Manuscript received Oct. 18, 2007; revised Dec. 28, 2007

University Timetabling Using Evolutionary Computation

SUPACHATE INNET [1], NAWAT NUNTASEN [2]
Department of Computer Engineering
School of Engineering, University of the Thai Chamber of Commerce
Bangkok 10400 THAILAND[1]
Computer Science and Information Technology Department
Faculty of Science and Technology, Rajabhat Mahasarakham University
Mahasarakham 44000 THAILAND[2]
Tel./Fax: +662-275-4892
E-mail: supachate_inn@utcc.ac.th [1]
E-mail: nuntasen26@hotmail.com [2]

Abstract:- University timetabling problems have been interested by many researchers for more than a decade. However, there is no appropriated solution or computation model available to solve these problems successfully. This is because of many different version of timetabling problems. In this paper, a noval approach of Genetic Algorithm (GA) for solving educational timetabling problem is proposed, including the constraints statements, the definition of a hierarchical structure for the fitness function, and the generalized genetic operators, which can be applied to matrices representing timetables. The paper focuses on lecturing timetables only, but not the examinational timetabling. The crossover rate and mutation rate were varied to conduct effective results and they shows that the given appropriate crossover rate of 50% and mutation rate of 50% is the best ratio to solve university timetabling problems.

Key-Words: Scheduling, Genetic Algorithm, Evolutionary Computation, University Timetabling Problems

1 Introduction

Course (school or university) timetabling problems have been interested by many researchers for more than four decade. This is because of different forms of timetabling problem. Usually, these problems are often solved by leveraging human resource in universities or institutes. This requires a couple weeks or more to be done and the result is often not satisfied. Although, there are different problems, there exists many problem solving methods to save a lot of man-hours work, which usually concepts of employ optimization algorithms. Such methods include Genetic Algorithms [1]-[8], Tabu Search [9], Simulated annealing [10], and Constraints Logic Programming [11]. However, there is currently no appropriate computer tool.

To obtain great quality university timetable, optimal constraints satisfaction and optimization of the timetable's objectives at the same time is introduced [4]. Problems of university timetabling are NP-complete for reaching high quality solutions [2][3][5], so that solving it satisfactorily is often hard. A Genetic Algorithm (GA) is a powerful algorithm to find optimized solution; hence, it is employed to solve these problems [3][1].

In the past, many researchers have studied applications of GA for solving universities timetabling problems [1]-[8]. They are mostly interesting in element-level conditions. However, for the best of research, nobody yet has totally considered separation of period between the same subjects, excepted on the previous paper [12]. This problem occurs when; for instance, a subject requires two connected periods, which one period may be positioned one place and the other can be located in the other. The algorithm focuses on this problem as well as to reach high efficient solution with less computation memory employment.

The paper is organized as follows. After the Introduction, the problems of universities timetabling is stated in details, and the constraints employed to gain effectiveness are identified.

Then, the development of GA for solving university timetabling problems are discussed. In addition, the subsection includes a discussion of proposed model of chromosomes, required fitness function, and the employed genetic operators, i.e., crossover and mutation. The results session is then provided following with the conclusion session.

2 Problems Formulation

University Timetabling problems are NP-hard problem which mean that there is high quality solution. This is because such problems have several forms, and each university has their own and very specific requirement. The requirements or constraints include hard- and soft constraints.

Hard Constraints are unacceptable problems that need compulsory treatment. It is not allowed to occur at any percentages. These problems were not allowed to be happened. While Soft Constraints are ones, that can be accepted within minimization of frequency, more as representing preferences. They will maximize the perfectiveness of timetabling, e.g., there should not be time space more than two to three periods between the nearby subjects of students' schedule. Both of them are focused in this paper.

The constraint identification was gained from the timetabling principles of the University of the Thai Chamber of commerce, which applied for bachelor degree of engineering regular course, year 2004-2006. The schedule is from Monday to Friday within 6 periods a day (30 periods are the maximum for timetabling in a week).

2.1 Hard Constraints

1 Lecturers are not allowed to teach different subjects in the same period of a day.
2 A room is not allowed to be occupied by different subjects in the same period of a day.

