# A Noval Approach of Genetic Algorithm for Solving Examination Timetabling Problems

a case study of Thai Universities

Supachate Innet
Department of Computer Engineering and Multimedia
University of the Thai Chamber of Commerce
Bangkok, Thailand
supachate_inn@utcc.ac.th

***Abstract*— Arranging examination timetable is problematic. It differs from other timetabling problems in terms of conditions. A complete timetable must reach several requirements involving course, group of student sitting the exam in that course, etc. It is similar to the course's timetable but not the same. Many differences between them include the way to create and the requirements. This paper proposes an adaptive genetic algorithm model applied for improving effectiveness of automatic arranging examination timetable. Hard constraints and soft constraints for this specific problem were discussed. In addition, the genetic elements were designed and the penalty cost function was proposed. Three genetic operators: crossover, mutation, and selection were employed. A simulation was conducted to obtain some results. The results show that the proposed GA model works well in arranging an examination timetable. With 0.75 crossover rate, there is no hard constraints appeared in the timetable.**

***Keywords— genetic algorithm; examination timetable; evolution computing***

## I. Introduction

Currently, education institutes, especially private universities in Thailand open several courses in order to serve students' requirements. Each year, the number of students in each course is changeable, depending on fashion and nation's requirements. Arranging course and exam timetable are, therefore, becoming more complicated. Examination timetabling problems are differing from other timetabling problems in terms of conditions [1] and are also differing from institute to institute. Several parameters must be considered in order to obtain a satisfied solution. Such parameters include courses, groups of student, exam rooms, time slots, duration of exam, etc. In addition, time to accomplish this task depends on personal experiences. One who is familiar with this task or has knowledge in exam scheduling may propose more effective solutions.

Examination timetabling problem are generating considerable interest from many researchers across the fields of Operation Research and Artificial Intelligence. A large number of different schemes have been proposed in the literature for automating effective timetable [2] – [9]. [2] provides a good survey results of search methodologies and automated approaches for examination timetabling. Some metaheuristic algorithms for solving examination timetabling problem have been compared using different size of datasets [3]. [4] and [5] engages a genetic algorithm to implement the exam timetable, but [5] has more focusing on final examination. [6] develops an approach by mimicking processes from cell biology and finds solutions to the incapacitated examination timetabling problem, and became known as the Carter benchmark set which is used generally to compare the performance of several methodologies in solving the incapacitated examination timetabling problem [7], [8] – [7]. Additionally, [7] occupied Flex-Deluge algorithm. Furthermore, [8] and [9] have employed different method for solving problems: [8] develops a hybrid variable neighborhood method, while [9] deploys local search based scheme. It can be seen that solving examination problem can be accomplished by many mechanisms.

However, in this paper, it is of interests to engage a genetic algorithm for solving examination timetabling problem, which extended from the work proposed in [10] – [12]. The proposed model was tested using the real-world datasets of University of the Thai Chamber of Commerce, a Thailand's private university, involving the weekly scheduling of all courses in 2012 - 2013 curriculum of bachelor degree of School of Engineering of this university. The objective is to minimize penalties from proximity constraints.

The paper is organized as follows. It starts with a general discussion about how this research comes from and some previous work related to this problem. Section II provides more discussion about examination timetabling problems in details. The proposed model of GA used in this work is explained in Section III. The section provides how the GA elements and chromosome are designed, what the definition of penalty cost function used in the simulation is, and which GA operators are employed to solve the problems. In section IV, the computational results are included and some discussions of the results are stated. Finally, the conclusion of this work is presented in Section V.

978-1-4673-5580-3/13/$31.00 © 2013 IEEE

# II. Problem Statement

To compete with large and famous public universities, private universities, where having limits number of classrooms, and instructors, must provide varieties of curriculums and facilities in enabling to meet students' interests. As curriculums are changing every year, arranging an exam timetable must be done appropriately semester by semester to cover changed conditions. This creates an important problem to the schedulers.

Until recently in UTCC, scheduling exam timetable is taken manually by staffs from all departments who make a decision based on their experience with a little guidance from spreadsheet computer software to avoid clashes. They only concerns themselves with some simple constraints such as no student attend more than one exam in a given timeslot, the room's capacity is bigger than the number of attendants. Though, this process still requires more than two weeks. Still, there are many complaints and requests for change from students after the schedule is announces. Additionally, they again re-schedule manually with as few adjustments to the rest of the timetable as possible. However, the problem may circulate to the other students. Usually, if the problem cannot be solved, the nighttime timeslot is taken, and the student may request to sit more than one or two exams during a day.

