Vietnam Journal of Science and Technology ***57*** *(6) (2019) 734-748*

doi:10.15625/2525-2518/57/6/13501

AN ENHANCED GENATIC ALGORITHM BASED COURSES TIMETABLING METHOD FOR MAXIMAL ENROLLMENTS USING MAXIMUM MATCHING ON BIPARTITE GRAPHS

Duong Thang Long

Hanoi Open University, B101 Nguyen Hien street, Hai Ba Trung district, Ha Noi

Email: *duongthanglong@hou.edu.vn*

Received: 24 December 2018; Accepted for publication: 8 October 2019

Abstract. Universities usually use academic credit systems for holding all training courses. They have to establish a suitable timetable for enrollment by students at beginning of every semester. This timetable must be met to all hard constraints and it is satisfied to soft constraints as high as possible. In some universities, students can enroll to the established timetable so that among of their courses is as much as possible. This leads to finish their studying program earlier than normally cases. In addition, this also leads to well-utilized resources such as facilities, teachers and so forth in universities. However, a timetable usually has so many courses and some its courses have same subjects but different time-slots. These may cause difficulties for manually enrolling by students. It may be fall into conflict of time when choosing two courses at same time-slots. It is difficult for enrollment with high satisfied. In this paper, we design a genetic algorithm based method for university timetable with maximal enrollments by using maximum matching on bipartite graphs.

Keywords: university timetables, genetic algorithm, bipartite graph, maximum matching.

Classification numbers: 4.10.2, 5.6.2.

1. INTRODUCTION

University timetabling is typical scheduling problem and it is also a classical problem. Universities usually use academic credit systems for holding all training courses. They have to establish a suitable timetable for enrolling by students at beginning of every semester. However, this problem has many complicated factors. They may be capable teaching and time inquired of teachers, a lot students, many kinds of classrooms and subjects, and especially major constraints within these elements. This problem also includes many relevant factors which should be considered such as examinations, practice, lecture halls, etc. Authors in [1-5] show that the timetabling problem is a kind of NP-hard. Typically, timetabling problems are conducted in traditional ways by intuitive and direct calculation of human. Currently, due to diversities and many relations between elements, this problem often takes a lot of time and labor. Using

computers for dealing with this problem is not only much interesting to researchers, but also allowance achieving superior results despite many more constraints. Obviously, this leads to save a lot of time and effort.

Solving methods of this problem have been researched by many authors. In [6], authors pointed out that, Hertz proposed using Tabu search with including two stages (TATI / TAG) and it is an appropriate method for scheduling problems with large-scale implementation. Nothegger [7] suggests ant colony optimization (ACO) for solving this problem. Tassopoulos and Beligiannis use swarm optimization to establish a timetable for various schools in Greece. Al. Betar et al. [7] propose a hybrid method (HHS) to solve scheduling problems for universities. HHS is an integrated algorithm with optimization and climbing hills swarm to balance space exploration and searching.

Due to efficiencies of genetic algorithms (GAs) [8], a lot authors use GAs for timetable problems to improve performance of traditional methods [1-7,9-11]. Authors in [3] indicated that GAs can be used as a properly universal method for complicated optimization problems, which they almost have no deterministic solution. Enhanced GA based methods can achieve high performance by adjusting genetic operations. Authors can use an alternative strategy in order to avoid falling into local optimum. In facts, M. Abbaszadeh in [3] used GA with changed structure of performing gene sequence which allows transferring 15% of better individuals to next generation in mutation operators. Moreover, to avoid falling into local optimum, they considered impact of their parameters to mutations. They also removed repetitive genes and replaced by better gene sequences. Results of this method are high performance and maximum accuracy. Authors in [11] proposed a GA with binding elements of this problem to adapt practical constraints such as requirements of faculties for time, teaching expertise. They use fuzziness measurements in some genetic operations. In [2], we used hedge algebras based fuzziness measure for presenting school time of teachers. In [12], we also adjusted some genetic operations such as selection, crossover, mutation and replacement by using the temperature factor in simulated annealing. This has achievements of improving performance.

However, the above authors mainly focus on how to improve efficiency of solutions with having no violation of hard constraints and maximal satisfied soft constraints. They do not consider resulted timetable which it gives the best case of enrollment for every student. There are two things that can be treated well. Firstly, how to generate a good timetable so that students can have more opportunities of enrollment. Students can enroll as many courses as possible. Secondly, once a timetable is generated, how can students enroll suitable courses by their self so that they enroll courses as many as possible. In this paper, we propose enhanced GA based method for timetabling problems with maximal capability of enrollment. We use maximum matching on bipartite graphs to get maximal enrollment of every student. This article consists of 5 sections, Part 1 is introductions to universities timetabling problems. Part 2 is detail of genetic algorithms based method for this problem. Part 3 proposes an enhanced GA based method with using maximum matching on bipartite graphs. Part 4 is about computer program and testing on real data in the Hanoi Open University. The final section is conclusion.

