Domain Expert Module for Step-By-Step Linear Equation Solving

Marina Issakova
marina.issakova@ut.ee
University of Tartu, Institute of Computer Science
Estonia

Abstract: T-algebra is an intelligent learning environment for step-by-step solving of algebra problems in four areas of school mathematics, including linear equations. The knowledge the system has of its subject domain is one of key places for intelligence in learning systems. T-algebra domain expert module follows the 'glass-box' principle and can produce step-by-step solutions similar to pencil-and-paper solutions. In addition to solving the problems, the domain expert module in T-algebra can also check the student's solution steps and answers, give advice, etc. This paper will describe a domain expert module built in T-algebra intelligent environment and its applications using the domain of linear equations as an example.

1. Introduction

According to the glossaries, a domain expert is a person with special knowledge or skills in a particular area. In the school, the mathematics teacher is a domain expert for mathematics (including the theme of linear equations) teaching using pencil and paper. The teacher explains solution algorithms and their steps to the students. However, there is only one teacher for the whole class and she is not able to give prompt advice or draw attention to errors using traditional instruction technology. The problems often include a great number of details and if the student receives the corrected solution from the teacher only a week after the assignment, he may not remember his thoughts at the moment of making the error or the causes of error. Ideally, we would have one domain expert for each student to correct the solution, explain errors and give advice to the student immediately after a mistake was made. Unfortunately we do not have such schools. This is the reason why the use of computers could be of assistance in this matter.

One possible solution would be the use of computer algebra systems like Maple, Mathematica, etc. However, these programs have not been developed specifically for educational purposes and they use such advanced methods for solving linear equations, for example, that the student can get only an answer from it. Their domain expert cannot explain how this answer is obtained, cannot demonstrate step-by-step solution of the problem and, of course, the student cannot make mistakes and receive feedback.

The other possibility is the use of interactive learning environments. There are different kinds of interactive learning environments available, which allow building step-by-step solutions.

In the first kind of environments, for example Aplusix [10], the students can produce step-by-step solution themselves, because a solution step consists simply of entering the next line. Yet the domain expert module of such programs does not handle the solution algorithms of different types of problems and does not provide a precise diagnosis of the errors made. The current version of the Aplusix program diagnoses only the non-equivalence of the new expression with the previous one. The authors of Aplusix are already working on adding some Artificial Intelligence components to help the students with difficulties [11].

The second kind of learning environments (such as Mathpert [3], EPGY [12], AlgeBrain [1]) is based on the principle that the student selects only a part of the expression and the transformation rule. The transformation itself is made by the computer. In such environments the computer performs more work than the user and the student is not given the possibility to make mistakes. Therefore, their domain expert module can help when the student is stuck (is not able to choose the appropriate rule), but it is not intended to diagnose the students' knowledge gaps.

T-algebra is an environment enabling to solve algebra problems step-by-step in four areas: calculation of the values of numerical expressions; operations with fractions; solving of linear equations, inequalities and linear equation systems; simplification of polynomials. In the design of the T-algebra learning environment, we have been guided by the principle that all the necessary decisions and calculations at each step should be made by the student. The program should contain such domain expert module, which would be able to not only give an answer, but to show a solution path using the designed interface and would be intelligent enough to check the knowledge and skills of the student, understand the student's mistakes and offer feedback. The domain expert module should be capable of generating and explaining problem-solving behaviour, which is comparable with that produced by human experts – teachers.

This paper will describe a domain expert module built in the T-algebra intelligent environment using the domain of linear equations as an example. The second part of this article gives an overview of the T-algebra environment, our step dialogue and domain expert module. The third part describes applications of the domain expert in T-algebra.

2. Domain Expert Module in T-algebra Environment

Problem solving in T-algebra environment takes place step-by-step. To make the diagnosis of mistakes more complete and the program more intelligent, our own rule dialogue was designed ([7], [9]). Each solution step in T-algebra consists of three stages: selection of the transformation rule, marking the parts of expression, entering the result of the operation. The last stage can proceed in three different modes [8]: free input, structured input and partial input. Free input mode is easily comprehensible (it is similar to working on paper). Structured and partial input modes are more specific. The program helps the user in a certain way, either by indicating the structure of the result or even filling out a part of the result. The presented scheme improves the ability of the program to check the student's solutions and respond to the errors made by the student. The program monitors whether the student works according to the algorithm, and supports it with the respective dialogue, diagnoses transformation errors, offers advice and, if necessary, performs the next step by itself.

