Moodle Plugins for Quiz Generation using Genetic Algorithm

Muhammad Rian Fakhrusy
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
13511008@std.stei.itb.ac.id

Yani Widyani
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
yani@informatika.org

Abstract— **This paper describes the development of Moodle plugins to generate quiz using Genetic Algorithm. Moodle is an open source learning management system. One of its features is to generate a quiz consisting random questions which is chosen form question bank. If the quiz is created randomly, it can't accommodate specific constraints. Genetic algorithm provides an opportunity to generate quiz which approximate the constraints. Our research follows previous researches done by Xiaoqin & Yin (2009) and Huang & Wang (2008), modifies them to be applied in the development of Moodle quiz generation plugins. Several attributes which are selected from previous researches, are augmented to the each question in question bank. The generated quiz is also augmented by corresponding attributes. Quiz generator needs constraints as input. Genetic algorithm will generate a quiz whose attributes approximate the constraints. The generated quiz can be used in tests after this step. At the moment, there is no option of using genetic algorithm for quiz generation in Moodle. The plugins consist of one main quiz plugin and several question type plugins. The main quiz plugin was developed so that genetic algorithm can be used in quiz generation. Meanwhile, several question type plugins were developed to provide augmented questions which quiz plugin is going to use. These plugins have been successfully implemented and integrated to Moodle..**

Keywords—genetic algorithm; Moodle; plugins; quiz generation

I. Introduction

Learning management system (LMS) has been used widely by universities to assist learning activities. Assignments, forums, quizzes and lessons are example of these activities. Some LMS are open source, thus can be modified and distributed freely. Moodle is the most popular LMS in Europe [1]. There have been 103,634,021 registered Moodle users by May 30, 2017 [2]. Moodle is open source, thus can be customized by anyone. Customization is done by creating plugins so that it does not interfere with Moodle main functionality. There have been over 1000 plugins developed in Moodle. These plugins were developed in many categories: activities, reports, repositories, themes, etc. [3].

Plugins can enhance learning activities in Moodle with a new functionality. One of the activities is creating a new quiz. The creation is done by selecting some questions from question bank manually then adding them into the quiz [4]. However, creating the quiz manually is cumbersome and time-consuming. Moodle also provide a way to generate the quiz questions automatically by using random algorithm [4]. However, the quiz generated by random algorithm can't be configured according to user input.

Generating quiz from question bank according to user input is a constraint optimization problem [6]. Constraints in this problem belong to soft constraints. These constraints have to be taken into consideration when generating a quiz. The generated quiz must be as close as possible to the input constraints.

Genetic algorithm is one of approach in solving optimization problem. It is suitable for problem which has multiple local optima and many parameters [5]. Some study using genetic algorithm are conducted to generate quiz questions which approximate the constraints. In [6], genetic algorithm produces better results than random algorithms to satisfy the constraints. However, Moodle has yet to implement quiz generation using genetic algorithm as its functionality. It might be possible to add the functionality to Moodle as plugins.

II. Related Topics

A. Quiz Generation

Quiz generation is creation of a quiz consisting of a set of questions chosen from question bank automatically. Question bank must be filled with questions before a quiz is generated from it. Each question in question bank has some attributes which is needed in quiz generation. The quiz which is generated also has some attributes according to the attributes of all of its questions.

Quiz generation is begun after user insert input, which are soft constraints of the quiz which is generated. Soft constraints are not like hard constraints where the conditions are required to be satisfied. Soft constraints will penalize value in the objective function based on the extent that the condition on the variable are not satisfied [7]. Quiz questions generation system take the constraints into consideration when generating a set of questions which is used by the quiz. The generated quiz attributes must be as close as possible to the soft constraints. Fig. 1 displays the diagram of quiz generation system [8]. There are some terminologies in quiz generation:

Questions – Statements that should be responded with the best answers
Quiz – A test consisting of some questions
Attributes – The properties of quiz or questions, some quiz attributes can be derived from all its questions attributes.
Constraints – User input on how the quiz attributes should become after the quiz is generated

