The development of a web application for assessment by tests generated using genetic-based algorithms

Doru Anastasiu Popescu[1], Victor Tița[2], Nicolae Bold[1]

[1]University of Pitesti, Department of Mathematics and Computer Science, Romania
[2]University of Agronomic Sciences and Veterinary Medicine Bucharest, Faculty of Management, Economic Engineering in Agriculture and Rural Development, Slatina Branch, Romania
dopopan@gmail.com, victortita@yahoo.com, bold_nicolae@yahoo.com

Abstract. The multitude of the technology-based tools used for educational purposes is now a common thing to be seen. These tools can help within the educational process either for the organizational purposes or these are included in the materials used in education. This paper presents Kromatine, a generator of assessment tests which are obtained using a genetic algorithm, which includes it in the first category, organizational purposes. The genetic algorithm uses basic genetic operations and structures and it is presented in a form of a web application. It eases the organizational tasks of the teacher by giving him the opportunity to generate tests that will be used further in assessment. The questions are stored in a database and the user has the possibility to add questions to database and to generate tests that can be used later. The questions are characterized by a degree of difficulty and are multiple-choice type. The choice of the genetic algorithm is due to the fact that the problem can be summarized in generating an arrangement of question summing a given total degree of difficulty (comparable to the subset sum problem), which includes the issue in the category of NP-complete problems. Also, the problem structure can be easily modeled based on a general genetic algorithm structure.

Keywords: genetic, tool, web application, education, assessment.

1 Introduction

As technology advances more and more and the modalities of easing organizational tasks are more numerous and following several recent breakthrough researches. All these research is based on nature functionalities and structures. Also, education is an extremely important field within the domains of the people, from the primary school to adult training [15]. This importance is obvious, because education is a basis for every human activity.

We will present in this paper a primary version of a tool that generates tests used for assessment based on a genetic algorithm. As we will see, the questions are multiple-choice type and are selected from a database which is built overtime. Section 2 introduces a theoretical base formed from notions and operations used to build the tool. In section 3 we will present the actual implementation of the tool, in the form of a web application, and section 4 contains an example of obtaining assessment tests using this tool.

2 Research and related work

Genetic algorithm theory is rapidly developing due to advances in technology and research. Known for their large applicability, their approximate nature is a both a feature and a drawback that is currently studied in order to increase the accuracy of the solutions obtained. Thus, state of the art research on genetic algorithm is aiming to solving both classical theory problems and unusual particular issues.

Given the first direction, the applicability of the genetic algorithms to fuzzy problems is a candidate for solving matrix problems (implying chessboard-like structures such as the queen problem [1]), which refer to the larger component of combinatorics. Also, the genetic algorithms are widely used as a second solution for NP-complete problems and one of the closest to education area is the generation of a timetable or a schedule [2], given certain constraints. Besides that, the genetic algorithm may be combined with neural network notions in order to help in pattern and classification problem [11].

The problem studied in this paper is part of the second set of problems. Given the fact that the problem of generating tests formed of question with a given requirement is not a common issue found in the literature, the existing papers which deal with the problem deal with the problem of efficiency of genetic-based generation [4]. Other types of generators use random-based generators [5] or ant-colony algorithms (ACA), where is shown that effectiveness is slightly greater in terms of generating time. However, time generation is not necessarily a key-parameter within the problem of tests generation, but the precision of results. The precision is close either an ACA or genetic algorithms are used [6].

Another issue regarding the studied problem is that this can be classified as NPcomplete, due to its reduction to the subset sum problem, of generating subsets of finite cardinal whose sum of difficulty degree is close to a given parameter, which is known for being NP-complete [3]. This is why an evolutionary approach is preferred.

Issues regarding the generation of tests which are secondary in this paper are also consisting in the type of the question that is generated, whose number can be extended using existing methods based on word analyze and NLP algorithms [7] and the automatic determination of the degree of difficulty of a given question [8]. These issues are forming new fields of research and integration in future research. Furthermore, the questions that form tests can be seen as nodes in a complex network, which would consist in the possibility of using graph-based structures [14] and introduce the concept of linked questions within the implementation of the algorithm.