The problem solution window of T-algebra is shown on Figure 2.1. The main part of the window contains solution steps and a virtual keyboard that can be used for active input. On the right side is the menu of possible actions. The lower part includes instructions for the student in this particular situation.

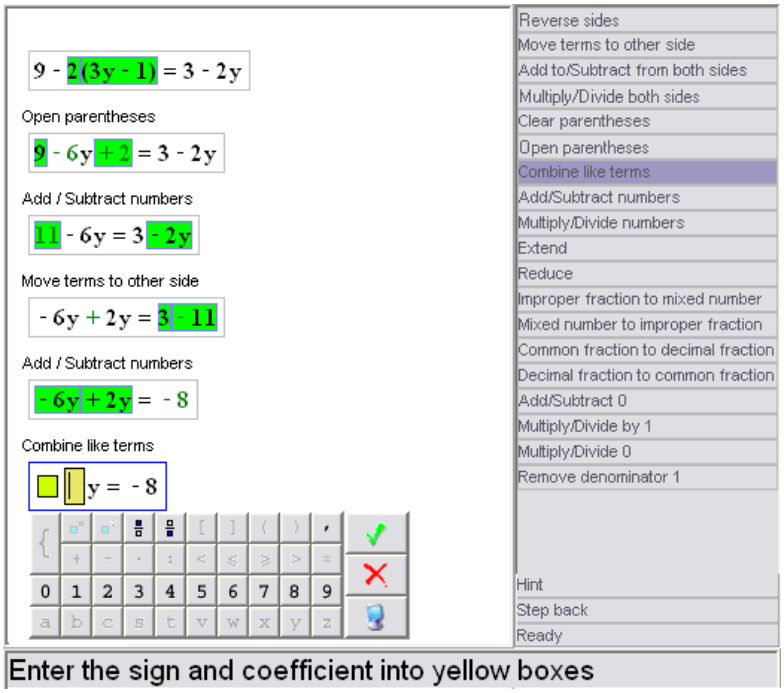

Figure 2.1 The problem-solution window of the T-algebra program

According to Wenger [16] Intelligent Tutoring System consists of 4 major components: Domain knowledge, Student model, Pedagogical knowledge and Interface. Many studies have been conducted to build an intelligent tutoring system based on pre-existing software (using computer algebra system instead of the Domain knowledge) (see [12], [13]). The reason for that is "… recognition that programs like Maple represent massive programming efforts coupled with the feeling that to repeat such an effort would be a waste of resources" ([12] p. 78). However, like Beeson we believe that "… if we start with an educational purpose, and enunciate some simple design principles that more or less obviously follow from that purpose, these principles have ramifications that run through to the computational core of the system, so that it is impossible to achieve ideal results by tacking on some additional "interface" features to a previously existing computation system" ([3] p. 90). That is why we built our own domain expert module, which we believe, is one of the advantages of T-algebra.

According to Anderson [2] the knowledge the system has of its subject domain is one of key places for intelligence in learning systems. The intelligence in a domain is provided by the expert module of the system. A powerful expert module must have an abundance of knowledge. Expert modules range from completely opaque or 'black-box' representations, where only the final results are available, to fully transparent or 'glass-box' (or 'white-box' [4]) ones, where each step of reasoning can be inspected and interpreted. On the one hand these modules serve as the source of knowledge to be presented to the student, which includes generating questions, explanations and responses, and on other hand they provide a standard for evaluating the student's performance by generating comparable solutions to the problems in the same context. The modules must also be able to detect common systematic mistakes and any resulting gaps in the student's knowledge. The expert modules must also be able to generate sensible, and possibly multiple, solution paths to compare intermediate steps and achieve student monitoring.

Problem types, solution algorithms and algorithm steps (rules) are described in our domain expert module. We have implemented in the domain expert module not only the problem types based on a

known solution algorithm like linear equation solving, but also the problems based on single solution algorithm steps, for example, move all variable terms to the left side of equation and all constant terms to the right, divide equation sides by variable coefficient, multiply equation sides by the common denominator of all terms. The T-algebra domain expert module follows the 'glass-box' principle and can produce step-by-step solutions similar to pencil-and-paper ones, not only answers like 'black-box' systems. It solves problems using the designed rule dialogue: it selects a transformation rule corresponding to a certain operation in the algorithm (or some simplification or calculation rule), selects the operands (the whole equation or certain parts of equation) for this rule and replaces them with the result of the operation.

