Intelligent Testing in Kyiv: Analytical and Deductive Paradigms and their Implementation

Vitaly Klimenko[1] and Alexander Lyaletski[2]

[1] Institute for Problems of Mathematical Machines and Systems
Glushkov avenue, 42, 03187 Kyiv, Ukraine
klimenko@immsp.kiev.ua
[2] Institute Faculty of Cybernetics, Kyiv National Taras Shevchenko University
Glushkov avenue, 4D, 03680 Kyiv, Ukraine
lav@unicyb.kiev.ua

Abstract. We describe an experience of the Kiev schools on analytical transformations and automated deduction with respect to intelligent testing of knowledge obtained through distant learning with or without the use of electronic textbooks. Leaving aside the simple "choose from prescribed answers" tests, intelligent testing of two kinds is considered: one of them relies on symbolic computation and the other on deduction.

Keywords: Analytical transformations, automated deduction, electronic learning, knowledge testing, intelligent testing. on symbolic computation and the other on deduction.

Key Terms. Development, Machine Intelligence, Software Engineering Process

1 Introduction

At the beginning of the 1960th, Academician V.M.Glushkov initiated two ways of the development of computer-aided mathematics in Kiev: one was concerned with symbolic computations (i.e. computer algebra systems in the modern terminology) and the other with the automatization of deduction steps (i.e. automated reasoning systems). Now, these two ways play important role in information technologies. That is why it is interesting to know impact they can have on the development of intelligent testing in e-learning in current days.

There exist a great number of software systems for authoring of "electronic textbooks" for various disciplines being taught in secondary schools, colleges, and universities. Their common feature is their orientation towards a broad spectrum of educational branches and, owing to this, towards the simplest kind of examination of trainee's knowledge based on choosing a right answer from a number of alternatives proposed by an examiner. The downside of this technique is that it gives the trainee an incentive to guess a right answer rather than really look for it, which does not allow a tutor to estimate trainee knowledge correctly. A list of "prescribed answers" is notably inconvenient for mathematical disciplines, where a solution to a problem often consists in deriving an analytic (symbolic) expression or in a formal proving when a chain of deductive steps assuring the validity of a statement under consideration must be constructed. Thus, we have the following ways for the computer-aided testing of knowledge obtained by a trainee in the (e-)learning of a subject: the query-answering method, the analytical transformation, the deductive construction, and, perhaps, their combinations.

The state of the art in symbolic computation and automated reasoning has initiated transition from the simple "choose-an-answer" testing to the more intelligent and complex ones: the analytical and deductive reasoning. The first approach is suitable for testing on the base of finding analytical solutions of tasks in algebra, trigonometry, physics, and so on (for both secondary schools and higher education institutions). The second one is useful in studying a mathematical theory allowing its complete formalization for computer checking of logical steps of a trainee proving a certain sentence of a theory under consideration (the same concerns any knowledge domain, where formalization and deduction are admissible).

The present paper is devoted to the brief description of some of the approaches of Ukrainian researchers relating to symbolic (analytical) computation and automated reasoning that can be used for the construction of software tools for intelligent testing (other than the "choose-an-answer" method).

2 Symbolic Computation

Testing on the base of symbolic computation is needed when a solution for an equation must have the form of an analytical (symbolic) expression, for example, a root of an algebraic equation or an equation in partial derivatives. In order to perform such a testing, we need a "shell" that can assure that the symbolic expression proposed by the examinee is correct, that is, it can be transformed into a formula, given by an examiner by means of symbolic computation. Such analytical verification is very appropriate for testing in various domains of physics, trigonometry, algebra, and so on. In the first place, it requires the following generic tools:

– tools for performing arbitrary-precision computation for integers, rationals and complex numbers;
– methods for determining whether two symbolic expressions are equal; these are usually based on various systems of term rewriting rules;
– methods for term normalization (in particular, normalization to certain conventional mathematical forms);
– tools for analytical transformations of mathematical expressions defined in the terms of hierarchical data structures of arbitrary complexity.

The Institute for Problems of Mathematical Machines and Systems National, Academy of Sciences of Ukraine, (IPMMS NASU) started research in this domain in 1960s and created a family of hierarchically developing computer algebra systems in the frame of the "Analitic" project under the leadership of Academician

V.M.Glushkov, which was strongly oriented to assisting in performing analytical transformation operations such as algebraic simplification, mathematical derivation and integration, etc. The specialized computer series "MIR" (Mashina dlya Inzhenernyh Raschetov - Engineering Computation Machine): "MIR-1", "MIR-2", and "MIR-3" having their input languages of the three first version of the "Analitic" language [1]. Later, the developed algorithms were implemented into the SM 1410 computers ("Analitic-79" [2]) and into the usual IBM PC ("Analitic-93" [3] and "Analitic-2000" [4]). Now, the modern project versions called "Analitic-2007" [5] and "Analitic-2010" [6] are in progress and usage. Let us give a brief description of some of their features and implementation.