2. GENETIC ALGORITHS BASED METHOD FOR TIMETABLING PROBLEMS

In general, course-based timetabling problems (CTP) consists of assigning appropriate time-slots, teachers and rooms to all given courses. This is done to ensure that all hard constraints are met and take satisfactions of soft constraints as high as possible. However, in academic credit training systems, it depends on characteristics of each university. CTP will be

deployed with certain differences. Some universities let students enroll subjects firstly. Then they use these enrollments as parameters of the problem. They divide students into each class and fix them in every class. These classes are treated as resources and we assign classes to courses. So, all students of each class will be assigned to such courses. In this case, students cannot enroll for any further courses and they are always fixed in a class. This is not flexible for students and it does not give many choices for them such as time-slots, lecturers. In this paper, we use resources including time-slots, teachers and classrooms. This would be appropriate to the case that a timetable will be implemented before students enroll.

Steps for solving CTP can be described as follows: At the beginning of a semester, from learnable subjects of students, we propose all possible courses. Then we establish a timetable by assigning resources (time-slots, teachers and rooms) to every course. We announce generated timetable to students for enrolling. So, CTP problem consists of assigning teachers, time-slots, rooms to courses so that all hard constraints (H) must be met and soft constraints (S) can be satisfied as much as possible.

In universities, hard and soft constraints are composed of various factors. They can be following requirements:

(H1) Each teacher or room is not assigned to more than one course at a time-slot.

(H2) Rooms must be assigned to appropriate courses. A practical room cannot be assigned to a theory course and vice versa, a hall room should not be assigned to small courses, etc.

(H3) Each teacher must be assigned to courses so that he or she has sufficient knowledge and capability of teaching for courses.

(H4) Teachers must be assigned to courses with time-slots so that they are present at the school. Each teacher has a list of time-slots for presenting at school.

(S1) Teachers are assigned to courses so that their expertise of courses is as high as possible.

(S2) Teachers are priority assigned to theirs expected time-slots as high as possible.

(S3) It should be to balance number of courses for every teacher, i.e., the minimum and maximum number of courses of every teacher should be taken.

(S4) It should be given priority courses with prerequisite of a subject to same time-slot. This aims to increase abilities of enrollments on the timetable.

(S5) Abilities of enrollment for every student on the timetable is as high as possible.

In this paper, we assume that it is not necessary to design curriculums with fixed mandatory subjects of every semester. Instead, we design a diagram of pre-requisite subjects for curriculums. When students want to choose subjects for learning in a semester, he or she must be passed all pre-requisite subjects belonging to the chosen subjects. For this case, soft constraints S4 and S5 have significant meanings. Soft constraint S4 means the more courses of same time-slot, the more chance of enrollment for students. However, these courses must be together pre-requisite. For S5 constraint, when students choose more subjects for learning then they can early graduated. In addition, if we reach high satisfied S5 then we many students at school in a semester. This means that school financing will be increasing, facilities are much used, etc.

Depending on particular academic credit training systems of a university, soft constraints can be adjusted some parameters for suitable reality. CTP can be formalized as an optimization problem model and we now describe its input data. In this model, we use following symbols:

- $\{C_1, C_2, \ldots, C_{n_C}\}$ denotes set of courses, n_C is number of courses;
- $\{L_1, L_2, \ldots, L_{n_L}\}$ denotes set of teachers (lectures), n_L is number of teachers;
- $\{R_1, R_2, \ldots, R_{n_R}\}$ denotes set of rooms, n_R is number of rooms;
- $\{T_1, T_2, \ldots, T_{n_T}\}$ denotes set of time-slots, n_T is number of time-slots;
- $\{S_1, S_2, \ldots, S_{n_S}\}$ denotes set of students, n_S is number of students.

Normally, in universities, CTP should have weekly time-slots. A day of week can be divided into two sessions such as morning and afternoon. We assume that there are 6 days of a week from Monday to Saturday, then we have 12 time-slots of a week. However, we can also divide a day into many periods of time and weekly time-slots can be more than 12.

In [2], we analysed these constraints in details, then H3, S1 and S3 constraints can be easily met by manually assigning teachers to every course based on experts. H1 constraint can be only obtained during progress of CTP by checking this constraint on a timetable. The remaining constraints are represented by matrices as the following:

(H2) $C \times R = \{CR_{i,k} \mid i = \overline{1, n_C}; k = \overline{1, n_R}\}$ defines constraints between rooms and courses. The value of this matrix is {0,1}, $CR_{i,k} = 1$ implies that R_k room can be assigned to C_i course and 0 is not.

(H4, S2) $L \times T = \{LT_{j,l} \mid j = \overline{1, n_L}; l = \overline{1, n_T}\}$ defines constraints between teachers and time-slots. We integrate H4 and S2 constraints. In which, H4 is strictly priority of time-slots for assigning with teachers and S2 is as high as possible. Therefore, this matrix will receive values in the form of language. For example, NO, NORMAL, GOOD, VERY GOOD, etc. NO value denotes a teacher being not present at schools during at that time-slot.

