

**SOLVING OF LINEAR EQUATIONS,
LINEAR INEQUALITIES AND SYSTEMS
OF LINEAR EQUATIONS IN INTERACTIVE
LEARNING ENVIRONMENT**

MARINA ISSAKOVA



TARTU UNIVERSITY
PRESS

Faculty of Mathematics and Computer Science, University of Tartu, Estonia

Dissertation is accepted for the commencement of the degree of Doctor of Philosophy (PhD) on June 15, 2007, by the Council of the Faculty of Mathematics and Computer Science, University of Tartu.

Supervisor:

Cand. Sc., Associate Professor Rein Prank
University of Tartu
Tartu, Estonia

Opponents:

PhD, Professor Jean-Francois Nicaud
Joseph Fourier University
Grenoble, France

Cand. Sc., Associate Professor Jaak Henno
Tallinn University of Technology
Tallinn, Estonia

Commencement will take place on September 6, 2007.

ISSN 1024-4212
ISBN 978-9949-11-680-5 (trükis)
ISBN 978-9949-11-681-2 (PDF)

Autoriõigus Marina Issakova, 2007

Tartu Ülikooli Kirjastus
www.tyk.ee
Tellimus nr. 298

CONTENTS

List of original publications	8
1 Introduction	10
1.1 Motivation	10
1.2 Related works	11
1.3 Problem statement	16
1.4 Contribution of the thesis	17
1.5 Structure of the thesis	18
2 T-algebra interactive learning environment	19
2.1 Expressions in T-algebra	20
2.2 Description of the problem-solution window	21
2.3 General dialogue scheme in T-algebra	22
2.3.1 Selection of the rule and marking the parts of expression	24
2.3.2 Entering the result of the application of the rule.....	25
2.3.2.1 Three input modes	26
2.3.2.2 Additional input	28
2.4 Domain expert module in T-algebra environment.....	31
2.4.1 Solution engine.....	33
2.4.2 Applications of domain expert module	33
2.4.2.1 Giving advice.....	33
2.4.2.2 Checking the stages of the step.....	34
2.4.2.3 Checking the completion	37
2.4.2.4 Checking the answer.....	38
2.4.2.5 Checking the initial expression.....	39
2.4.2.6 Checking the equivalence of two expressions.....	39
2.5 Student statistics.....	39
2.6 The program for teachers	42
2.7 Implementation.....	43
3 Problems and algorithms in the domain of linear equations, linear inequalities and systems of linear equations in school textbooks and in T-algebra	46
3.1 Problems and algorithms in school textbooks	46
3.1.1 Linear equations.....	46
3.1.2 Linear inequalities.....	48
3.1.3 Systems of linear equations	50
3.2 Designed rules in T-algebra	51
3.2.1 Rule <i>Multiply/Divide both sides</i>	55
3.2.2 Rule <i>Move terms to other side</i>	61

3.2.3	Rule <i>Reverse sides</i>	63
3.2.4	Rule <i>Add to/Subtract from both sides</i>	65
3.2.5	Rule <i>Express variable</i>	66
3.2.6	Rule <i>Substitute variable</i>	69
3.2.7	Rule <i>Add equations</i>	72
3.2.8	Rule <i>Multiply fraction with variable by number</i>	74
3.2.9	Rule <i>Open parentheses</i>	77
3.3	Designed problem types in T-algebra	79
3.3.1	Problem type <i>Move terms to correct side of equation and combine</i>	83
3.3.2	Problem type <i>Solve linear equation</i>	84
3.3.3	Problem type <i>Check the solution of inequality</i>	85
3.3.4	Problem type <i>Solve by elimination using addition</i>	87
4	Conducted experiments	89
4.1	First experiment	89
4.2	Second experiment	92
4.3	Third experiment	94
4.3.1	Comparison of student mistakes made during solving on paper and in T-algebra	97
4.3.1.1	Mistakes made during multiplying both sides of equation	98
4.3.1.2	Mistakes made during dividing both sides of equation	102
4.3.1.3	Mistakes made during moving terms to the other side of equation	103
4.3.1.4	Mistakes made during reversing the equation sides	105
4.3.1.5	Mistakes made during combining like terms and during adding/subtracting numbers	106
4.3.1.6	Mistakes made during opening the parentheses	107
4.3.2	Conclusions	107
4.4	Fourth experiment	108
4.4.1	Results of experiment	109
4.4.2	Conclusions	111
4.5	Fifth experiment	112
4.5.1	Results of experiment	114
4.5.2	Conclusions	118
	Conclusions	120
	References	122
	Summary in Estonian	127
	Acknowledgements	129