A complete exam timetable must reach as many requirements as possible if not all. The requirements involves in this problem include requirements of student, room, lecturer, period, and time space.

Requirements are usually stated in terms of constraints. In this paper, two types of constraints are under consideration: hard constraints and soft constraints. Hard constraints are unacceptable problems which need obligatory treatment. They are not allowed to occur at any percentages. In opposition of hard constraints, soft constraints are conditionally acceptable, but with a minimization of frequency.

Hard constraints and soft constraints specified in work are as follows:

### A. Hard Constraints

*1) All course must be scheduled.*

*2) The number of student taking the examination must not exceed the seat capacity of the examination room.*

*3) For all group of student that taking the same corse must be taken at the same time slot of the same day.*

*4) Lecturer's perference concerning his/her examination timeslot must be meet.*

Note that the constraint where a student attends more than one examination will not be considered in this study because it is assumed that the exam timetable is scheduled and announced before the enrolment days. Therefore it is possible to know who enroll for a course.

### B. Soft Constraints

*1) Night time timeslots should not be taken.*

*2) A Student of the same group should have a day-off (2 timeslot off) after sitting an exam.*

These constraints are employed to define a fitness function as will be discussed in Section III (B).

# III. Proposed GA Model

This section proposed an adaptive genetic algorithm model for improving effectiveness of automatic arranging examination timetable. The model is developed from the previous work presented in [10] – [12]. The element of chromosome is re designed for more appropriate to exam schedule. The penalty cost function is determined for evaluating the timetable. The following subsection explains all topics mentioned above in details.

### A. Elementary Design

The chromosome of exam timetable's elements is constructed into a group of elements (E). According to this paper, elements comprise two types of data: course (C), and group (G). A set of E will represent as constituent elements of chromosomes. So, E = {C, G} will represent the elements of chromosome. Each of C, and G has subset of each own. It can be shown as $C = \{C_0, C_1, C_2, \ldots, C_c\}$ which will represent a set of c courses. In the same way for a set of g student groups writing the same exam will do as $G = \{G_0, G_1, G_2, \ldots, G_g\}$. Note that $\{C_0, G_0\}$ means unoccupied period.

The chromosomes will then be operated by means of the proposed genetic algorithm illustrated in Fig. 1.

Fig. 2 shows an example of chromosome of exam timetable's elements. An exam chromosome contains D(p) days, and in one day there are R(q) room available room. Each room may have different seat capacity but all room can provide three exam time slots. The last time slot of each room occurs at night time.

### B. Structure of the Penalty Cost Function

In order to adhere with real world issues, the chromosome will be examined by a fitness function called "Penalty Cost Function (F)". This function constructs from hard constraints and soft constraints as discussed in Section II. Simply, it is the summation of penalty from all hard constraint violations and soft constraint violations. The objective of this function is to determine how well the chromosome meets requirements. The chromosome that provides the minimum penalty cost value will be chosen to create an appropriate examination timetable. Equation (1) is the Penalty Cost Function defined to solve problems specified in this work.

$$F_{cost}(x) = \sum_{i=1}^{4} w_i^{hrd} \times cost_i^{hrd}(x) + \sum_{i=1}^{5} w_i^{sft} \times cost_i^{sft}(x) \quad (1)$$

*where,*

$F_{cost}(x) : Penalty\ Cost\ of\ x^{th}\ generation$

$x : the\ generation\ of\ exam\ timetable\ under\ evaluation$

$w_i^{hrd} : weight\ of\ i^{th}\ hard\ constraint$

$w_i^{sft} : weight\ of\ i^{th}\ soft\ constraint$

In addition, the hard constraints and soft constraints are formulated as follow.

