A semantics-based approach to automatic generation of test questions and automatic verification of user answers in the intelligent tutoring systems

Wenzu Li
Belarussian State University Informatics and Radioelectronics
Minsk, Belarus
lwzzggml@gmail.com

***Abstract*—The article is dedicated to the problem of test question generation and user answer verification in the intelligent tutoring systems. The approach of using knowledge base to automatically generate various types of test questions in the intelligent tutoring systems developed using OSTIS Technology and the approach of realizing automatic verification of user answers based on various semantic structures of described knowledge are introduced in detail in this article.**

***Keywords*—test question generation, user answer verification, OSTIS Technology, intelligent tutoring systems, semantic structure, knowledge base**

I. Introduction

As an activity of the progress and development of human society, education has made a unique contribution to the progress of human civilization, especially with the development of science and technology, education is playing an increasingly important role in modern society. In recent years, with the development of modern information technology such as artificial intelligence, computer researchers have begun to apply artificial intelligence technology to the field of education. The application of artificial intelligence technology in the field of education can not only improve the learning efficiency of learners, but also an important means to ensure the fairness of education. Among them, the most representative product combining artificial intelligence technology and education is the intelligent tutoring systems (ITS) [5].

Compared with the traditional multimedia training system (MTS), ITS has the following characteristics:

- able to conduct free man-machine dialogue;
- providing personalized learning strategies;
- automatic solution of test questions;
- automatic generation of test questions;
- automatic verification of user answers;
- etc.

Among them, automatic generation of test questions and automatic verification of user answers are the most basic and important functions of ITS. It allows automation of the entire process from test question generation, exam paper generation to automatic verification of user answers and scoring of exam papers. This can not only greatly improve the efficiency of testing the user's knowledge level, but also reduce their learning cost, while eliminating human factors to ensure the fairness and justice of the testing process as much as possible.

Although some approaches and systems for automatic generation of test questions and automatic verification of user answers have been proposed and developed by some scientific research teams in recent years with the development of related technologies such as semantic web and natural language processing (NLP), these approaches and systems have many shortcomings, for example:

- only simple objective questions can be generated;
- most of the existing answer verification approaches and systems only support the verification of user answers to objective questions;
- some existing approaches to verifying user answers to subjective questions are based on keyword matching and probability statistics and do not consider the semantic similarity between answers;
- partially semantic-based verification approaches to user answers to subjective questions can only calculate the similarity between answers with simple semantic structures;
- components developed using existing approaches to test question generation and user answer verification can only be used in the corresponding systems and are not compatible with each other;
- automated implementation of the entire process from test question generation to user answer verification is not supported [1], [2], [6].

Objective questions refer to a type of question that has a unique standard answer. In this article, objective questions include: multiple-choice questions, fill in the blank questions and judgment questions. Objective questions differ from subjective questions, which have more than one potential correct answer and sometimes have room for a justified opinion. Subjective questions in this article include: definition explanation questions, proof questions and problem-solving task [8].

For the above reasons, an approach to automatic generation of test questions and automatic verification of user answers in ITS developed using OSTIS Technology (Open Semantic Technology for Intelligent Systems) is proposed in this article, and the implementation process of the approach is

described in detail in this article, that is, the development of a universal subsystem for automatic generation of test questions and automatic verification of user answers. The basic principle of automatic generation of test questions in this article is to first summarize a series of test question generation strategies based on the structural characteristics of the ostis-system (system built using OSTIS Technology) knowledge base and the knowledge representation structure therein, and then use these test question generation strategies to extract corresponding semantic fragments from the knowledge base and generate semantic models corresponding to test questions [1], [4]. The basic principle of test question answer verification is to first calculate the similarity between the semantic graph of the standard answer and the semantic graph of the user answer, and then realize the automatic verification of the user answer based on the calculated similarity and the evaluation strategy of the corresponding test question. A semantic graph is a network that represents semantic relationships between concepts. In the ostis-systems, the semantic graph is constructed using SC-code (as a basis for knowledge representation within the OSTIS Technology, a unified coding language for information of any kind based on semantic networks is used, named SC-code) [4], [6]. It should be emphasized that the semantic graph corresponding to the test question and its corresponding natural language description are converted to each other using the natural language interface [7]. The approach proposed in this article needs to solve the following tasks:

- automatic generation of a number of test questions from the knowledge base and storage in the corresponding sections of the subsystem knowledge base;
- design and build subsystem knowledge bases for storing generated test questions;
- according to the needs of users, the corresponding types of test questions are extracted and composed of exam papers;
- calculating the similarity between the semantic graphs of the answers to the objective questions;
- calculating the similarity between the semantic graphs of the answers to the definition explanation questions;
- calculating the similarity between the semantic graphs of the answers to the proof questions and the problem-solving task;
- automatic verification of test question answers and automatic scoring of exam papers based on the calculated similarity and the evaluation strategy of the corresponding test questions.

It should be emphasized here that in the previous articles, we have introduced the implementation process of the corresponding approaches by module (automatic test question generation module and user answer automatic verification module). For example, in the literature [6] we detail the approach to automatically generate various types of test questions from the knowledge base of the ostis-systems, and in the literature [8] we detail the approach to automatically verify user answers in the ostis-systems (including verification of user answers to subjective questions and verification of user answers to objective questions). Therefore, this article focuses on the automation of the entire process from test question generation to user answer verification, and the development of a universal subsystem for automatic generation of test questions and automatic verification of user answer. The approach proposed in this article does not rely on any natural language, but in order to explain how the proposed approach works, the semantic fragments and illustrations selected in this article are presented in English. Among them, the discrete mathematics ostis-system and the euclidean geometry ostis-system will be used as demonstration systems for the subsystem developed using the proposed approach.

II. Existing approaches and problems

A. Automatic generation of test questions

Approach to automatic generation of test questions mainly studies how to use electronic documents, text corpus and knowledge bases to automatically generate test questions quickly and flexibly. Among them, the knowledge base stores highly structured knowledge that has been filtered, and with the development of semantic networks, using the knowledge base to automatically generate test questions has become the most important research direction in the field of automatic generation of test questions [5], [9], [13]. Some of the research results are listed below:

- an approach to using classes, instances, attributes and relationships between them in the OWL ontology for generating multiple-choice questions is presented in the literature [12]. The W3C Web Ontology Language (OWL) is a Semantic Web language designed to represent knowledge about things, groups of things, and relations between things. Ontology is a type of knowledge, each of which is a specification of the corresponding subject domain, focused on describing the properties and relations of concepts that are part of the specified subject domain;
- an approach to automatically generate objective questions using an ontology created by Protégé [11] is presented in the literature [10].

These approaches mainly have the following problems:

- the approach of using electronic documents to automatically generate test questions requires a large number of sentence templates;
- the creation of text corpus requires a lot of human resources to collect and process various knowledge;
- existing approaches can only be applied in the corresponding systems and are not compatible;
- existing approaches only allow to generate simple objective questions.