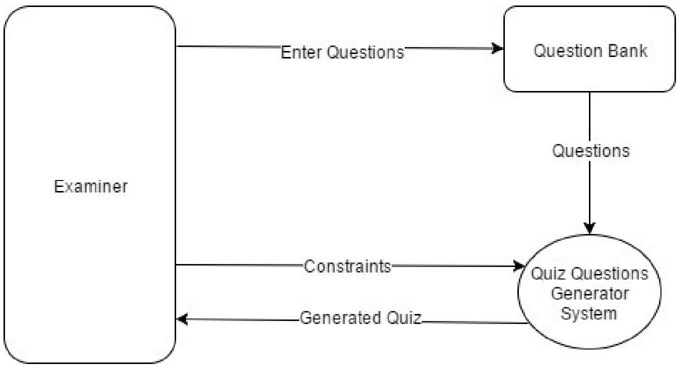

Fig. 1. Quiz generation system [8]

B. Genetic Algorithm

Genetic algorithm is a method to solve the optimization problem. The algorithm is inspired by the principles of natural selection [9]. Genetic algorithm uses a lot of biological science terms: selection, crossover, mutation, fitness, population, individual, chromosome, gene, etc.

Individual is a candidate for problem solution. Population is group of all individuals. Chromosome is a code that forms an individual. Chromosome consists of several genes. Each gene in chromosome represents an attribute which is part of a solution. Chromosome is usually encoded by a binary string consisting only of 0 and 1, even though it can also be encoded in other ways. Every chromosome is assigned by a value called fitness. Fitness is a value describing how well the candidate solution solving the problem in consideration. Fitness is determined by an equation called fitness function. Fitness function must be good so that the candidate solution converge to the real solution [10].

Genetic algorithm begins with an initial population which is created randomly. Each chromosome in the population is assign by a fitness value determined by fitness function. The higher the fitness, the higher the chance a chromosome is being selected as a parent of next generation population. The selected chromosome will go through a process called crossover and mutation.

. Crossover is genetic algorithm operator which produces a new chromosome from the genes of the parent chromosomes combination. Mutation is a genetic algorithm operator that changes genes value in a chromosome. Fig. 2 and Fig. 3 display the process of mutation and crossover respectively.

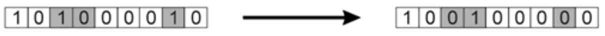

Fig. 2. Mutation in genetic algorithm [11]

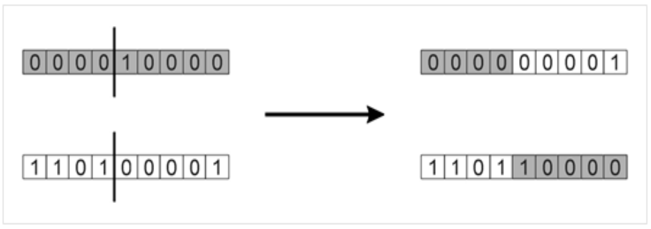

Fig. 3. Crossover in genetic algorithm [11]

The process of evolution to the next generation is repeated until certain stop condition [11]. Fig. 4 displays genetic algorithm cycles.

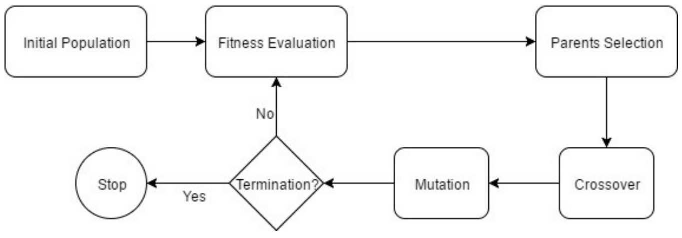

Fig. 4. Genetic algorithm cycles [12]

C. Quiz Generation using Genetic Algorithm

There are some related works about generating quiz using genetic algorithm [6,15]. In [15], several attributes are used in every question: id, score, type, knowledge, diff, div, level, time. These attributes were being considered as input by genetic algorithm in quiz generation. In the research, quiz generation was done using constraints below as input:

1) *Total score:* Sum of every questions score in a quiz.
2) *Average diff:* Average of every quiz questions diff in proportion with score attribute.
3) *Each type score:* The number of questions which have a certain value of each type attributes within a quiz.
4) *Each knowledge score:* The number of questions which have a certain value of each knowledge attributes within a quiz.
5) *Each level score:* The number of questions which have a certain value of each level attributes within a quiz.
6) *Div value:* Average of every quiz questions div in proportion with score attribute.
7) *Total time:* Sum of every questions time in a quiz.

