JOURNAL OF APPLIED
COMPUTER SCIENCE
Vol. 18 No. 1 (2010), pp. 39–48

The Gentzen Sequent Calculus in E-Testing. Part I: Foundations

Adam Niewiadomski[1], Andrzej Indrzejczak[2]

[1]*Institute of Information Technology*
Technical University of Łódź
Wólczańska 215, 90-924 Łódź, Poland
aniewiadomski @ ics.p.lodz.pl

[2]*Department of Logics and Methodology of Sciences*
University of Łódź
Kopcińskiego 16/18, 90-232 Łódź, Poland
indrzej @ filozof.uni.lodz.pl

Abstract. *Modern methods and trends in education, especially the so-called* e-learning*, are undeniably more and more popular and widely applied in recent years. The paper is focused on the subdomain of the e-learning area which is named* e-testing *and it concerns the problem of replacing the manual checking and evaluating of distance exams with automated and intelligent procedures run by dedicated computer systems.*

A popular system of automated theorem proving by Gentzen [1] is applied to mechanise the process of checking and evaluating e-tests in set theory and logics. The paper presents the general rules according to which such an e-test should be built, then the evaluating algorithm based on the Gentzen sequent calculus is presented and exemplified, finally the detailed comments on the implementation and the results are given.

Keywords: *E-learning, e-testing methods, Gentzen's sequent calculus, e-tests evaluating and scoring, decision support.*

1. Motivation and Background

Present forms of academic education are being more and more extensively supported by information technologies, particularly by Internet applications. Profits generated by replacing the so-called *campus-based education* with distance education are obvious – they are expressed not only as financial income and/or cost reduction, but also in ethical and methodological aspects, such as the access to education for disabled people (usually strongly limited by classic infrastructure of schools), and as the possibility given to students of learning, studying, and taking exams at any time, any place, and at their own speed [2, 3]. Currently, many high schools are using different Internet-based tools to transfer curricula and programmes to students. There are dedicated Web-systems – *platforms*[1] – which enable running virtual lectures, classes, labs, and tests.

However, while various computer techniques are being employed to provide the communicativeness and attractiveness of the presented materials (multimedia, videoconferences, 3D graphics, video and animations, etc.), evaluating and scoring e-tests still must be done manually by a tutor. It obviously engages his/her perception to a great extent, and, in addition, is highly uncomfortable and inefficient. Moreover, this process (checking and scoring) must be repeated by a teacher at least the number of times equal to the number of students in a course; it may cause the participation of the "human factor" in scoring and evaluating process, particularly visible in assigning unequal notes to the same answer but coming from different students. In fact, the only support that is given to a tutor by e-learning platforms, is the storage of answers and scores in a database and the remote (i.e. via network) access to them. Nevertheless, current state of the art and information technologies still lack of methods of intelligent decision support which could remarkably improve e-tutors' and examiners' work efficiency.

Thus, the gist of this article is to present an application of the original methods, based on Gentzen's sequent calculus, employed in the automation of those elements of e-learning which presently are based mostly on human "live and online" participation, and engage perception to a great extent: examining, testing, and evaluating results of tests, termed *electronic testing*, or *e-testing* for short.

The solution presented here may efficiently and almost at all replace teacher's participation in e-testing process in chosen domains. The new role of e-tutor is foreseen as that of a supervisor of automated scoring and evaluating e-tests, rather

[1] e.g. Lotus LearningSpace, Oracle iLearning.

than as that of a ”clerk” who has to control and review students tests thoroughly as well as to propose and discuss scores for each question/task (notice that, till now, even if he/she is supported by multimedia and network applications, scoring and commenting are performed manually). The presented purpose is analogous to the well-known applications of expert systems in medical diagnosing where data coming from interviews and examinations are processed by computer procedures, and the final result must be accepted or modified to a competent diagnosis and treatment by doctors.

2. Methods of Automated Scoring

2.1. Literature references

As for now, very few works on automated and/or intelligent controlling, evaluation, and scoring of tests on e-learning platforms [4, 5] is present in literature. The approaches for theory and application in modelling of e-students behaviour, knowledge representation, evaluation of tutoring systems, teaching and learning strategies are widely presented in [6].