Finally, the problem described in the paper is a new integration of technology tools within the vast domain of education. We should not exclude the social part of the education [10] and the implications of the usage of the technology [13], which are immensely influencing the educational development of the students. Thus, a future development would be the inclusion of a social aspect within the tool, either in selection of the test or regarding the interaction between users.

3 Theoretical notions and application structure

Before the actual presentation of the tool to be made, we will present the notions that led to its creation. The tool has been developed based on a genetic algorithm, meaning that the structures used are the gene and the chromosome. Also, the questions and the generated tests are stored in a database. The definitions that follow present the particular notions and clarify the terminology used in this paper. The database has four tables which are basic for the needs of the generator:

- table Questions, which contains fields storing the identification number of the question, the statement, the number of choices, the degree of difficulty of the question, the correct choice(s) and the user who proposed it;
- table Choices, which contains the question identification number, the choice letter from „a" to „z" and the choice text;
- table Tests, containing fields storing the identification number of the test, the questions, the total degree of difficulty of the test, the user who generated it, the generation timestamp and the generation time. The latter field is used entirely for monitoring and research purposes;
- table Users, containing fields related to the user who uses the generator, such as user identification number or alias. The table is designed to store user data and has an organizational purpose.

A detailed perspective on the database tables is presented in Table 1.

Table 1. A detailed perspective of the database structure.

Table	Field name	Type	Values	Requirements
Questions	question_id	int	id ≥ 0	distinctive
	statement	text	set of characters	-
	diff	int	0 ≤ diff ≤ 5	-
	choice_no	int	0 ≤ diff ≤ 27	-
	correct	varchar	Variable length-combos of ‘a’ to ‘z’	-
	user_id	int	id ≥ 0	-
Choices	question_id	int	id ≥ 0	-
	choice_x	varchar	{‘a’, ‘b’, ‘c’, …, ‘z’}	-

Back to Table of Contents

	choice_text	text	set of characters	-
Tests	test_id	int	id ≥ 0	distinctive
	questions	text	length = 2 × $\|S\|$[1]	serialized data
	total_diff	int	0 ≤ total_diff ≤ 5 × $\|S\|$	-
	user_id	int	id ≥ 0	-
	timestamp	datetime	dd:mm:yyyy / H:i:s	-
	generation_time	float	seconds	-
Users	user_id	int	id ≥ 0	distinctive
	alias	text	set of characters	-

The structure of the database DBQ containing the tables and the connections between them is presented in figure 1.

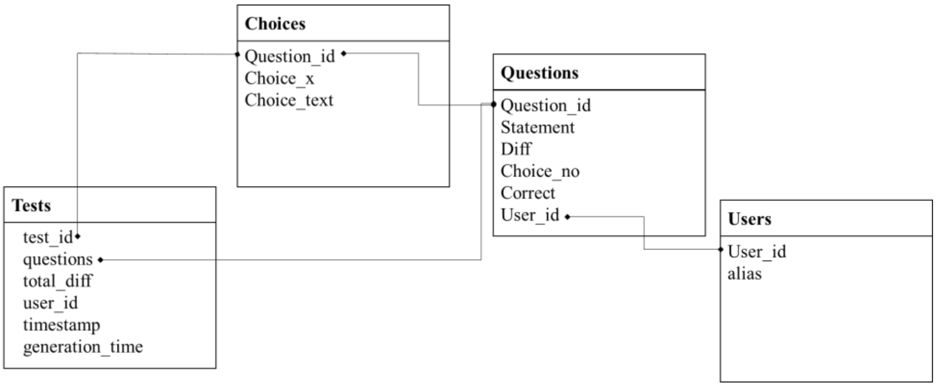

Fig. 1. Visual concept of the database

Specification 1. A question *q* (*id*; *st*; *dd*; *V*) is an object formed of the next components:

— the identification number of the question id;
— the statement *st*;
— the degree of difficulty *dd*, *dd* ϵ {1, 2, 3, 4, 5};
— choices set *V*.