The T-algebra expert module knows the algorithm for every problem type. Algorithms in T-algebra are implemented as an ordered list of rules. Firstly, an algorithm contains rules for simplification, which are not algorithm steps, but which the student may use at any moment on paper, like *Add/Subtract 0*, *Multiply/Divide by 1*, etc. Secondly, the rules for arithmetic operations and manipulation with numbers were added, for example, *Extend*, *Reduce*, *Improper fraction to mixed number*, etc. And finally, an algorithm contains rules, which correspond to pencil-and-paper algorithm steps in such order as they should be applied in pencil-and-paper solution algorithm. The order is very important, because the expert module examines this list of rules from the beginning, finds the first rule that it can apply and applies this rule. Then it examines this list again from the beginning and finds and applies new (or the same) rule. This cycle continues until no more rules can be applied or the expression/equation is in the solved form. This is the way the expert module gets an answer to the problem. In the domain of linear equation, the expert module works according to the following algorithm (list of rules) for linear equation solving:

- rules for simplification (not algorithm steps: *Add/Subtract 0*, *Multiply/Divide by 1*, etc.),
- rules for arithmetic operations and manipulation with numbers (*Extend*, *Reduce*, etc.),
- rule *Multiply/Divide both sides* for multiplying (removing fractions),
- rule *Open parentheses*,
- rule *Add/Subtract numbers*,
- rule *Combine like terms*,
- rule *Move terms to other side*,
- rule *Multiply/Divide both sides* for division.

The expert module checks whether it can apply the rule by trying to find suitable operands for this rule. If it finds operands, then it can apply the rule to these operands. After the operands are found, the expert module calculates some necessary parts of the result depending on the input mode, puts them together with unchanged parts and gets new expression/equation.

Figure 2.1 shows how the expert module would solve the equation according to the algorithm described above. The T-algebra domain expert module is cognitive faithful, i.e., it solves the problem in the same way as the student should. However, this is not the only way to solve this equation. The expert module will accept all other solution paths as well if the student takes them. The expert module can solve the problem from any point of the solution, not only the initial expression. The student can take some steps and the expert module is still able to finish the problem from there.

Figure 2.2 shows the semantic network representation of the domain knowledge in T-algebra.

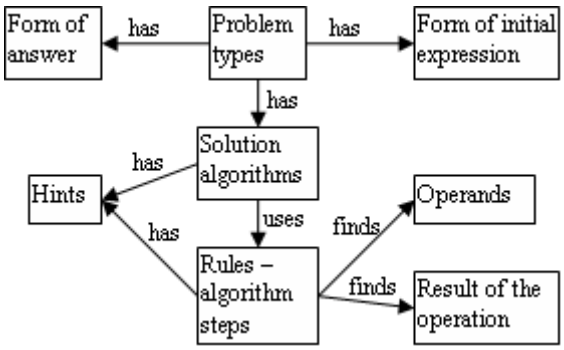

Figure 2.2 Semantic network of domain knowledge

3. Applications of Domain Expert Module

In addition to solving problems, the domain expert module in T-algebra can check the student's solution steps and answers, give advice, etc. This part of the article describes the applications of our domain expert module in T-algebra.

3.1. Giving Advice

There are systems that can help when the user is at a loss. Mathpert is one of such systems. Mathpert has 3 different possibilities to help the student: the buttons *AutoFinish*, *AutoStep* and *Hint*. *AutoFinish* finishes the solution, *AutoStep* generates one step of the solution and *Hint* tells which rule should be applied next. In this environment the student can learn and practice a solution algorithm, but the learning of algorithm steps is passive, because the program never shows to which parts of the expression the rule should be applied (does not show the operands for the operation) and how. We want the student always to participate in the solution process and learn all the stages of each step; therefore, T-algebra does not show the whole step automatically, but only the next stage of the step.

At any stage of the step it is possible to ask the program for help and let the program complete certain stages automatically. Before every step the student can ask the program which rule should be applied at this moment according to the algorithm by pushing the button *Hint* (Fig. 2.1). The expert module will check the current expression and problem type, find which rule should be applied and display appropriate help message (Fig. 3.1). The same button will tell if the problem is solved.