B. Automatic verification of user answers

Automatic verification of user answers is divided into verification of answers to objective questions and verification of answers to subjective questions. The basic principle of verification of answers to objective questions is relatively

simple, i.e., it is enough to determine whether the string of the standard answer and the string of the user's answer match. The answers to subjective questions are usually not unique, so the basic principle of verification of answers to subjective questions is to calculate the similarity between standard answers and user answers, and then to implement automatic verification of user answers based on the calculated similarity and the evaluation strategy of the corresponding test questions. The more similar the standard answer and the user answer are, the higher the similarity between them [14], [16], [17]. Verification of answers to subjective questions is divided into the following categories according to the approach used to calculate similarity:

- Based on keyword phrases
  This type of approach first allows to split the sentences into keyword phrases and then calculate the similarity between them according to the matching relationship of keyword phrases between sentences. Representative approaches include:
  - N-gram similarity
  - Jaccard similarity
- Based on vector space model (VSM)
  The basic principle of VSM is to use traditional machine learning algorithms to first convert sentences into vector representations, and then use the distance calculation formula between vectors to calculate the similarity between them [15]. Representative approaches include:
  - TF-IDF
  - Word2vec
  - Doc2Vec
- Based on deep learning
  This type of approach allows the use of neural network models to calculate the similarity between sentences [18]. Representative neural network models include:
  - Tree-LSTM
  - Transformer
  - BERT
- Based on semantic graph
  The basic principle of calculating the similarity between answers (i.e., sentence or short text) using this type of approach is to first convert the answers into a semantic graph representation using natural language processing tools (such as syntactic dependency trees and natural language interfaces), and then calculate the similarity between the semantic graphs (i.e., similarity between answers). In ITS knowledge is stored in the form of semantic graphs, so this type of approach provides the possibility to compute the similarity between any two semantic graphs in ITS. The main advantage of this type of approach is computing the similarity between answers based on semantics. One of the most representative approaches is SPICE (Semantic Propositional Image Caption Evaluation) [19].

These approaches mainly have the following problems:

- the keyword phrase-based approach does not take into account the order between words in a sentence;
- the VSM-based approach leads to the generation of high-dimensional sparse matrices, which increases the complexity of the algorithm;
- semantic graph-based approaches supporting only the description of simple semantic structures;
- these approaches cannot determine whether the sentences are logically equivalent to each other;
- these approaches are dependent on the corresponding natural language.

Therefore based on the existing approaches to automatically generate test questions using knowledge bases, approaches to calculate the similarity between answers using semantic graphs, and OSTIS Technology, an approach to automatically generate test questions and automatically verify user answers using semantics is proposed in this article.

III. Proposed approach

The main task of this article is to detail an approach to automatic generation of test questions and automatic verification of user answers in the ostis-systems and to develop a universal subsystem based on this approach. Where the universality of the subsystem means that the subsystem can be easily transplanted between different ostis-systems. The proposed approach can be divided into two parts according to the functions to be implemented, i.e., automatic generation of test questions and automatic verification of user answers [8]. Therefore, we will introduce the implementation process of these two parts separately.

A. Automatic generation of test questions

The basic principle of automatic generation of various types of test questions (including objective questions and subjective questions) in the ostis-systems is to first extract the corresponding semantic fragments from the knowledge base using a series of test question generation strategies summarized based on the knowledge representation approach and the knowledge description structure in the framework of OSTIS Technology, then add some test question description information to the extracted semantic fragments, and finally store the semantic fragments describing the complete test questions in the corresponding section of the universal subsystem [1]. When exam papers needs to be generated, the subsystem allows to extract some corresponding test questions from the subsystem knowledge base according to the parameters input by the user and combine them into exam papers. The test questions and exam papers in the form of semantic graphs are converted into natural language descriptions using a natural language interface. Since in the literature [6] we have detailed some of the strategies used for automatic generation of test questions in the ostis-systems, we next select only some of the test question generation strategies for introduction.

- Test question generation strategy based on class
  This type of test question generation strategy is used to automatically generate objective questions based on

various relations between classes. It is further divided into:

- Based on "inclusion*" relation
  The inclusion relation is one of the most frequently used relations in the knowledge base of the ostis-systems, which is satisfied between many classes (including subclasses), so that the inclusion relation between classes can be used to generate objective questions. The set theory expression form of inclusion relation between classes is as follows: $S_i \subseteq C(i \geq 1)$, ($S$-subclass, i-subclass number, C-parent class). The following shows a semantic fragment in the knowledge base that satisfies the inclusion relation in SCn-code (one of SC-code external display languages) [1], [4]:

  binary tree
  ⇐ *inclusion**:
  directed tree
  ⇒ *inclusion**:
  - *binary sorting tree*
  - *brother tree*
  - *decision tree*

  Consider the example of a multiple-choice question generated using this semantic fragment according to the strategy of inclusion relations, which has the natural language form shown below:
  <<Binary tree does not include ( )?>>
  A. directed tree B. brother tree
  C. decision tree D. binary sorting tree
  Similarly, other types of objective questions can be generated using this strategy;
- Based on "subdividing*" relation
  The result of set subdivision is to get pairs of disjoint sets, and the union of these disjoint sets is the original set. The subdividing relation is also an important relation in the knowledge base, so that semantic fragments in the knowledge base that satisfy this relation can be used to generate objective questions;
- Based on "strict inclusion*" relation
  Strict inclusion relation is a special form of inclusion relation ($S_i \subset C(i \geq 1)$). Using strict inclusion relation to automatically generate objective questions is similar to using inclusion relation.

Other strategies used to generate objective questions include:

- Test question generation strategy based on elements;
- Test question generation strategy based on identifiers;
- Test question generation strategy based on axioms;
- Test question generation strategy based on relation attributes;
- Test question generation strategy based on image examples.

The process of generating subjective questions using subjective question generation strategy is as follows:

- searching the knowledge base for semantic fragments describing the definition, proof or solution of the question using logic rules (i.e. templates constructed using SC-code);
- storing the found semantic fragments in the corresponding section of the knowledge base of the subsystem;
- using manual approaches or automatic approaches (such as natural language interfaces) to describe the definition, proof process or solution process of the corresponding test question according to the knowledge representation rules (i.e. standard answers to subjective questions). Among them, standard answers to subjective questions are represented using SCg-code (SCg-code is a graphical version for the external visual representation of SC-code) or SCL-code (a special sub-language of the SC language intended for formalizing logical formulas) [1], [4].

Using these test question generation strategies described above allows various types of test questions to be generated automatically from the knowledge base. These automatically generated test questions are stored in the knowledge base of the subsystem according to their type and the corresponding test question generation strategy, this type of storage allows to quickly and dynamically generate exam papers according to the needs of user needs. In the next section we will describe in detail the construction of the knowledge base of the subsystem and the way in which test questions are stored in it. The proposed approach to generating test questions has the following advantages:

- OSTIS Technology supports uniform knowledge representation approaches and knowledge description structures, so the proposed approach to generating test questions can be used in different ostis-systems;
- the generated test questions are described using SC-code, so they do not rely on any natural language;
- using the proposed test question generation approach, not only objective questions but also subjective questions can be generated.

B. Automatic verification of user answers

In the ostis-systems, test questions are stored in the knowledge base in the form of semantic graphs, so the most critical step of user answer verification is to calculate the similarity between the semantic graph of standard answer and the semantic graph of user answer, and when the similarity is obtained and combined with the evaluation strategy of the corresponding test questions, the correctness and completeness of user answers can be verified [2], [8].

User answer verification is classified according to the type of test questions: 1. verification of answers to objective questions; 2. verification of answers to subjective questions. Although the most critical step of answer verification is all about calculating the similarity between the semantic graphs of answers, the knowledge types (factual knowledge and logical knowledge) and and knowledge structures used to describe different types of test questions are not the same, so the approach to calculate the similarity between the semantic

graphs of answers to different types of test questions are different. Factual knowledge refers to knowledge that does not contain variable types, and this type of knowledge expresses facts. Logical knowledge usually contains variables, and there are logical relations between knowledge. In the ostis-systems SCL-code is used to represent logical knowledge. In this article objective questions, proof questions and problem-solving task are described using factual knowledge, and definition interpretation questions are described using factual knowledge and logical knowledge together.