(S4) $C \times C = \{CC_{i_1,i_2} \mid i_1 = \overline{1, n_C};\ i_2 = \overline{1, n_C}\}$ describes prerequisite subjects between courses. It will receive binary values, 0 if there are not prerequisite of two subjects or 1 if there is a prerequisite subject of another. Courses with prerequisite subjects should be assigned at the same time-slots in order to increase ability of enrollment.

(S5) $S \times C = \{SC_{u,i} \mid u = \overline{1, n_S};\ i = \overline{1, n_C}\}$ describes constraints between students and courses. It receives binary values, 1 if student S_u can learn course C_i and 0 is not.

Now we represent a timetable as the following Table 1. For this table, columns are courses and rows are lecturers, time-slots and rooms, correspondingly.

Table 1. Representation of the timetable.

Courses	C_1	C_2	...	C_i		C_{n_C}
Lecturers	L_{j1}	L_{j2}	…	L_{ji}		L_{jn_C}
Rooms	R_{k1}	R_{k2}	…	R_{ki}		R_{kn_C}
Time-slots	T_{l1}	T_{l2}	…	T_{li}		T_{ln_C}

In this table, as above mentioned, we assign lecturers to every course in order to certainly

satisfy requirements of H3 constraint. This is also to balance number of courses for every teacher. However, a course can only continue running if number of its enrolled students is enough large. This is a dynamic factor, so universities should take somehow to ensure that the number of deployed courses is as high as possible.

Underlying this, CTP could be condensed in a shorten form. We just need assigning time-slots and rooms to courses on a timetable. In [2], we use linguistic terms of hedge algebras for $L \times T$ matrix values. If x denotes a term, then semantic quantitative function of x - $\upsilon(x)$ can be the following triangle for satisfying measurement of H4 and S2 constraints. For example, Figure 1 describes satisfying measurement of three time-slots for a teacher.

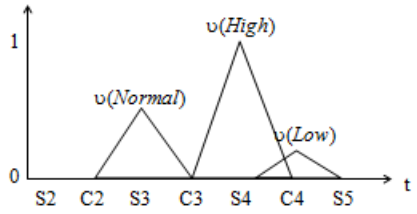

Figure 1. Selection of priority-time-slots of a teacher.

In [2], for S2, we set F_k^2 being total time-satisfaction measure of L_k teacher by evaluating satisfaction of assigned time-slots. S4 can be easily obtained by counting number of same time-slot assigned courses which they are together prerequisite. We set F^4 being totally this counting for measurement of S4 constraint. For S5, we evaluate it based on capability of enrollments according to $S \times C$ matrix. Once a timetable is generated, each student S_q can enroll some courses for learning. Therefore, we can automatically build an enrollment solution for every student from $S \times C$ matrix and generated timetable. This solution can be obtained by applying optimal method which its objective is satisfaction of S5 as high as possible. However, this is a quite difficult sub-problem because of many factors and relations between elements. We will describe details in next section. From here, we just denote F_q^5 being satisfaction of enrollments. It is now can be stated a model of CTP as a formalization of multi-objectives 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,

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

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

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

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

which $z_{i,l,j}$ is a binary variable for defining course C_i be assigned to timeslot T_l and classrooms R_j if its value is 1 or otherwise. ω(.) is a function to determine values of LT matrix, it may be NO (ω = 0) or other linguistic terms (ω = 1). $w_{i,k,l}$ is a binary variable for determining course C_i be assigned to teacher L_k and timeslot T_l or not?

In [2], we proposed a genetic algorithm based method for solving CTP. This method uses temperature factors from simulated annealing (SA) as parameters for increasing convergence of the algorithm, and avoiding fall into local optimum as well.

a) Chromosome encoding

In general, for applying GA, we have to encode problem solutions into an appropriate gene sequence. By using directly encoding method, each gen of chromosome represents a parameter of solutions as a real number. For a timetable as in Table 1, as mentioned above, we manually assign teachers to every course by user expertise. Then, it need to encode parameters of rooms and time-slots, thus, a chromosome has $2n_C$ length of gens as in Figure 2.

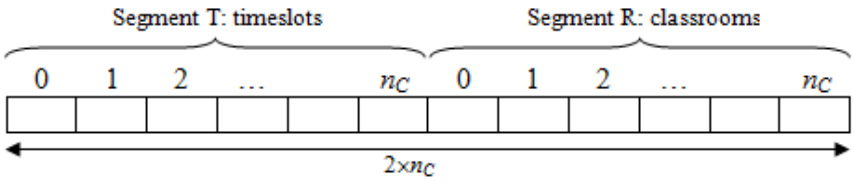

Figure 2. Chromosome of gene encoding of solution.