Appendix A	130
Backus-Naur Form full description of expressions.....	130
Appendix B	132
Quick start to T-algebra student's program	132
Appendix C	138
Full list of designed problem types	138
Appendix D	163
User questionnaire.....	163
Appendix E	164
Problem file.....	164
Curriculum Vitae	166

LIST OF ORIGINAL PUBLICATIONS

1. Issakova, M. and Lepp, D. (2004). Rule dialogue in problem solving environment T-algebra. In *Proceedings TIME–2004: Montreal International Symposium on Technology and its Integration into Mathematics Education*, 16 p., Montreal, Canada.
2. Issakova, M., Lepp, D. and Prank, R. (2005). Input Design in Interactive Learning Environment T-algebra. In *Proceedings ICALT–2005: The 5th IEEE International Conference on Advanced Learning Technologies*, pp. 489–491, Kaohsiung, Taiwan.
3. Issakova, M. (2005). Possible Mistakes During Linear Equation Solving On Paper And In T-algebra Environment. In *Proceedings of the 7th International Conference on Technology in Mathematics Teaching (ICTMT7)*, volume 1, pp. 250–258, Bristol, UK.
4. Prank, R., Issakova, M., Lepp, D., Vaiksaar, V. and Tõnisson, E. (2006). Problem solving environment T-algebra. In *Proceedings of 7th International Conference Teaching Mathematics: Retrospective and Perspectives*, pp. 190–197, Tartu, Estonia.
5. Prank, R., Issakova, M., Lepp, D. and Vaiksaar, V. (2006). Designing Next-Generation Training and Testing Environment for Expression Manipulation. In *International Conference on Computational Science (ICCS 2006)*, Part I, LNCS 3991, pp. 928–931, Springer-Verlag.
6. Issakova, M. (2006). Learning Linear Equation Solving Algorithm and Its Steps in Intelligent Learning Environment. In *ITS 2006 Proceedings*, LNCS 4053, pp. 725–727, Springer-Verlag.
7. Issakova, M. (2006). Intelligent Environment for Learning Linear Equation Solving Algorithm and Its Steps. In *Proceedings of the Student Track ITS 2006*, pp. 8–17, Jhongli, Taiwan.
8. Issakova, M., Lepp, D. and Prank, R. (2006). T-algebra: Adding Input Stage To Rule-Based Interface For Expression Manipulation. *International Journal for Technology in Mathematics Education*, 13(2): 89–96.
9. Issakova, M. (2006). Comparison of student errors made during linear equation solving on paper and in interactive learning environment. In *Proceedings DES–TIME–2006: Dresden International Symposium on Technology and its Integration into Mathematics Education 2006*, 20 p., Dresden, Germany.

10. Issakova, M. (2006). Domain Expert Module for Step-By-Step Linear Equation Solving. In *Proceedings of The 11th Asian Technology Conference in Mathematics (ATCM 2006)*, pp. 193–202, Hong Kong, China.
11. Issakova, M. (2007). Do First Year Students Know how to Solve Simple Linear Equations? An Experiment with T-algebra. In *Proceedings of the 8th International Conference on Technology in Mathematics Teaching (ICTMT8)*, 6p., Hradec Králové, Czech Republic.
12. Prank, R., Issakova, M., Lepp, D., Tõnisson, E. and Vaiksaar, V. (2007). Integrating rule-based and input-based approaches for better error diagnosis in expression manipulation tasks. In *Symbolic Computation and Education*, World Scientific Publishing Co., (to appear).

1 INTRODUCTION

This thesis belongs to the field of interactive learning environments for step-by-step solving of expression manipulation problems, particularly for solving linear equations, linear inequalities and systems of linear equations. The main contribution of the thesis is design, implementation and testing of the environment with a novel step dialogue for proper learning and diagnosis of knowledge gaps in solving linear equations, linear inequalities and systems of linear equations.

1.1 Motivation

Expression manipulation (incl. solving of linear equations, inequalities and systems of linear equations, simplification of polynomials, etc.) is one of the key skills needed for solving problems in practically all fields of mathematics. The students solve in the school hundreds of technical exercises with fractions, monomials, polynomials, equations, inequalities and systems of equations. However, expression manipulation is also an element of the mathematics curriculum that poses difficulties to many students and is relatively labour-intensive for the teachers while the results of learning in this area are often unsatisfactory.