C. Verification of answers to objective question

The semantic graphs used to describe objective types of test questions and their answers in the knowledge base have the same semantic structure, so the similarity between answers to such types of test questions can be calculated using the same approach. Since the user answers in the natural language to the objective questions are already aligned with the existing knowledge in the knowledge base when they are converted to semantic graphs using the natural language interface, that is, the elements representing the same semantics in the knowledge base have the same main identifier (identifier is a file that can be used to denote (name) an entity in the framework of external language) [7]. Therefore, it is not necessary to consider the differences between concepts at the natural language level when calculating the similarity between semantic graphs of answers to objective questions, that is, the similarity between answers is calculated based on the semantic structure. The basic principle for calculating the similarity between semantic graphs of answers to objective questions is shown below:

- the semantic graph of standard answers (s) and the semantic graph of user answers (u) are decomposed into substructures according to the rules of representation of factual knowledge;
- using formulas (1), (2), and (3) to calculate the precision P_{sc}, recall R_{sc} and similarity F_{sc} between semantic graphs.

$$P_{sc}(u,s) = \frac{|T_{sc}(u) \otimes T_{sc}(s)|}{|T_{sc}(u)|} \tag{1}$$

$$R_{sc}(u,s) = \frac{|T_{sc}(u) \otimes T_{sc}(s)|}{|T_{sc}(s)|} \tag{2}$$

$$F_{sc}(u,s) = \frac{2 \cdot P_{sc}(u,s) \cdot R_{sc}(u,s)}{P_{sc}(u,s) + R_{sc}(u,s)} \tag{3}$$

The main calculation parameters in the formulas include:

- $T_{sc}(u)$ — all substructures after the decomposition of the user answers u;
- $T_{sc}(s)$ — all substructures after the decomposition of the standard answers s;
- $\otimes$ — binary matching operator, which represents the number of matching substructures in the set of two substructures.

Once the similarity of the answers is obtained, the correctness and completeness of the user answers to the objective questions can be verified by combining them with the evaluation strategy of the objective questions. The evaluation strategy of the objective questions is shown below:

- if there is only one correct option for the current test question, only if the standard answer and the user answer match exactly, the user answer is considered correct and the user gets the maximum score (Max_{score});
- if the current question has multiple correct options (multiple-choice question with multiple correct options and partially fill in the blank questions):
  - as long as the user answer contains an incorrect option, the user answer is considered incorrect and the user score is 0;
  - if all the options in the user answer are correct, but the number of correct options is less than the number of correct options in the standard answer, the user answer is considered correct but incomplete. At this time, the user answer score is $R_{sc} * Max_{score}$;
  - if all the options in the standard answer match exactly with all the options in the user answer, the user answer is exactly correct, and the user score is Max_{score}.

Fig. 1 shows an example of verification of user answer to subjective question in SCg-code.

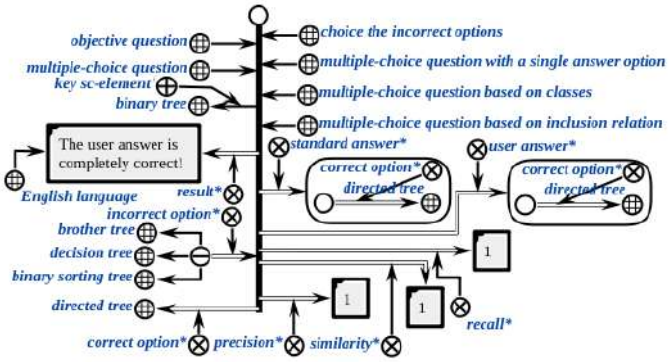

Figure 1. An example of verification of user answer to subjective question.

D. Verification of answers to subjective questions

The most critical step of verification of answers to subjective questions is also the calculation of similarity between semantic graphs of answers, but the knowledge types and knowledge structures used to describe different types of subjective questions and their answers are not the same in the ostis-systems. Thus the approach to calculating the similarity between the semantic graphs of the answers to the subjective questions is further divided into: 1. the approach to calculating the similarity between answers to definition explanation questions; 2. the approach to calculating the similarity between answers to proof questions and problem-solving task.

Calculating the similarity between answers to definition explanation questions

The answers to the definition explanation questions in the ostis-systems are described in the form of logical formulas

using factual knowledge and logical knowledge (SCL-code). Logic formulas are powerful tools for formal knowledge representation in the framework of OSTIS Technology, which are expanded based on the first-order predicate logic formulas and inherits all the operational properties of first-order predicate logic formulas [4]. It is to be emphasized that when calculating the similarity between the answers to the definition explanation questions, the factual knowledge in the semantic graph of the user answers has been aligned with the existing knowledge in the knowledge base (using natural language interfaces) [7]. In order to calculate the similarity between the semantic graphs of the answers to the definition explanation questions the following tasks need to be solved:

- automatic selection of potential equivalent standard answer;
- establishing the mapping relationship of potential equivalent variable sc-node pairs between the semantic graphs of the answers;
- calculating the similarity between semantic graphs;
- if the similarity between semantic graphs is not equal to 1, they also need to be converted to the prenex normal form (PNF) representation separately, and then the similarity between them is calculated again [23].

Because some definition explanation questions sometimes have multiple standard answers, but the logical formulas used to represent them formally are not logical equivalents (described according to different conceptual systems). For example, the definition of equivalence relation: 1. in mathematics, an equivalence relation is a binary relation that is reflexive, symmetric and transitive; 2. for any binary relationship, if it is a tolerant relationship and is transitive, then it is an equivalence relation. The logical equivalence between semantic graphs in the ostis-systems is divided into two types: 1. logical equivalence between semantic graphs described based on logical formulas; 2. logical equivalence between semantic graphs based on different conceptual systems. This type of equivalence is further classified according to the type of knowledge:

- logical equivalence between semantic graphs based on factual knowledge;
- logical equivalence between semantic graphs based on logical knowledge (for example, the definition of equivalence relation).

Therefore, when calculating the similarity between answers, it is necessary to filter a standard answer that best matches the user answer from multiple possible standard answers in advance. Therefore an approach to filter a standard answer that best matches the user answer according to the predicate similarity between answers is proposed in this article. This working principle of this approach is shown below:

- finding all the predicates in each answer (non-repeating);
- calculating the predicate similarity between the user answer and each standard answer using formulas (1), (2) and (3);
- the standard answer that is most similar (maximum similarity) to the user answer is selected as the final standard answer.

Since the semantic graphs used to describe the answers to the definition explanation questions are constructed based on logical formulas, the variables sc-nodes (equivalent to the bound variables in the predicate logic formula) are included in the semantic graphs. In order to calculate the similarity between semantic graphs, the most critical step is to establish the mapping relationship of potential equivalent variable sc-node pairs between semantic graphs. Therefore, based on the existing ontology mapping methods and mapping systems (for example, ASMOW, RiMOM, etc.) an approach to establish the mapping relationship of potential equivalent variable sc-node pairs between semantic graphs according to semantic structures (various sc-constructions) are proposed in this article [20], [21], [22].

In the ostis-systems, the sc-construction composed of sc-tuple, relation sc-node, role relation sc-node and sc-connector is used to describe logical connectives (such as negation ($\neg$) and implication ($\rightarrow$), etc.) and quantifiers (universal quantifier ($\forall$) and existential quantifier ($\exists$)), atomic logic formula (various sc-constructions) or multiple atomic logic formulas that satisfy conjunctive relation are contained in the sc-structure and connected with the corresponding sc-tuple, and these sc-elements together constitute the semantic graph used to represent the user answer [1], [22]. All sc-tuples and sc-connectors form a tree, which completely describes the logical sequence between connectives and quantifiers in the logical formula. Because the sc-structure containing the atomic logical formula is connected to the corresponding sc-tuple, as long as the position of each sc-tuple and sc-structure in the semantic graph is determined, the position of each variable sc-node in the semantic graph can be determined. An approach to numbering each sc-tuple and sc-structure in the semantic graph according to a depth-first search strategy (DFS) is proposed in this article. The working process of this approach is shown below:

- starting from the root of the tree structure composed of sc-tuples, each sc-tuple node in the tree is numbered in turn according to the DFS strategy and the priority of the current sc-node (for example, the sc-node priority of the ”if’ ” condition is higher than the sc-node of ”else’ ” conclusion) (the numbering sequence starts from 0);
- according to the numbering sequence of sc-tuple, each sc-tuple in the tree is traversed from small to large, and the sc-structure connected to the current sc-tuple is numbered while traversing (the numbering sequence starts from 1).