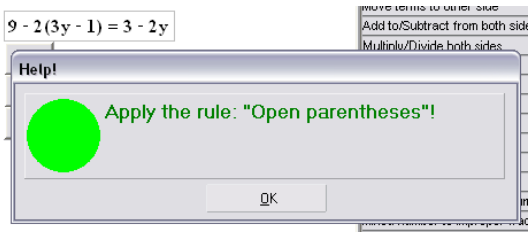

Figure 3.1 Help message for choosing the rule

During the marking of the operands the student can press the special button with computer image and the program will select the appropriate operands itself. The same help button is also available when entering the result – the program will put the right answers into the boxes (Fig. 2.1).

3.2. Checking the Stages of the Step

In existing interactive learning systems the student either can err and the program diagnoses only the non-equivalence (like Aplusix) or the student cannot make mistakes at all. For example, in Mathpert the student even cannot select an unsuitable rule. In this environment the student first selects some part of the expression and only then the program offers "… those operations that make sense for what you've selected" ([15] p. 35).

In T-algebra the student is left the possibility to make mistakes at all three stages of the step. If a mistake can be made, then T-algebra can respond to it as well.

First, the student could err in choosing the rule. If the application of the selected rule is impossible, the program does not immediately inform the student about the error, because the student will not find suitable objects for applying this rule or will make an error by choosing unsuitable objects. This gives the student a chance to correct the error without assistance. If the user cancels the step before confirming the marking of the parts of the expression, the error counter does not increase. If the application of a rule is possible but leads to a completely wrong direction (for example, in the problem on reversing the equation sides the student is choosing the rule for multiplication both sides of equation), then the program will show appropriate error message and will not proceed to the next stage. If the application of this rule is just unreasonable – leads to right answer, but with longer solution path (for example, in the problem on reversing the equation sides the student is choosing the rule for moving terms to other side), the program allows proceeding and leaves a possibility for the teacher to evaluate the solution process.

Secondly, the student can make mistakes in marking the parts of the expression. First the program checks whether a syntactically correct part of the expression has been marked. Second, it is checked whether the marked parts are appropriate for the implementation of the selected operation (for example, the term for combining should be a monomial). Third, the program checks whether the marked parts are compatible with the selected rule (for example, the terms for combining should be similar to each other). Finally the position of operands is checked (for example, the term for moving to other side should not be a member of sum in parentheses (Fig. 3.2)). When wrong parts have been selected, the program does not permit to continue. Some rules (especially in the field of linear equations) are applicable to the whole expression/equation and it is not necessary to mark anything. However, the possibility to mark is preserved. If something is marked, the program checks whether the whole expression/equation is selected.

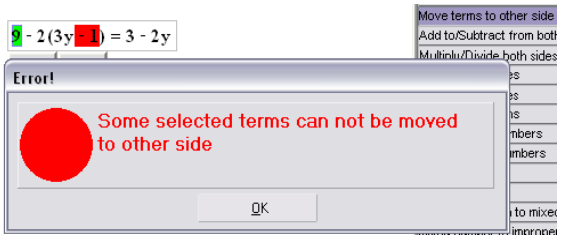

Figure 3.2 Error message displayed when marking the operands

The input stage has the largest selection of potential mistakes, because the student must apply the rule for the marked parts and enter the result. When the user has finished entering, the program first checks the syntactical correctness of the entered part and the correctness of applying the rule (whether the answer is in appropriate form, for example, the result of combining like terms should be a monomial). The program checks whether the entered parts are equivalent to the parts calculated by the expert module. If the expressions are not equivalent, the program checks the correctness of each entered part to produce a more specific diagnosis. Finally the program checks the equivalence of the complete new line with the previous line (in some cases the position of operands causes additional requirements, for example, combining like terms *-3x* and *5x* in equation *2-3x+5x=6* the student should enter *+2x* even if *2x* is right answer for operation).

During all the checking phases at the input stage the program tries to determine whether the student has made a standard error, which occurs often in student solutions (for example, changing all signs when reversing sides is a very common mistake made by Estonian students). If the mistake is in the set of standard mistakes (some studies have been conducted to collect the students' mistakes made on paper [5], [6], [14]), then T-algebra is able to diagnose it and offer an appropriate error message (Fig. 3.3). If the mistake is not in the set of standard mistakes, then T-algebra tells about the non-equivalence of equations.

