Generation of tests of various complexity levels in e-learning system based on educational text formalization model

Nerodenko V. Generation of tests of various complexity levels in e-learning system based on educational text formalization model / Nerodenko V., Tytenko S. // Modern Aspects of Software Development: Proceedings of VI International Scientific and Practical Virtual Conference of Software Development Specialists, June, 24 2019 p. – Kyiv: Igor Sikorsky KPI, 2019. – pp. 121-133.

Introduction

Modern learning tools are widely based on the use of Internet space, stimulating further development of distance learning systems components. One of the key components of distance learning is monitoring and verification of knowledge. The most common way of testing knowledge in educational systems is testing. A significant amount of research in the field of testing knowledge focuses on quality and validity of tests [1]. However modern speed of updating professional branches determines the significance of automated test composition and test tasks, which will accelerate the preparation of training courses and environments in modern areas of study. The method of automated composition of testing tools on the basis of didactic text formalization model [2,3] was proposed. The method represents formalization of educational materials on the basis of conceptual-theses model with further application of algorithms for generation and test tasks testing. Further studies of CTM application for knowledge control, its modifications and close alternatives were mainly focused on improving the quality of generated tasks [4–6]. This article [5] describes the method of generating test tasks based on decomposition of statements from the educational text using the system of semantic classes. It offers tools for managing complexity of tasks. Therefore the presented method greatly complicates the previous formalization of didactic text in comparison with the basic model of CTM and requires a large number of homogeneous language constructs in the educational text. The matter of differentiation of tests complexity in the CTM requires additional research.

In accordance with revised taxonomy of Bloom [7], the results of educational activities are differentiated by an ordered scale of cognitive purposes. Learning objectives are arranged in the following ways: to know, understand, apply, analyze, evaluate and create [7]. There's actual functional possibility of a testing system that will determine the level of the subject assimilation in accordance with taxonomy of pedagogical purposes. The full accomplishment of this task by means of testing, especially on the basis of automatically generated tasks, is difficult to achieve, while progress in this direction is necessary and relevant. Consequently, the generation of test tasks on CTM basis needs to be revised to implement the differentiation of tests complexity, which will allow assessing the level of knowledge in accordance with cognitive goals.

Goal description

The purpose of this work is to improve the methods of automated construction of tests of various complexity on the basis of the conceptual-theses model. The task is to ensure that the types of test tasks correspond to different cognitive levels of the Bloom scale [7], to formulate test models that differentiate according to the level of complexity, and to carry out an empirical review of the proposed solutions.

The conceptual-theses model of the formalization of didactic text

CT-elements include concepts and theses [2]. The concept describes the meaning of the subject matter. The set of concepts: $C = \{c_1, c_2, ..., c_n\}$. Abstracts are a natural expression of knowledge in the subject area in the form of fragments of educational text and media content. The thesis is an info or statement about the concept. A set of abstracts: $T = \{t_1, t_2, ..., t_n\}$. Each thesis relates to one concept, and this connection is given by the relation: $CT: T \rightarrow C$. In return, each concept can have an arbitrary number of abstracts, which is described by the relation: $TC: C \rightarrow 2T$.

Templates and types of test tasks

To automate the creation of test tasks, we will use the templates which we will describe with the set: $TTempl = \{tt_1, tt_2, ..., tt_n\}$. In previously proposed implementation of testing on the basis of CTM the templates were divided into two main types: 1) the basis of the question — a thesis, based on the variety of answers — the concept; 2) the basis of the question— a concept, variants of answers are theses [2]. Templates [2] describe the essence of the questioning element (concept or thesis), admissible classes for the questioning element and the response elements. It is proposed to diversify the use of a similar template structure to generate different types of test tasks.