For a detailed procedure for numbering sc-tuples and sc-structures, please refer to the literature [8]. In answer verification, if the standard answer and the user answer are exactly equal, it means that the atomic logic formulas with the same semantics between the answers have the same position in the semantic graph (That is, the numbering sequence of sc-structure is the same). Therefore, in this article, the mapping

relationship of potential equivalent variable sc-node pairs will be established based on the matching relationship of the sc-constructions in the same position between the answers. The establishment of mapping relationship of the potential equivalent variable sc-node pairs between answers mainly includes the following steps:

1) according to the numbering sequence of the sc-structure in the semantic graph, each time a sc-structure pair with the same number is found from the standard answer and the user answer;
2) according to the priority order (from high to low) of the various types of sc-constructions used to describe the atomic logic formula, it is determined in turn whether the current sc-structure pair contains this type of sc-construction at the same time. If the current sc-structure pair contains this type of sc-construction at the same time, then, according to the matching relationship of each sc-element between the current sc-construction in the standard answer and the current sc-construction in the user answer, the mapping relationship of the potential equivalent variable sc-node pairs between the current sc-construction pair is established;
3) repeat step 1 — step 2 until all mapping relationships between semantic graphs are established [8].

Fig. 2 shows an example of establishing the mapping relationship between semantic graphs in SCg-code.

In Fig. 2, the definition of the inclusion relation is described $(\forall A \forall B((A \subseteq B) \Longleftrightarrow (\forall a(a \in A \rightarrow a \in B)))$.

When the mapping relationship between the potential equivalent variable sc-node pairs between the semantic graphs is established, the similarity between the answers can be calculated. The process of calculating the similarity between the semantic graphs of the answers to the definition explanation questions is shown below:

- decomposing the semantic graph of standard answer and the semantic graph of user answer into substructures according to the rules of representation of factual knowledge and logical knowledge;
- numbering the sc-tuples and sc-structures in the semantic graphs of the answers, respectively, and establishing the mapping relationship of potential equivalent variable sc-node pairs between the semantic graphs;
- using formulas (1), (2) and (3) to calculate the precision P_{sc}, recall R_{sc} and similarity F_{sc} between semantic graphs.

Since the semantic graphs of answers to definition explanation questions are described based on logical formulas, if the similarity between semantic graphs is not equal 1 ($F_{sc} < 1$), it is also necessary to determine whether their logical formulas are logically equivalent. There is such a theorem in predicate logic: any predicate logic formula has a PNF that is equivalent to it. Because the logical formulas in the framework of OSTIS Technology are extended based on predicate logical formulas, it also has such a property. Therefore we can consider converting semantic graphs based on logical formula descriptions to PNF descriptions, and then determine whether logical equivalence is satisfied between them [23], [24]. However, the PNF of the logic formula is not unique, and the reasons why the PNF is not unique include:

- the used order of different logical equivalence formulas (conversion rules). For example, converting $(\forall x F(x) \wedge \neg \exists x G(x))$ to PNF:
  - $\forall x F(x) \wedge \neg \exists x G(x)$
    $\Leftrightarrow \forall x F(x) \wedge \forall x \neg G(x)$
    $\Leftrightarrow \forall x(F(x) \wedge \neg G(x))$, (equivalence rule)
  - $\forall x F(x) \wedge \neg \exists x G(x)$
    $\Leftrightarrow \forall x F(x) \wedge \forall y \neg G(y)$, (renaming rule)
    $\Leftrightarrow \forall x \forall y(F(x) \wedge \neg G(y))$, (rule of expansion of quantifier scope)
- the order of the quantifiers in PNF;
- etc.

Therefore, based on the approach to convert predicate logic formulas into PNF and some characteristics of logic formulas in ostis-systems, an approach to convert logic formulas into unique (deterministic) PNF according to strict restriction rules is proposed in this article. The strict restrictions mainly include the following:

- in order to solve the problem that PNF are not unique due to the order in which the logical equivalence formulas are used, we specify that the renaming rule is preferred when converting logical formulas to PNF;
- in order to solve the problem that the PNF is not unique due to the order of the quantifiers, an approach to move all quantifiers to the forefront of the logical formula strictly according to the priority of the quantifiers is proposed in this article. The movement process of quantifiers is shown below:
  - if no quantifiers exist at the front of the logical formula, all existential quantifiers are moved to the front of the logical formula in preference;
  - if the last quantifier at the forefront of the logical formula is a universal quantifier, the universal quantifiers in the logical formula will be moved preferentially to the front of the formula;
  - if the last quantifier at the forefront of the logical formula is a existential quantifier, the existential quantifiers in the logical formula will be moved preferentially to the front of the formula.
- the logical formula used to represent the answer to the definition explanation question can usually be expressed in the following form: $(Q_1 x_1 Q_2 x_2 ... Q_n x_n (A \leftrightarrow B))$, where $Q_i (i = 1, ...n)$ is a quantifier [8], [25]. A is used to describe the definition of a concept at a holistic level, and it does not contain any quantifiers. B is used to explain the semantic connotation of a definition at the detail level, and it is usually a logical formula containing quantifiers (also known as a logical sub-formula). Therefore, based on the characteristics of the logical formula and in order to simplify the knowledge processing, it is only necessary to convert the logical formula B to PNF;

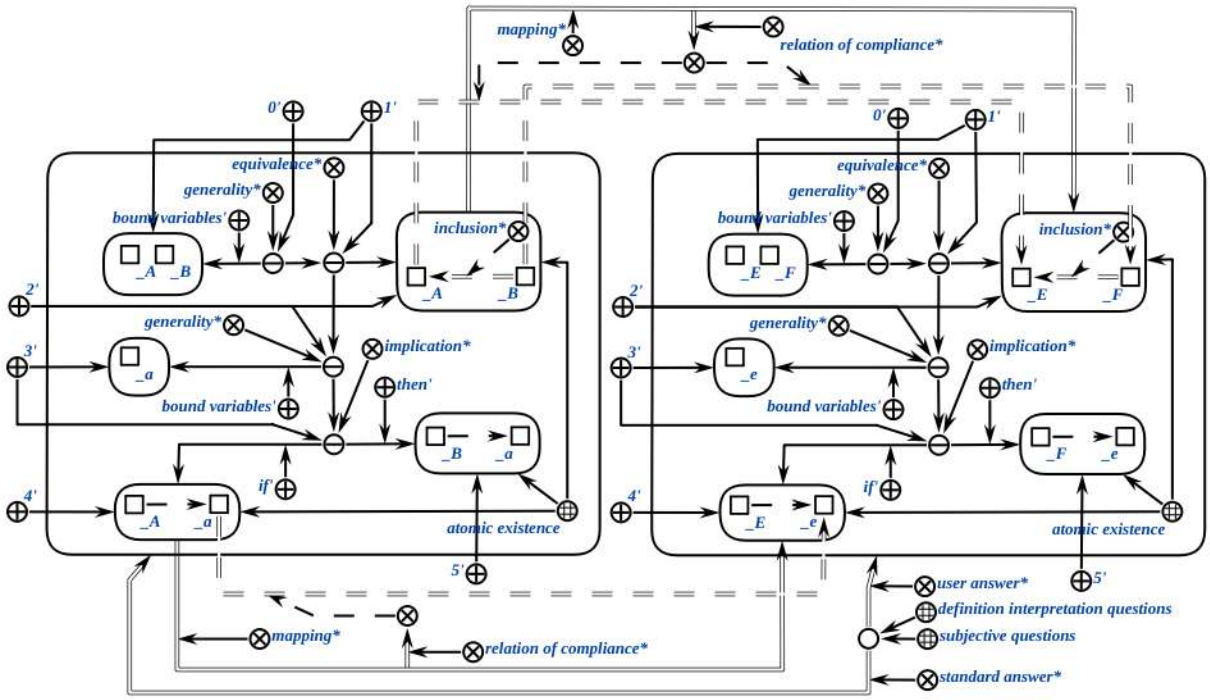

Figure 2. An example of establishing the mapping relationship between semantic graphs.

