OPTIMIZATION OF TEST KEEPER SCHEDULING USING GENETIC ALGORITHM AT INFORMATICS DEPARTMENT PETRA CHRISTIAN UNIVERSITY

Kristo Radion Purba
Informatics Engineering, Petra Christian University
kristo@petra.ac.id

***Abstract*— At Informatics Department, Petra Christian University, before mid or final exam, there will be a manual process to schedule the test keepers for every exam session. The test keepers are lecturer assistants (assistant is an appointed student to help lecturer in class). For an exam session, the keeper can be 1 up to 3 people, depending on the exam's participant. These manual process is considering many factors, i.e. the assistant's batch (year), the average of exam's participant batch(year), gender combination of the keeper, evenness of the exam keeping of every assistant, the character of the assistant itself, and the exam schedule of the assistant. These factors are considered upon picking every exam sessions' keeper, which is taking a lot of time and knowledge, and this process is done twice a semester by an exam coordinator (lecturer). In this paper, will be designed an application that is using genetic algorithm to automatically assign the test keepers for every exam. The result of the application is tested during the mid-exam and final-exam early semester of 2016, and the application is giving a good result, with the accuracy of 90.23%, in which the 9.77% is some minor changes that is required to make the test keepers more suitable.**

***Index Terms*—About; Genetic algorithm; Test keeper; University exam.**

I. INTRODUCTION

At Informatics Department, Petra Christian University, like any other university, there are 2 **exam term** for every semester, mid exam and final exam. Some classes can have or not an exam session. For example, game development class is having mid exam but not final exam, because the final exam is replaced with a game project. So usually, almost all classes are having mid exam, but final exam is usually 80-90% of all classes.

Usually there are 3 **exam sessions** for each day, and the exam term will run for 7 days. In each session, usually there are 2 up to 4 **exams**. The total number of exams in an exam term is usually around 70-90, depends on the count of the opened class in the semester.

In every exam, there will be 1 up to 3 test keepers, depending on the number of participants. Test keepers are lecturer assistant. In an exam term, there are usually 130-150 keepers.

These test keepers will be scheduled by an exam coordinator (lecturer). These processes are taking a lot of time and effort, because there are so many constraints that will be considered during the scheduling, i.e. assistant's and participant batch (year), gender combination of the keeper, evenness of the exam keeping of every assistant, the character of the assistant itself, and the own exam schedule of the assistant itself.

In this research, an implementation of genetic algorithm will be proposed to optimize the scheduling of test keepers for exam sessions at Informatics Department Petra Christian University.

II. LITERATURE REVIEW

In this chapter, genetic algorithm will be discussed, as it is the algorithm that will be used in this research. Also, there will be a brief introduction about the Informatics Department Petra Christian University, and the exam system.

A. Genetic Algorithm

In the computer science field of artificial intelligence, a genetic algorithm (GA) is used to find solutions of problems that are not obvious, or not having a certain formula, or the searching space of the problem is not clear. [1]

The complete process of GA is described below: [2]

a. Generate the initial chromosomes, usually 10-20 chromosome in a generation. Chromosome is a set of genes. Chromosome contains the solution. [3]
b. Create fitness function and assign to each chromosome. The fitness value is determined depending on what kind of problem needed to be solved. [4]
c. Record the best gene before the copulation process. This will be used later in the elitism process.
d. Perform selection to the chromosomes, here we can use tournament selection, roulette, proportionate, rank, steady state selection, etc. In this research, roulette will be used. In the roulette, chromosomes that are having bigger fitness will have more chance.
e. Crossover process. The crossover process is done by swapping the genes of chromosome A with B, from start offset to end offset. The start and end will be picked randomly.
f. Mutation process. For a chromosome, genes in several positions (random position) will be changed to a new random gene.
g. Elitism process. Pick the best chromosome before the copulation process, and it will be used to replace the worst chromosome after the copulation process. This process is to make sure that the GA will never create worse generation after copulation.

h. Repeat the process from b-f, until a stopping condition is met. These stopping conditions can vary depending on the case, it can be number of generations, time limit, acceptable quality, or some specific condition i.e. more than 50% of the population is having the same fitness value. [5]

B. Related Work

There are many researches that are using genetic algorithm to solve problems, such as in medical field [6], where GA is used to improve disease screening, treatment planning, diagnosis, etc. In the medical field, data are really big, thus a meta-heuristic like genetic algorithm is suitable to be used.