All test tasks belonged to the type that involves choosing one correct answer, which is not enough to cover the Bloom scale [7]. The set of tasks is denoted as follows: $Task = \{task_i\}$, and the set of their types: $TType = \{CO, CS, WA, MI, GF, RW\}$. We will provide a description for each type of task:

- *CO* — multiple-choice task demanding only one correct answer;
- *CS* — multi-choice task demanding several correct answers;
- *WA* — is an open-type task in which you need to insert the concept-answer to the thesis definition that is specified in the question;
- *MI* — task for matching sets of corresponding CT-elements;
- *GF* — is an open type task where it is necessary to fill the gap in the statement;

● *RW* — is a multi-choice task with the requirement to specify incorrect answer variants.

Types of tasks available for generating on the basis of this template, we will describe the following way: $TTemplTypes: TTempl \rightarrow 2^{TType}$

Thus, the attributes of the template tti:

● $TQEntity(tt_i)$ — the CT-element that underlies the issue;

● $TQStr(tt_i)$ — template text that corresponds to the query of the test task;

● $TQClasses(tt_i)$ — allowable classes of CT-elements, which underlie the question that can be used for the template;

● TQNotClasses(tt_i) — inappropriate CT-element classes underlying the issue;

● $TAClasses(tt_i)$ — allowable classes of the CT-element, which are the basis of the answer;

● $TANotClasses(tt_i)$ — the classes of the CT-element that underlies the response that can not be used for the template;

● $TTemplTypes(tt_i)$ — is a new field in the template structure, which specifies the set of tasks types that can be generated on the basis of this template.

In each template, the *TQStr* field also stores the marker to fill the CT-element. If there is no such marker, then the CT-element is inserted at the end of the line $TQStr(tt_i)$. Note that for queries *{MI, GF, RW}*, the query line will not contain the CT-element.

Generation of a test task with differentiation in terms of complexity

We will describe different cognitive goals according to the modified Bloom scale [7] by the set *KT = {Remember, Understand, Apply, Analyze, Evaluate, Create}*. The level of complexity of the tests will be divided into low, medium and complex, we will denote them by the following set: *Complexity = {Low, Middle, Hard}*. This type of division will allow the task to be divided in terms of complexity, and then to conduct a differentiated testing. We can indicate the connection between the complexity of the test depending on the type of knowledge per the Bloom scale [7]: $ComplexityKT: Complexity \rightarrow 2^{KT}$. This dependence can be described as a system as well:

$$ComplexityKT(cl) = \begin{cases} \{Remember, Understand\}, cl = Low \\ \{Remember, Understand, Apply\}, cl = Middle \\ \{Remember, Understand, Apply, Analyze\}, cl = Hard \end{cases}$$

Types of tasks that are generated, depending on the complexity of the test are described as follows:

$$TypesByComplexity(cl) \begin{cases} \{CO, CS, WA\}, cl = Low \\ \{CO, CS, WA, MI\}, cl = Midlle \\ \{CO, CS, WA, MI, GF, RW\}, cl = Hard \end{cases}$$

Modified algorithm to create a test task contains the following steps:

1. **Select a control area of the content**. A selected set of content elements from a set of content *V* form a control area of the test *test: SuffControlA* $\subseteq V$. Here we can also perform check of the data sufficiency to create the test:

$$(|\{t: t \in T \wedge VT(t) \in SuffControlA\}| \geq rta) \rightarrow content\_valid(SuffControlA),$$

where *rta*—minimum required number of thesis to ensure the generation of test tasks, is established heuristic.

2. **Selelect complexity of the test** $cl_{test} \in Complexity$.

3. **Generation of the required number of test tasks according to the complexity of the test**. From the set of admissible types of tasks for this complexity level, we form the required number of test tasks in a loop.

Generation of test task.

3.1. **Choosing one of the acceptable types of test tasks** *tType:*

$$tType \in TypesByComplexity(cl_{test}).$$

3.2. **Select a template that is suitable for generating a type task** *tType*. Looking for a template tt_i, satisfying the condition $tType \in TTemplTypes(tt_i)$.