Value of each gene (g_i) is real number in [0,1], thereby we determine value of real domain of time-slots and rooms as well, by the following functions (f^T and f^R):

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

in which, symbol $\lceil . \rceil$ is the nearest upper integer of values.

b) Fitness function designing

We design a fitness function by integrating hard constraints and soft constraints. In directly encoding, we use penalty coefficients in fitness function to avoid violations hard constraints and get high satisfaction of soft constraints. Objectives of problems can be converted to minimization. In this case, we divide fitness function into two parts of soft and hard constraints. First part is satisfying measurement of soft constraints, it can be formulated as the following function:

$$F^S = w_2 . \left(\frac{1}{n_L} \sum_{k=1}^{n_L} \left(1 - F_k^2\right) \right) + w_3 . (F^4)^{-1} + w_4 . (F^5)^{-1}$$

where, w_1, w_2 and w_3 are weights of soft constraints in objective function.

Second part is measurement of violation hard constraints, it is can be formulated as the following function:

$$F^H = w_1 . \left(1 - \frac{1}{1 + c(H_1) + c(H_2) + c(H_4)}\right)$$

where, c(.) is a function of counting violations number of hard constraints H1, H2, H4.

The fitness function is weighted sum of F^H and F^S as the following:

$$fitness = F^H + F^S$$

In this case, hard constraints can be met and fulfilled by setting weight of F^H is much greater than others (F^S). It means that soft constraints may be satisfied after eliminating violations of hard constraints.

c) Genetic operators implementation

We use genetic operations with integrated temperature T_k as an additional parameter (where k is the index of current generation). The probability for selection and mutation operations is changed through each generation by applying this temperature [2]. Genetic operations of selection, crossover, mutation and replacement are designed in [12] for generating new chromosomes during evolution.

In facts, the better timetable, the more enrollments of students. In order to get a good timetable, beside partly F^2 and F^4 of fitness function, we should consider details of F^5 because it causes the satisfaction of enrollments on a timetable. Thus, in next section, we apply maximum matching on bipartite graph for solving sub-problem of maximum enrollment.

3. MAXIMIZE ENROLLMENTS USING MAXIMUM MATCHING ON BIPARTITE GRAPH

In objectives of CTP, we have to maximize enrollments on a timetable. In [2], our method could reach high results of these objectives during evolution. However, there was an optimal sub-problem in the last objective of CTP. It was not solved in our method because of quite difficult. In facts, once a timetable is generated, every student enrolls courses based on subjects that he or she can learn. This sub-problem becomes determining maximal courses that each student can enroll.

In general, at the beginning of a semester, we get all subjects which can be enrolled for every student. For each subject, we get number of students which can enroll for learning. Then, we propose all possible courses of every subject and put them into Table 3 as the following:

Table 2. Proposed courses with corresponding subjects.

Courses	C_1	C_2	...	C_i		C_{n_C}
Subjects	Sub_1	Sub_2	…	Sub_i		Sub_{n_C}

where, a subject can belong to more than one course, i.e. $C_i \neq C_j \wedge Sub_i = Sub_j$. Students want to learn a subject, they can enroll a course belonging to this subject. Constraints between students and courses are represented in $S \times C$ matrix. Since E_q^5 denotes number of enrolled courses by a student S_q, we have to generate a timetable so that every student can get maximum E_q^5.

For each student S_q, we build a bipartite graph which left side of vertices are subjects and right ones are time-slots (Figure 3). Firstly, we determine subjects that student S_q can learn based on generated timetable. Then, each of these subjects is connected to time-slots which they

are assigned to courses of this subject. These connections are edges of the bipartite graph as the following picture:

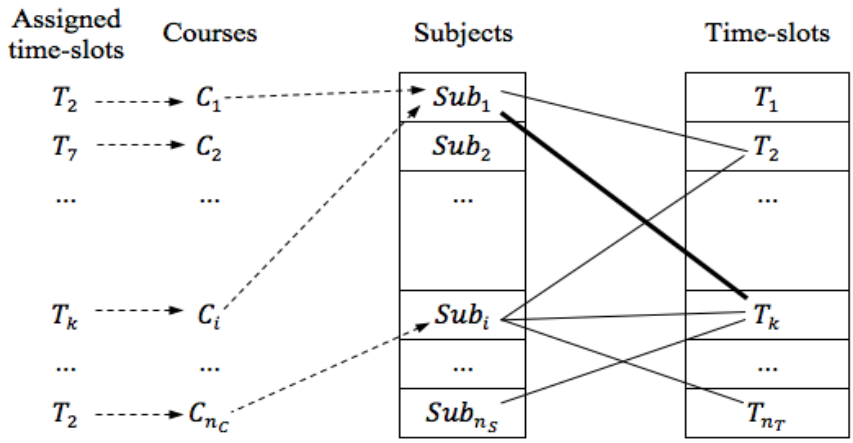

Figure 3. Bipartite graph for enrolling of a student.

A time-slot may be assigned to more than one course in a timetable, so we have connections between a time-slot and many subjects. Furthermore, a subject may belong to more than one course and a time-slot is assigned to one course, we also have connections between a subject and many time-slots. For example, in Figure 3, time-slot T_k is connected to three subjects of Sub_1, Sub_i and Sub_{n_S} or Sub_1 is connected to two time-slots of T_2 and T_k because of subject Sub_1 belongs to two courses of C_1 and C_i.