Back to Table of Contents

Observations:

The degree of difficulty dd is subjective for each question and it is considered to be input data given by a human operator. This degree is considered to situate on a scale from 1 to 5, where 1 is the least difficult and 5 means the most difficult. In order to normalize the difficulty and cancel to a certain extent the subjectiveness of the appreciation of the difficulty, a short explanation is given to the users.

The set *V* contains objects structuralizing a choice vi (*id*; *l*; *cst*), $i = 1, |V|$ of the question, as follows:

- question identification number *id*;
- choice identification particle *l*. We choose as choice identification letters from the English alphabet, thus $l \in$ {„a“, „b“, …, „z“}. The number of choices is thus limited to 27;
- choice statement *cst*.

Observation:

a) A test *T* (*S*, *GD*) is a set of questions q_i, $i = 1, |S|$, where *S* is the set of questions that form the test and *GD* is the degree of difficulty of the test:

$$GD = \sum_1^{|S|} q_{dd_i} \qquad (1)$$

Specification 2. Given the database question set *Q* and the selected test question set *S* for a given set of input data, a gene g_i is an integer particle and a member of the set $\{1, \ldots, |Q|\}$, $i \in \{1, \ldots, |S|\}$.

Observations:

a) Basically, a gene stores the order number for a question (*g* is equivalent to q_{id}).
b) |*S*| is an input data and used in the algorithm.
c) The elements of set *S* are unknown before the generation, being an output data.

Specification 3. Given the database question set *Q*, the selected test question set *S* at a given state, the population set *NC* and the desired total degree of difficulty *MGD*, a chromosome *C* is an object formed of:

- order number *id*, $id \in \{0, \ldots, |NC|\}$;

- the gene set $G_j = \{g_i \mid i \in \{1, \ldots, |S|\}\}$, where $G = S$; $j = 1, |NC|$;

- the fitness function *f* defined as follows

$$f(C_j) = \frac{1}{\left|\sum_1^{|S|} g_i - MGD\right|} \qquad (2)$$

Observations:

a) G_j is equivalent to q_{id}.
b) We can easily observe thatMGD= [|S|,5×|S|].
c) The fitness function checks if the sum of the difficulty degrees of each question within a chromosome are lower and as close as the value MGD.
d) The chromosome contains the order numbers of questions that form a test. If we denote the test questions set by T, then $T = S = \{G_i | i = 1, |S|\}$.

Proposition 4. Given a chromosome C_i (i = 1, |NC|) and random positions a and b (a, b = 1, |S|), the mutation operation is defined as the shift of the genes found on the positions a and b.

Observation. The mutation has as result the generation of a new chromosome.

Proposition 5. Given two chromosomes C_i and C_j and a random position p, the crossover operation is defined as a succession of steps as follows:

- The two chromosomes are split at the position p.
- The first part of the chromosome C_i is combined with the second part of the chromosome C_j and the first part of the chromosome C_j is combined with the second part of the chromosome C_i.
- Two new chromosomes C_i' and C_j' are obtained, as follows:

$$C_{i'} = (g_{i1}, g_{i2}, \dots, g_{ip-1}, g_{jp}, \dots, g_{jS}) \quad (3)$$

$$C_{j'} = (g_{j1}, g_{j2}, \dots, g_{jp-1}, g_{ip}, \dots, g_{iS}) \quad (4)$$

Within the algorithm, the order of the operations is:

a. Generation of the initial population
b. Sort of chromosomes based on fitness
c. Mutation of chromosomes
d. Crossover of chromosomes

Operations b), c) and d) are repeated for a previously-set number of generations. The final result is a list of tests from which we store a finite number of tests which have the highest value of the fitness.