2.2 Soft Constraints

1 There should not be more than two periods free between the nearby classes during the day of study.
2 Period 3 or 4 should be vacant for lunch time each day.
3 There should not be more than 4 continuing periods of study in a day.
4 There should not be more than 4 continuing periods occupied for lecturers in a day.

3 New Approach Genetic Algorithm

3.1 Developed Genetic Algorithms

In this section, development of GA for solving university timetabling problems. The algorithm has been developed from the previous work of the authors [13] in an order of getting higher effectiveness. The developed algorithm is explained as follows.

1. *Produces two prototypes chromosomes, called parents P1, and P2*
2. *Introduces crossover process to produce two children chromosomes, denoted as Ch1 and Ch2.*
3. *Examines fitness value of each chromosome with the ranking by fitness function.*
4. *A mutation or crossover technique is randomly chosen*

4.1. In case of crossover, the two best chromosomes that give lowest fitness value to gain parents chromosomes of next generation (Fig.1).

4.2 In case of mutation (Fig.2), one best chromosome that gives lowest fitness value following with comparison with previous

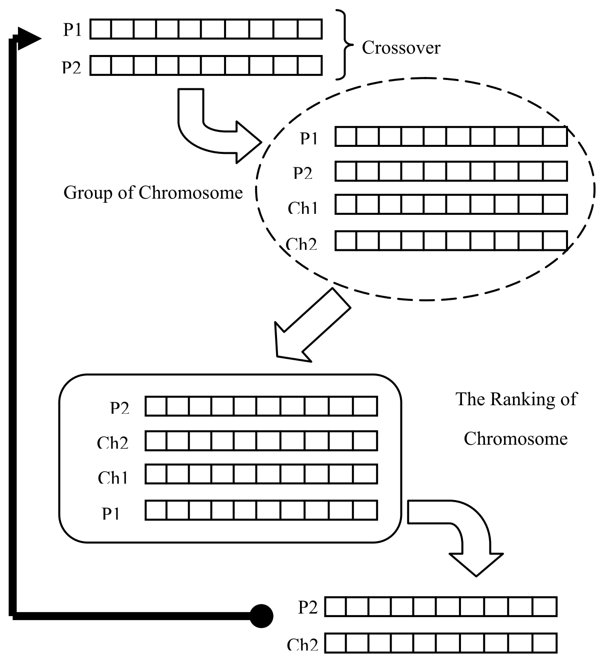


Fig.1. Crossover was randomly selected for timetabling

values to be selected mutation under the criteria of lesser or equal of previous minimize value. Then put it back to the pool and continues 4 again.

5. *Continues the process until the fitness value is met the requirement or the user end the process.*

Fig.1. shows how to produce desired chromosomes from their parents**.** Fig.2 shows the process of mutation.

3.2 Timetabling Chromosomes applied by GA

Timetabling Chromosomes applied by GA are constructed into tuple or group of element (E) to be spaces for selected input [6][7][8]. According to this paper, elements comprise three types of data, which are lecturer (L), Subject (S), and Room (R). In each element, it can be presented in a form of E = {L, S, R}. A set of E will represent as constituent elements of chromosomes. So, E = {E1,…, En} will represent as the elements of input to timetabling, while each of E(1-n) comprise of consequent set of L, S, and R. Each of them has subset of each own. It can be shown as $L = \{L_1, L_2, L_3, \ldots, L_n\}$, which will represent lecturers in that semester. In the same way of a set of Subject and a set of Room in that semester will do as $S = \{S_1, S_2, S_3, \ldots, S_m\}$ and $R = \{R_1, R_2, R_3, \ldots, R_n\}$, respectively. To enter the timetabling by elements, it can be happened in three types of phenomena. They are a period for an element, several periods for an element, and a blank period or more for no element as there is no class in the schedule as shown in Table.1.

A chromosome will be stringed following the sequences of periods that yield the length of that chromosome as shown in equation (1)

$$lChrom = G * P(n) \qquad (1)$$

where, lChrom: the length or number of bits of each chromosome G: Group of students who register the subjects following the plan in each major and P(n): number of periods in the timetabling.