Other implementation is [7], where GA is used to schedule precedence-constrained task that is using two fitness function, the first one is to minimize the total execution time, and the second is satisfy the load balance.

III. Web and Mobile Application Design

In this research, a web and mobile application is build to help the scheduling process. The flow for the scheduling process can be seen in Figure 1.

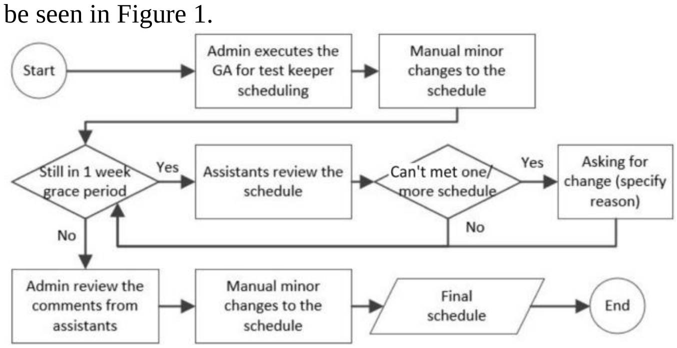

Figure 1: The Scheduling Process

The web application is used by the administrator to schedule automatically using GA, or manually. What will be expected to be done by the web application is that the GA will schedule automatically, and web administrator will do some minor changes if needed.

After the scheduling process, the schedule will be reviewed by assistants, using a mobile application, for 1 week of grace period. During this period, assistant will check for their own daily schedule whether it can be fulfilled by them. If it can't be met, because of some private reasons, they can give comment to the administrator (exam coordinator) whether they can be replaced by someone else.

After the grace period is finished, the schedule is finalized, and the assistant can't no longer give comments. But still, the administrator can change if it's really needed. The mobile application is still needed for the assistant to check for their own schedule during the exam term.

IV. Genetic Algorithm For Scheduling

The Genetic Algorithm (GA) will automatically schedule test keepers, using considerations that are replicated from human knowledge.

A. Chromosome Design

First, exam sessions will be converted to test keeper slots, as seen in Figure 2. An exam with 0-20 participants will be kept by 1 assistant, 21-45 kept by 2, 46 or more kept by 3. One slot will later be filled with one assistant.

Date	Start	Class	Sem	Lecturer	Place	Participant	Keeper
04-04-2017 (Tue)	7:30	Internet of Everything (A)	2	Ir. Resmana Lim,	P.703	62	Keeper 1 Keeper 2 Keeper 3
	7:30	Internet of Everything (B)	2	Indro Setiawan, S.T.	P.704	64	Keeper 4 Keeper 5 Keeper 6
	7:30	Computer Security	8	Agustinus Noertjahy	P.710	39	Keeper 7 Keeper 8
	10:30	Risk Management	8	Adi Wibowo,	P.710	42	Keeper 9 Keeper 10
	10:30	Risk Management	8	Adi Wibowo,	P.620	23	Keeper 11 Keeper 12
	10:30	Algorithm	1	Anita	Lab SI	12	Keeper 13

Figure 2: Test Keeper Slots

The keeper slots are usually around 130-150, depending on the number of opened class in that semester. After making these slots, every assistant will be assigned an ID starting from 1, as seen in Figure 3. Also, here the properties of each assistant is assigned. The number of assistants are usually around 20-30, every year there will be graduating assistants and replaced with the new batch. A student can become assistant starting from semester 2, and have to met certain criterions.