One of the reasons for poor performance is repetition of incorrect solution attempts without getting feedback. The difficulties experienced by the students while solving the problems can be quite variable and require a thorough thought effort from the teacher in order to understand all details. When using traditional instruction technology, the teacher is not able to give prompt advice or draw attention to errors in time. Thus, the mistakes are repeated many times and can become habitual. The exercises often include a great number of details and if the student receives the corrected solution from the teacher only a week after the assignment, she/he may not remember her/his thoughts at the moment of making the error or the causes of error. Sometimes even a principal error can be regarded as an error caused by oversight and this can prevent the student from analysis of its real causes. For the teacher, checking of assignments in this field is very labour-intensive and she/he may not be able to discover all errors made in written assignments. The need for quick analysis of large volumes of information indicates that the training and testing of expression manipulation skills could be improved by using computerized environments.

The existent environments do not entirely meet all the required principles, as we will see in the next section. Some environments can be used only for training (do not allow making common mistakes) and others mainly for testing (do not provide help and precise diagnosis), but there is no single program that would be suitable for all purposes.

1.2 Related works

In this section I review the following computerized environments for expression manipulation:

- computer algebra systems;
- systems based on computer algebra systems;
- interactive learning environments.

In these environments I observe the following points:

- what a student does in this environment and what the environment does;
- what knowledge and skills should a student have to reach the solution or answer in this environment;
- what diagnosis and feedback the environment provides, if the student makes a mistake.

In many countries, the schools use *computer algebra systems* (DERIVE (Kutzler, 1996; Derive by Texas Instruments), Maple (Char et al., 1986; Maple by Waterloo Maple Inc.), Mathematica (Mathematica by Wolfram Research), etc.) to work with algebraic problems. However, these programs have not been developed specifically for educational purposes. Generally, they use such advanced methods for expression manipulation that the student can get only an answer from them, and their domain expert cannot explain how this answer was obtained and cannot demonstrate step-by-step solution of the problem. Even systems like WIRIS (Xambo et al., 2002) that are designed and advertised specifically for using in schools do not have sufficiently detailed commands for construction of stepwise solutions. Computer algebra systems (CAS) are not designed for the cases where the student solves problems, makes mistakes, requires feedback and advice, etc.

A further step for helping students learn expression manipulation was made in *systems, which are based on CAS*. Some of these systems allow, but do not require entering stepwise solutions (for example, ActiveMath (ActiveMath by ActiveMath Team; Melis et al., 2001; Büdenbender et al., 2002)), others ask to enter only the answer (AiM (AiM by AiM Team; Sangwin, 2004), STACK (STACK by Chris Sangwin; Sangwin, forthcoming)). As these systems are based on CAS, they accept input in the language of the underlying system, usually linear input. These systems use CAS for controlling student's input and providing feedback depending on the answer. These systems usually check only whether the input is correct/incorrect/impossible and whether the problem is solved (is sufficiently simplified or completely factorized). In the case of an error, such systems do not provide explicit feedback and cannot highlight the erroneous part. The help provided in these systems is restricted to showing only the answer or solution predefined by the author of the problem.

The most suitable systems for expression manipulation are interactive learning environments. There are different kinds of interactive learning environments available with various dialogues, which allow building step-by-step solutions. We can classify them as follows, depending on the type of dialogue they use:

- rule-based or command-based environments;
- input-based environments.

Rule-based environments are based on the principle that the student selects the transformation rule and in some cases a part of the expression; the transformation itself is made by the computer. The order of selection is different in different systems: in some systems the student should first select some part of expression and the system offers suitable rules, in others the student should first select the rule. In many cases it is even sufficient to select the whole expression for the operation as the program itself selects the required operands. The program executes all operations as a black box. In such environments, the student learns and practices the solution algorithm, but the learning of performing algorithm steps (details of operations) is passive, because the computer performs more work than the user. In addition, the student is not given the possibility to make certain mistakes; many typical mistakes are simply impossible. The only mistakes possible are selection of unsuitable rule and selection of unsuitable (or incorrect) part. 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 gaps in the student's knowledge and skills.