$$cost_1^{hrd} = \sum_{c=1}^{n_c}\sum_{g=1}^{n_g} CourseNotInTimeSlot\left[(C_c, G_g)\right]$$

$$cost_2^{hrd} = \sum_{c=1}^{n_c}\sum_{g=1}^{n_g}\sum_{t=1}^{n_t} SameCourseNotSameTime[(C_c, G_g), T_t]$$

$$cost_3^{hrd} = \sum_{t=1}^{n_t}\sum_{c=1}^{n_c}\sum_{g=1}^{n_g} CapacityOverload[(C_c, G_g), |G_g| > |R_g|]$$

$$cost_4^{hrd} = \sum_{t=1}^{n_t}\sum_{c=1}^{n_c}\sum_{g=1}^{n_g} LecturerPref[(C_c, G_g), T_t]$$

$$cost_1^{sft} = \sum_{t=3,6,9\ldots}^{n_t} HasNightTimeSlot[T_t]$$

$$cost_2^{sft} = \sum_{t=1}^{n_t}\sum_{c=1}^{n_c}\sum_{g=1}^{n_g} ContinueExam[(C_c, G_g), T_t]$$

*where the value of each function can be explained as seen below:*

$CourseNotInTimeSlot\left[(C_c, G_g)\right]$
Equal 1 : $(C_c, G_g) == true$
Equal 0 : otherwise

```
begin
  Define fitness function
  Form prototype of chromosome
  gen ← 0
  // randomly create parents chromosome - P1(gen)
  // and P2(gen)
  Initial chromosome P1(gen), P2(gen)
  while (not terminate condition) do
    begin
      // random a probability for operation
      if (random < crossover probability)
        // do crossover
        C1(gen) = Crossover P1(gen)
        C2(gen) = Crossover P2(gen )
      else
        // do mutation
        C1(gen) = Mutate P1(gen)
        C2(gen) = Mutate P2(gen)
      // examine fitness value of each chromosome
      Examine P1(gen), P2(gen), C1(gen), C2(gen)
      gen ← gen + 1  // next generation of chromosome
      // select two best chromosome to be new parents
      Rank P1(gen), P2(gen), C1(gen), C2(gen)
      P1(gen) = SelectBestRank1
      P2(gen) = SelectBestRank2
    end
end
```

Fig. 1. Adaptive Genetic Algorithm for solving automatic examintion timetabling problem.

$SameCourseNotSameTime[(C_c, G_g), T_t]$
Equal 1 : $(C_c, G_g), T_t\ != (C_c,\ G_g), x$
Equal 0 : otherwise

$CapacityOverload[(C_c, G_g), R_r]$
Equal 1 : $\sum_{c=1}^{n_c}\sum_{g=1}^{n_g}[(C_c, G_g), |G_g|] > capacity == true$
Equal 0 : otherwise

$LecturerPref[(C_c, G_g), T_t]$
Equal 1 : $(C_c, G_g), T_t,\ != (C_{\acute{c}},\ G_{\acute{g}}), T_{\acute{t}}$
Equal 0 : otherwise

$HasNightTimeSlot[(C_c, G_g), T_t]$
Equal 1 : $(C_c, G_g)\ != (C_0, G_0)$
Equal 0 : otherwise

$ContinueExam[(C_c, G_g), T_t]$
Equal 1 : $(C_c, G_t), T_t\ == (C_c, G_t), T_{t+1}$
Equal 0 : otherwise

*where,*
$n_c$ : *number of course*
$n_g$ : *number of group of student enrolled in the course*
$n_t$ : *number of time slot*
$T_t$ : $t^{th}$ *Time slot*
$|G_g|$ : *size of* $g^{th}$ *group*
$|R_g|$ : *size of* $g^{th}$ *room*

## C. Genetic Operator

Three genetic operators are employed to solve this problem: Crossover, Mutation, and Selection.

### 1) Crossover

Crossover is a method that exchanges element of two parent chromosomes, say Parent 1 and Parent 2. The process of crossover is as follows:

1. Select a number of positions of element in one of the prototype chromosome randomly. These elements will be operated by crossover method. Fig. 3 shows that bit 1, 3, 4, and 7 of Parent 1 were randomly chosen, and the value of those bit are E(1), empty E(0), E(4) and E(7) respectively.