	1	2	3	4	5	6
Mon	E1	E1				
Tue			E2		E9	E9
Wed	E7 $\{L_9,S_{12},R_5\}$			E2 $\{L_1,S_4,R_2\}$		
Thurs						
Fri				E21	E21	

Table 1. Types of phenomena happened in timetabling

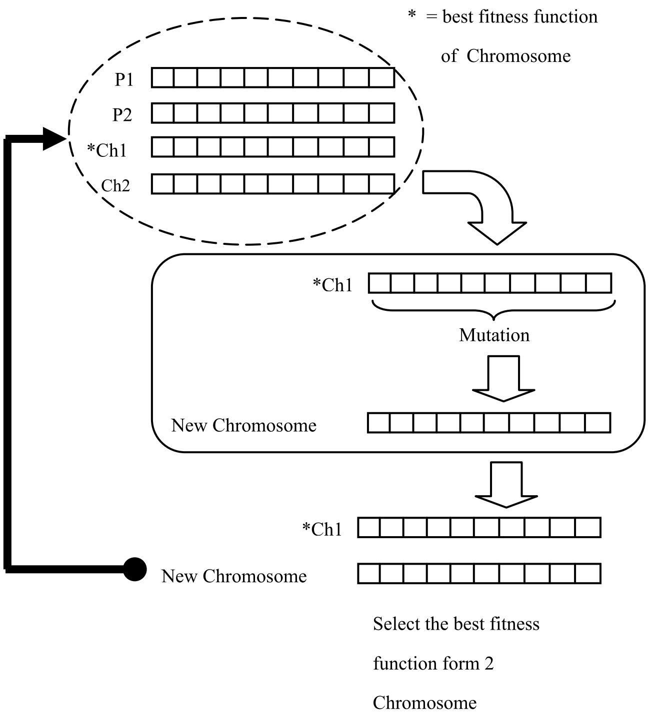


Fig.2. Mutation was randomly selected for timetabling

Bit	1	2	…	…	9	9	…	17
Chro	E1	E1	…	…	E2	E2	…	E9

Fig.3. Timetabling chromosomes

Fig.3 shows timetabling chromosomes resulting from equation (1).

The chromosome of university timetable is the arrangement of chromosome of each group of student as shown below:

$$Chromosome = T_{d(1)p(1)g(1)}, T_{d(1)p(1)g(2)}, T_{d(1)p(1)g(3)}, \ldots, T_{d(1)p(1)g(N)}, T_{d(1)p(2)g(1)}, \ldots, T_{d(1)p(2)g(N)}, T_{d(1)p(3)g(1)}, T_{d(1)p(3)g(2)}, \ldots, T_{d(1)p(M)g(1)}, T_{d(1)p(M)g(2)}, \ldots, T_{d(L)p(M)g(N)}$$

where $T_{d(x)p(y)g(n)}$ *: the chromosome of university timetable at d(x), p(y), g(n) and*
d(x) : the day of x where 1<x<L
p(y) : the period of y where 1<y<M
g(n) : the group of n where 1<n<N
L : the maximum number of day to be arrange in the timetable
M : the maximum number of period
N : the maximum number of group

3.3 Fitness Function

Fitness function is used to measure appropriateness of timetabling and search to find the best one, which capable to be the raised problems. Fitness function constructed from timetabling conditions. A simple form of fitness function can be written as shown in equation (2).

$$\sum_{i=1.n} W_i^h \times \cos t_i^{hard}(x) + \sum_{i=1...n} W_i^s \times \cos t_i^{soft}(x) \quad (2)$$

