Research Article

Irene Brito*, José João Almeida, Gaspar J. Machado

Mvgen: Multi Version Question Generation for Math Courses

https://doi.org/10.1515/edu-2019-0010
received July 12, 2019; accepted September 20, 2019.

Abstract: A question generating system developed in Perl that uses LaTeX as typographical language is presented, which, besides random generation of different question types, allows a random creation of a large number of different test versions. Examples of question types in mathematics (Linear Algebra and Discrete Mathematics) are provided and the associated generation strategies are explained.

Keywords: question generation, random generation, mathematics, STEM education, science education.

1 Introduction

Nowadays, with the development of informatics resources and tools, most of which are freely available online, and with the increasing interest in a both automatic and controlled form of teaching, learning, and assessment, the search for powerful exercise generation systems is growing. Different exercise generation systems have been proposed in the literature, see e.g. Belmonte et al., 2002; Tomás and Leal, 2003; Sangwin & Grove, 2006; Almeida et al., 2013a; and references therein. For example, Stack (see Sangwin & Grove, 2006; Sangwin & Harjula, 2017) is a free, open source, computer aided assessment system for Mathematics (and Science and related disciplines), which uses the computer algebra system Maxima to generate random questions, establish the properties of answers and provide sophisticated feedback. A similar system to Stack, although less widely used and developed, is the system Passarola (see Almeida et al., 2013a). Passarola permits perhaps a more complex exercise creation and this also in completely different domains, e.g. Music (Almeida et al., 2013b). Maple TA (http://www.maplesoft.com/products/mapleta/), another online testing and assessment software designed for science, technology, engineering, and mathematics (STEM) courses, relies on Maple and it is, in contrast to the other two systems, not freely available.

Concerning the various types of questions and tests obtained from those systems, the multiple choice test reveals to be the most common and widely used assessment technique for measuring the learning outcomes in all areas of science. However, the usage of multiple choice questions is controversial. Their advantages and disadvantages have widely been discussed and suggestions to overcome possible difficulties have been made, see e.g. Azevedo, 2015; Chesbro, 2010; Cruz et al., 2012; Scharf, 2007; Torres et al., 2009. Due to the large number of students in courses, instructors often design several versions of this kind of examination. Rearranging the questions and the corresponding answer choices manually can be a difficult task.

In this paper we present a system, Mvgen, where the random generation of questions and choices is achieved in a straightforward and fast way. On the one hand, the system gathers automatically one correct answer (from a set of possible answers) and a subset of different distracters and, on the other hand, it permutes automatically the possible answers, so that with less effort a large number of different questions is achieved. Also the question order is changed randomly for each test version.

Comparing the system Mvgen with the other systems mentioned before, we can say the following. The main difference is that Mvgen was designed above all as a tool to support the multiple question and test generation, whereas the other systems are predominantly designed to work as online assessment tools. The system Mvgen is a simple system, depending only on Perl and LaTeX, it is more self-contained and requires less computational needs than the other systems. It has the flexibility to

***Corresponding author: Irene Brito,** Departamento de Matemática e Centro de Matemática, Universidade do Minho, 4800 Guimarães, Portugal, E-mail: ireneb@math.uminho.pt
José João Almeida, Departamento de Informática, Universidade do Minho, 4710 Braga, Portugal
Gaspar J. Machado, Departamento de Matemática e Centro de Física, Universidade do Minho, 4800 Guimarães, Portugal

generate different question and test versions through various mechanisms (by random permutations, using tables), as it will be seen in the Examples section. If an instructor has a set of questions written in LaTeX and if his aim is to obtain in a simple and fast way a high number of different question and test versions, then Mvgen reveals to be more appropriate than the other systems, which are more complex to use and more sophisticated.

The random generation of tests and exams, consisting partially of multiple choice questions, with the system Mvgen has already been applied to the examination in Discrete Mathematics and Linear Algebra of engineering courses, where 180 different exam versions where generated, one for each student.

This system allows also to generate many other question types. We will consider the following: multiple choice, numeric response, matching, and ordering questions. Here, we recall their definitions:

- A multiple choice question consists of a stem (the text of the question) and of several answer options or choices containing the key (correct answer) and distractors (incorrect answers). One answer must be chosen from the number of supplied choices.
- A numeric response question requires a numeric answer, which is compared with correct answer. The matching of both is defined by specifying a tolerance criterium.
- A matching question contains a set of elements, which must be matched against another set of elements.
- An ordering question consists of a list of items displayed in a random order. The items must then be placed in a correct sequential order (or the correct sequential order must be indicated).

