).

4.2. Experiment with real-world problems

Currently, in the Faculty of Information Technology - Hanoi Open University, we establish three semesters in a year with two main semesters and one optional semester. We have more than 1000 students. The number of subjects in curriculum is 42 and it has many of requisite constraints of subjects. We also use system management software for supporting the list of all students with ability enrolled in each subject at the beginning of a semester. We have 8 classrooms of theory with a capacity about 70 students per room, and also have 3 computer labs for practicing with holding 35 students per room, 2 halls holding 140 students. We use weekly time-slots, so it has 7 days for learning (both Saturday and Sunday), 3 time-slots per day (morning, afternoon and evening, except for Thursday night), so the total number of time-slots is 20.

The data in the experiment is for the first semester (HK1) of 2015-2016 school years. Based on the statistical capabilities of the 1044 student registration, the expected open courses are 86 corresponding 36 subjects. The constraints matrix is set up consisting of $C \times R$, $L \times T$, $S \times C$ (as too big to detail here). The teachers are predefined to courses depend on their expert by users. The parameters for running is same as Table 7, but some parameters are properly changed, such as $N_{pop} = 250$, $G_{\max} = 1000$, and weights of fitness function are $w_1 = 0.9$, $w_2 = 0.01$, $w_3 = 0$, $w_4 = 0.09$.

We run the experiment in three times. Result of the first time is showed in Table 12. In which, the symbol "S" is the morning, "C" is the afternoon of time-slots with suffixes indexed is day of week (from 2 to 7 corresponding Monday to Saturday respectively), teachers index, courses index and classrooms index are $\#L$, $\#C$ and $\#R$ column respectively. This timetable gives a total ability of all registering students is 4697 respectively, equivalent to 4.5 courses per student (approximately 14 credits per student), it is properly with minimum credits of regulations.

We use the result of first run for implementing in practice to students enrolled and results are 87.2% success of courses (i.e. the number of courses was canceled because of low student enrollment of 12.8%). This no success is depending on student's registration, we give students the suggestions for registration. Compared to the previous semesters, this no success of rate is quite low, such as in 2014-2015 school year, the no success rate of 3^{rd} semester (HK3) is 27.4%, 2^{nd} semester (HK2) is 18.95%, 1^{st} semester (HK1) is 14, 6% (see Figure 3). This shows that the timetable generated by our method with registration suggested can increase the capability of students registering, leading to reduction of canceled courses than before.

On the other hand, as mentioned, this is a random search algorithm, we run three times with the average time is 185 seconds per each run. The graph in Figure 4 shows the fitness value of the best individuals in each generation and the hard constraints are violated through evolution of the algorithm in Figure 5. In Figure 5, it also shows that we reach a timetable with no violate hard constraints at about 700^{th} generation of evolution. This assures the timetable of results which has no violations hard constraints.

5. CONCLUSION

In this article, we review the timetabling problem especially in the credits training at universities. We also study genetic algorithms as a useful tool in soft computing to solve complex problems with many constraints and multi-objective. Therefore, the paper has proposed methods for solving timetabling problem in credits training at universities based on genetic algorithms. Some of the constraints of the problem have been considered for using the fuzzy parameters of hedge algebras to meet the actual requirements.

The results of experiments in practice at the Faculty of Information Technology - Hanoi Open University show the effectiveness of the method. That's the running time of the