Two hard-constraints and four soft-constraints discussed in section 2.1 and 2.2 can be equated as shown in equation (3) below:

$$\begin{aligned} & w_1 \sum_{G=1}^{G=g} \sum_{P=1}^{P=30} \sum_{S=0}^{S=s} \sum_{L=0}^{L=l} \text{Bound Lecturer Clash} \left[\left(\left(Lecture_L, Subject_S\right), Period_P\right), Group_G\right] \\ & + w_2 \sum_{G=1}^{G=g} \sum_{P=1}^{P=30} \sum_{S=0}^{S=s} \sum_{R=0}^{R=r} \text{Bound Room Clash} \left[\left(\left(Room_R, Subject_S\right), Period_P\right), Group_G\right] \\ & + w_3 \sum_{G=1}^{G=g} \sum_{P=1}^{P=30} \sum_{S=0}^{S=s} \text{Break not more than 2 period} \left[\left(Subject_S, Period_P\right), Group_G\right] \\ & + w_4 \sum_{G=1}^{G=g} \sum_{P=3}^{P=4} \sum_{S=0}^{S=s} \text{Break Period 3 or Period 4} \left[\left(Subject_S, Period_P\right), Group_G\right] \\ & + w_5 \sum_{G=1}^{G=g} \sum_{P=1}^{P=30} \sum_{L=0}^{L=l} \text{Lecturer Load} \left[\left(Lecture_L, Period_P\right), Group_G\right] \\ & + w_6 \sum_{G=1}^{G=g} \sum_{P=1}^{P=30} \sum_{S=0}^{S=s} \text{Student Load} \left[\left(Subject_S, Period_P\right), Group_G\right] \end{aligned} \quad (3)$$

where weight of condition for each constraints are as follows:

- w_1 *: weight of condition which a lecturer is not allowed teaching different subjects in the same period.*
- w_2 *weight of condition which a room is not allowed to be used to teach different subjects in the same period.*
- w_3 *: weight of condition which a group of students should not have more than two periods free before any subjects in timetabling schedule.*
- w_4 *: weight of condition which the period 3-4 should be free for lunch time in a day.*
- w_5 *: weight of condition which a lecturer student should not have classes more than 4 continuing periods.*
- w_6 *: weight of condition which a group of students should not have classes more than 4 continuing periods.*

In addition, the value of each function can be explained as seen below:

$$\text{Bound Lecturer Class}\left[\left(\left(Lecture_L, Subject_S\right), Period_P\right), Group_G\right]$$

- *Equal 0:* *when* $Period_P$ *are as* L_0 *(no lecturer) or there is the same* $Lecture_L$ *and the same* $Subject_S$ *in the same* $Period_P$
- *Equal 1:* *when the same* $Lecture_L$ *happens in the same* $Period_P$ *but different* $Subject_S$

$$\text{Bound Room Class}\left[\left(\left(Room_R, Subject_S\right), Period_P\right), Group_G\right]$$

- *Equal 0:* *when as others*
- *Equal 1:* *when the same* $Room_R$ *happens in the same* $Period_P$ *but different* $Subject_S$

$$\text{Breaknot more than 2 Period}\left[\left(Subject_S, Period_P\right), Group_G\right]$$

- *Equal 0:* *when as others*
- *Equal 1:* *when* $Subject_S$ *happens in* $Period_P$ *and* $Period_{P-4}$ *or* $Subject_S$ *happens in* $Period_P$ *and* $Period_{P+4}$

$$\text{Break P3 or P4}\left[\left(Subject_S, Period_P\right), Group_G\right]$$

- *Equal 0:* *when* $Subject_S$ *happens in* $Period_3$ *or when* $Subject_S$ *does not happen in* $Period_3$ *and* $Period_4$

- *Equal 1:* *when S_S happens in* $Period_3$ *and* $Period_4$

Lecturer Load $\left[(Lecture_L, Period_P), Group_G \right]$

- *Equal 0:* *when* $Period_P$ *happens in* $Lecture_L$ *while* $P \leq 4$
- *Equal 1:* *when* $Period_P$ *happens in* $Lecture_L$ *while* $P > 4$

Student Load $\left[(Subject_S, Period_P), Group_G \right]$

- *Equal 0:* *when* $Subject_S$ *happens in* $Period_P$ *of each* $Group_G$ *while* $Subject_S \leq 4$
- *Equal 1:* *when* $Subject_S$ *happens in* $Period_P$ *of each* $Group_G$ *while* $Subject_S > 4$

The Best answers of timetabling by GA application comprise the minimum target function, which is not allowed to have the hard constraints (w_1, w_2) and the one which should yield minimize soft Constraint (w_3, w_4, w_5, w_6)

3.4 Genetic Operator

Process of Crossover and Mutation in timetabling will be run inside each string of chromosomes, which occupied by groups of students. It can be shown as the following picture of chromosome Fig.4.