3.3.**Find valid CT-elements for a given** tt_i **template located in the control area** *SuffControlA*. The set of concepts and theses of the *SuffControlA* area will be defined as follows:

$$SCControlA = \{c: VC(c) \cap SuffControlA \neq \emptyset\},$$
$$STControlA = \{t: VT(t) \in SuffControlA\}.$$

For the set of tasks of all types, except for *MI*, the operator of permissible questionable CT-elements choice for a given template tt_i for the case when $TQEntity(tt_i) = Thesis$, looks like this:

$$STQEntities(tt_i) = \begin{cases} \left\{ \begin{array}{l} t : t \in T \wedge t \in STCControlA \wedge \\ \left(\begin{array}{l} TClass(t) \in TQClasses(tt_i) \\ \vee\, TQClasses(tt_i) = 0 \end{array} \right) \wedge \\ TClass(t) \notin TQNotClasses(tt_i) \end{array} \right\}, TQEntity(tt_i) = Thesis \\ \\ \left\{ \begin{array}{l} c : c \in C \wedge t \in SCControlA \wedge \\ \left(\begin{array}{l} CClass(t) \in TQClasses(tt_i) \\ \vee\, TQClasses(tt_i) = 0 \end{array} \right) \wedge \\ CClass(c) \notin TQNotClasses(tt_i) \end{array} \right\}, TQEntity(tt_i) = Concept \end{cases}$$

If the set of $STQEntities(tt_i)$ is nonempty, we select a random element from the $STQEntities(tt_i)$ set, denote its STQEntity.

For a type of task in comparison *MI* $TQEntity(tt_i)$ can only be a concept, and within a single task it is necessary to get a set of different concepts for which the corresponding set of theses for matching will be chosen similarly to the next steps of the algorithm.

If the set of *STQEntities(tt$_i$)* was empty, then the tti template was either inadmissible or in the educational fragment relating to the *SuffControlA* set, the CT-elements did not fit the selection criteria. In this case, the transition to p.3.2 takes place. If you can not find any template for the task *tType*, the transition to p.3.1. takes place to select another type of task.

3.4. **Search for the right answer variants**. The CT-essence for the answer variants is conventionally denoted as *Entity*:

$$Entity = \left(ent: ent \in \left(CTEntity \backslash TQEntity(tt_i)\right)\right).$$

The set of corresponding CT-elements:

$$E = \begin{cases} C, Entity = Concept \\ T, Entity = Thesis \end{cases}.$$

Conditionally we denote the corresponding sets of classes of CT-elements as *EClass*.

Correctness of variants is specified by their relation to the questionable CT-element. The predicate *correcponds(e,a)* formalizes the statement that the argument-thesis really relates to the argument-concept:

$$\begin{pmatrix} (e \in C \wedge a \in T \wedge CT(a) = e) \\ (e \in T \wedge a \in C \wedge CT(e) = a) \end{pmatrix} \rightarrow corresponds(e, a).$$

An operator of choice the set of possible correct answer variants for all types of tasks except *tType=GF*:

$$RTAEntities(tt_i) = \left\{ \begin{matrix} e: e \in E \ \wedge corresponds(e, STQEntity) \wedge \\ \left(\begin{matrix} EClass(e) \in TAClasses(tt_i) \vee \\ TAClasses(tt_i) = \emptyset \end{matrix} \right) \wedge \\ EClass(e) \notin TANotClasses(tt_i) \end{matrix} \right\}.$$

For the type of task to fill the gap in the statement, where $tType = GF, TQEntity(tt_i) = Concept,$ generalized search operator for admissible theses-statements:

$$RTAEntities(tt_i) = \{e: e \in E \ \wedge corresponds(e, STQEntity) \wedge Entity = Thesis \wedge isSubstring(STQEntity, e) \wedge (EClass(e) \in TAClasses(tt_i) \vee TAClasses(tt_i) = \emptyset) \wedge EClass(e) \notin TANotClasses(tt_i)\}.$$

3.5. **Search for distractors**. Distractors are searched for types of tasks *CO, CS, RW*. The search operator looks like that:

$$WTAEntities(tt_i) = \{e: e \in E \wedge (EClass(e) \in TAClasses(tt_i) \vee TAClasses(tt_i) = \emptyset) \wedge EClass(e) TANotClasses(tt_i) \wedge \neg corresponds(e, STQEntity)\}.$$

The test task is constructed. Transition to the construction of the next test task (p.3). If the required number of test tasks is constructed, the test contruction is complete.