Based on this bipartite graph, enrollments of a student can be reached by finding a set of edges which each edge indicates an enrollment of a subject and the corresponding time-slot. However, these edges have no common vertices, i.e. a student can only enroll a subject and a time-slot at once in a semester. For example, in Figure 3, if a student enrolls subject Sub_1 with corresponding T_k (the bold line) then he or she cannot enroll Sub_1 and T_k any more. This suggests that we can apply maximum matching methods on bipartite graphs to solve this sub-problem, the more edges we find, the more subjects that student S_q can enroll for learning.

We now can let F_q^5 be number of possible enrolled subjects by student S_q. With maximum matching of every student, we can get total satisfactions of enrollments for all students as the following:

$$F^5 = \sum_{q=1}^{n_S} F_q^5.$$

In maximum bipartite matching, a set of edges with no common vertices is called a matching - M, thus this problem become finding M with maximum cardinality. We apply Hopcroft-Karp algorithm for this problem [#1]. A vertex has two statuses of matched and unmatched, and an edge is also matched or unmatched. It is said that an alternating path of M is a path of a graph so that it has interleaved matched edge and unmatched edge in turn of M. We also call an augmenting path with respect to M if it is an alternating path of M with two endmost unmatched edges. For example, in Figure 4, the red edges are matched, we have three alternating paths but there is only one augmenting path at the middle (4.b) of this Figure.

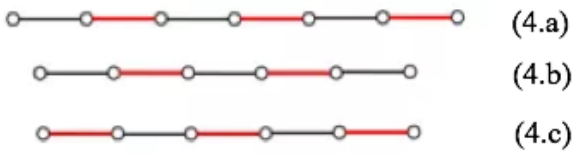

Figure 4. Alternating paths and augmenting paths.

It is known that If M is a matching and P is an augmenting path with respect to M, then $M \oplus P$ is a matching containing one more edge than M. In which, we use notation $A \oplus B$ to denote symmetric difference of two sets A and B, i.e. a set of all elements so that each element belongs to only one of two sets. For example, the augmenting path (4.b) with respect to M (two red edges), we can get new one $M \oplus P$ with one more edge than M (on the right of Figure 5).

Figure 5. The new augmenting path with one more edge.

In addition, a matching M in a graph G is a maximum cardinality matching if and only if it has no augmenting path.

The Hopcroft-Karp algorithm is based on that each time for searching augmenting paths, instead of finding an augmenting path, we find a blocking set of augmenting paths with respect to M which called $\{P_1, P_2, \dots, P_k\}$ so that they are vertex-disjoint. Then we extend M by applying $M \oplus P_1 \oplus P_2 \dots \oplus P_k$. This procedure is repeated until no augmenting path exists.

Next section, we develop a computer program for our proposed method, then it is tested on an examples and real data sets in Faculty of Information Technology - Hanoi Open University.

4. COMPUTATIONAL EXPERIMENTS

4.1. Experiment with sample problems

We assume that a generated timetable in the following Table 3 with 6 courses, 3 subjects, 4 teachers, 3 time-slots, 3 rooms and 3 students.

Table 3. The sample timetable with corresponding subjects.

Subjects	Sub_1	Sub_2	Sub_1	Sub_1	Sub_3	Sub_2
Courses	C_1	C_2	C_3	C_4	C_5	C_6
Teachers	L_1	L_2	L_1	L_3	L_4	L_2
Time-slots	T_1	T_1	T_2	T_3	T_1	T_2
Rooms	R_1	R_2	R_1	R_1	R_3	R_2

For illustrating maximum number of possible enrollments, we leave out all constraints matrix unless $S \times C$ matrix (Table 4). Three students S_1, S_2 and S_3 can learn $\{Sub_2, Sub_3\}$,

$\{Sub_1, Sub_2, Sub_3\}$ and $\{Sub_1, Sub_3\}$, respectively.

Table 4. Constraints between students and courses ($S \times C$).

Subjects	Sub_1	Sub_2	Sub_1	Sub_1	Sub_3	Sub_2
Courses / Students	C_1	C_2	C_3	C_4	C_5	C_6
S_1	0	1	0	0	1	1
S_2	1	1	1	1	1	1
S_3	1	0	1	1	1	0

We build three bipartite graphs as in Fig. 6, Fig. 7 and Fig. 8 for students S_1, S_2 and S_3, respectively.

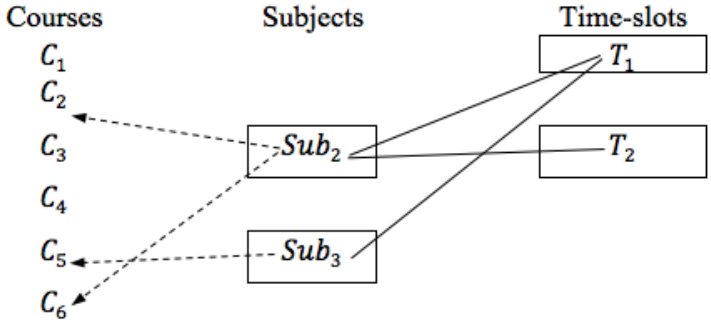

Figure 6. Bipartite graph for student S_1.