3.4.1 Crossover

Crossover is an exchange element of two parents' chromosomes. Its process comprises:

1 In the first chromosome of parents, Random position to run crossover.

	Group 1					Group 2				
Bit	1	2	3	…	30	31	32	33	…	n
period	1	2	3	…	30	31	32	33	…	n
Ch1	E1	E1		…	E10		E9	E9		En
	Genetic Operator					Genetic Operator				
Ch2		E7	E7	…		E25		E9	…	En

Fig.4. Chromosome grouping for genetic process

2 At the right position from the random, elements were examined by the 2 following steps:

2.1 From the random position, elements, which bordered to each other were examined

In case of such positions have equal value, both of them will be saved and put in the first child Chromosomes. In case of not equal, one from the random will be selected. It can be shown as the following picture of chromosome Fig.5.

3. At the second chromosome of parents, delete every element, which same with the random position in the first chromosome of parents and put the residual elements in to the next space of the chromosome of the first child.

Then, use the same principles to the second chromosome of parents to yield the chromosome of the second child. by altering the elements randomly. This can be explained in detail as follows:

1. Random two positions in a group of students to run mutation in selected chromosomes.
2. At the right position from the random, elements were examined. Then two following steps were processed.

3.4.2 Mutation

The process of chromosome mutation was done

Bit	1	2	3		30	31	32	33	…	n
Period	1	2	3		30	31	32	33	…	n
Parent 1	E1	E1		E5	E10		E9	E9		En
	* ↓				* ↓			* ↓		
Ch1	E1	E1			E10		E9	E9		

Fig.5. Chromosome random and Selection for Crossover

Bit	1	2	3	…	30	31	32	33	…	n
period	1	2	3	…	30	31	32	33	…	n
Parent 2	E5	~~E1~~	~~E1~~	…		~~E9~~	~~E9~~	E2		En
Ch1	E1	E1	E5		E10	E2	E9	E9		

Fig.6. Deleted and Selected Element from the Second Parents of Chromosome

2.1 At positions, which randomly bordered to each other, elements were examined.

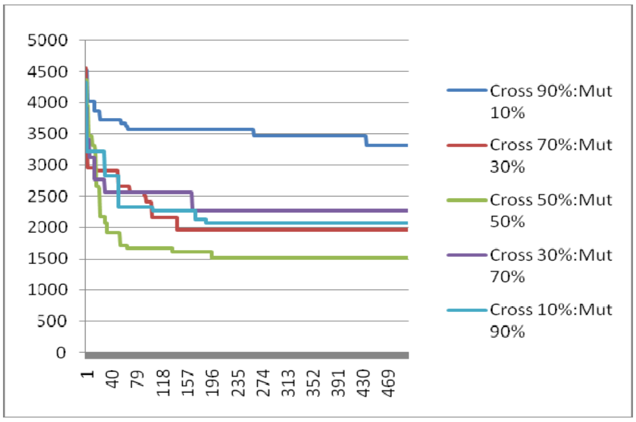
2.2 In case of such positions have equivalent value, both of them will be saved to insert at the first position, which makes the sequent shifting of previous chromosomes to the next position. It can be shown as the following picture of chromosome.

Bit	1	2	3	4	5	…	27	28	29	30
period	1	2	3		30	31	32	33	…	n
Ch1	E1	E1		E5	E10		E9	E9		En
	*						*	*		
Ch1	E9	E9	E1	E1	E5					

Fig.7. Chromosome random and selection for Mutation

4 Results

This trail uses data for timetabling of School of Engineering, University of the Thai Chamber of Commerce. They comprise 80 subjects, 26 rooms, 42 lecturers and 25 groups of students.500 generations of GA application were conducted via three different proportions between crossover : mutation which are 90%: 10%, 80%:20%,50%:50% , 20%:80% and 10%:90%.