ID	Name	Batch	Firm	Discipline	Gender (1 male, 0 female)
1	Michael S	2013	3	3	1
2	Andre	2014	3	3	1
3	Darian	2014	2	2	1
4	Calvin	2014	2	2	1
5	Liliana	2014	3	3	0
6	Vannesa	2014	2	2	0
7	David	2015	2	0	1

Figure 3: Assistant Data

The chromosome design is based on the keeper slots. Each slot will be filled with the ID of the assistant. So the chromosome length will be around 130-150 small integers. The integer length is 2 digits. The example of chromosome can be seen in Figure 4. This figure is related with Figure 2.

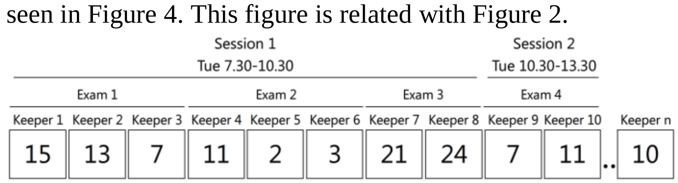

Figure 4. Chromosome design

In the test keeper slots, several global constraints must be fulfilled, such as there should be no duplicated test keeper in the same session. The global factors are mentioned sub-chapter B.

B. Fitness Value Factors

The factors for the considerations are split into two parts, individual factors (consideration of each exam) and global factors. The individual factors are:

a. Assistant's batch (year). Usually, assistant will be prioritized to do exam keeping for students one-year younger than the assistant's batch. An assistant will not keep an exam if the participant is older than him/her. For this, at least the younger assistant will be accompanied by the older one. But still, it's better if all test keepers are older than the participants. It's also not good if the assistant is far older than the participants, because we will run out of old assistants very soon. So, one year older or same age is the ideal one. The assistant batch (ab) value is the average of assistants' batch in the exam. If for example there are 2 people keeping an exam, from batch 2013 and 2014, the average is 2013.5.

b. The majority of exam's participant batch (year). If for example, the majority of participants are of batch 2015, the test keeper should ideally come from batch 2014.
The assistant batch (ab) and the participant's batch (pb) will be compared, resuting the batch difference score (bd), using the following lookup Table 1. Here, older and younger refers to age. If for example the ab = 2013.5, and the pb = 2014 (majority), the (pb-ab) = 0.5, thus the bd will be 1. Higher is better.

Table 1
Batch difference score

pb - ab	bd
<=0 (participant is older)	0
>0 to <=1	1
>1 to <=2	0.67
>2 to <=3	0.33
>3	0

c. Gender combination of the keeper. Female will be prioritized not to be alone, and male will be prioritized with female. If male if almost always with male, we will run out of male quickly. Male is considered more responsible and firm. The gender score (g) is described in Table 2, where higher score means more prioritized.

Table 2
Gender Score

Gender combination	g (score)
Male	1
Female	0.67
Male, male	0.67
Male, female	1
Female, female	0.33
Male, male, male	0.33
Male, male, female	0.67
Male, female, female	1
Female, female, female	0.3

d. Character of the assistant. Firmness and discipline will take place into consideration. These numbers will be set by administrator, with the scale of 0 to 3. An assistant that is known to be not firm will be combined with a firm one. Discipline factor is also the same. An assistant that is known to be a late-comer is considered as not discipline. We want the exam to start on time, so at least one should be a discipline person.
The character of each assistant will be scored, the value is between 0 and 1, and then will be averaged across all keeper in the exam, calculated using a simple formula in equation 1.

$$c = \frac{\sum_{i=0}^{n-1}(firmness_i/6 + discipline_i/6)}{n} \quad (1)$$

Where n is test keeper slot

The global factors are:

a. Evenness of the exam keeping of every assistant. Usually, each assistant will have 6-8 exam keeping. These number should be even across all assistants. So the deviation of these numbers should be as small as possible. The number of exam keeping of an assistant usually maxed to 8 in an exam term. So, the deviation of exam keeping count should be no more than 8. The formula for ev (evenness) is shown in equation 2 and 3.

$$s = \sqrt{\frac{\sum_{i=0}^{n-1}(x_i - \bar{x})^2}{n-1}} \text{, s will be limited to 0-8} \quad (2)$$