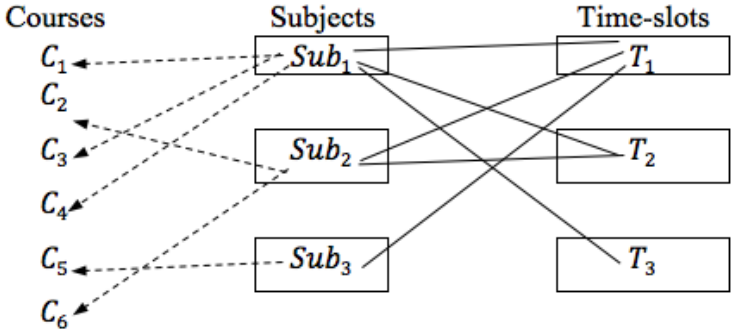

Figure 7. Bipartite graph for student S_2.

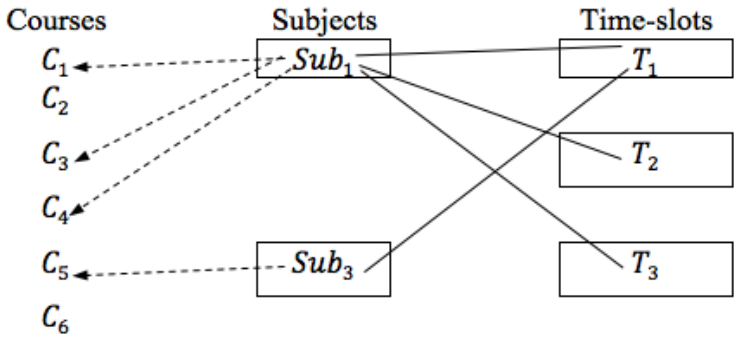

Figure 8. Bipartite graph for student S_3.

For maximum matching of these graphs, in this simple case, we manually determine

$$M_1 = \{(Sub_2, T_2), (Sub_3, T_1)\}, M_2 = \{(Sub_1, T_3), (Sub_2, T_2), (Sub_3, T_1)\}$$

and

$$M_3 = \{(Sub_1, T_2), (Sub_3, T_1)\}$$

for students S_1, S_2 and S_3 respectively.

In facts, in Fig. 6, if we choose an edge of (Sub_2, T_1) then there is no more edge can be chosen, so it is not a maximum matching for student S_1. Similarly, in Fig. 7 and Fig. 8, if we choose the edge of (Sub_1, T_1), then there is no more edge can be chosen for Sub_3 subject, in this case, we cannot reach to a maximum matching for student S_2 and S_3. In Table 5, we give all matching of each student, it shows that which one is maximum matching. The largest number of possible learning subjects for a student is number of edges in maximum matching. The maximum matching also indicates subjects and time-slots for enrollments of corresponding student.

Table 5. Matching with its size for each bipartite graph of students.

Fig. of student / Number of edges	Fig. 6 of S_1	Fig. 7 of S_2	Fig. 8 of S_3
1	(Sub_2, T_1)	(Sub_1, T_2)	(Sub_1, T_1)
2	(Sub_2, T_2) (Sub_3, T_1)	(Sub_1, T_1) (Sub_2, T_2)	(Sub_1, T_2) (Sub_3, T_1)
3		(Sub_1, T_3) (Sub_2, T_2) (Sub_3, T_1)	

4.2. Experiment with a real-world problem

In [2], we used a real-world dataset of Faculty of Information Technology - Hanoi Open University. The detail of this dataset is also showed in [2]. In this paper, we summarize dataset and parameters for running in Table 6. We establish an experiment running with plugged maximum bipartite matching into fitness function of GA for getting the largest number of possible learning subjects.

Table 6. Summary dataset of running experiments.

Name of parameters	Values
Number of lecturers	75
Number of time-slots	20
Number of rooms	15
Number of students	1044
Number of subjects	42
Number of courses	86
$\boldsymbol{\alpha}$ – Temperature decreasing factor in genetic operations	0.7
$\boldsymbol{\gamma}_{max}$ – Maximal temperature in genetic operations	9
N_{pop} – Population size	250
G_{max} – Maximal generation in evolution	1000
p_c – Crossover probability	0.9
p_m – Mutation probability	0.1
(w_1, w_2, w_3, w_4) – Weights of components in fitness	(0.9, 0.01, 0.01, 0.08)