This paper is organized as follows. In section 2, the language of the question generation system Mvgen is described. Section 3 contains examples of question types built in Mvgen. Finally, in section 4, some conclusions are presented.

2 Language

Mvgen is a Domain Specific Language for multi version test generation, designed to be single-file, small, and open-source. It is written in Perl with no extra dependencies and generates LaTeX. Internally, Perl is used to parse the Mvgen text, process, and rewrite it, calculating dynamically some expressions when needed. The questions can be written in a text file using LaTeX language and stored in a single

```
\documentclass[a4paper]{article}
...
\begin{document}
...
#(1)
...
\end{document}
#base{Al.base}
```

Figure 1: LaTeX file.

base file (e.g. “Al.base”). This file forms a data base for the construction of exercise sheets, tests, or exams. The latter are created in LaTeX using common layout and style files. Thereby, the desired questions are extracted from the base file using the command “#(.)”, where inside the brackets appears the number of the question (e.g. “#(1)”), and including the instruction “#base(Al.base)” after “\end{document}” (see Figure 1).

One can then generate a PDF file with the desired number of tests, a PDF file containing the questions and the corresponding corrections used in the test, and, for tests with multiple choice questions, also a .txt file is generated with the correction code for the different test versions.

The generation strategy of multiple choice questions is the following. In the text file, the question is formulated after the “#q” operator, where an additional number is used to distinguish between different questions, e.g. “#q1”, “#q2”. Then one creates a set of correct answers by listing them below the operator “#v” (which stands for verity) and a set of distractors below the operator “#f” (which stands for falsity), where in each line below “#v” and “#f” appears only one option. When a test file containing a multiple choice question is generated, Mvgen selects randomly one correct option and three distractors in order to build the multiple choice question. By default the number of distractors is three, so that the multiple choice question will have four options, but this number can be changed. As for the number of different question versions that can be generated in this way, one can observe the following. Let n be the number of options in a multiple choice question, n_v the number of correct answers and n_f the number of distractors. In order to set up a multiple choice question one must only take into account that $n_v \geq 1$ and $n_f \geq n - 1$. The number of different question versions, which can be generated from a multiple choice question, is then

$$N_q = \frac{n n_v n_f!}{(n_f - n + 1)!}. \quad (1)$$

By default, the number of options in a multiple choice question is n=4, so that one must provide $n_f \geq 3$ distractors and at least one correct option and this implies the following number of different question versions:

$$N_q = \frac{4n_v n_f!}{(n_f-3)!}. \quad (2)$$

3 Examples

In this section we will consider examples of questions from Linear Algebra and Discrete Mathematics and explain how they can be generated.

3.1 Multiple choice questions

Consider the example of multiple choice question in Linear Algebra presented in Figure 2, which was generated using the instructions given in Figure 3.

The stem is formulated after "#q1". Below "#v" are listed three correct answers and below "#f", six distractors. When the test file containing this question is generated, one of the correct options and three distractors will be randomly selected. When more than one test version is generated, the choices will be randomly permuted in each version. In this case, n=4, n_v=3, n_f=6 and, applying (2), one can generate N_q=1440 different question versions.

The following example of multiple choice question in Discrete Mathematics (see Figure 4) has been generated using the code indicated in Figure 5.

In this example, two correct answers are listed below the operator "#v" and ten distractors below "#f". Therefore, one has n=4, n_v=2, n_f=10 and, substituting these values in formula (2), one concludes that one can generate N_q=5760 different question versions.

3.2 Matching question

Figure 6 contains an example of a matching question in Linear Algebra, where the matrices of linear transformations on the right (represented by the numbers from 1 to 4) must be matched to the corresponding linear transformations indicated on the left.

This question has been generated using the code given in Figure 7. The corresponding pairs (linear transformations and matrices) are identified in each line after "#pairs" using the operator "::". Taking into account the different ordering of pairs and shuffling of ordered pairs, one can generate 576 different question versions.