In [6], several quiz attributes is used: code, points, difficulty, time, ability level and distinguishing degree. In the research, constraints below were used as input:

1) *Total points:* Sum of every questions points in a quiz.
2) *Difficulty factor:* Average of every quiz questions difficulty in proportion with score attributes.
3) *Solution time:* Sum of every questions time in a quiz.
4) *Ability level test questions points:* The number of questions that have a certain value of each ability level attributes within a quiz.
5) *Distinguishing degree factor: Average of every quiz questions distinguishing degree in proportion with score attributes.*

Fitness function used in [15] to determine whether a chromosome is a good candidate solution is defined as below.

$$f = \frac{1}{1+\sum_{i=1}^{m} w_i|e_i|} \quad (1)$$

e_i is normalized relative error (NRE) of question attribute i. w_i is weight for question attribute i. m is number of attributes that is used by genetic algorithm as a quiz generation constraints. NRE is calculated by

$$NRE = \frac{|Constraint_i - Candidate_i|}{Constraint_i} \quad (2)$$

$Constraint_i$ is the value for attribute i which examiner want to achieve. $Candidate_i$ is the real value which is calculated from all questions used in a candidate quiz.

D. Moodle Plugins

Moodle is abbreviation of modular object-oriented dynamic learning environment. Modular means it groups codes and data depend on their functionality [13]. Creating plugin is the most maintainable way to add new functionality into Moodle. Moodle plugins are divided into several types. Some types of plugin are correlated to the quiz creation: activity modules, quiz reports, question access rule, question types, question behaviors and question import/export formats [14].

Quiz generation needs questions and constraints as inputs. Questions are needed as the source of generated quiz questions. In Moodle, these questions are stored in question bank. Constraints are needed as a guide to produce a desired quiz. Fig. 5 displays the role of plugins in quiz generation.

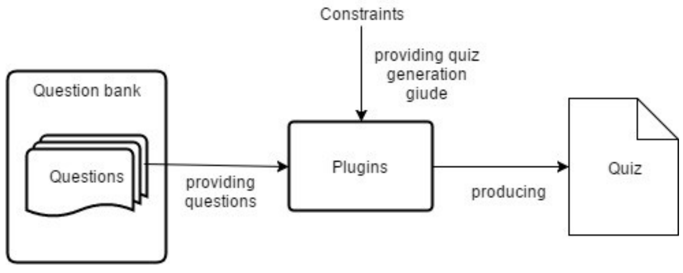

Fig. 5. Plugins role in quiz generation

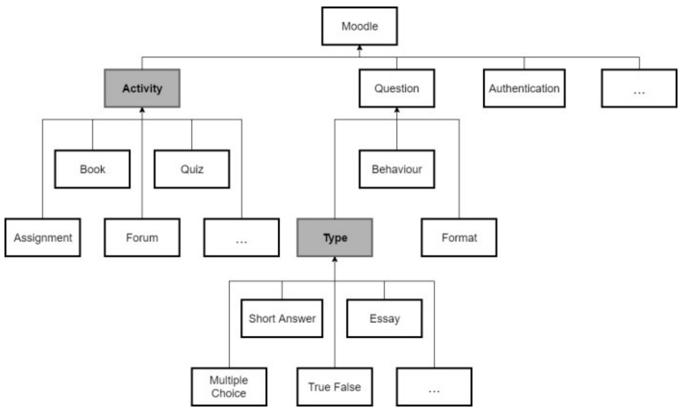

Fig. 6. Location of activity and question types plugins in Moodle structure

Quiz generation plugin belong to activity modules plugins in Moodle. This plugin can't be used independently. Some question type plugins are needed to support additional attributes which every question has. Both of these kinds of plugins work together in Moodle quiz generation. Fig. 6 displays the location of these kinds of plugins in Moodle structure.