Where:
x = number of exam keeping of assistant i
n = number of assistants
The result of s will be limited to 0-8

$$ev = \frac{8-s}{8} \quad (3)$$

From equation 3, can be seen that if the deviation is minimum or zero, the ev will be 1, which is the best.

b. Exam schedule of the assistant itself. Because assistant is a student, they are having their own exam sessions as participant. Therefore, they will not be an exam keeper when they have their own exam. If this factor is happening, this chromosome will be reshuffled.

c. Test keepers will not be duplicated in the same exam session. In a day, there are 3 exam sessions, i.e. the first session is 07.30-10.30, the second is 10.30-13.30, the third is 13.30-16.30. If this factor is happening, this chromosome will be reshuffled.

d. An assistant can't have multiple exam keeping for the same exam session. So if this is happening, this chromosome will be reshuffled.

C. Fitness Value

A fitness value will be calculated for a chromosome, based on the fitness value factors previously. The fitness value is shown in equation 4.

$$fitness = \left(\frac{\sum_{i=0}^{n-1}(bd_i + g_i + c_i)}{n} + ev\right) : 4 \quad (4)$$

Where:
bd = batch difference score
g = gender score
c = character score
ev = evenness of exam keeping
n = number of exam

From this equation, can be seen that the bd, g, c is coming from the individual factors, or the factors coming from one exam. One exam can consist 1 up to 3 genes (genes is test keeper slot). The individual factors will be averaged first, and then will be added with ev as the global factor. After that, they will be divided by 4, thus creating the fitness value between 0 and 1 floating point. The 3 other global factors will directly re-shuffle the chromosome if happened.

D. Crossover and Mutation

The crossover rate of a GA process is usually higher than mutation and can be above 50% of probability. Mutation rate should be kept low, usually just around 10%. High crossover rate will lead to a global optimum, while too low will make the

GA process longer. Mutation is good to prevent local optimum, but a too high number will lead to too many variations of the genes and causing the GA process to be longer.

The mutation and crossover rate is determining the success of genetic algorithm, and usually done using trial and error process [8], as can be seen in Table 3.

In this research, upon several testing, the ideal crossover rate is 60%, and mutation is 10%. The crossover will exchange gene from a random start and random end offset between two chromosomes by 60% chance. The mutation will randomize a gene in some random positions picked by 10% chance.

V. SCHEDULING TESTING RESULT

In this chapter, the genetic algorithm will be tested. First, we will test the basic parameters i.e. mutation rate and crossover rate. The number of population for each generation is 20. Changing this number doesn't affect much. The algorithm will be stopped if it's having the same highest fitness value for 5 times in a row. The chromosome that is having the highest fitness will be picked as the solution. The GA will be tested 3 times to check whether it is resulting the same chromosome after each run.

A. Mutation and Crossover Rate

Changing the mutation and crossover rate doesn't affect much, and can vary depends on the initial generation, but we will pick the best rate after the testing. Table 3 show how these rates affects how long the GA will run before it met the stopping condition.

Table 3
Mutation and Crossover rate testing

Crossover %	Mutation %	# of Generation	Time (minute)
55	15	531	13.14
55	10	544	12.98
60	15	522	12.32
60	**10**	**509**	**12.16**
70	15	514	13.06
70	10	525	11.92

From table 3, the 60% and 10% rate will be picked for the crossover and mutation rate, respectively, based on the number of generations before finding the best.

B. Genetic Algorithm Execution

In this section, we will run the GA for 3 times each for 2 exam terms (during the mid-exam and final-exam early semester of 2016). The result shows that it usually generates 3 different solutions, but the fitness value is almost the same. Because of the chromosome can vary really much, there's not a problem if we pick one of these solution, because they are equally good.

For the mid exam, the result can be seen in Figure 5 and Table 4. For the final exam, the result is in Figure 6 and Table 5. Because it is from the same semester, the number of exams are the same which is 76 exams, and needed 154 test keepers for the mid exam (all classes are having exam), and 131 test keepers for final exam (only 64 exams are having exam, the others are project based). From these charts, can also be seen the growth of fitness values. The initial generation is set the same, so the result is not dependent on the initial generation.