The point of departure to automated evaluating and scoring checking single- and multiple-choice e-tests via the Hamming distance computed between the set of correct answers and the set of answers given by a student (the smaller the distance is, the higher the score for the test is). Nevertheless, such a method does not allow to score or evaluate the answers to more complicated (with respect to their form) questions like mathematical or logical proofs (e.g. *Prove that* $(p \to q) \wedge (q \to p) \to (p \leftrightarrow q)$), textual and/or open questions (e.g. *What is the highest peak of Himalaya?*), or mixed word tests in foreign languages (e.g. *Put the given words into a correct order: die Zeitung / Ich / lesen / heute / will*).

The method of automated checking the correctness of logical proofs via the Gentzen’s Sequent Calculus has been introduced in [7]; in this article, we describe technical details and the application to a prototype e- learning platform. Methods of automated evaluation according to multi-expert criteria via linguistic summaries of databases are described in [8, 9]. Methods of retranslating data via interval-valued fuzzy sets in e-test management are given in [10, 11]. An interesting application of n-gram string matching algorithms to checking e-tests at programming and at German language are given in [12] and in [13], respectively.

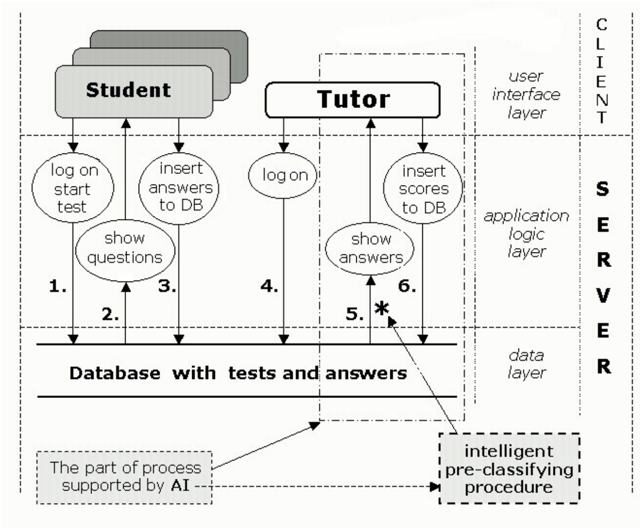

Figure 1. Data flow in e-testing process

2.2. An idea of automated scoring

Implementations of e-learning platforms are usually Web applications with Graphical User Interface (GUI), e.g. Lotus LearningSpace, Oracle iLearning, Tegrity WebLearner Studio, R5, WebCity, Moodle, etc. These applications are designed according to the requirements of the client-server architecture, in detail: the *thin client* architecture, which means that all functions and objects, that the application logic layer consists of, are implemented and run on a server, see Fig. 1. The client application, usually the Web browser collaborating with the server, is responsible for running a personalized GUI only. The thin client architecture is required with respect to the safety of data, in particular, templates of correct answers.

The data flow diagram (DFD) of an application for distance testing through the Web is presented in Fig. 1. The communication between a student and a tutor is, in general, supported by the WWW interface. The schema approximates the data exchange between a student and a teacher during an e-test performed on a platform. Let us briefly comment on this schema. Each student taking a test through the Internet must log on to the platform to obtain rights for solving his/her test[2], and,

[2]Those rights may sometimes depend on the scores already obtained by student during his/her learning, but it is not necessarily an assumption.