III. The Proposed Plugins

A. System Architecture

Two kinds of plugins were implemented in quiz generation system: activity and question types. Both of these kinds of plugins were implemented by modifying existing modules in Moodle. Activity plugin is the main plugin of the system. The plugin was modified from quiz modules. It was renamed as "generated quiz". Question types plugins is the supporting plugins. There are five question type plugins which was modified from existing Moodle question type: essay, multiple choice, matching, true false and short answer. These plugins were renamed by adding "for generated quiz" at the end of the question type names. Architecture of the system is shown in Fig. 7. Cyan-colored parts in the diagram are parts of the system which were added or modified from Moodle.

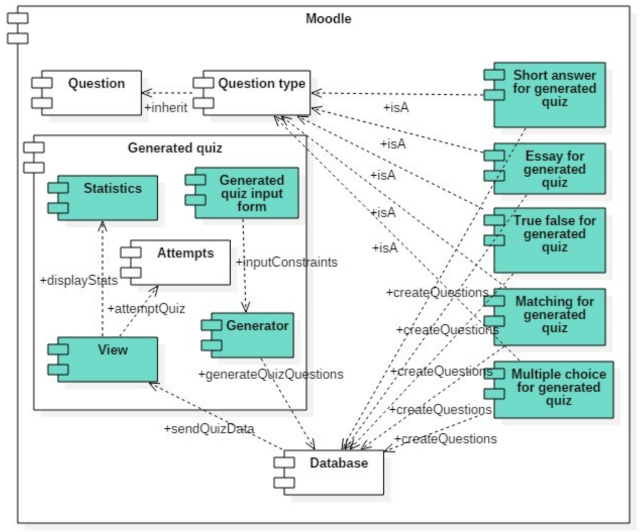

Fig. 7. Quiz generation system architecture

Modification in generated quiz was done by adding generator and statistics parts to quiz module. Input form and view parts of the generated quiz were also modified from quiz modules. Generator is a part of generated quiz which uses genetic algorithm to generate questions used in a quiz. Statistics is part of generated quiz which displays the comparison between constraints as input and generated quiz information as output. Modification in input form was done so that the generated quiz was able to get quiz information including constraints from examiner. View parts were modified so that there is an option to view statistics after a quiz is being generated. However, attempts part of the quiz wasn't modified.

Modification in questions type was done by adding four more attributes to the question type database. These attributes are difficulty, distinguishing degree, time and level. The question type database primary key is a foreign key in Moodle question database. A way to import/export questions from/to external file was also added to the question type plugins. Moodle XML is the format of the external file.

B. Plugins Capabilities

Plugins are capable of doing tasks below:

1) Creating questions: plugins can create questions having attributes which is needed by genetic algorithm in generating quiz.

2) Exporting questions: plugins can save questions data from Moodle question bank to external file.

3) Importing questions: plugins can load questions data from external file to Moodle question bank.

4) Creating quiz: plugins can create a quiz which has constraints information.

5) Generating quiz questions: plugins can use genetic algorithm to generate quiz questions which approximate the constraints.

6) Sorting quiz questions: plugins can sort quiz questions based on one of the questions attributes in ascending or descending order

7) Viewing quiz generation statistics: plugins can display statistics comparing constraints as an input and generated quiz information as an output.

C. Constraints and Attributes of Generated Quiz

Plugins can only accept certain constraints as input. All of the constraints are soft constraints. The constraints are depended on quiz attributes. Quiz attributes value is depended on the value of all questions it consists of. [6,15] were used in deciding what attributes the plugins were going to use. These researches were using different name for some attributes with the same purpose. There are three kinds of equations about how quiz attributes calculated from questions attributes:

1) Sum equation (SUM):

$$\text{SUM} = \sum_{i=1}^{n} att_{ij} \tag{2}$$

Sum equation calculates the total value of each attribute j in a quiz which has n questions.

2) Average equation (AVG):

$$\text{AVG} = \frac{\sum_{i=1}^{n} score_i att_{ij}}{\sum_{i=1}^{n} score_i} \tag{3}$$

Average equation calculates the average value of each attribute j in a quiz which has n questions in proportion with score attribute.

3) Count equation (COUNT):

$$\text{COUNT}_k = \sum_{i=1}^{n} c_i a_{ij} \quad ; c_i = \begin{cases} 1 & a_{ij} = k \\ 0 & a_{ij} \neq k \end{cases} \tag{4}$$

Count_k is the number of quiz questions that have j attribute with the value of k.