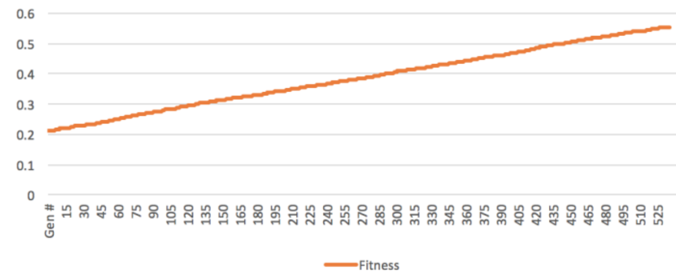

Figure 5a. Mid Exam early 2016: Execution #1

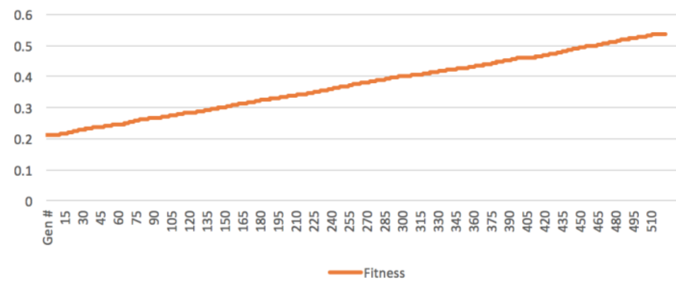

Figure 5b. Mid Exam early 2016: Execution #2

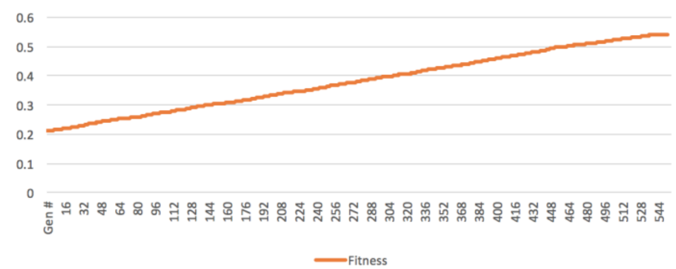

Figure 5c. Mid Exam early 2016: Execution #3

In table 4, shown the details of the GA execution.

Table 4
Mid Exam early 2016: Execution Details

Execution #	# of Generation	Fitness result	Time (minute)
1	**537**	**0.552228571**	**13.78**
2	523	0.537457143	12.69
3	553	0.541885714	14.62

The execution 1 result is used for the solution for mid exam early 2016, which is the names of the test keepers for every exam. The solution is checked manually by the exam coordinator based on usual experience and knowledge, and it requires 10.38% of changes (16 of 154 keeper names) to perfect the result. So here we can say that the accuracy is 89.62%.

The result for GA execution for final exam can be seen below. The GA execution slightly faster, because the number of exam are lower, thus the chromosomes are shorter.