- to simplify the knowledge processing, only the implication connective need to be eliminated when converting logic formulas to PNF;
- multiple atomic logic formulas connected using the same conjunctive connective are preferentially merged into one whole (i.e. they are merged into the same sc-structure).

The process of converting the semantic graphs of answers to definition explanation questions into PNF descriptions according to strict restriction rules is shown below:

- if there are multiple sc-structures in the semantic graph connected by the same conjunctive connective, the sc-constructions contained in them are merged into the same sc-structure;
- eliminating all the implication connectives in the semantic graphs;
- moving all negative connectives in the semantic graphs to the front of the corresponding sc-structure;
- using renaming rules so that all bound variables in the semantic graphs are not the same;
- moving all quantifiers to the front of the logical formula;
- merging again the sc-structures in the semantic graphs that can be merged.

Fig. 3 shows an example of converting a semantic graph into PNF representation in SCg-code ($\forall A \forall B((A \subseteq B) \leftrightarrow \forall a(a \in A \rightarrow a \in B)) \Leftrightarrow \forall A \forall B((A \subseteq B) \leftrightarrow \forall a(\neg(a \in A) \vee (a \in B))))$.

It should be emphasized that if the calculated similarity between the semantic graphs of PNF representation is not 1 ($F_{sc} < 1$), the similarity between the semantic graphs

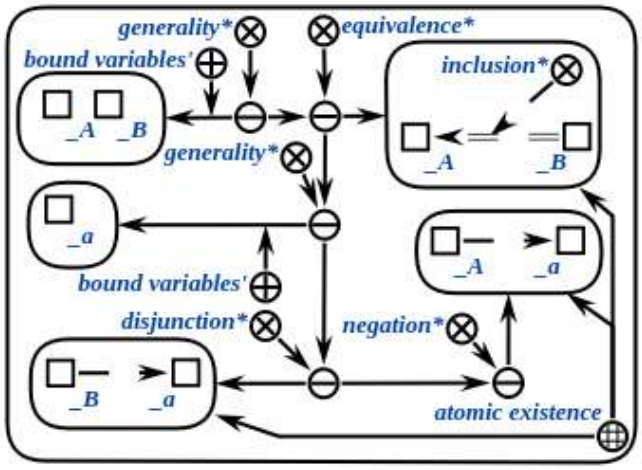

Figure 3. An example of converting a semantic graph into PNF representation.

calculated for the first time is used as the final answer similarity. When the similarity between the answers is obtained and then combined with the evaluation strategy of the subjective questions, the correctness and completeness of the user answers can be verified [8].

Calculating the similarity between answers to proof questions and problem-solving task

Both proof questions and problem-solving task in mathematics follow a common task-solving process:

1) the set (Ω) of conditions consisting of some known conditions;
2) deriving an intermediate conclusion using some of the known conditions in Ω and adding it to Ω. Each element in Ω can be regarded as a solving step;
3) repeat step 2) until the final result is obtained [26], [27].

This task-solving process is abstracted as a directed graph, whose structure is in most cases an inverted tree (in special cases the directed graph will contain cycle), and is called a reasoning tree (i. e. the reasoning tree of the standard answer) [26]. Fig. 4 shows an example of a reasoning tree.

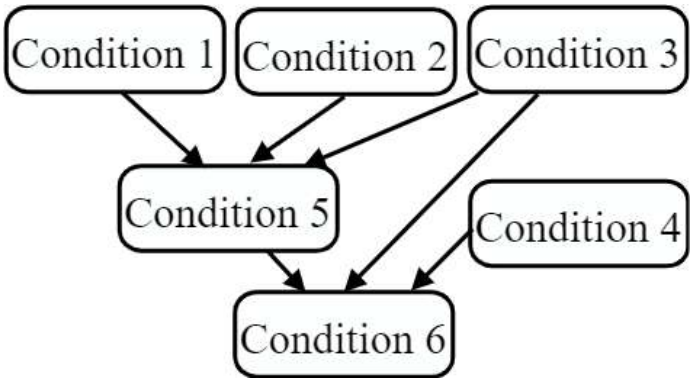

Figure 4. An example of a reasoning tree.

The user answer to the proof question or problem-solving task is a linear structure consisting of some solving steps (i.e. known conditions, intermediate conditions or conclusions), each of which satisfies a strict derivation relationship and logical relationship if the user answer is completely correct. The automatic verification process of user answers to this type of test questions is the same as the traditional manual answer verification process, i.e., verifying whether the current solving step of the user answer is a valid conclusion of the partial solving step preceding that step. This means whether the solving step in the user answer corresponding to the parent node in the reasoning tree always is located after the solving steps in the user answer corresponding to the child nodes.

The semantic graphs of user answers to proof questions and problem-solving task in the ostis-systems are linear structures consisting of some semantic sub-graphs for describing the solving steps and some semantic fragments for describing the logical order and transformation processes between the semantic sub-graphs [1], [4]. The construction process and semantic specification of semantic graphs of user answers to proof questions and problem-solving tasks are described in detail in the literature [3]. The semantic graph of standard answers to this type of test questions is an reasoning tree consisting of a number of search templates (which can be abstracted as the nodes in the tree). Each search template is constructed strictly according to the solving steps of the corresponding test question (i.e., according to the known conditions, intermediate conditions and conclusions in Ω). The search template in the ostis-systems is used to search in the knowledge base for all semantic fragments corresponding to it, and it is constructed based on the SCL-code. The following takes a real problem-solving task as an example to introduce the constructing of the semantic graph of its standard answer (reasoning tree). Description of the problem-solving task: <<Two equal circles externally tangent to other and a third circle the radius of which is 4. The segment that connects the tangent points of the two equal circles to the third circle is 6. Find the radii of equal circles>>. Fig. 5 shows the explanatory picture of the task.

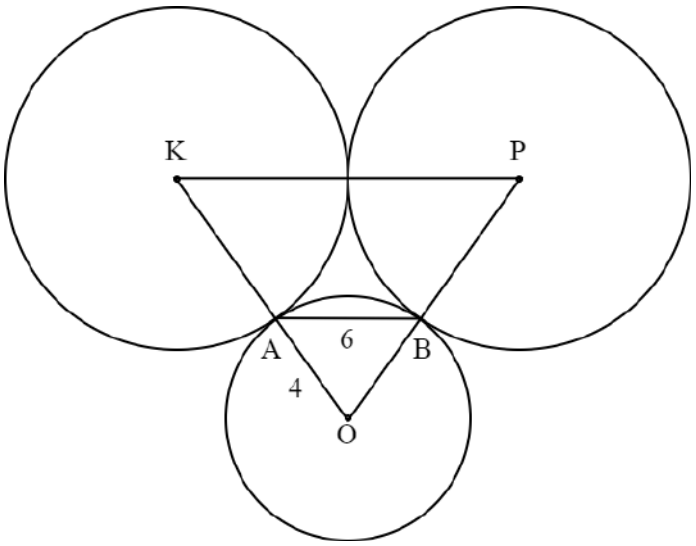

Figure 5. Explanatory pictures for problem-solving task.

Description of user answer in natural language:
1) $\because KP = 2 * R$
2) $\because KO = 4 + R$
3) $\therefore \Delta AOB \backsim \Delta KOP$
4) $\therefore KA = R = 12$

Fig. 6 shows an example of the semantic graph of the standard answer in SCg-code.

The user answers in natural language are converted into semantic graphs using natural language interfaces. Therefore, when calculating the similarity between the semantic graphs of the answers, it is not necessary to consider the differences of the concepts at the natural language level [7]. Fig. 7 shows an example of the semantic specification of a segment in the knowledge base in SCg-code.