Question attributes used in the plugins is chosen from question attributes used in [5,14] researches. New names for the attributes were made because these researches used different names for some attributes that have similar purpose. The quiz constraints which were compared with the constraints in generating quiz involving these attributes were also analyzed. Table I. contains the list of new names for the attributes. The table also mentions the equation used in quiz attributes which are depended on questions attributes.

TABLE I. COMPARISON OF ATTRIBUTES THAT IS USED IN [6,15]

Question attributes universal name	**[15]**		**[6]**	
	Questions attributes name	***Equation of quiz attributes***	***Questions attributes name***	***Equation of quiz attributes***
id	id	-	code	-
score	score	SUM	points	SUM
type	type	COUNT	type of question	-
difficulty	diff	AVG	difficulty factor	AVG
chapter	knowledge	COUNT	knowledge points	-
distinguishing degree	div	AVG	distinguishing degree	AVG
time	time	SUM	solution time	SUM
level	level	COUNT	ability level	COUNT

All of these question attributes were used in quiz generation plugins. Id, score, type and chapter attributes have already been implemented in Moodle, although some of them have different names. In Moodle, score is default mark, type is question type and chapter is category. Time limit is one of the information that can be inputted when creating a quiz in Moodle. It is also possible to make time limit as constraints for the time attributes. However, Moodle question types don't have time as its attribute. So, the implementation of new question type plugins was needed. Difficulty, distinguishing degree and level attributes were also need to be implemented because Moodle question types haven't implemented them yet. They were added as one of the attributes question type plugins used, along with time attribute.

Constraints are user input, quiz attributes are the property of the quiz. The data of constraints and quiz attributes are saved in the database with different name. The following names are given for the constraints: total score, number of questions for each type, number of questions for each chapter, average difficulty, average distinguishing degree, total time, and number of questions for each level. Quiz attributes have slightly different names from the constraints. In quiz attributes, all of those names were added by "real" in the beginning, meaning that these are the real value of the quiz attributes rather that value which user wanted (constraints).

D. Applying Genetic Algorithm

In quiz generation, a genetic algorithm population represents several candidate quizzes. An individual represents a candidate quiz. A gene represents a question in the quiz. Genes are encoded by a set of numbers that represent question ids in question bank. This set of numbers is also called a chromosome. Individual and chromosome are often used interchangeably. The difference is that individual is the candidate solution (in this case, a quiz) and chromosome encodes the individual (in this case, a set of question ids which form the quiz) [16]. Fig. 8 displays a population used in quiz generation.

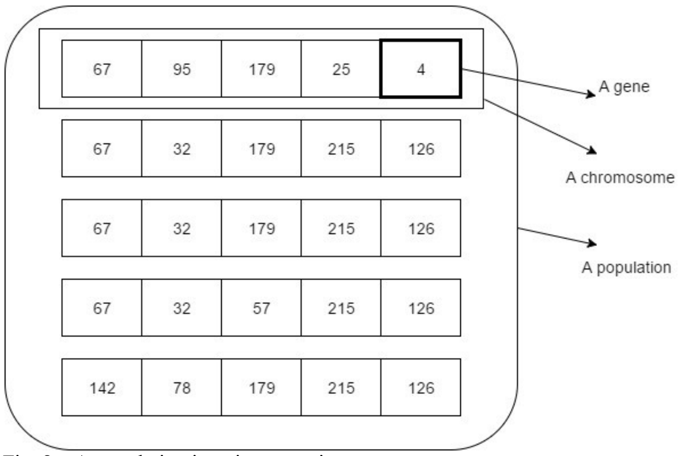

Fig. 8. A population in quiz generation

Genetic algorithm process is begun by creating an initial population randomly. This population consists of several chromosomes. The number of chromosomes in a population is one of the configurable genetic algorithm settings. Each gen in chromosomes represents a different question in a quiz. That's why the number of gen in every chromosome is identical to the number of questions in a quiz. Every chromosome in initial population is assigned by a fitness value by using (1). In this step, quiz attributes is compared with the constraints for every chromosomes in the population. The more identical value in the quiz attributes and the constraints, the higher the fitness that the chromosomes get. In the plugins, the value of weight in (1) can be set by user.