Testing results

Testing was performed for 29 second-year students in engineering software disciplines. The subjects covered by testing were mostly related to subjects learned by this group during the previous year of study, some topics were related to the current semester. The test results were evaluated as a percentage of the correct and false responses. When passing a test based on *Low* complexity model, the results of testing in the majority corresponded to a score of 90%. At the *Middle* level – 75%, while on *Hard* – 65%. The *Low* level checks knowledge type of *Understand and Remember*. At this test level difficult tasks were rarely encountered, mostly they needed skills of memorization and orientation in the control area. The *Middle* level, due to the task of *MI* comparing, verifies knowledge type of *Apply*. It took more time to comprehend the issue, as well as skills were necessary for knowledge application. This can be explained that concepts from various training fragments of a controlled set of content were used, that required a student to have a deep understanding of the subject area. The *Hard* level because of *GF* tasks and, in particular, *RW* ones tests Knowledge type *Analyze*. The user needed to understand the principle necessary to remove the extra answers, that required the ability to analyze the statements and possess analytical skills in the subject area. It should be noted that the students who showed the highest level of semester education, in general, had a higher percentage in *Hard* level testing. So, we can confirm the success of differentiated testing on the basis of the proposed model and the need for its further research to fully cover the modified Bloom scale [7].

Conclusion

In this paper the modification of the method of automated test construction based on the conceptual-theses model was proposed, that provided differentiation of control per the level of complexity with verification of cognitive goals achievement in accordance with the modified Bloom scale [7]. New types of tasks were introduced, the structure of the test task templates was modified and a formal model of the test generation based on the complexity level was presented. A software implementation of the proposed formal apparatus was carried out and initial test was performed, which confirmed the adequacy and the perspective of the implemented modifications. Among the areas of further research there is the detailed formalization of the task solving of data sufficiency for the generation of the test, the improvement of the quality of test tasks of different types and the construction of new task templates. Particular attention is required to provide additional means for checking the achievement of cognitive goals in accordance with the entire Bloom scale [7].

References

1. Avanesov V.S. The composition of the test tasks. Educational book for university professors, school teachers, graduate students and students of pedagogical universities. 2 ed.,rev.. etc. M.:Adept. 1998 - 217p.
2. S. V.Tytenko “Test tasks generation in the distancelearning system on the basisof didactic text formalization model”, Research Bulletin of NTUU" Kyiv Polytechnic Institute", no1(63), pp. 47–57, 2009

3. Tytenko S.V, Gagarin O.O. Practical implementation of automation testing technology based on conceptual model. Education and virtuality - 2006. Collection of scientific works of the 10th International Conference of the Ukrainian Association of Distance Education / In common. edit VA Grebenyuk, Dr. Kinshuk, V.V. Semenets .- Kharkiv-Yalta: UADO, 2006.- P. 401-412.
4. Petrova L.G. Using the modified conceptual-thesaurus model for the automated formation of the test questions database in the systems of computerization of education / L.G. Petrova, S.A. Petrov // Information technologies and teaching aids. - 2012. - No. 4 (30). - P. 1-13.
5. Melnyk A.M. Method of generation of test tasks based on the system of semantic classes / Melnyk AM, Pasichn R.M. // Vestnik TSTU. - 2010. - Volume 15. - No. 1. - P. 187-193
6. Tanchenko S. S. Elimination of language discrepancy in test tasks generated on the basis of a conceptual-thesaurus model / S. S. Tanchenko, S. V. Titenko, O. Gagarin // XII International scientific conference named after T. A. Taran "Intellectual Analysis of Information of IAI-2014", Kyiv, May 14-16, 2014: cb. tr. / ch. edit SVSirota. - K.: Prosvita, 2014. - P. 196-200.
7. Krathwohl, D. A revision of Bloom's taxonomy: an overview, Theory into Practice, 41(4). - 2002.- 212 - 218.