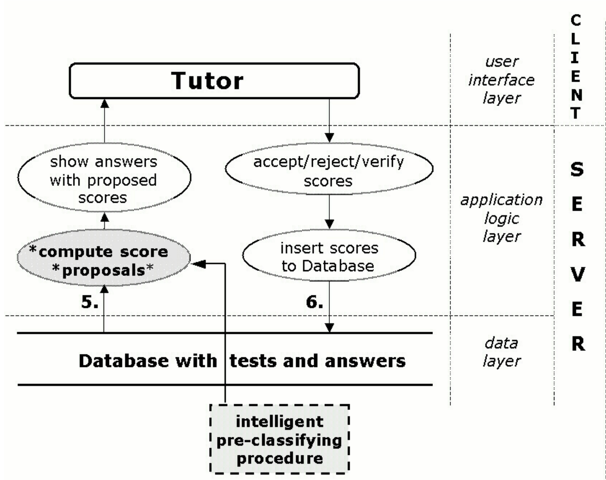

Figure 2. The part of e-testing process supported by AI

as the consequence, to see questions/exercises to be answered/solved (steps 1.,2.). The student solves the test, and then his/her answers are saved in the database (step 3.). This is the end of the test for the student. From that moment, a tutor must check all the answers from students. He/she confirms his/her rights to access the database (step 4.). Then the answers are read (step 5.), and the scores are returned and commented via the application interface, and saved in the database in order to make them accessible for students (step 6.).

Nevertheless, evaluating and scoring depicted in Fig. 1 as steps 5. and 6. are still manual jobs to be done by a tutor. Therefore, the original idea of mechanising the e-testing process is explained in Fig. 2, which is the zoom of the part of Fig. 1 – before the answers are delivered to a tutor, an intelligent procedure is revoked in order to compute suggestions of scores (step 5.). A tutor then accepts and/or modifies each machine-computed score, and finally publishes all the scores on the platform (step 6.).

The clou of the method presented in this article is the point depicted as the shadowed ellipse in Fig. 2; we propose an original solution for the e-tests in logics and set theory. It is based on Gentzen's sequent calculus and it enables the mechanised and automated evaluating and scoring students' answers. The proposed algorithms are intended to be applied as the step 5. in Fig. 2.

3. Gentzen Sequent Calculus

The methods based in distances or similarities, cf. section 2.2, are not sufficient for checking correctness when we deal with logical deductions. One of the most interesting tools that can help is a method proposed by Gentzen [1] commonly called *Sequent Calculus*. The original form of the calculus was not meant by the author as a method for automatic proof search but as a theoretical tool for investigation on Natural Deduction. It was observed however that it is also a handy formalisation for practical applications, especially for decidability results. Below we present original form of Sequent Calculus for Classical Propositional Logic according to Gentzen. Basic units of a calculus are sequents of the form $\Gamma \Rightarrow \Delta$, where Γ and Δ are finite (possibly empty) lists of formulae in the language of propositional logic. One of the common intuitive interpretation of a sequent is a statement that the conjunction of elements of Γ implies disjunction of elements of Δ but other interpretations are also possible. The calculus consists of the rules:

$(AX)\ \varphi \Rightarrow \varphi$

$(Cut)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \varphi, \Pi \Rightarrow \Sigma}{\Gamma, \Pi \Rightarrow \Delta, \Sigma}$

$(W\Rightarrow)\ \dfrac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow W)\ \dfrac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi}$

$(C\Rightarrow)\ \dfrac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow C)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi}$

$(P\Rightarrow)\ \dfrac{\Pi, \varphi, \psi, \Gamma \Rightarrow \Delta}{\Pi, \psi, \varphi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow P)\ \dfrac{\Gamma \Rightarrow \Delta, \psi, \varphi, \Pi}{\Gamma \Rightarrow \Delta, \varphi, \psi, \Pi}$

$(\neg\Rightarrow)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi}{\neg\varphi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow\neg)\ \dfrac{\varphi, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi}$

$(\wedge\Rightarrow)\ \dfrac{\varphi, \psi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow\wedge)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}$

$(\vee\Rightarrow)\ \dfrac{\varphi, \Gamma \Rightarrow \Delta \quad \psi, \Gamma \Rightarrow \Delta}{\varphi \vee \psi, \Gamma \Rightarrow \Delta}$ $\quad (\Rightarrow\vee)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi, \psi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}$

$(\rightarrow\Rightarrow)\ \dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \psi, \Pi \Rightarrow \Sigma}{\varphi \rightarrow \psi, \Gamma, \Pi \Rightarrow \Delta, \Sigma}$ $\quad (\Rightarrow\rightarrow)\ \dfrac{\varphi, \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi}$

The proof of a sequent in SC (Sequent Calculus) is a binary tree where each leaf is labelled with (*AX*), the root is labelled with a proven sequent and every edge of a tree is obtained by the above rules.