After that, elitism selection is done to preserve some percentage of the chromosomes for the next generation. Elitism selection selects a percentage of the best chromosome from the current generation to carry over to the next generation unaltered [17]. Then, crossover and mutation is done with a certain probability. The process is repeated until it reaches certain number of population generations. All of these settings (elitism ratio, crossover ratio, mutation ratio and number of population generations) are configurable.

There are three conditions where genetic algorithm terminates. If one of the conditions is satisfied the process terminates. These conditions are fitness reaches maximum value, the population converges or the maximum number of generation has been reaches. The population converges if the value of best chromosome has not been changed within 0,001 difference for 100 generations.

E. Plugins Implementation

Implementation was done by using PHP language because it is a language that Moodle use. The specification of the system implementation environment is shown in Table II.

TABLE II. SYSTEM IMPLEMENTATION ENVIRONMENT SPECIFICATION

Aspect	Value
Processor	Intel Core i7-5500U CPU @ 2.40 GHz
Memory	12GB
Server	Apache 2.4.17
Client	Google Chrome
Operating System	Windows 10 64-bit
Software	Moodle 3.1
Programming Language	PHP 5.6.16
Database	MySQL 5.7.9

IV. EVALUATION

Dummy data were used in the evaluation. The data consisted of 389 questions which were obtained from Moodle quiz questions repository [18]. Data transformation was needed so that it can be used by generated quiz plugin. Transformation was done by XML file injection. Injection changed all question types in the repository into question types which is supported by generated quiz plugin. Injection was done by adding four more attributes: difficulty, distinguishing degree, time and level. Each of those attributes was assigned with a random value in the attributes value range. Essay questions data weren't provided in the repository, so that it was created manually.

The testing is done for every capability the plugins have. Every capability was tested using black box functional testing. In this test, scenario were made and done. Then, output of the scenario was compared with expected output. If the output is the same as expectation, then it means that the capability runs well.

The capability of creating questions was tested with the following scenario. First, user went to input form page of one of the question type plugins. User input data which cause errors, saw whether error messages were shown and input the correct data after that. Then, user viewed the question database to see if the question had been saved. In the testing, data that cause error showed error messages and the correct question data were stored in the database. It means that creating questions capability run well.

The capability of exporting questions was tested with the following scenario. First, user went to question bank export page. Then, user selected Moodle XML format and the category where the questions were stored. Then, user pressed export button and selected the path where the question backup shall be stored. Based on the testing, the question data was saved in external file in Moodle XML format. It means that exporting questions capability also run well.

The capability of importing questions was tested with the following scenario. First, user went to question bank import page. Then, user selected Moodle XML format and the category where the questions should be restored. After that, an external Moodle XML file was selected. The import button was pressed and the questions were restored from the external file to the database. Importing questions capability run well.

The capability of creating quiz was tested with the following scenario. First, user opened the main page of generated quiz, the form data input page. User input data which cause errors, saw whether error messages were shown and input the correct data after that. Then, user viewed the quiz database to see if the quiz had been saved. Based on the testing, data that cause error showed error messages and the correct quiz data were stored in the database. It means that creating quiz capability run well.

The capability of generating quiz questions was tested with the following scenario. First, user opened the main page of generated quiz, the form data input page. Then, user input the constraints of quiz which were going to be generated and clicked save button. In the test, questions were generated successfully and stored in quiz slots database. It means that generating quiz questions capability run well.

The capability of sorting quiz questions was tested with the following scenario. First, user open the main page of generated quiz, the form data input page. Then, user input data, selected one of the attributes which the sorting were based on and selected the option whether the sorting will be done in ascending or descending way. In the test, every attribute which the sorting based on worked perfectly. It means that sorting quiz questions capabilities run well.

The capability of viewing quiz generation statistics was tested by viewing statistics page after quiz had been generated. In the test, statistics page shows fitness data, execution time and the details of constraints as input and generated quiz as output. Viewing quiz generation statistics run well.

V. CONCLUSION

Conclusions of this research are described as follow:

1) Quiz generation plugins is successfully implemented and integrated in Moodle. These plugins consist of an activity plugin called generated quiz and five question type plugins. Generated quiz plugin is the main plugin which generate a quiz from input constraints using genetic algorithm. Question types plugin provide questions which can be used by the main plugin. All of these plugins capabilities run well.