I.1 Consider the matrix $A = \begin{bmatrix} -1 & 3 \\ -4 & 3 \end{bmatrix}$. Then:

A. $\det(-3A^{-1}) = -27$.

B. $\det(4A^{-1}) = 36$.

C. $\det(4A^{-1}) = \frac{16}{9}$.

D. $\det(2A^{-1}) = 18$.

Figure 2: Multiple choice question in Linear Algebra.

```
#q1 Consider the matrix $A=\smleft -1 & 3 \\-4 & 3\smright$. Then:
#v
$\det(2A^{-1})=\frac{4}{9}$.
$\det(3A^{-1})=1$.
$\det(4A^{-1})=\frac{16}{9}$.
#f
$\det(2A^{-1})=18$.
$\det(-2A^{-1})=-\frac{4}{9}$.
$\det(4A^{-1})=36$.
$\det(-4A^{-1})=-36$.
$\det(3A^{-1})=\frac{1}{3}$.
$\det(-3A^{-1})=-27$.
```

Figure 3: Generation code for multiple choice question.

II.4 Let $A = \{(1, 2), \{1, 2\}\}$, $B = \{(2, 1), \{2, 1\}\}$, $C = \{(1, 2), (2, 1)\}$, and $D = \{\{1, 2\}, \{2, 1\}\}$. Then:

A. $C \subseteq B$.

B. $A \subseteq D$.

C. $D \subseteq B$.

D. $A \subseteq C$.

Figure 4: Multiple choice question in Discrete Mathematics.

```
#q4
Let $A=\{(1,2),\{1,2\}\}$, $B=\{(2,1),\{2,1\}\}$, $C=\{(1,2),(2,1)\}$, and $D=\{\{1,2\}, \{2,1\}\}$.
Then:
#v
$D\subseteq A$.
$D\subseteq B$.
#f
$A\subseteq B$.
$A\subseteq C$.
$A\subseteq D$.
$B\subseteq A$.
$B\subseteq C$.
$B\subseteq D$.
$C\subseteq A$.
$C\subseteq B$.
$C\subseteq D$.
$D\subseteq C$.
```

Figure 5: Generation code for multiple choice question in Discrete Mathematics.

II.2 Match each linear transformation $T : \mathbb{R}^2 \to \mathbb{R}^2$ to its matrix A_T.

$T(x, y) = (x - y, 0)$ a ☐ 1 $A_T = \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix}$

$T(x, y) = (0, x - y)$ b ☐ 2 $A_T = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

$T(x, y) = (-y, x)$ c ☐ 3 $A_T = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$

$T(x, y) = (x, -y)$ d ☐ 4 $A_T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

Figure 6: Matching question in Linear Algebra.

For a matching question the number of different question versions which can be generated is $N_q=(n_p!)^2$, where n_p is the number of pairs listed in the generation code after #pairs.

```
#q2
Match the linear transformation $T:\mathds{R}^{2}\rightarrow\mathds{R}^{2}$
to its matrix $A_T$.
#pairs
$T(x,y)=(x-y,0)$ :: $A_T=\smleft 1 & -1\\0 & 0\smright$
$T(x,y)=(x,-y)$ :: $A_T=\smleft 1 & 0\\0 & -1\smright$
$T(x,y)=(0,x-y)$ :: $A_T=\smleft 0 & 0\\1 & -1\smright$
$T(x,y)=(-y,x)$ :: $A_T=\smleft 0 & -1\\1 & 0\smright$
```

Figure 7: Generation code for matching question in Linear Algebra.

11.1 Consider the matrix $A = \begin{bmatrix} -1 & 3 \\ -4 & 3 \end{bmatrix}$. Then, $\det(A) =$ ☐

Figure 8: Numeric response question in Linear Algebra.

```
#q3
Consider the matrix $A=\smleft #linha \\-4 & 3\smright$. Then, $\det(A)=$
  \ansbox{888}
#v
  #det
#tab
linha  :: det
-1 & 3 :: 9
-2 & 5 :: 14
-3 & 3 :: 3
-4 & 3 :: 0
```

Figure 9: Generation code for Numeric response question in Linear Algebra.

11.5 Order the following sets by set inclusion
(indicating in the table the number of the sets from the left to the right in ascending order).

1 $\mathbb{Z}$
2 $\mathbb{N}$
3 $\mathbb{N}_0$
4 $\mathbb{Q}$
5 $\mathbb{R}$

Figure 10: Ordering question in Discrete Mathematics.

```
#q5
Order the following sets by set inclusion \\
(indicating in the table the number of the sets from the left to the right in ascending order).
#sort
$\mathds{N}$
$\mathds{N}_0$
$\mathds{Z}$
$\mathds{Q}$
$\mathds{R}$
```

Figure 11: Generation code for ordering question in Discrete Mathematics.

3.3 Numeric response question

The numeric response question from Linear Algebra presented in Figure 8 has been obtained from the code given in Figure 9.