Table 12. Timetable of 1st experiment running

#C	#L	#T	#R
0	9	S3	B31
1	12	C6	B31
2	10	S4	B31
3	5	S6	B32
4	12	C3	B31
5	12	C5	B31
6	1	C5	B42
7	16	S2	B44
8	1	C2	B41
9	16	S3	B22
10	21	S4	B21
11	17	S4	B24
12	4	C4	B42
13	2	S3	B24
14	3	S4	B44
15	0	S3	B44
16	10	C4	B24
17	4	C3	B24
18	15	C6	B51
19	15	C5	B51
20	15	C3	B52
21	7	C3	B44
22	14	S6	B43
23	14	S5	B41
24	19	S5	B24
25	7	S7	B44
26	14	C3	B41
27	1	S7	B22
28	27	S4	B51
29	26	S6	B51
30	23	S5	B52
31	14	C5	B24
32	7	C6	B41
33	0	S6	B23
34	10	C6	B43
35	24	S6	B42
36	2	C5	B23
37	2	C2	B42
38	7	S2	B43
39	7	S3	B21
40	19	S4	B22
41	0	C3	B51
42	9	S7	B41
43	8	S6	B44
44	1	S5	B44
45	17	C5	B44
46	17	C4	B41
47	4	S3	B23
48	9	C6	B22
49	5	C2	B24
50	6	S4	B43
51	1	C7	B44
52	20	C7	B22
53	1	C4	B44
54	4	C5	B43
55	6	C4	B43
56	5	S3	B43
57	5	C4	B23
58	0	C2	B43
59	0	S5	B42
60	0	S4	B42
61	8	S3	B42
62	9	S6	B41
63	6	C6	B52
64	18	C5	B22
65	11	S2	B52
66	2	C3	B42
67	13	S7	B24
68	10	C2	B44
69	5	C6	B44
70	3	S6	B24
71	18	C3	B22
72	3	C5	B41
73	4	C6	B24
74	8	C6	B42
75	8	C5	B52
76	25	C4	B51
77	3	C2	B23
78	13	S4	B41
79	2	S6	B21
80	11	C4	B52
81	22	S3	B51
82	3	C3	B43
83	11	S5	B23
84	6	S5	B51
85	6	S2	B51

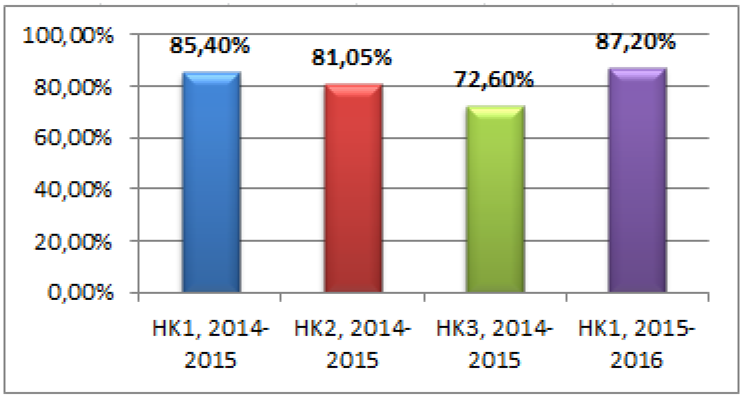

Figure 3. The successful rate of courses in the timetable given by proposed method