2) Genetic algorithm is used in the plugins to generate a quiz that approximates the input constraints. The generation is done by inserting constraints as input. Genetic algorithm takes the constraints as consideration when generating a quiz. The objective of the genetic algorithm is to generate a quiz whose attributes as close as possible to the input constraints. Quiz attributes is depended on every question attributes the quiz has. In this way, genetic algorithm successfully selects suitable questions for the quiz.

VI. FUTURE RESEARCHES

Some suggestions for future researches are described as follow:

1) It may be possible to implement quiz generation as an external tool in LMS using LTI (learning tools interoperability). LTI have advantage in portability because it is supported by several LMS: Moodle, Canvas, Sakai, etc. [19].

2) Another algorithm such as simulated annealing algorithm is possible to be used instead of genetic algorithm in quiz generation plugin.

3) Fitness function calculation in genetic algorithm is possible to be done asynchronously. It is possible to use asynchronous distributed genetic algorithm in quiz generation researches.

REFERENCES

[1] P. Hill and M. Feldstein, "New Release of European LMS Market Report" (http://mfeldstein.com/new-release-european-lms-market-report), December 1, 2016 (Accessed on February 12, 2017).

[2] Moodle, "Moodle Statistics" (https://moodle.net/stats/), accessed on May 20, 2017.

[3] Moodle, "Moodle Plugins Search" (https://moodle.org/plugins/search.php?s=&search=Search+plugins), accessed on June 4, 2016.

[4] Moodle, “Building Quiz”, (https://docs.moodle.org/33/en/Building_Quiz), accessed on June 5, 2017

[5] P. Burns, Benefits of using genetic algorithm, URL (version: 2012-06-04): https://stats.stackexchange.com/q/23107

[6] W. Xiaoqin and S. Yin. "Research on Intelligent Auto-Generating Test Paper Based on Improved Genetic Algorithms". Computational Intelligence and Software Engineering, CiSE 2009. International Conference, pp. 1-4. 2009.

[7] K. Naik, S. Sule, S. Jadhav and S. Pandey, “Automatic Question Paper Generation using Randomization Algorithm,” International Journal of Engineering and Technical Research, vol. 2, issue 12, December 2014.

[8] M. Mitchell, “An Introduction to Genetic Algorithms”. Cambridge, Mass: MIT Press, 1996. M. Chang, L. Ratinov, and D. Roth. “Structured Learning with Constrained Conditional Models”. Machine Learning, Vol. 88 No. 3, 399-431. 2012.

[9] M. Chang, L. Ratinov, and D. Roth. “Structured Learning with Constrained Conditional Models”. Machine Learning, Vol. 88 No. 3, 399-431. 2012.

[10] D. Whitley, "A genetic algorithm tutorial," in Statistics and Computing, Vol. 4, pp. 65-85, 1994.

[11] A. E. Eiben and J. E. Smith, "Introduction to Evolutionary Computing," Springer-Verlag Berlin Heidelberg, 2003.

[12] R. Singh and S. Arya, "Genetic Algorithm for the Design of Optimal IIR Digital Filters," Journal of Signal and Information Processing, Vol. 3 No. 3, pp. 286-292, 2012.

[13] S. R. Schach, "Object-Oriented Software Engineering," New York: McGraw-Hill, 2008.

[14] Moodle, "Moodle Dev docs" (http://docs.moodle.org/dev/Developer_documentation), accessed on August 5, 2016.

[15] W. Huang and Z. Wang, "Design of Examination Paper Generating System from Item Bank by Using Genetic Algorithm," Computer Science and Software Engineering, Vol. 5, 1323-1325, 2008.

[16] K. Man, W. K. Tang, and S. Kwong, "Genetic Algorithms: Concepts and Applications [in Engineering Design]", IEEE Trans, Industrial Electronics, 1996.

[17] S. Baluja and R. Caruana, "Removing the genetics from the standard genetic algorithm". PA: Carnegie Mellon University Pittsburgh, 1995.

[18] Moodle, “Quiz Questions,” (https://moodle.net/mod/data/view.php?id=23) accessed on June 6, 2017.

[19] IMS Global Learning, "Learning Tools Interoperability'" (https://www.imsglobal.org/activity/learning-tools-interoperability) accessed on June 1, 2017.