4 Implementation

The implementation was made in the form of a web application. The implementation was based on Bootstrap framework, used for display and structural components. The back-end component is based on PHP combined with MySQL used for database storage. The customizable parameters, i.e. the ones which influence the performance of the final output (the size of the initial population, the mutation rate, the crossover rate) can be modified, but they have default values that guarantee a close-to-optimum solution. Thus, if the user in unaware of the definition of these parameters, he can as well ignore giving them values. Regardless the situation, the technical details are presented in a help section.

The main page of the application is shown in Figure 2 (a-d).

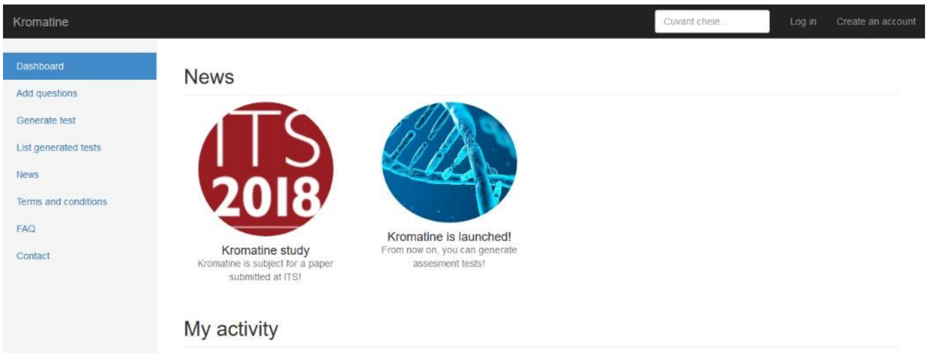

Fig. 2. (a) Main panel of the application

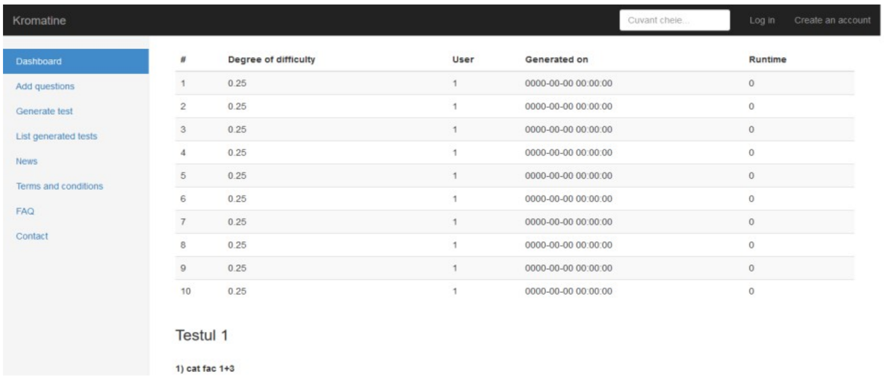

Fig. 2. (b) Activity page.

Back to Table of Contents

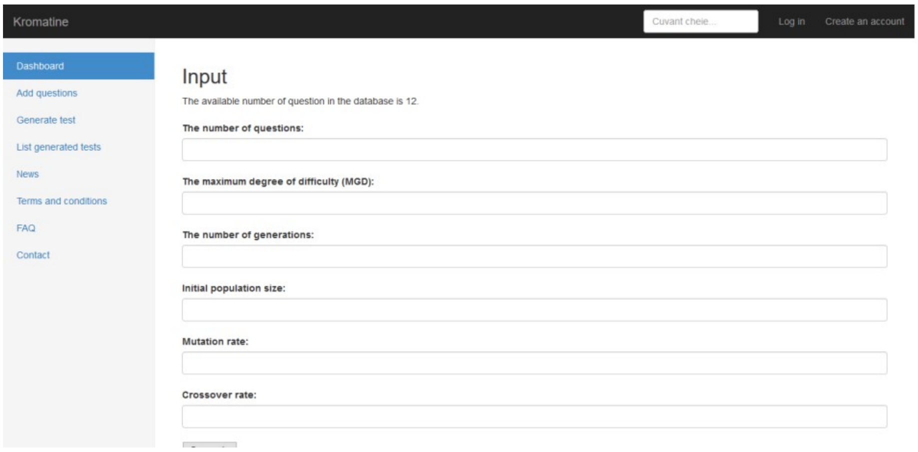

Fig. 2. (c) Generation form.