The command "#tab" permits constructing a table with options for the first row of matrix A, appearing in column "linha", and with the corresponding answers, appearing in the column "det". Each time this question is generated, a different row is substituted in matrix A in "#linha". The answer introduced in the question box can then be compared with the correct answer contained in the selected table row, which can be displayed for the instructor with "(#det)". In this case, one can generate four different question versions (corresponding to the different lines of the table). The "#tab" command can also be applied to other question types.

3.4 Ordering question

In the following ordering question from Discrete Mathematics (see Figure 10), the correct order of the six sets must be indicated in the table taking into account the set inclusion.

The generation code for this question is given in Figure 11. The six sets are listed below the operator "#sort" in correct ascending order, from the top to the bottom, taking into account the set inclusion. When the pdf file is generated, the order of the sets is shuffled and an ordering number is associated to each set, which will appear in the pdf file together with a table (see Figure 10). The correct order must then be indicated by writing the number of each set in the table, from the left to the right in ascending order, observing the set inclusion. With this example, one can generate 720 different question versions.

The number of different question versions which can be generated in this way from an ordering question is $N_q = n_s!$, where n_s is the number of elements to be sorted.

4 Conclusions

The Mvgen question generating system reveals to be an useful tool for assessment, in order to obtain in a very simple and fast way a large number of different question versions and test versions. In the multiple choice questions, randomization was used in the selection of choices and in the display of options. In the numeric response questions, the selection of question elements was randomized. Considering the matching questions, randomization was used in the selection of question pairs and in the shuffling of pairs, while in the ordering questions, the ordering of elements was randomized. The questions were generated in a controlled way, in the sense that, due to the manner in which the questions are set up, the instructor knows in advance the layout of the possible question versions and the correct answers. In the future, we intend to complement this system with an online assessment tool using the system Passarola (see Almeida et al., 2013a).

Concerning, in general, the exercise generation, we do not claim that the system Mvgen is better than the other existing systems, which are, of course, more developed,

more sophisticated and more widely used. However, Mvgen, due to its simplicity and lesser computational requirements, can be an alternative to those systems and be of advantage, when one aims to obtain a high number of different question and test versions in a straightforward way.

References

Almeida, J. J., Araújo, I., Brito, I., Carvalho, N., Machado, G. J., Pereira, R. M. S., Smirnov, G. (2013a). PASSAROLA: High-Order Exercise Generation System. CISTI'2013, pp. 1-5.

Almeida, J. J., Araújo, I., Brito, I., Carvalho, N., Machado, G. J., Pereira, R. M. S., Smirnov, G. (2013b). Exercise Generation with the System Passarola. ICAICTE, Atlantis Press, pp. 322-326.

Azevedo, J. (2015). E-assessment in mathematics courses with multiple-choice questions tests. CSEDU2015, pp. 260-266.

Belmonte, M., Guzman, E., Mandow, L., Millan, E., Perez de la Cruz, J. (2002). Automatic Generation of Problems in Web-based Tutors. *Virtual Environments for Teaching and Learning*. World Scientific Publishing, pp. 237-281.

Chesbro, R. (2010). Strategies for the Meaningful Evaluation of Multiple-Choice Assessments. Science Scope, 34, pp. 12-15.

Cruz, P., Oliveira, P., Seabra, D. (2012). Exercise templates with SAGE (tcms.org.ge). *Tbilisi Mathematical Journal, 5*, pp. 37-44.

Sangwin, C. J. and Grove, M. J. (2006). STACK: addressing the needs of the neglected learners. 1st WebALT Conference and Exhibition, Technical University of Eindhoven, Netherlands, WebALT Inc, University of Helsinki, pp. 81-95.

Sangwin, C. J. and Harjula, M. (2017). Online assessment of dimensional numerical answers using STACK in science. *European Journal of Physics, 38,* 035701 (12pp.).

Scharf, E., Baldwin, L. (2007). Assessing multiple choice question (MCQ) tests – a mathematical perspective. *Active Learning In Higher Education, 8*, pp. 31-47.

Tomas, A. P. and Leal, J. P. (2003). A CPL-based tool for Computer aided generation and solving of Maths exercises. PADL2003, Springer-Verlag, Lecture notes in Computer Science 2562, pp. 223-240.

Torres, C., Lopes, A.P., Babo, L., Azevedo, J. (2009). Developing multiple choice questions in Mathematics. ICERI'2009, pp. 6218-6229.