The calculus, as it stands, is not very handy for actual proof search but it admits a variety of modifications that make actual proving very easy. From the logical point of view the most important feature of all the rules except (*Cut*) is the *subformula property*: every formula in premises is preserved in a consequence of a rule, either as a subformula of some new formula or without any changes. Concerning (*Cut*), Gentzen has proved that every proof that applies this rule may be transformed into proof with no use of (*Cut*). This is his famous Cut-elimination theorem which has many important consequences, including proof of decidability of classical (and intuitionistic) propositional logic. The procedure, roughly, goes like this: we start with the sequent we want to prove and systematically decompose all formulae applying suitable rules. Finally, we obtain either a proof or a tree where at least one branch ends with nonaxiomatic sequent containing only variables. Cut-elimination has opened the door to invention of many useful methods of automated proof-search like Beth's diagrams [14] or Smullyan's truth-trees [15], generally called as *Tableau Methods* (cf. Indrzejczak, [16]). SC has been also generalised in many ways to plenty of non-classical logics, like modal or temporal ones (e.g. Indrzejczak, [17]).

In the area of automated deduction the most popular system is obviously the method of resolution due to Robinson [18]. But it should be noted that first provers were based on sequent calculus or on tableau methods. In 1960 Hao Wang [19] implemented a prover for first-order logic on IBM 704-computers. At the same time Dag Prawitz, Hakan Prawitz and Neri Voghera [20] independently developed a similar program. It ran on a computer Facit EDB manufactured by AB Advidabergs Industrier. Also nowadays one can find many provers based on sequent calculus, for example: HARP theorem prover due to Oppacher and Suen [21] implemented in LISP, Tatzelwurm due to Kaufl and Zabel [22] or GAZER due to Barker-Plummer and Rothenberg [23] (both implemented in Prolog). Many of such programs have much more extensive application than resolution-based provers; for example HLM (Helsinki Logic Machine), prolog-implemented prover covers about 60 different calculi for many non-classical logics.

By the way, the method of resolution is also closely connected with SC. In a sense it is a particular form of sequent calculus operating on clauses and using a form of cut (namely resolution rule). Hence, in resolution, proof-search is not based on the elimination of cut but, contrary, on the constant use of a special form of this rule. These short remarks show how powerful and general tool SC is, thus it is not surprising that it can be also applied to automated test evaluation. In what follows, we try to show that not only logical matters but also easiness of implementation call for the application of SC.

4. Summary

This paper is the first part of two. It describes the foundations and background of the Gentzen Sequent Calculus and of the e-learning, especially the problems appearing in examining over network platforms, i.e. the so-called e-testing.

The original contribution by the authors is an innovative method of automated checking and scoring e-tests. The details of algorithms, their implementations, and the final conclusions are introduced in Part II [24].

References

[1] Gentzen, G., *Untersuchungen Über das Logische Schliessen*, Mathematische Zeitschrift, Vol. 39, 1934, pp. 176–210, 405–431.

[2] Hameed, C., Clements, M., and Birch, C., *Virtual campus for e-learning*, E-mentor, Vol. 3, 2004, pp. 37–39, (in Polish).

[3] Zając, M., *E-learning from the perspective of a teacher*, E-mentor, Vol. 3, 2004, pp. 4–6, (in Polish).

[4] Guzman, E. and Conejo, R., *Linguistic Data Summaries for Intelligent Decision Support*, Lecture Notes in Computer Science, Vol. 3220, 2004, pp. 12–21.

[5] Melis, E. and Siekmann, J., *ActiveMath: An Intelligent Tutoring System for Mathematics*, Lecture Notes in Artificial Intelligence, Vol. 3070, 2004, pp. 91–101.