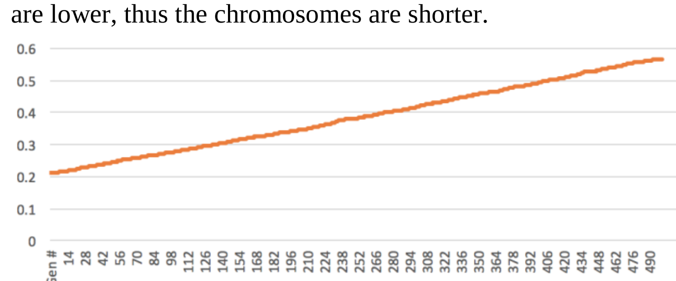

Figure 6a. Final Exam early 2016: Execution #1

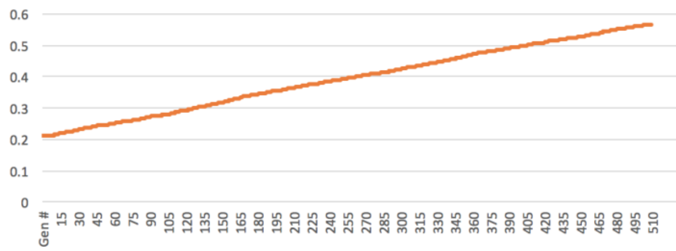

Figure 6b. Final Exam early 2016: Execution #2

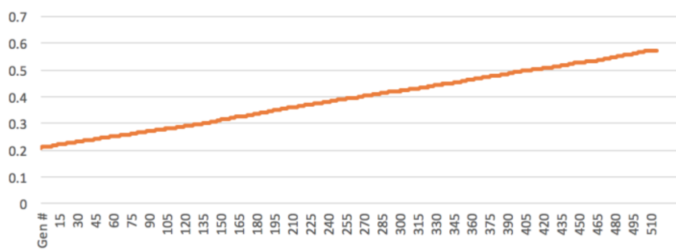

Figure 6c. Final Exam early 2016: Execution #3

Table 5
Final Exam early 2016: Execution Details

Execution #	# of Generation	Fitness result	Time (minute)
1	498	0.565514286	10.93
2	505	0.565085714	10.48
3	**514**	**0.573228571**	**11.08**

The execution 3 result is used for the solution for final exam early 2016. After some manual checking, it requires 9.16% of changes (12 of 131 keeper names) to perfect the result. So here we can say that the accuracy is 90.84%.

Here is the example result of the test keeper schedules, exported to Excel file, can be seen in Figure 7.

Date	Start	Class	Sem	Lecturer	Place	Participant	Keeper
04-04-2017 (Tue)	7:30	Internet of Everything (A)	2	Ir. Resmana Lim,	P.703	62	Yunike,Glenn, Gunawan
	7:30	Internet of Everything (B)	2	Indro Setiawan, S.T.	P.704	64	Michael S, Albert
	7:30	Computer Security	8	Agustinus Noertjahy	P.710	39	Gloria, Yohanes
	10:30	Risk Management	8	Adi Wibowo,	P.710	42	Glenn, Jeremy
	10:30	Risk Management	8	Adi Wibowo,	P.620	23	Aaron, Hansen
	10:30	Algorithm	1	Anita	Lab SI	12	Ricky,Fransisco

Figure 7. The Schedule, exported to Excel

VI. Conclusion

From the genetic algorithm execution for scheduling test keepers, can be seen that the accuracy is pretty good, but still needing some knowledge to fix the schedule. The accuracy is 90.23%, the average from 2 exam terms in the same semester.

To make the schedule more perfect, it will require some more work, such as adding neural network which involves training process which also can make the process faster. Also, the GA fitness value can be tweaked more to produce better result.

VII. Bibliography

[1] S. Sivanandam and S. Deepa, Introduction to Genetic Algorithms, New York: Springer, 2008.

[2] R. Bhattacharjya, Introduction To Genetic Algorithms, Guwahati: Indian Institute of Technology Guwahati, 2013.

[3] R. Chakraborty, Fundamentals of Genetic Algorithms, Guna, 2010.

[4] M. O. Odim, B. O. Oguntunde and O. O. Alli, "On the Fitness Measure of Genetic Algorithm for Generating Institutional Lecture Timetable," *Journal of Emerging Trends in Computing and Information Sciences 4.4,* 2013.

[5] M. Hristakeva and D. Shrestha, "Solving the 0-1 Knapsack Problem with Genetic Algorithms," *Midwest Instruction and Computing Symposium,* 2004.

[6] A. Ghaheri, S. Shoar, M. Naderan and S. S. Hoseini, "The Applications of Genetic Algorithms in Medicine," *Oman Medical Journal,* pp. 406-416, 2015.

[7] F. A. Omaraa and M. M. Arafa, "Genetic algorithms for task scheduling problem," *Journal of Parallel and Distributed Computing,* pp. 13-22, 2010.

[8] K. Deb and S. Argrawal, "Understanding Interactions Among Genetic Algorithm Parameters," *Foundations of Genetic Algorithms 5,* 1998.