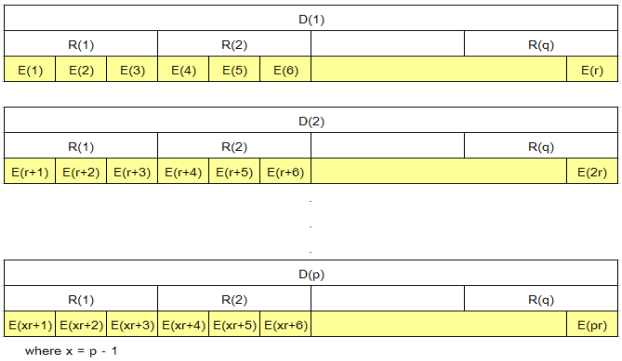


Fig. 2. Example of chromosome of exam timetable's elements

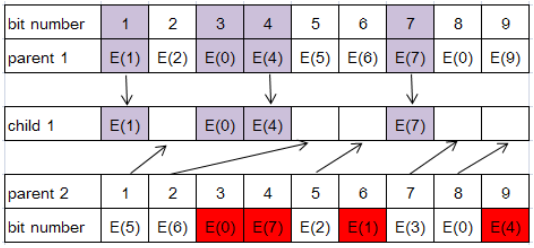


Fig. 3. Example of Crossover Process.

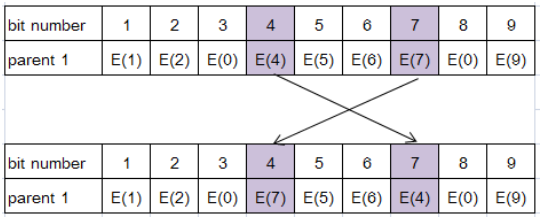


Fig. 4. Example of Mutation Process.

2. Duplicate those elements to a child chromosome, say Child 1.
3. For the other parent chromosome (Parent 2), delete all the elements (not the bit position) of the same value as were selected in Parent 1. From Fig. 3, E(1), E(4), and E(7) were deleted.
4. Put all the residual element of Parent 2 into the next space of the Child 1.
5. Use the same principles to the second chromosome of parent (Parent 2) to yield the chromosome of the other child (Child 2).

*2) Mutation*

The process of chromosome mutation was done by altering the elements in the chromosome to create a new chromosome. The process of mutation can be explained in detail below.

1. Select two positions of element in one of the prototype chromosome randomly. In Fig. 4, bit 4 and bit 7 of Parent 1 were randomly chosen as an example.
2. Swap those bits to create the new Parent 1.

*3) Selection*

Selection is the method to choose required chromosome from the mating pool. The chromosomes that are selected will be the parents' chromosome of the next generation. In this paper, two chromosomes that have best (or lowest) fitness value are.

TABLE I. SCHEDULING PROBLEM SPECIFICATION

| No | Description | Quantity |
|---|---|---|
| 1 | Number of courses | 71 |
| 2 | Number of group | 135 |
| 3 | Number of room for exam | 10 |
| 3 | Number of timeslot per room per day | 3 |
| 4 | Number of examination day | 11 |

# IV. Computational Results

The problem considered in this paper is specific to real-practice courses from University of the Thai Chamber of Commerce (UTCC), a Thailand's private university, and involves the final exam scheduling of all courses in 2012 - 2013 curriculum of bachelor degree of school of Engineering of this university. The specifications of this problem are shown in Table I. Table II represents the value of weight for each constraint that is used for obtaining the computational results. The values of weight do not significant in terms of quantity, but they represent each constraint. They are used for differentiating; and hence, counting the quantity of constraints appeared in the results.

An effect of crossover probability on the fitness value was study by varying it from 0.00, 0.30, 0.40, 0.50, 0.60, 0.65, 0.70, 0.75, and 1.00. Five hundred generations of process were conducted. The penalty cost values were determined when it was stable. The minimum process generations that provide stable penalty cost values were recorded. All the results were tabulated in Table III.

From the results shown in Table III, it can be seen that the lower fitness value are occurred when the crossover probability are in range 0.65 to 0.75. Therefore, the simulation was repeated at 0.70 crossover probability to average the penalty cost value. The result is illustrated in Table IV. The result shows that by using 0.70 crossover rate, hard constraints will never occurred, but some soft constraints will be unavoidable. To get better results, some facilities, such as more examination rooms, more examination day, or more room's capacity may require.