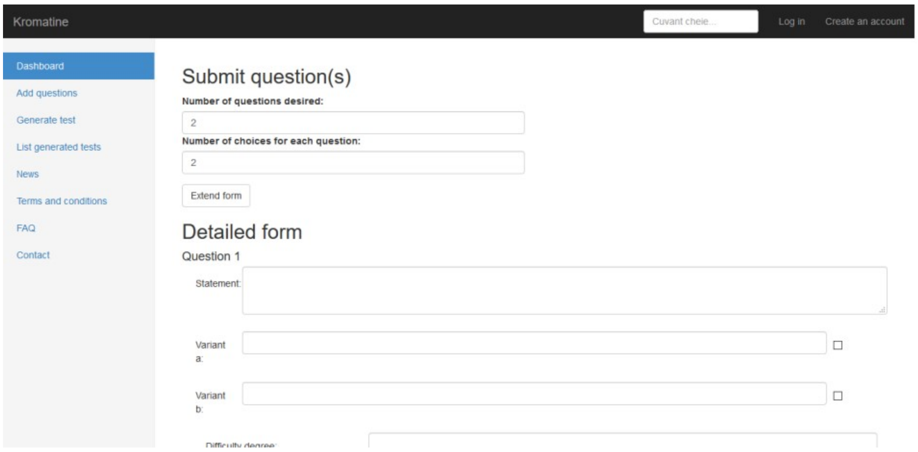

Fig. 2. (d) Submission form.

The application was built of the following components:

- the dashboard, which shows a summary of the user activity;
- the script for proposing questions, consisting in an extended form;
- the page for generating questions, which is the core of the entire application and where the input data is set;
- the page used for showing the generated tests for a given user, where he can choose some of the tests generated before.

The visual representation of the application scheme is presented in figure 3.

Back to Table of Contents

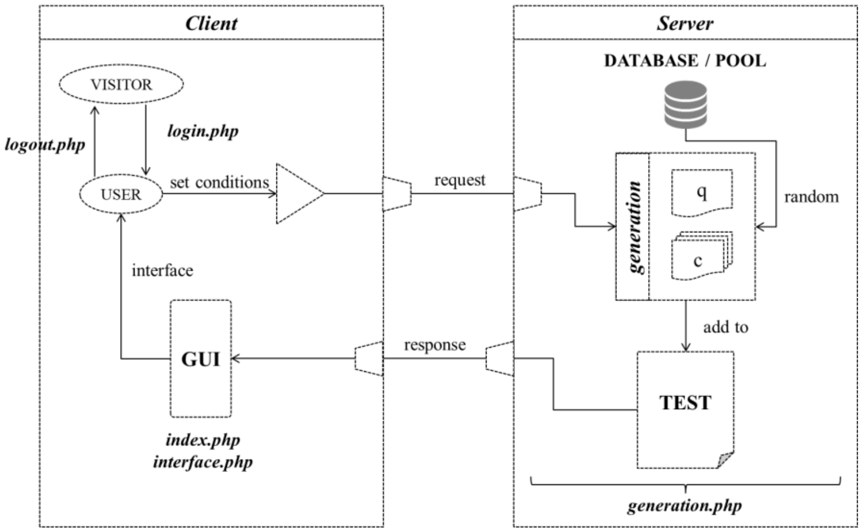

Fig. 3. Visual representation of the application scheme

5 Conclusions

The presented application is basically a core for a future development of an assessment aid tool for a teacher. The implemented tool can be in this matter included in a long list of technology-based tool that are used in education, widely developed [12] on different supports, even mobile [9]. Given the fact that the foundation theory of the problem relates to NP-completeness, the chosen genetic approach is legitimate due to user requirements. Future work would obviously consist in the development of the existing tool in directions of functionalities for the user, such as the automatic output of the test in a desired form (document), and theoretical basic structure, such as adding requirements to the fitness function.