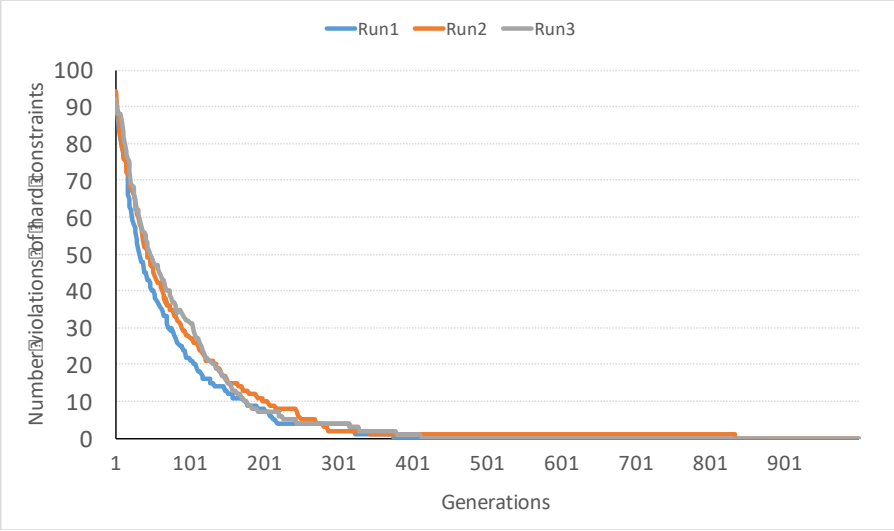

Figure 9. Number violations of hard constraints decreasing in generations.

By directly encoding, an individual is represented for a timetable. We use maximum bipartite matching for counting number of learnable subjects based on individuals. So, values of F^5 is evaluated for fitness function. Then, we compute the total number of subjects for all students and get maximum, minimum and average of them in all individuals of each generation. These are compared with the version of no applying maximum bipartite matching in [2].

We run this experiment in three times which are denoted by Run1, Run2, Run3. The number violations of hard constraints are decreasing from about nearly 100 down to zero at about 850th generation for all three running times (Figure 9). We set high weighting for avoiding violations of hard constraints, it is set by 0.9. Number violations of hard constraints are much decreasing at early generations of evolution, then it is kept decreasing in priority to zero.

Number of all possible enrollments of the best individual in every generation is showed in Figure 10. For early generations, this number is quite high due to there are some violations of hard constraints. It is also quite much changed at each generation after that, and stability of this number is reached at about 800th generation.

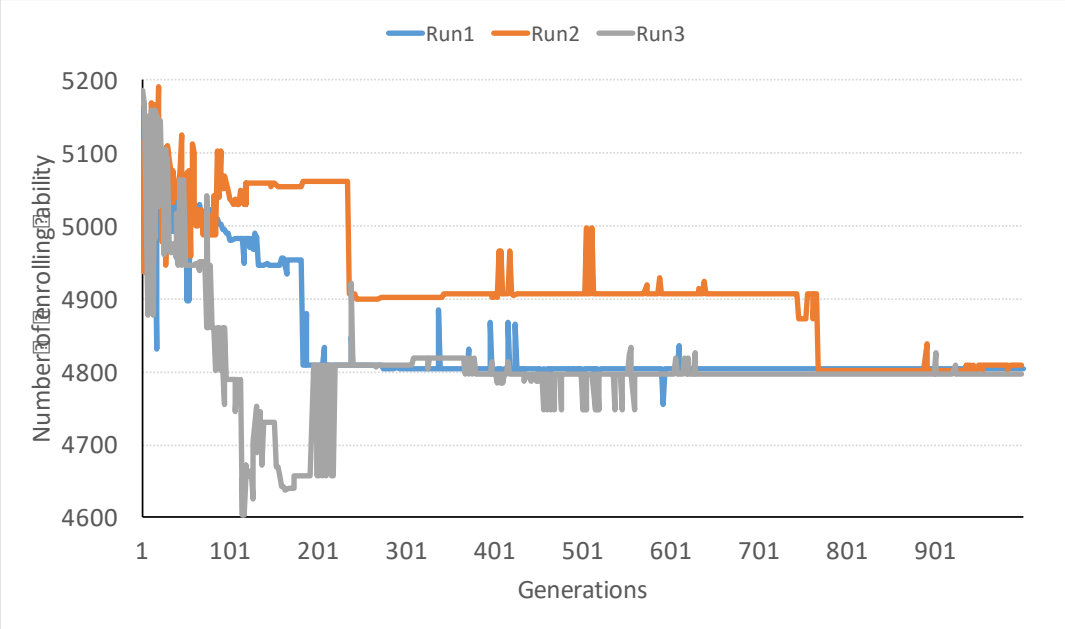

Figure 10. Number of possible enrollments of the best individual in generations.

The most important result in this paper is number of possible enrollments of the best individual in each experiment running. In all running, this number is higher than our method in [2] which it does not apply maximal bipartite matching for evaluating possible enrollments of students on timetables. In Figure 11, this number of first, second and last running is higher than those in [2] by 51, 144, 143, respectively.

For this result, it shows that the good performance of our enhanced method in number of possible enrollments for students. This can make decreasing number of canceled courses when a generated timetable is used for enrolling by students, because, it gives more chances for enrolling and we use maximal enrollments of every student for suggestion in reality.