Rule-based environments have been designed since the eighties of the last century. The earlier examples of rule-based environments are EXPRESSIONS (Thompson and Thompson, 1987), ALGEBRALAND (Brown, 1985), DISSOLVE (Oliver and Zukerman, 1990), Mathpert (Beeson, 1990), Aplusix (Nicaud et al., 1999). EXPRESSIONS presents expressions in two formats: in the usual form and as a tree. In order to perform a step, the student should select the rule (the button) and then an operation from the tree, which defines the expression to be transformed. The program changes the expression and the tree accordingly. In ALGEBRALAND the student just selects the operation and operands. Earlier versions of Aplusix environment allowed practicing factoring polynomials without making calculations, i.e., choosing action, selecting expression and in some cases entering additional information. Most of these systems are no longer in development with the exception of Aplusix, which is now an input-based environment (Aplusix by IMAG-Leibniz laboratory), and Mathpert, which was developed into MathXpert (Beeson, 2002; MathXpert by Help With Math). In the MathXpert system, the student should first select some expression and after that the program displays the list of rules, which can be applied to the selected expression (or some part of it). One of the main keynotes

of this system is that “It is not possible to make a mistake” (Beeson, 2002). An analogous idea is used in the comparatively new system L’Algebrista (Cerulli and Mariotti, 2002). In this system, the student chooses a part of the expression and the operation – the button with axiom, but “L’Algebrista will not carry out a transformation that is invalid” (Stacey et al., 2004). For example, the student will not succeed if she/he wants to select $a+3$ in the expression $2*a+3$; the system will automatically extend the selection to the whole expression.

There are also some other systems available now, which are working in terms of rules. In the web-based intelligent tutoring system for solving equations AlgeBrain (Alpert et al., 1999), the student should select operands (term can be selected by clicking on its primary operator) and operation; the system does not allow selecting syntactically incorrect parts. The system proposes hints and animated feedback. A similar scheme is used in the Education Program for Gifted Youth (EPGY) (Ravaglia et al., 1998), but in this system the student should sometimes highlight or input some additional information. EPGY uses the kernel of Maple, but has its own semantic machinery. The current version of Cognitive Tutor Algebra I (Cognitive Tutor by Carnegie Learning Inc.) includes an equation-solving system, where the student has to choose an operation. For some operations, the system asks to enter some additional information (for example, number to be added to both sides of equation), for others, the system asks whether the operation should be carried out to the left side, to the right side or to both sides of equation. The solving process in E-tutor: An Equation Solving Tutor (Razzaq and Heffernan, 2004) is analogous, but this small system offers dialogue-based feedback instead of hints. The tutorial dialogues in E-tutor draw upon Ms. Lindquist (Heffernan and Koedinger, 2000; Ms. Lindquist by Neil Heffernan), an algebra tutor for word problems.

Some commercial products also utilize rule-based interface, for example LiveMath (LiveMath by MathMonkeys) and The Learning Equation (The Learning Equation (TLE) by ITP Nelson) (particularly in the incorporated program Algebra Tiles). Even though LiveMath is described as being a CAS, this system is rule-based in its essence, because it does not enable getting the answer in one step; for solving the student should select a part of expression and the operation from the menu.

Now let us describe the second kind of interactive learning environments. *Input-based systems* use paper-and-pencil-like dialogue design where a transformation step consists mainly of entering the next line. The student can work according to the algorithm and perform the algorithm steps by himself, but the student has the possibility to perform whatever steps and as much as she/he wants in one step. The domain expert module of such programs usually does not handle the solution algorithms of different types of problems and does not check whether the student works according to the algorithm. The student is

given the possibility to make arbitrary mistakes; input is restricted only by the syntax of expressions. The computer is now in the same situation as the teacher who should check a solution on the paper: there is no explicit information on the student's decisions about operations and operands. Without knowing what operation was applied to what part(s) of previous expression and without restrictions on the number of operations applied during one step, it is very hard to diagnose errors more precisely than "expressions are not equivalent". The domain expert module of input-based environments usually does not provide a precise diagnosis of the errors made.