From the above example, it can be seen that Segment AB and Segment BA are represented by the same sc-node, they are just two identifiers of the sc-node.

Therefore based on the previously introduced principles of automatic verification of user answers to proof questions and problem-solving task and the semantic models of answers in the ostis-systems, an approach to calculate the similarity between the semantic graphs of answers to proof questions and problem-solving task according to the reasoning tree of standard answer (semantic graph of standard answer) is proposed in this article. The calculation process of similarity between semantic graphs is shown below:

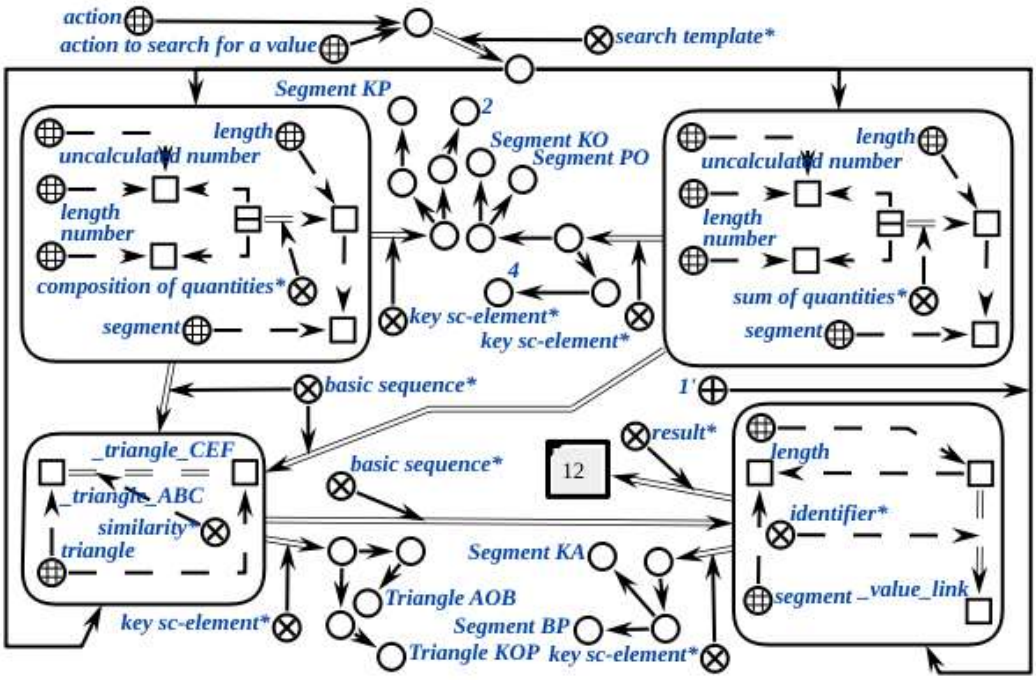

Figure 6. An example of the semantic graph of the standard answer.

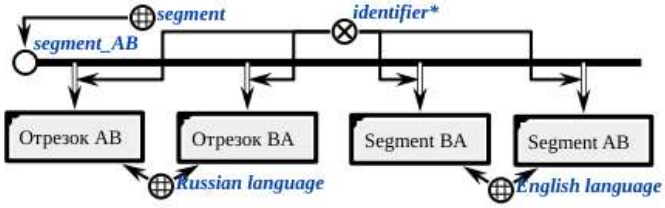

Figure 7. An example of the semantic specification of a segment.

1) numbering each semantic sub-graph (solving step) in the semantic graph of user answers (the numbering order started from 1);
2) each node in the reasoning tree (the search template) is traversed in turn according to the DFS strategy. At the same time, the corresponding semantic sub-graph that is included in the semantic graph of the user answer are searched in the knowledge base using the search template currently being traversed. If such a semantic sub-graph exists, then determine whether the searched semantic sub-graph number is smaller than the semantic sub-graph number corresponding to the search template of the current search template parent node (except for the root node of the reasoning tree), and if so, the searched semantic sub-graph is considered correct;
3) repeat step 2) until all search templates in the reasoning tree have been traversed and the number of correct semantic sub-graphs is counted at the same time;
4) using formulas (1), (2) and (3) to calculate the precision P_{sc}, recall R_{sc} and similarity F_{sc} between answers. Parameters in the formula are redefined:
   - $|T_{sc}(u)|$ — the number of all semantic sub-graphs in the semantic graph of the user answer u;
   - $|T_{sc}(s)|$ — the number of all search templates in the reasoning tree s;
   - $|T_{sc}(u) \otimes T_{sc}(s)|$ — the number of correct semantic sub-graphs.

Once the similarity between the answers to the proof questions and the problem-solving task is obtained, the correctness and completeness of the user answers can be verified combined with the evaluation strategy for the subjective questions.

The evaluation strategy for the subjective questions is shown below:

- if the similarity between the answers is equal to 1 ($F_{sc} = 1$), the user's answer is completely correct and the user gets the maximum score (Max_{score});
- if the similarity between the answers is less than 1 ($F_{sc} < 1$) and the precision is equal to 1 ($P_{sc} = 1$), the user answer is correct but incomplete and the user score is $R_{sc} * Max_{score}$;
- if the similarity between the answers is greater than 0 and less than 1, and the precision is less than 1 ($0 < F_{sc} < 1$ and $P_{sc} < 1$), then the user answer is partially correct and the user score is $F_{sc} * Max_{score}$;
- if the similarity between the answers is equal to 0 ($F_{sc} = 0$), the user answer is wrong and the user score is 0 [8].

The proposed approach to automatic verification of user answers has the following advantages:

- verifying the correctness and completeness of user answers based on semantics;
- the correctness and completeness of user answers to any type of test question can be verified and logical equivalence between answers can be determined;
- allowing the calculation of the similarity between any two semantic graphs in the knowledge base;
- the proposed approach can be used in different ostis-systems.

IV. Knowledge base of the subsystem

The knowledge base of subsystem is mainly used to store automatically generated test questions of various types, and it also allows to automatically extract a series of test questions and form exam papers according to user requirements. Therefore, in order to improve the efficiency of accessing the knowledge base of the subsystem and the efficiency of extracting the test questions, an approach to construct the knowledge base of the subsystem according to the type of test questions and the generation strategy of the test questions is proposed in this article.

The basis of the knowledge base of any ostis-system (more precisely, the sc-model of the knowledge base) is a hierarchical system of subject domains and their corresponding ontologies [1], [3], [4]. Let's consider the hierarchy of the knowledge base of subsystem in SCn-code:

Section. Subject domain of test questions
⇐ *section decomposition**:
{
- *Section. Subject domain of subjective questions*
  ⇐ *section decomposition**:
  {
  - *Section. Subject domain of definition explanation question*
  - *Section. Subject domain of proof question*
  - *Section. Subject domain of problem-solving task*
  }
- *Section. Subject domain of objective questions*
  ⇐ *section decomposition**:
  {
  - *Section. Subject domain of multiple-choice question*
  - *Section. Subject domain of fill in the blank question*
  - *Section. Subject domain of judgment question*
  }
}

Next, taking the subject domain of the objective questions as an example, let us consider its structural specification in SCn-code:

Subject domain of objective questions
∈ *subject domain*
∋ *maximum class of explored objects'*:
objective question
∋ *not maximum class of explored objects'*:
- *multiple-choice question*
- *fill in the blank question*
- *judgment question*

In this article objective types of test questions are decomposed into more specific types according to their characteristics and corresponding test question generation strategies. Next, taking the multiple-choice question as an example let us consider its semantic specification in SCn-code:

multiple-choice question
∈ *maximum class of explored objects'*:
Subject domain of multiple-choice question
⇐ *subdividing**:
{
- *multiple-choice question based on relation attributes*
- *multiple-choice question based on axioms*
- *multiple-choice question based on image examples*
- *multiple-choice question based on identifiers*
- *multiple-choice question based on elements*
  ⇐ *subdividing**:
  {
  - *multiple-choice question based on role relation*
  - *multiple-choice questions based on binary relation*
  }
- *multiple-choice question based on classes*
  ⇐ *subdividing**:
  {
  - *multiple-choice question based on subdividing relation*
  - *multiple-choice question based on inclusion relation*
  - *multiple-choice question based on strict inclusion relation*
  }
}
⇐ *subdividing**:
{
- *multiple-choice question with multiple answer options*
- *multiple-choice question with a single answer option*
}
⇐ *subdividing**:
{
- *choice the incorrect options*
- *choice the correct options*
}

V. Problem solver

The problem solver of any ostis-system (more precisely, the sc-model of the ostis-system problem solver) is a hierarchical system of knowledge processing agents in semantic memory (sc-agents) that interact only by specifying the actions they perform in the specified memory [1].