2.4 Analitic-2007

The "Analitic-2007" version was implemented at the beginning of 2007. It inherited all the best features of all its predecessors and di_ered from the previous versions by deeper algebraic transformations, more detailed classification of algebraic tools, sophisticated facilities of calculations control, and improved interactive methods.

The "Analitic-2007" programming system is intended for IBM PCs and is operated as an application for the operating system Windows-2000 and higher versions. It consists of the system kernel and a number of program packs. The compact kernel provides a user with a large quantity of programs, supports the semantic integrity of the "Analitic" language, the universality of its functional properties, and the operability of the "Analitic-2007" system in the environment of the different Windows operating systems. It performs compiling and recompiling programs and data, executing programs, transforming language objects (including programs being considered as objects of the language).

The system automatically determines the size of memory accessible for performing a program and occupies the maximal scope of accessible memory by default. A user has a possibility for determining the size of memory necessary for the normal execution of his programs. In the case of exceeding the existent memory size, the "Analitic-2007" programming system uses a virtual memory.

2.5 Analitic-2010

The last version "Analitic-2010" has significant changes in the kernel of "Analitic-2007" and it is partially transferred to the .NET platform [6]. A new user-friendly graphic interface missing in the previous versions was constructed.

When implementing "Analitic-2010", the main efforts were directed to the improvement of the operating stability of the kernel. For this, the parser was recoded without any changes in the language "Analitic". As a result, the software of the previous versions was transferred to the new one.

The new interface is oriented to the efficient handling of data in the interactive mode and the faster generation of new programs. It is equipped with a complete code editor supporting intelligent input and all the possibilities inherent in a modern integrated development environment.

Finally, we note that all the "Analitic" family systems are used actively for finding analytical solutions of tasks in mechanics, astronomy, differential equations theory. Besides, a number of experiments were performed in automated performing of analytical transformation in various fields of mathematical learning and testing, for example, in checking of algebraic and trigonometric identities. Thus, they can be very useful in e-learning.

3 Deductive Computation

Deductive testing consists in logically verifying a chain of reasoning steps expressed in a formal language. The Ukrainian approach is based on a so-called evidential paradigm [7] being the modern vision of the Evidence Algorithm advanced by V.M.Glushkov in [8]. In accordance with it, the language should be close to a natural language and it should be used in a course of training. Deductive testing is to be used in mathematical disciplines having the form of formal axiomatic theories containing logical inference rules. It may also be successfully applied in jurisprudence, where the testing consists in performing legally and logically valid reasoning steps, or in creating legal documents consistent with the current legislation.

3.1 Evidential Paradigm

The evidential paradigm [7] itself is based on the declarative way of representation and logical processing of knowledge having the form of formalized texts (containing axioms, definitions, propositions and so on, when we deal with mathematical problems). Systems exploiting this paradigm are called automated reasoning systems or, in particular, systems for automated theorem proving. Note that this approach turns out to be the most adequate for the automated logical inference search as well as for verification of a formal text (mathematical or not), namely, for checking the validity of all the reasoning steps in it.

For the purposes of deductive testing in the frame of the evidential paradigm, we adhere to the following requirements for a testing environment:

- For presentation of reasoning, a trainee must use a (semi-)formal language which is close to the natural language of mathematical publications. This language preserves the structure of the problem in question and the texts in this language can be translated into some representation convenient for computer processing.
- Each reasoning step (in a natural form) from the text under verification must be "obvious" to a computer in the sense that is can be checked by it. A checking procedure must evolve for incrementing reasoning steps as much as possible. It must combine general methods of logical inference search with heuristic reasoning techniques such as induction, case reasoning, definition handling, and so on. Such collection of reasoning techniques must also grow and evolve.
- In the case of necessity, symbolic computation methods must be attracted to the testing of trainee knowledge;
- Formal knowledge accumulated in the system (and used in training) must be organized in a hierarchical information environment.

From 1998, the evidential paradigm is actively investigating in Ukraine, mainly, at the Faculty of Cybernetics of the Kiev National Taras Shevchenko University.

3.2 SAD System

As a result of the development of the evidential paradigm, the System for Automated Deduction (SAD) has been constructed. Its basic purpose is to assist to a man in "doing" mathematics, for example, in mathematical text verifying and/or theorem proving. It can be downloaded or accessed online on the Evidence Algorithm project site "http://nevidal.org" (see also [9] containing the history of the development of the Evidence Algorithm and SAD system).

In accordance with the evidential paradigm, the SAD system can perform the following chain of text transformations oriented to theorem proving/verifying [10]:

– At the first level, an input text written at the original formal language ForTheL [11] being close to the natural English language of mathematical publications is syntactically analyzed. After this, it can be translated into a first-order language, which permits to perform computer inference search in different logics;
– At the second level, the goal statements are processed one-by-one by a foreground reasoner of SAD. In the verification mode, this module is intended to reduce a given proof task to a number of subtasks for a prover. At present, its toolkit contains some simplification methods on propositional level such as decomposition of a problem under consideration, reasoning by general induction, and others;
– Inference search tasks are resolved at the third level with the help of a first-order prover, for example, with the native prover Moses [12] that is based on a special goal-driven sequent calculus for classical first-order logic with equality.