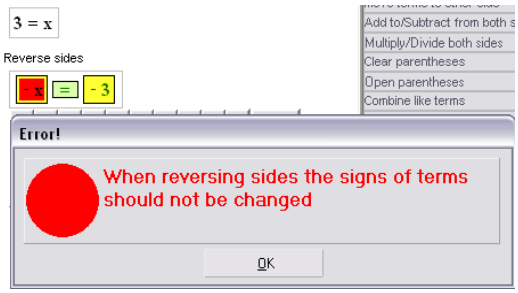

Figure 3.3 Error message displayed when entering the result

3.3. Checking the Completion

The solution algorithm and the form of answer depend on the problem type. The solved form of a linear equation should be *variable* = *number* or *number* = *number*. When student thinks that he has solved the problem he should push the button *Ready* (Fig. 2.1). The program checks whether the answer is in an appropriate form. If the current equation is not in the solved form, then the program tries to determine which steps of the algorithm were not performed and displays the respective error message (Fig. 3.4).

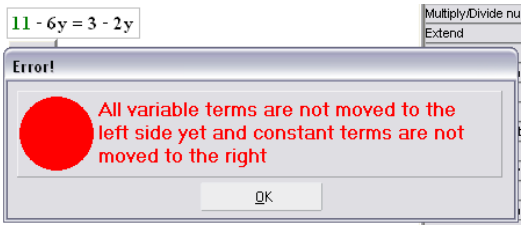

Figure 3.4 Error message displayed when giving an answer

In problems based on single steps the student should not solve the equation to the end but should practice only application of one or two rules. If the student did not recognise the answer and tries to modify the equation further (for example, in the problem on multiplying equation sides by common denominator of all terms the student tries to combine like terms after multiplying), the program will not permit this. Each time when the student tries to choose a new rule the program displays an error message "The problem is solved. Give an answer".

The domain expert module of T-algebra is intelligent enough to avoid loops in checking the completion of the solution. It understands that the equation $x = \frac{5}{6}$ should be multiplied if the problem type is 'multiply equation sides by common denominator of all terms', because the answer should be without fractions. The expert module will also suggest to multiply the equation $\frac{5}{6} = x$ in the same problem type. However, if the problem type is 'solve linear equation', it will never suggest multiplying these equations even though the first step in linear equation solving according to school algorithm is to use the multiplication property to remove fractions if present. If it would suggest multiplying to remove fractions according to the algorithm, then dividing to isolate variable and then again multiplying, etc., it would enter in a loop. The program first checks the form of an answer and only then seeks the rule for application. The equation $x = \frac{5}{6}$ is already an answer; the sides should be reversed to get an answer from the equation $\frac{5}{6} = x$.

3.4. Checking the Answer

If the student pushed the button *Ready* and the program checked that the equation is in appropriate form, then it means that this is the correct solved form, because it is impossible to produce incorrect solution steps in T-algebra. If an error message was displayed at any checking stage during solving the problems, the student had to correct the error in order to proceed to the next stage.

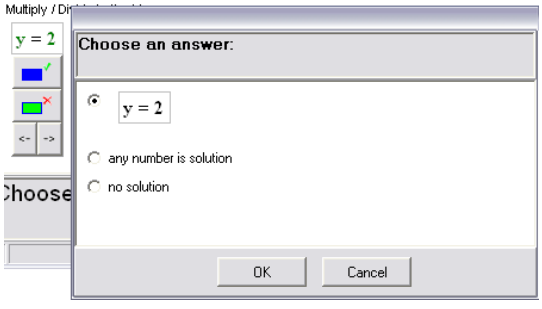

Figure 3.5 Giving an answer

However, in some problem types the student should specify some additional properties of the answer. When solving a linear equation, the solved form of the equation can be *variable* = *number* or *number* = *number*. In this problem type after pushing the button *Ready*, a separate window with three possibilities (Fig. 3.5) will appear where the student should select one of the possible answers (whether the one number is solution, there is no solution, or any number is solution). After the answer is confirmed, the program will check whether the appropriate choice was made.

3.5. Checking the Initial Expression

During problem composition the problem type must be selected first and then the initial expression must be entered. First, the initial expression should be syntactically correct. Second, the initial expression for every problem type should be in a known form, e.g., in the linear equation theme it must be a linear equation. Finally, most problem types have some restrictions. For example, the initial expression of problem type 'multiply equation sides by common denominator of all terms' must contain at least one fraction to be removed, the initial expression of combining like terms must contain like terms to combine, etc. The domain expert module first checks whether the expression is in appropriate form (is suitable for this problem type), then solves the problem and displays the solution path and answer to the compiler.