Early input-based systems were created already in the seventies of the last century: BUGGY/DEBUGGY system (Brown and Burton, 1978; Burton, 1982), LMS (Sleeman and Smith, 1981), EMMA (Quigley, 1989), Algebra tutor (Anderson et al., 1990). BUGGY was the first naive diagnostic system based on "the Buggy model" proposed by Brown and Burton (Brown and Burton, 1978), where student's errors are seen as symptoms of a "bug", a discrete modification to the correct skills. The system tried to find one bug that could explain the student's answers. DEBUGGY, a development of BUGGY, took into account that more than one bug can cause the student's errors during one step. The Leeds Modeling System, LMS, used rules and associated mal-rules for modeling students as they learn to solve linear equations in one variable. The student could give an answer in one step, but could simplify the equation. The system did not indicate whether or not the answer was correct. The tutor for solving linear equations EMMA used an engine similar to LMS (rules and mal-rules), but the system checked each step for correctness. If the step corresponded to a rule or mal-rule, then student was informed about that. The student could also get different kinds of help (a generated solution; a list of rules that are applicable to the current state; an explanation of how the last step was obtained). The passive tutor created by McArthur (McArthur et al., 1987) used student input to create a reasoning tree. The student could use the menu items *Answer Ok?* and *Step Ok?* for getting feedback. Using *Step Ok?* the student got a hint whether the step is acceptable, mathematically invalid or inappropriate. The student could also look at the step generated by the system, as well as how this step was elaborated. Algebra tutor, an early version of Cognitive tutor Algebra 1 (Anderson et al., 1990), also used purely entering of the result and the program tried to figure out what step was performed and to give appropriate feedback. But the authors of cognitive algebra tutors found that "The problem was that the students' error might well have occurred at some intermediate step that the students were no longer fixated upon. It was very difficult to communicate to the student what the problem was." (Anderson et al., 1990, p. 42). These systems are no longer in development.

The current version of the Aplux program (Nicaud et al., 2004) is input-based. The program copies the content of previous line (expression, equation or system of equations) to the next line and the student should edit it into the result

of the step. The current version of Aplusix diagnoses only the non-equivalence of the new expression with the previous one: the program displays between two lines the indicator of equivalence, giving the student feedback about correctness of the step. The authors of Aplusix are developing the program further. They are building a library of correct and incorrect rules, which can describe how one expression was transformed by the student to the next expression, and adding student modeling using conceptions (the models will be provided only for teachers, not students) (Nicaud et al., 2006). They are also planning to provide good feedback for the student from the calculated conceptions.

Nowadays there are available some systems, which use pure input for solving. In Math-Teacher (Math-Teacher by MATH-KAL) the expressions should be entered in the old-fashioned linear form. The program provides feedback: correct (colored green), incorrect (colored red) or syntax error. Some help (like hint on the last answer line) is provided. Another example of an input-based system is Treefrog (Strickland and Al-Jumeily, 1999). In case of incorrect input, this system colors the input red and provides a hint, indicating what the student should do (for example, Bring the x s together), even if the input is absolutely nonsensical. The e-learning tool for solving systems of linear equations The Equation Solver (Passier and Jeuring, 2006) uses a set of rewrite rules for providing feedback about syntactic errors, semantic errors and about progression (for example, how many variables have been solved). The system of equations is entered into ordinary textbox, i.e., linearly. The authors of this system have made a major restriction: they assume that the student performs one step in a step (applies only one rewrite rule per submitted system of equations).

Some attempts were made to supplement selection by rules with entering the result. The intermediate version of Cognitive Tutor: Algebra (Anderson et al., 1995) was a system where the student could decompose a result calculation into substeps recursively until primitive steps were reached. At each substep, the student had to choose the operation that should be performed on equation, enter the arguments to pass to this operation, and enter the result. The tutor embedded boxes on top of boxes to indicate the levels of embedded goals. But after evaluation the authors found that the tutor did not give positive results and "... the major reason for the lack of effect was that there was a large difference between the tutor interface and the interface used in class (i.e., paper and pencil). It was just not obvious how to map the boxed representation of algorithmic decompositions to the linear line-by-line transformations..." (Anderson et al., 1995, p. 183).

The author of the thesis tried in real life the following of the abovementioned systems: DERIVE, Maple, Mathematica, WIRIS, ActiveMath, AiM, STACK, MathXpert, the current version of Aplusix, EPGY, E-tutor: An equation solving tutor, Ms. Lindquist, LiveMath, TLE, Math-Teacher, Treefrog. Other systems

are described on the basis of referred articles and the author of the thesis cannot take responsibility for the correspondence of these articles to the reality.