The original notion of admissible substitution used in the calculus permits to preserve the initial signature of the task so that special accumulated equations can be sent to a specialized solver, e.g. an external computer algebra system. In particular, some of the tools of the "Analitic" programming systems can be used for this in the case of the existence of an according interface.

Note that the SAD system was implemented in such a way that SAD can be connected with an external first-order prover such as Otter [13], SPASS [14], or Vampire [15], E prover [16], and Prover 9 [17].

Therefore, the SAD is a good candidate for deductive testing in e-learning.

4 Conclusion

The above-given analysis of Kiev approaches to symbolic computation and deductive reasoning demonstrates that the advances made by researches at IPMMS and Kiev National Taras Shevchenko University allow introducing and implementing various forms of distant e-learning based on a more thorough and unbiased evaluation of an examinee, which can improve the quality of learning for disciplines which admit (at least partial) formalization. The next step consists in integration of analytical and deductive testing in a common framework, allowing these two forms of intelligent testing to complement and enforce each other.

When the described methods and tools for intelligent testing will be implemented and approved in practice, the next milestone can be proceed. First, these tools can be incorporated into the existing e-learning systems (taking into account the specifics of a domain under study). Second, one can use the proposed framework to design and implement entire electronic courses and textbooks, containing learning material as well as exercises for simple and intellectual testing for objective evaluation of student's knowledge.

References

1. Glushkov V.M., Bodnarchuk V.G., Grinchenko T.A., Dorodnizyna A.A., Klimenko V.P., Letichevsky A.A., Pogrebinsky S.B., Stogniy A.A., and Fishman Yu.S.: ANALITIK (an algorithmic language for description of computational processes using analytical transformations) (in Russian). Cybernetics 3, pp.102--134 (1971).
2. Glushkov V.M., Grinchenko T.A., Dorodnizyna A.A., Drakh A.M., Klimenko V.P., Pogrebinsky S.B., Savchak O.N., Fishman Yu.S., and Tsaryuk N.P.: ANALITIK-79 (in Russian). Technical report, Institute of Cybernetics, Kiev, USSR (1979).
3. Morozov A.A., Klimenko V.P., Fishman Yu.S., Bublik B.A., Gorovoy V.D., and Kalina E.A.: ANALITIK-93 (in Russian), Mathematical Machines and Systems 5, pp. 127--157 (1995).
4. Morozov A. A., Klimenko V. P., Fishman Yu. S., Lyakhov A. L., Kondrashov S. V., and Shvalyuk T. N.: ANALITIK-2000 (in Russian). Mathematical Machines and Systems 1,2, pp. 66--99 (2001).
5. Morozov A.A., Klimenko V.P., Fishman Yu.S., and Shvalyuk T.N.: ANALITIK-2007 (in Russian). Mathematical Machines and Systems 3,4, pp. 8--52 (2007).
6. Klimenko V.P., Lyakhov A.L., Gvozdik D.N., Zakharov S.A., and Shvalyuk T.N.: On the implementation of a new version of the Analitic family CAS (in Russian). In: International conference CMSEE-2010, Poltava (2010).
7. Lyaletski A. and Morokhovets M.: Evidential paradigm: a current state. In: Programme of the International Conference "Mathematical Challenges of the 21st Century", University of California, Los Angeles, USA (2000).
8. Glushkov V.M.: Some problems of automata theory and artificial intelligence (in Russian). Cybernetics 2, pp. 3--13 (1970).
9. Lyaletski A. and Verchinine K:. Evidence Algorithm and System for Automated Deduction: A retrospective view. LNCS, vol. 6167, pp. 411--426 (2010).
10. Anisimov A. V. and Lyaletski A. V.: The SAD system in three dimensions, In: International symposium SYNASC'06, Timisoara, Romania, pp. 85--88 (2006).
11. Vershinin K. and Paskevich A.: ForTheL – the language of formal theories. International Journal of Information Theories and Applications, vol. 7, no. 3, pp. 120--126 (2000).
12. Lyaletski A., Paskevich A., and Verchinine K.: Theorem proving and proof verification in the system SAD. LNCS, vol. 3119, pp. 236--250 (2004).
13. The Otter automated deduction system, http://www.mcs.anl.gov/research/projects/AR/otter/.
14. The SPASS Prover, `http://www.spass-prover.org/`
15. The Vampire prover, `http://www.vprover.org/`
16. The E Theorem Prover, `http://www4.informatik.tu-muenchen.de/~schulz/E/E.html`
17. The Prover9 prover, `www.cs.unm.edu/~mccune/prover9/`