Therefore, in order to implement the corresponding tasks, the problem solver for the automatic generation of test ques-

tions and automatic verification of user answers is developed in this article, and its hierarchy is shown as follows in SCn-code:

Problem solver for the automatic generation of test questions and automatic verification of user answers
⇐ *decomposition of an abstract sc-agent**:
{
- *Sc-agent for automatic generation of test questions*
  ⇐ *decomposition of an abstract sc-agent*:*
  {
  - *Sc-agent for quick generation of test questions and exam papers*
  - *Sc-agent for generating single type of test questions*
  - *Sc-agent for generating a single exam paper*

  }
- *Sc-agent for automatic verification of user answers*
  ⇐ *decomposition of an abstract sc-agent*:*
  {
  - *Sc-agent for automatic scoring of exam papers*
  - *Sc-agent for calculating similarity between answers to objective questions*
  - *Sc-agent for calculating the similarity between answers to definition explanation questions*
  - *Sc-agent for converting a logical formula into PNF*
  - *Sc-agent for calculating the similarity between the answers to proof questions and problem-solving task*

  }

}

The main function of the sc-agent for quick generation of test questions and exam papers is to automate the whole process from test question generation to exam paper generation by initiating the corresponding sc-agents (sc-agent for generating single type of test questions and sc-agent for generating a single exam paper). The main function of the sc-agent for generating single type of test questions is to automatically generate a series of test questions from the knowledge base using logical rules built on the basis of SC-code [4]. The logical rules for generating test questions are constructed strictly in accordance with the strategies for generating test questions described earlier. Fig. 8 shows an example of a logic rule for generating multiple-choice question constructed based on a strategy of inclusion relation.

The main function of the sc-agent for automatic scoring of exam papers is to implement automatic verification of user answers to various types of test questions and automatic scoring of exam papers by initiating sc-agents for calculating the similarity between user answers and sc-agents for converting a logical formula into PNF.

VI. Conclusion and further work

A semantic-based approach to automatic generation of test questions and automatic verification of user answers in the ostis-systems is proposed in this article. And based on the proposed approach a universal subsystem for automatic generation of test questions and automatic verification of user answers is developed. The developed subsystem supports automating the entire process from test question generation, to the scoring of exam papers.

The basic principle of automatic generation of test questions is the automatic generation of objective and subjective questions from the knowledge base using some rules constructed based on the structural features of the knowledge base of the ostis-systems. The basic principle of automatic verification of user answers is to first calculate the similarity between the semantic graphs of answers, and then combine it with the evaluation strategy of the corresponding test question to achieve automatic verification of user answers (including the logical equivalence judgment between answers). The proposed approach to calculate the similarity between answers also supports the calculation of the similarity between any two semantic graphs in the knowledge base, so the approach can be used in other tasks in the future as well (such as ontology mapping, knowledge fusion, etc.).

The effectiveness of the developed subsystem will be evaluated in future work.

Acknowledgment

The work in this article was done with the support of research teams of the Department of Intelligent Information Technologies of Belarusian State University of Informatics and Radioelectronics. Authors would like to thank every researcher in the Department of Intelligent Information Technologies.

References

[1] V. V. Golenkov and N. A. Guljakina, “Proekt otkrytoj semanticheskoj tehnologii komponentnogo proektirovanija intellektual’nyh sistem. chast’ 1: Principy sozdanija project of open semantic technology for component design of intelligent systems. part 1: Creation principles],” Ontologija proektirovanija [Ontology of design], no. 1, pp. 42–64, 2014.

[2] V. Golenkov, N. Guliakina, I. Davydenko, and A. Eremeev, ”Methods and tools for ensuring compatibility of computer systems,” in Otkrytye semanticheskie tehnologii proektirovanija intellektual’nyh sistem [Open semantic technologies for intelligent systems], V. Golenkov, Ed., BSUIR. Minsk, BSUIR, 2019, pp. 25–52.

[3] D. Shunkevich, “Metodika komponentnogo proektirovaniya sistem, upravlyaemyh znaniyami,” in Otkrytye semanticheskie tehnologii proektirovanija intellektual’nyh sistem [Open semantic technologies for intelligent systems], V. Golenkov, Ed., BSUIR. Minsk, BSUIR, 2015, pp. 93–110.

[4] (2022, NOV) Ims.ostis metasystem. [Online]. Available: https://ims.ostis.net

[5] Xu G. P., Zeng W. H., Huang C. L. Research on intelligent tutoring system. Application research of computers, 2009, Vol. 26(11), pp. 4020-4030.

[6] Li W., Grakova N., Qian L. Ontological Approach for Question Generation and Knowledge Control. Communications in Computer and Information Science, 2020, Vol. 1282, pp. 161-175.

[7] Qian L., Sadouski M., Li W. Ontological Approach for Chinese Language Interface Design. Communications in Computer and Information Science, 2020, Vol. 1282, pp. 146-160.

[8] Wenzu Li, ”Development of a problem solver for automatic answer verification in the intelligent tutoring systems,” in Otkrytye semanticheskie tehnologii proektirovanija intellektual’nyh sistem [Open semantic technologies for intelligent systems], V. Golenkov, Ed., BSUIR. Minsk, BSUIR, 2021, pp. 169–178.