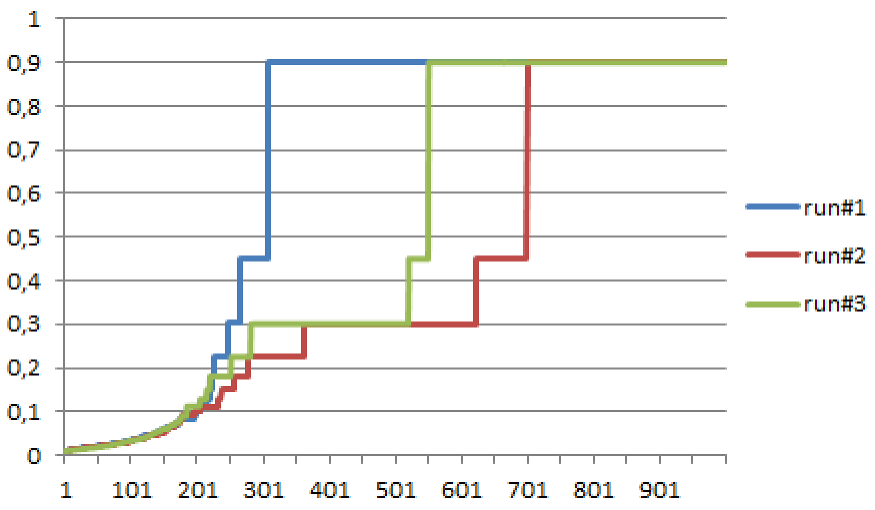

Figure 4. The changing fitness of individuals through evolutionary time

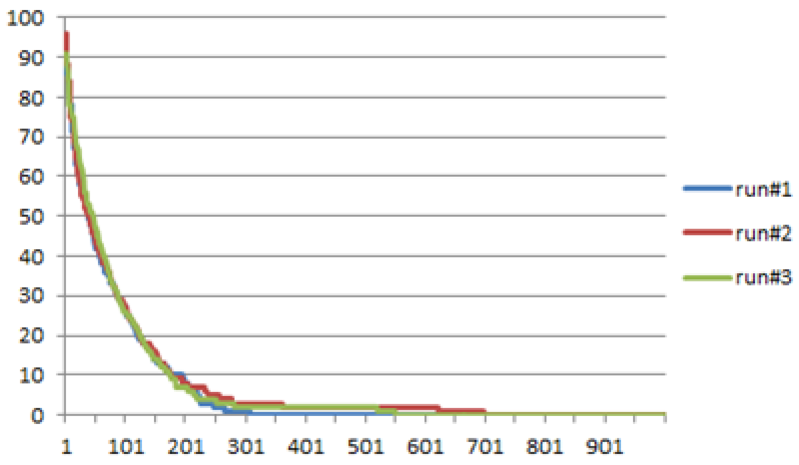

Figure 5. Hard constraints violations through evolutionary time

algorithm is quite fast, compared to the traditional solution, we have to change a lot of times to obtain the final results of timetable, by applying this results of our method, we have a stable timetable and put into student for registration. Moreover, the result shows that the timetable gives opportunity for students to register well, besides the reasonable suggestions are output from the algorithm. Compared to the prior semesters, the rate of canceled courses is lower. These demonstrated the potential effectiveness of the algorithm in practical application.

The method proposed in the paper can be extended to apply for practicing in many situations. However, the hard and soft constraints may put more different assessments to show the suitable of each assessment, especially based on fuzzy parameters with more suitable for the purpose of actual use, thereby potentially increasing the efficiency up. These will be studied further and announced in the next research.

REFERENCES

[1] M. Abbaszadeh and S. Saeedvand, "A fast genetic algorithm for solving university scheduling problem," *IAES International Journal of Artificial Intelligence*, vol. 3, no. 1, p. 7, 2014.

[2] Z. Bratković, T. Herman, V. Omrčen, M. Čupić, and D. Jakobović, "University course timetabling with genetic algorithm: A laboratory excercises case study," in *European Conference on Evolutionary Computation in Combinatorial Optimization.* Springer, 2009, pp. 240–251.

[3] R.-M. Chen and H.-F. Shih, "Solving university course timetabling problems using constriction particle swarm optimization with local search," *Algorithms*, vol. 6, no. 2, pp. 227–244, 2013.

[4] V. Kolonias, G. Goulas, C. Gogos, P. Alefragis, and E. Housos, "Solving the examination timetabling problem in gpus," *Algorithms*, vol. 7, no. 3, pp. 295–327, 2014.

[5] R. Malhotra, N. Singh, and Y. Singh, "Genetic algorithms: Concepts, design for optimization of process controllers," *Computer and Information Science*, vol. 4, no. 2, p. 39, 2011.

[6] A. O. Modupe, O. E. Olusayo, and O. S. Olatunde, "Development of a university lecture timetable using modified genetic algorithms approach," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 9, pp. 163–168, 2014.

[7] R. Perzina and J. Ramik, "Self-learning genetic algorithm for a timetabling problem with fuzzy constraints," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 11, pp. 4565–4582, 2013.

[8] ——, "Timetabling problem with fuzzy constraints: a self-learning genetic algorithm," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 3, no. 4, pp. 105–113, 2013.

[9] B. Sigl, M. Golub, and V. Mornar, "Solving timetable scheduling problem using genetic algorithms," in *Information Technology Interfaces, 2003. ITI 2003. Proceedings of the 25th International Conference on.* IEEE, 2003, pp. 519–524.

Received on March 28 - 2016
Revised on May 04 - 2017