The educational process depends on mathematical parameters that technology can use in order to ease the organisational tasks for the person who is in charge with the educational process (e.g., the teacher). Also, the technology has implications on the actual educational process by providing materials that create an interactive learning environment.

References

1. Alharbi, S., Venkat, I.: A Genetic Algorithm Based Approach for Solving the Minimum Dominating Set of Queens Problem. Journal of Optimization, Volume 2017 (2017)
2. Colorni, A., Dorigo, M., Maniezzo, V.: A Genetic Algorithm To Solve The Timetable Problem(1994).

3. Moon, B.: The Subset Sum Problem: Reducing Time Complexity of NP-Completeness with Quantum Search, Undergraduate Journal of Mathematical Modeling: One + Two: Vol. 4: Iss. 2, Article 2(2012).
4. Li Y., Li S., Li X.: Test Paper Generating Method Based on Genetic Algorithm, AASRI Procedia, Volume 1, Pages 549-553, ISSN 2212-6716 (2012).
5. Guang C.,Yuxiao D., Wanlin G., Lina Y., Simon S., Qing W., Ying Y., Hongbiao J.: A implementation of an automatic examination paper generation system, Mathematical and Computer Modelling, Volume 51, Issues 11–12, Pages 1339-1342, ISSN 0895-7177 (2010).
6. Liu, D. W., Jianmin Z., Lijuan.: Automatic Test Paper Generation Based on Ant Colony Algorithm. Journal of Software. 8. . 10.4304/jsw.8.10.2600-2606(2013).
7. Thessen A. E., Cui H., Mozzherin D.: Applications of Natural Language Processing in Biodiversity Science. Advances in Bioinformatics, 2012:391574 (2012).
8. Boopathiraj, C. Chellamani, K.: Analysis of Test Items on Difficulty Level and Discrimination Index in the Test for Research in Education. International Journal Of Social Sciences & Interdisciplinary Research, [S.l.], p. 189-193,ISSN 2277-3630 (2013).
9. Elvira Popescu, Constantin Stefan, Sorin Ilie, Mirjana Ivanovic, EduNotes - A Mobile Learning Application for Collaborative Note-Taking in Lecture Settings, Proceedings ICWL 2016, Lecture Notes in Computer Science, Vol. 10013, Springer, ISBN: 978-3-31947439-7, pp. 131-140, 2016.
10. Alex Becheru, Elvira Popescu, Design of a conceptual knowledge extraction framework for a social learning environment based on social network analysis methods, Proceedings ICCC 2017, ISBN: 978-1-5090-4862-5, pp. 177-182, 2017.
11. Doru Constantin, Emilia Clipici, "A new model for estimating the risk of bankruptcy of the insurance companies based on the artificial neural networks", Proceedings of the 16th edition of the SGEM International GeoConferences (International Multidisciplinary Scientific GeoConferences), 28 June-7 July, 2016.
12. C. Baron, A. Şerb, N.M. Iacob, C.L. Defta, IT Infrastructure Model Used for Implementing an E-learning Platform Based on Distributed Databases, Quality-Access to Success Journal, Vol. 15, Issue 140, pp. 195-201, 2014
13. C.L Defta, A. Șerb, N.M. Iacob, C. Baron, Threats analysis for E-learning platforms, Vol. 6 / Nr. 1, pp. 132–135, 2014
14. Domşa Ovidiu, Emilian Ceuca, Mircea Râşteiu, Algorithm to find a tree with Maximal Terminal Nodes, 1st Balkan Conference in Informatics, 21-23 November 2003, Thessaloniki, Greece, ISSN960-287-045-1, pag.113-122
15. Victor Ţiţa, Raluca Necula: Trends In Educational Training For Agriculture In Olt County, Scientific Papers Series Management, Economic Engineering in Agriculture and
16. Rural Development , Vol. 15, Issue 4, 2015, PRINT ISSN 2284-7995, E-ISSN 2285-3952

Back to Table of Contents