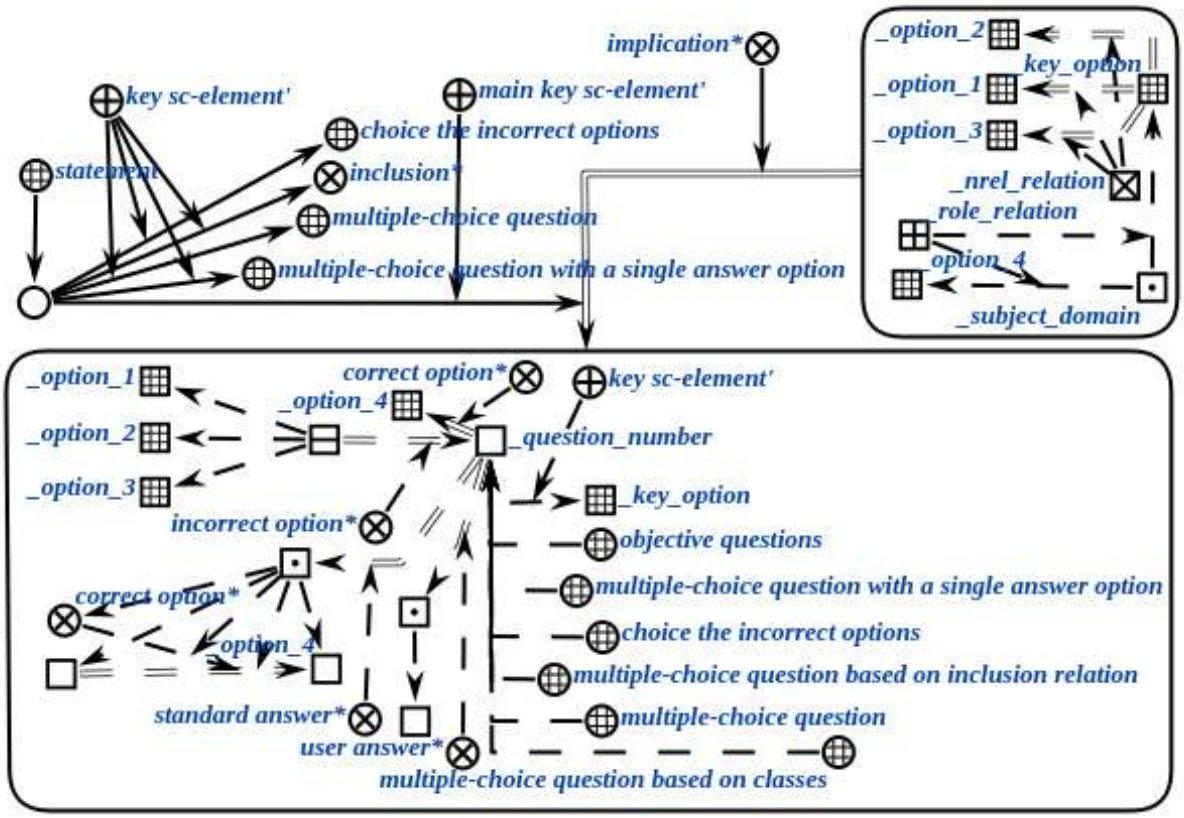

Figure 8. An example of a logic rule for generating multiple-choice question.

[9] Mousavinasab E., Zarifsanaiey N. R., Niakan Kalhori. S. Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. Interactive Learning Environments, 2021, Vol. 29(1), pp. 142-163.

[10] Li, H. Research on item automatic generation based on DL and domain ontology. Journal of Changchun University of Technology (Natural Science Edition), 2012, Vol. 33(04), pp. 461-464.

[11] (2022, NOV) Protégé [Electronic resource, Online]. Available: http://protege.stanford.edu

[12] Andreas P., Konstantinos K., Konstantinos K. Automatic generation of multiple-choice questions from domain ontologies. In: IADIS International Conference e-Learning, 2008, pp. 427-434.

[13] Arjun S. B., Manas K., Sujan K. S. Automatic Generation of Multiple Choice Questions Using Wikipedia. International Conference on Pattern Recognition and Machine Intelligence, 2013, Vol. 8251, pp. 733-738.

[14] Wan C. L., Yang Y. H., Deng F. A review of text similarity calculation methods. Information science, 2019, Vol. 37(33), pp. 158-168.

[15] Shahmirzadi O., Lugowski A., Younge K. Text similarity in vector space models: a comparative study. 2019 18th IEEE international conference on machine learning and applications (ICMLA), IEEE, 2019, pp. 659-666.

[16] Li X. J. Realization of automatic scoring algorithm for subjective questions based on artificial intelligence. Journal of Jiangnan University (Natural Science Edition), 2009, Vol. 08(03), pp. 292-295.

[17] Wan H. R., Zhang Y. S. Review on Research Progress of Text Similarity Calculation. Journal of Beijing Information Science (Technology University), 2019, Vol. 34(01), pp. 68-74.

[18] Mingyu Ji., Xinhai Zhang. A Short Text Similarity Calculation Method Combining Semantic and Headword Attention Mechanism. Scientific Programming, 2022.

[19] Anderson P., Fernando B., Johnson M., Gould S. Spice: Semantic propositional image caption evaluation. In: European Conference on Computer Vision, Springer, 2016, pp. 382-398.

[20] Su J. L., Wang Y. Z., Jin X. L., et al. Knowledge Graph Entity Alignment with Semantic and Structural Information. Journal of Shanxi University(Nat. Sci. Ed.), 2018, Vol. 42(1), pp. 23-30.

[21] Zhuang Y., Li G. L., Feng J. H. A Survey Entity Alignment of Knowledge Base. Journal of Computer Research and Development, 2016, Vol. 53(1), pp. 165-192.

[22] Wang X. Y., Hu Z. W., Bai R. J., et al. Review on Concepts, Processes, Tools and Methods Ontology Integration. Library and Information Service, 2011, Vol. 55(16), pp. 119-125.

[23] Fujiwara M., Kurahashi T. Prenex normal form theorems in semi-classical arithmetic. The Journal of Symbolic Logic, 2022, pp. 1-31.

[24] Kowalski R. Predicate logic as programming language. In: IFIP congress, 1974, Vol. 74, pp. 544-569.

[25] Pan M., Ding Z. A simple method for solving prenex disjunction (conjunction) normal forms. Computer Engineering and Science, 2008, Vol. 30(10), pp. 82-84.

[26] Jin-zhong Z., Xian-qing H., Xiao-shan G. Automated production of traditional proofs for theorems in euclidean geometry. Annals of Mathematics and Artificial Intelligence, 1995, Vol. 13(1), pp. 109–137.

[27] Zhang J., Peng X., Chen M. Self-evident automated proving based on point geometry from the perspective of Wu's method identity. Journal of Systems Science and Complexity, 2019, Vol. 32(1), pp. 78-94.

ОСНОВАННЫЙ НА СЕМАНТИКЕ ПОДХОД К АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ ТЕСТОВЫХ ВОПРОСОВ И АВТОМАТИЧЕСКОЙ ПРОВЕРКЕ ОТВЕТОВ ПОЛЬЗОВАТЕЛЕЙ В ИНТЕЛЛЕКТУАЛЬНЫХ ОБУЧАЮЩИХ СИСТЕМАХ

Ли Вэньцзу

Данная статья посвящена проблеме генерации тестовых вопросов и проверки ответов пользователей в интеллектуальных обучающих системах. В данной статье подробно представлен подход к автоматической генерации различных типов тестовых вопросов на основе базы знаний в интеллектуальных обучающих системах, разработанных с использованием Технологии OSTIS, и подход к реализации автоматической проверки ответов пользователей на основе различных семантических структур описанных знаний.

Keywords—генерация тестовых вопросов, проверка ответов пользователей, Технология OSTIS, интеллектуальные обучающие системы, онтология, база знаний, семантическая структура

Как деятельность прогресса и развития человеческого общества, образование внесло уникальный вклад в прогресс человеческой цивилизации, особенно с развитием науки и техники, образование играет все более важную роль в современном обществе. В последние годы, с развитием современных информационных технологий, таких как искусственный интеллект, компьютерные исследователи начали работать над применением технологии искусственного интеллекта в сфере образования. Применение технологии искусственного интеллекта в сфере образования может не только повысить эффективность обучения учащихся, но и стать важным средством обеспечения справедливости образования. Среди них наиболее представительным продуктом, объединяющим технологии искусственного интеллекта и образования, являются интеллектуальные обучающие системы (ИОС). Особенно после вспышки COVID-19 в 2020 году была подчеркнута важность и актуальность разработки ИОС. По сравнению с традиционной мультимедийной обучающей системой (МОС), ИОС имеет следующие характеристики:

- способен вести свободный человеко-машинный диалог;
- предоставление персонализированной педагогической услуги;
- автоматическое решение тестовых вопросов;
- автоматическая генерация тестовых вопросов;
- автоматическая проверка ответов пользователей;
- и т.д.

Среди них автоматическая генерация тестовых вопросов и автоматическая проверка ответов пользователей являются самыми основными и важными функциями ИОС. Она позволяет автоматизировать весь процесс от генерации тестовых вопросов, формирования экзаменационных билетов до автоматической проверки ответов пользователей и оценки экзаменационных билетов. Это может не только значительно повысить эффективность тестирования уровня знаний пользователей, но и снизить стоимость их обучения, при этом исключая человеческий фактор, чтобы максимально обеспечить справедливость процесса тестирования.

Received 01.11.22