Graph 1. Fitness function of GA with variation of crossover and mutation rate

Three ratios of crossover and mutation were presented to show the desired proportion of crossover and mutation. Graph 1 shows how fitness function values have been improved by generations with variations of ratio of crossover rate and mutation rate.

The results from graph 1 show that the value of fitness function is converts to a number. The results show that every time the chromosomes have been produced by mutation process, the value of fitness function can be decreased or same level.

The results in Table 2 also show that the more rate of crossover, the better the value of fitness function. However, it needs more generation of processing, while the more mutation rate will give the opposite results. The comparison among the 3 different proportions shows that the lowest fitness function value with minimize number of generations reachable to stable line of GA under crossover rate of 50% and mutation rate of 50 % are the most satisfactory one.

Ratio of GA Operator	Best Generation	Fitness Function value
Crossover 90%: Mutation 10%	436	3320
Crossover 80%: Mutation 20%	423	2070
Crossover 70%: Mutation 30%	142	1968.9
Crossover 60%: Mutation 40%	113	2171.8
Crossover 50%: Mutation 50%	197	1520.6
Crossover 40%: Mutation 60%	357	1669.7
Crossover 30%: Mutation 70%	169	2270.9
Crossover 20%: Mutation 80%	104	2318.2
Crossover 10%: Mutation 90%	388	2072.4

Table.2. Ratio of GA Operator, Best Generation and Fitness Function value

According to the loading of computer resource as shown in Fig. 8, the program use only 13 % of CPU loading and 15,816 Kb of memory. It is acceptable because very small loading of computer resource is required.

In addition, more than one period of separation between the subjects was not found after using this program under crossover rate of 50% and mutation rate of 50 % which is one of distinctiveness. It can be shown in Fig. 9 as follow.

5 Conclusion

This study represents genetic algorithm approach by use generation of living things which has construct chromosome group of answer and construct cycle of new genetic algorithm, which in constructing schedule of the course that is rapid collecting and using resource of the system. That has efficient ability to control cross over and mutation which has no affect to schedule desire in learning continuity more than one period. However, this approach should improve the process and be able to response of various conditions in the schedule which is appropriate and the environment of the problem occurred in constructing a class schedule which is different in each education institute.

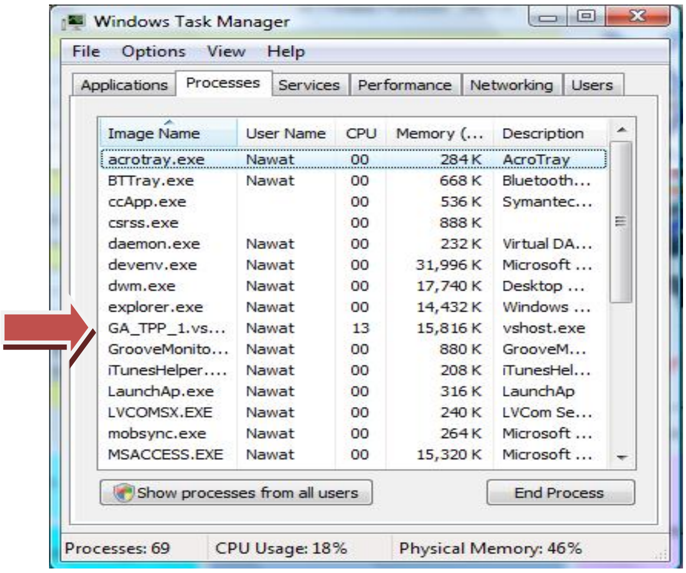
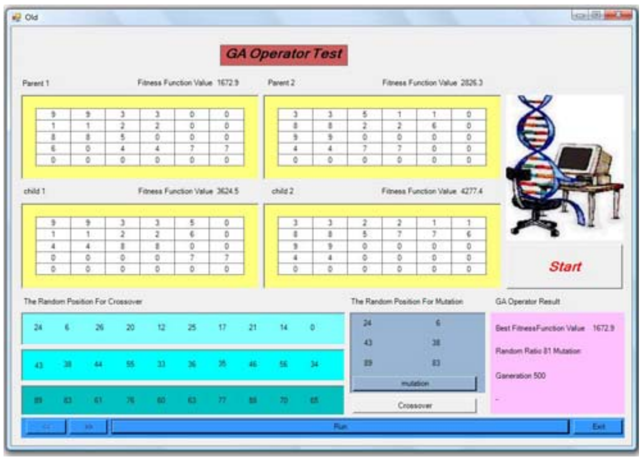
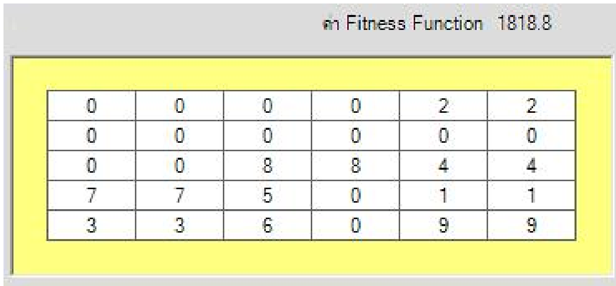