[6] Lester, J. C., Vicari, R. M., and Paraguacu, F., *Intelligent Tutoring Systems*, Lecture Notes in Computer Science, Vol. 3220, 2004.

[7] Niewiadomski, A. and Indrzejczak, A., *An application of Gentzen Sequent Calculues in automated evaluation of e-tests*, Scientific Notes of WSHE, Vol. 47, No. 5, 2004, pp. 51–58.

[8] Niewiadomski, A., *Rozmyte metody inteligentnej interpretacji danych*, Zeszyty Naukowe AGH, seria Informatyka, 2006, pp. 543–558.

[9] Niewiadomski, A., Bartyzel, M., and Szczepaniak, P. S., *Linguistic summaries of datasets in automated e-testing algorithm evaluation*, In: Proceedings of XV KKA Conference, Warsaw, Poland, June 27–30, 2005, Vol. 3, 2005, pp. 81–86, (in Polish).

[10] Niewiadomski, A., *Interval-valued data structures and their application to e-learning*, Lecture Notes in Computer Science, Vol. 3381, 2005, pp. 403–407.

[11] Niewiadomski, A. and Rybusiński, B., *Fuzzy Sets-Based Retranslation of Numerical Data in E-learning*, Lecture Notes in Artificial Intelligence, Vol. 3528, 2005, pp. 348–354.

[12] Niewiadomski, A., Jedynak, A., and Grzybowski, R., *Automated evaluation of e-tests*, In: Proceedings of 4th Polish-Ukrainian Conference Environmental Mechanics, Methods of Computer Science and Simulations, June 24–26, 2004, Lviv, Ukraine, 2004, pp. 133–140, (in Polish).

[13] Niewiadomski, A., Rybusiński, B., Sakowski, K., and Grzybowski, R., *An application of multivalued similarity relations in automated evaluation of grammar tests*, In: The Online Academy (Akademia On-Line), edited by J. Mischke, WSHE Press, 2005, pp. 149–154, (in Polish).

[14] Beth, E., *Semantic Entailment and Formal Derivability*, Mededelingen der Kon. Ned. Akad. v. Wet., Vol. 18, No. 13, 1955, pp. 309–342.

[15] Smullyan, R., *First-Order Logic*, Springer, Berlin, 1968.

[16] Indrzejczak, A., *Jaśkowski and Gentzen approaches to Natural Deduction and related systems*, In: The Lvov-Warsaw School and Contemporary Philosophy, edited by K. Kijania-Placek and J. Woleński, Kluver Academic Publishers, 1998, pp. 253–264.

[17] Indrzejczak, A., *Multiple Sequent Calculus for Tense Logics*, In: Abstracts of AiML and ICTL 2000, Leipzig, 2000, pp. 93–104.

[18] Robinson, J. A., *A Machine Oriented Logic based on the Resolution Principle*, Journal of the Assoc. Comput. Mach., Vol. 12, 1965, pp. 23–41.

[19] Wang, H., *Toward mechanical mathematics*, IBM Journal of Research and Development, Vol. 4, No. 1, 1960, pp. 2–22.

[20] Prawitz, D., Prawitz, H., and Voghera, N., *A mechanical proof procedure and its realization in an electronic computer*, Journal of the ACM, Vol. 7, No. 1-2, 1960, pp. 102–128.

[21] Oppacher, F. and Suen, E., *HARP: A tableau-based theorem prover*, Journal of Automated Reasoning, Vol. 4, 1998, pp. 69–100.

[22] Kaufl, T. and Zabel, N., *Cooperation of Decision procedures in a tableau-based theorem prover*, Revue d'Intelligence Artificielle, Vol. 4, No. 3, 1990, pp. 99–126.

[23] Barker-Plummer, D. and Rothenberg, A., *The GAZER Theorem Prover*, Lecture Notes in Computer Science, Vol. 607, 1992, pp. 726–730.

[24] Niewiadomski, A. and Indrzejczak, A., *The Gentzen Sequent Calculus in E-testing. Part II: Algorithms and Implementation*, Journal of Applied Computer Science, Vol. 18, No. 2, 2010, (in print).