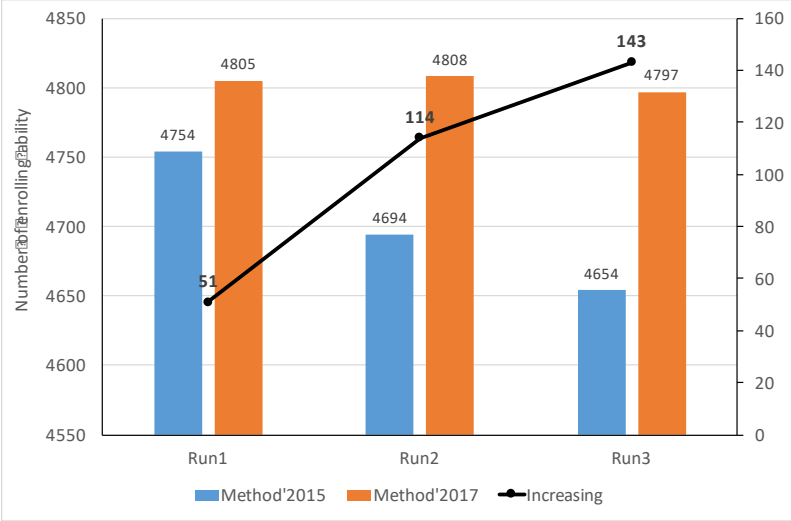

Figure 11. Number of enrolling ability of the best individual in generations.

5. CONCLUSIONS

In this paper, we propose a genetic algorithm based method for timetable problems in credits training at universities. Especially, we utilize maximal matching on bipartite graph for a sub-optimization problem in genetic algorithms, which is maximal enrollments of students.

The results of experiments in practice at the Faculty of Information Technology - Hanoi Open University show effectiveness of our proposed method. Running time of this algorithm is much faster than before, in comparing to traditional methods, it reduces much time to obtain the final timetable. It takes about 15 minutes while traditional method takes about 2 weeks by manually done. Moreover, results of this method show that final timetable gives much more opportunities for enrollments by students. Students also easily enroll courses for learning based on suggestions of maximal enrollments which is outputted by this algorithm. These also show potential effectiveness of this algorithm in practical application.

The proposed method in this paper can be extended to apply for practicing in many situations of credit courses training in universities. However, hard constraints and soft constraints may be considered more different assessments to show suitable of each assessment. Especially, we can use fuzzy parameters with more suitable for purpose of actual use, thereby efficiency is potentially increased. These will be studied further and announced in next research.

Acknowledgements. This research was supported by Hanoi Open University. I thank my colleagues from Faculty of Information Technology in Hanoi Open University, who provided collecting and preprocessing data of real problem for experimental running.

REFERENCES

1. Alade O. M., Omidiora E. O., Olabiyisi S. O. - Development of a University Lecture Timetable using Modified GA Approach, International Journal of Advanced Research in Computer Science and Software Engineering **4** (9) (2014) 163-168.

2. Duong Thang Long - A genetic algorithm based method for university course timetabling problems and application in Hanoi Open University, Journal of Computer Science and Cybernetics **32** (4) (2016) 285-301.
3. Mortaza A., Saeed S. - A Fast Genetic Algorithm for Solving University Scheduling Problem, IAES International Journal of Artificial Intelligence (IJ-AI) **3** (1) (2014) 7-15.
4. Kuldeep K., Sikander, Ramesh S., Kaushal M. - Genetic Algorithm Approach to Automate University Timetable, International Journal of Technical Research (IJTR) **1** (1) (2012) 47-51.
5. Wutthipong C., Soradech K., Nidapan S. - The Suitable Genetic Operators for Solving the University Course Timetabling Problem, Journal of Convergence Information Technology (JCIT) **8** (12) (2013) 60-66.
6. Meysam S. K. and Mohammad S. A. - Hybrid Genetic Algorithm for University Course Timetabling, IJCSI International Journal of Computer Science Issues **9** (2) (2012) 446-455.
7. Ruey-Maw C. and Hsiao-Fang S. - Solving University Course Timetabling Problems Using Constriction Particle Swarm Optimization with Local Search, Algorithms **6** (2013) 227-244.
8. Rahul M., Narinder S. and Yaduvir S. - Genetic Algorithms: Concepts, Design for Optimization of Process Controllers, Computer and Information Science **4** (2) (2011) 39-54.
9. Sanjay R.S. and Rajan S.B. - University Timetabling based on Hard Constraints using GA, International Journal of Computer Applications (0975 nts usi **42** (15) (2012) 1-5.
10. Vasileios K., George G., Christos G., Panayiotis A. and Efthymios H. - Solving the Examination Timetabling Problem in GPUs, Algorithms **7** (2014) 295-327.
11. Radomir P. and Jaroslav R. - Timetabling Problem with Fuzzy Constraints: A Self-Learning Genetic Algorithm, International Journal of Engineering and Innovative Technology **3** (4) (2013) 105-113.
12. Nguyen C.H., Witold P., Duong T.L and Tran T.S - A genetic design of linguistic terms for fuzzy rule based classifiers, International Journal of Approximate Reasoning **54** (1) (2012) 1p.1.