Fig.8. Computer resource utilization



(a)



(b)

Fig. 9. Timetabling results from GA applications under the satisfied trail.

Reference:

[1] E. K Burke, D. G. Elliman and R. F. Weare, A Genetic Algorithm based University Timetabling System, *the 2nd East-West International Conference on Computer Technologies in Education*, 1994.

[2] S. Győri, Z. Petres, A. R. Várkonyi-Kóczy, Genetic Algorithms in Timetabling. A New Approach, 2001, *http://www.mft.hu/hallg/200107.pdf.*

[3] P. Srinin, P. Rojanavasu, E. Pin-nguen, The Automatic Timetabling Problem by using Genetic, *Joint Conference on Computer Science and Software Engineering (JCSSE2005)*, 2005.

[4] S. Kazarlis, Solving University Timetabling Problems Using Advanced

Genetic Algorithms, *5th International Conference on Technology and Automation (ICTA'05)*, 2005.

[5] E. Ozcan, A. Alken, Timetabling using a Steady State Genetic Algorithm, *The 4th international conference on the Practice And Theory of Automated Timetabling*, Belgium, 2002.

[6] D. Abramson, J. Abela, A Parallel Genetic Algorithm for Solving the school Timetabling Problem, *15th Australian Computer Science Conference*, Hobart, 1992.

[7] K. Paitoonwattanakij, K. Vongvipaporn, GA For Scheduling School Timetable, *J.Natl.Res.Council Thailand*, 2000.

[8] S. Junnatat, Multi Objective Genetic Algorithm For School Timetabling Problem, *The National Conference on Computing and Information Technology*, 2005.

[9] A. Herth, Tabu Search for Large Scale Timetabling Problem, *European Journal of Operational Research*, 1992.

[10] A. Abramson, Constructing School Timetables Using Simulated Annealing: Sequential and Parallel Algorithms, *Management Science*, 1991.

[11] C. Gueret, N. Jussien, P. Boizumault, C. Prins, Building University Timetables Using Constraint Logic Programming. In: Burke, E, Ross, P. (Eds.) Practice and Theory of Automated Timetabling I (PATAT 1995, Edinburgh, Aug/Sept, selected papers), Vol. 1153. Springer-Verlag, Berlin Heidelberg New York, 1996

[12] N. Nuntasen, S. Innet, Application of Genetic Algorithm for Solving University Timetabling Problems: a case study of Thai University, 7th WSEAS International Conference on Simulation, Modelling, and Optimization, Beijing, China, 2007.

[13] N. Nuntasen, S. Innet, A Novel Approach of Genetic Algorithm for Solving University Timetabling Problems: a case study of Thai Universities, 6th WSEAS International Conference on System Science and Simulation in Engineering, Venice, Italy, 2007.