TABLE II. HARD CONSTRAINTS AND SOFT CONSTRAINTS WEIGHTING

| Weight | Value |
|---|---|
| $w_1^{hard}$ | 10000 |
| $w_2^{hard}$ | 10000 |
| $w_3^{hard}$ | 10000 |
| $w_4^{hard}$ | 10000 |
| $w_1^{soft}$ | 1 |
| $w_2^{soft}$ | 1 |

TABLE III. EFFECT OF CROSSOVER PROBABILITY TO PENALTY COST

| Crossover Probability | Number of Generations | Penalty Cost |
|---|---|---|
| 0.00 | 458 | 100012 |
| 0.30 | 310 | 62036 |
| 0.40 | 378 | 92020 |
| 0.50 | 474 | 57012 |
| 0.60 | 432 | 71014 |
| 0.70 | 445 | 31022 |
| 0.75 | 426 | 32 |
| 1.00 | 183 | 220039 |

TABLE IV. AVERAGE NUMBER OF CONSTRAINTS OCCURRED AT 0.75 CROSSOVER RATE.

| No | Generations | Penalty Cost Value | Average Hard Constraints | Average Soft Constraints |
|---|---|---|---|---|
| 1 | 426 | 32 | 0 | 24.4 |
| 2 | 431 | 23 | | |
| 3 | 421 | 25 | | |
| 4 | 454 | 26 | | |
| 5 | 422 | 18 | | |
| 6 | 463 | 19 | | |
| 7 | 441 | 24 | | |
| 8 | 459 | 18 | | |
| 9 | 444 | 28 | | |
| 10 | 469 | 31 | | |

# V. Conclusion

This paper proposes an adaptive genetic algorithm model applied for improving effectiveness of automatic arranging examination timetable. The algorithm was developed in enabling to obtain more optimized results. The elements of chromosome were designed. Hard constraints and soft constraints for this specific problem were discussed and the penalty cost function was proposed. In addition, three genetic operators: crossover, mutation, and selection were employed. The results show that the adaptive GA model works well in arranging a examination timetable. With 0.75 crossover rate, there is no hard constraints appeared in the timetable.

## *References*


[1] R,.R. Luis, O. Eugenio, "A Lauguage for Specifying Complete Timetabling Problems," PATAT '00 Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III, pp. 322 – 341, Springle, 2000.

[2] R. Qu, E.K. Burke, B. McCollum, L.T.G. Merlot, and S.Y. Lee, "A survey of search methodologies and automated approaches for examination timetabling," Technical Report NOTT-CSTR-2006-04, School of Computer Science & IT, University of Nottingham, 2006.

[3] Z.N. Azimi, "Comparison of Metaheuristic Algorithms for Examination Timetabling Problem, " Journal of Mathematics and Computing, vol. 16, no. 1 – 2, pp. 337 – 354, 2004.

[4] O.T. Arogundade, A.T. Akinwale, O.M. Aweda, "A Genetic Algorithm Approach for a Real-World University Examination Timetabling Problem," International Journal of Computer Applications, vol. 12, no. 5, December 2010.

[5] T. Wong, P. Cote, P. Gely, "Final Exam Timetabling: A Practical Approach," IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), vol, 2, pp. 726-731. IEEE Press, Los Alamitos, 2002.

[6] N. Pillay, W. Banzhaf, "A Developmental Approach to the Uncapacitated Examination Timetabling Problem," Lecture Notes in Computer Science, Springer Link, vol. 5199, pp. 276 – 285, 2008.

[7] E.K. Burke, J.P. Newall, R.F. Weare, "Solving Exam Timetabling Problems with the Flex-Deluge Algorithm," International Conference on the Theory and Practice of Automated Timetalbing (PATAT), pp. 370 – 372, 2006.

[8] E.K. Burke, A. Eckersley, B. McCollum, B. Petrovic, R. Qu, "Hybrid Variable Neighborhood Approaches to University Exam Timetabling," Technical Report NOTTCS-TR-2006, School of Computer Science and Information Technology, Nottingham, UK, 2006.

[9] M. Caramia, P. Dell'Olmo, G.F. Italiano, "Novel Local-search-based approaches to university examination timetabling," INFORMS Journalof Computing 20, pp. 86 – 99, 2006.

[10] S. Innet, N. Nuntasen, "University timetabling using evolutional computation," WSEAS Transaction on Advances in Engineering Education, issues 12, vol. 4, pp. 243 – 250., Dec. 2007.

[11] N. Nuntasen, S. Innet, "A Novel Approach of Genetic Algorithm for Solving University Timetabling Problems: a case study of Thai University," 7th Conference on 7th WSEAS International Conference on Applied Computer Science, vol. 7, pp. 246 – 252, Italy, 2007.

[12] P. Khonggamnerd, S. Innet, "On Improvement of Effectiveness in Automatic University Timetabling Arrangement with Applied Genetic Algorithm", International Conference on Computer Sciences and Convergence Information Technolgy, Seoul, Korea, 2009