4. Conclusion

Everybody agrees that computer can help in teaching algebra, but we still do not have system with suitable domain expert module. The domain expert module of computer algebra systems can give only answers for problems, which is not enough. The domain expert module of some interactive learning systems can build step-by-step solutions and give advice, but is not intended to diagnose the students' knowledge gaps (like MathXpert, AlgeBrain). The domain expert module of other systems can diagnose mistakes, but does not provide a precise diagnosis of the errors made (like Aplusix). We have seen that T-algebra domain expert module can not only give answers, but produce step-by-step solutions similar to pencil-and-paper solutions. We have succeeded to create such domain expert module in T-algebra that is intelligent enough to check the knowledge and skills of the student (the student's solution steps and answers), understand the student's mistakes, offer feedback and give advice. We built domain expert module, which we believe is one of the advantages of T-algebra.

Acknowledgements The author was supported by Estonian Doctoral School in Information and Communication Technologies.

References

[1] Alpert, S.R., Singley, M.K., and Fairweather, P.G. (1999). Deploying Intelligent Tutors on the Web: An Architecture and an Example. *International Journal of Artificial Intelligence in Education* 10(2) 183-197.

[2] Anderson, J.R. (1988). The expert module. In: Polson, M., Richardson, J. (eds.): *Handbook of Intelligent Training Systems*. Hillsdale, NJ: Erlbaum 21-53.

[3] Beeson, M. (1998). Design Principles of Mathpert: Software to support education in algebra and calculus. In: Kajler, N. (ed.): *Computer-Human Interaction in Symbolic Computation*. Berlin Heidelberg New York: Springer-Verlag 89-115.

[4] Buchberger, B. (1990). Should Students Learn Integration Rules? *ACM SIGSAM Bulletin*, Vol.24/1 10-17.

[5] Hall, R. (2002). An Analysis of Errors Made in the Solution of Simple Linear Equations. *Philosophy of Mathematics Education Journal* 15.

[6] Issakova, M. (2005). Possible Mistakes during Linear Equation Solving on Paper and In T-algebra Environment. *Proceedings of the 7th International Conference on Technology in Mathematics Teaching*, Vol. 1. Bristol, UK 250-258.

[7] Issakova, M. and Lepp, D. (2004). Rule dialogue in problem solving environment T-algebra. *Proceedings TIME-2004: Montreal International Symposium on Technology and its Integration into Mathematics Education*. Montreal, Canada.
[8] Issakova, M., Lepp, D., and Prank, R. (2005). Input Design in Interactive Learning Environment T-algebra. *Proceedings ICALT-2005: The 5th IEEE International Conference on Advanced Learning Technologies*. Kaohsiung, Taiwan 489-491.
[9] Lepp, D. (2005). Extended Solution Step Dialogue In Problem Solving Environment T-algebra. *Proceedings of the 7th International Conference on Technology in Mathematics Teaching*, Vol. 1. Bristol, UK 267-274.
[10] Nicaud, J., Bouhineau, D., and Chaachoua, H. (2004). Mixing microworld and cas features in building computer systems that help students learn algebra. *International Journal of Computers for Mathematical Learning* 5(2) 169-211.
[11] Nicaud, J., Chaachoua, H., Bittar, M., and Bouhineau, D. (2005). Student's modelling with a lattice of conceptions in the domain of linear equations and inequations. *Proceedings of AIED 05 Workshop 1*. Amsterdam, the Netherlands 81-88.
[12] Ravaglia, R., Alper, T., Rozenfeld, M., and Suppes, P. (1998). Successful pedagogical applications of symbolic computation. In: Kajler, N. (ed.): *Computer-Human Interaction in Symbolic Computation*. Berlin Heidelberg New York: Springer-Verlag 61-88.
[13] Sangwin, C.J. (2005). Making Mathematical Distinctions In CAA With Computer Algebra. *Proceedings of the 7th International Conference on Technology in Mathematics Teaching*, Vol. 1. Bristol, UK 292-299.
[14] Sleeman, D. (1984). An Attempt to Understand Students' Understanding of Basic Algebra. *Cognitive Science* 8 413-437.
[15] Walden, J. (1997). *Mathpert Calculus Assistant: User's Guide*. Santa Clara, USA: Mathpert Systems, an Imprint of Recognix.
[16] Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Los Altos, CA: Morgan Kaufmann Publishers.