Work related in some aspects to the topic of the thesis has been performed at the University of Tartu as well, where interactive learning environments with different step dialogues were developed for university students. In 1988–91, a program package for exercises in mathematical logic was developed under the instruction of the supervisor of the author (Prank, 1991). One of the programs was an interactive environment for stepwise solution of formula manipulation exercises in propositional logic. The first version of this program utilized a pure input interface. The student typed on the next line a new formula (having some copy-paste possibilities). The program checked the syntax, equivalence with the previous line and whether the target form of the expression was reached. It was noticed that the errors of misunderstanding the order of operations were most difficult. After the message “not equivalent” the students corrected the mistake easily if they had mistaken some conversion rule. But they did not understand the message when they had converted some substring that was not a proper subformula (for example, expressed in $X \supset Y \& Z$ the substring $X \supset Y$ through other operations). The program was unable to diagnose such mistakes without explicit information about the object of conversion. In the second version (Prank and Viira, 1991) the step dialogue was extended and a rule-based part was added. The student had to mark some subformula and then the step was performed depending on the working mode. The first working mode remained the same as in the first version of the program – input. The second mode consisted of the selection of a conversion rule from the menu. As a result, the program was able to verify separately the selection of operand and the performed conversion. This addition of a marking phase gave a level of feedback that was sufficient for that group of users (second-year students) and there was no need to make it more precise.

1.3 Problem statement

The general problem this thesis aims to solve is to design, implement and test an environment of new kind for proper learning of linear equations, linear inequalities and systems of linear equations as well as for assessment and diagnosis of gaps in the knowledge and skills. This environment should

- enable to solve linear equations, linear inequalities and systems of linear equations step-by-step and line-by-line as on paper;
- allow the student to make all the necessary decisions and calculations at each solution step;
- leave an opportunity for the student to make the same mistakes as on paper;
- give the possibility to learn both the algorithms and their steps in details;

- contain such dialogue that allows the program to understand all decisions made by students (chosen operation, selected operands, entered result);
- be intelligent enough to check the knowledge and skills of the student, understand the mistakes, offer feedback and advice;
- 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.

1.4 Contribution of the thesis

This thesis is a part of the larger project, the result of which is an interactive learning environment for working with numerical expressions, fractions, linear equations, inequalities, linear equation systems and polynomials. The thesis presents the interactive learning environment T-algebra with novel design, in which creation the author of the thesis participated. The design is novel, because it combines two known approaches: rule-based and input-based environments. The result of combination is named Action-Object-Input (A-O-I) scheme. It is hard to specify the particular contributions of different members of the T-algebra team to developing the general ideas of the A-O-I scheme. The main contribution of the author of the thesis to this project is design and implementation (programming) of and experiments with the rule dialogues enabling diagnosis of mistakes, and problem types with solution algorithms for the domain of solving linear equations, linear inequalities and systems of linear equations.

This thesis presents the work contributed by the author to the following design, implementation and experimenting efforts:

- division of problems solved at school in the domain of linear equations, linear inequalities and systems of linear equations into straitened computerized problem types;
- specification of initial and result conditions for checking by the program for each problem type;
- construction and programming of algorithms of automatic solving needed for generation of sample solutions and step advices for each problem type (algorithm creates a solution, which consists of application of rules);
- creation of list of rules, which are essential for solving each problem type, in conformity with the algorithms presented in school textbooks and skills learned beforehand;
- supplementary investigation for clearing up typical mistakes made by students on paper during solving of problems from the chosen domain;
- design and programming of dialogues, additional stages, initial and result conditions, diagnosis of mistakes for each rule from the domain (all rules are realized according to the A-O-I scheme);

- experimental validation of created dialogues with students and teachers;
- investigation of mistakes made by school and university students during linear equation solving in T-algebra and comparison with mistakes made on paper;
- evaluation of the created interactive learning environment (specifically the part dealing with the solving of linear equations).

1.5 Structure of the thesis

This thesis is based on the papers by the author presented in the List of original publications.

Chapter 2 (T-algebra interactive learning environment) thoroughly describes the design and general dialogue scheme of T-algebra. The first part introduces expressions allowed in the program. The second part of Chapter 2 depicts the problem solution window of the student's program. The third part presents the design of step dialogue. The fourth part of this chapter gives an overview of the domain expert module built in T-algebra using the domain of linear equations as an example and describes applications of the domain expert in T-algebra. The fifth part describes different statistics calculated and saved by T-algebra. The sixth part of this chapter gives brief introduction to the teacher's program. The last part describes implementation of T-algebra.

Chapter 3 (Problems and algorithms in the domain of linear equations, linear inequalities and systems of linear equations in school textbooks and in T-algebra) describes the domain of linear equations, linear inequalities and systems of linear equations. First, exploration of mathematics school textbooks is presented (definitions, algorithms, problem types). The second part of this chapter describes the rules designed for solving linear equations, inequalities and systems of linear equations in T-algebra. The last part represents composed problem types and their solving algorithms in the chosen domain.

Chapter 4 (Conducted experiments) describes five different experiments conducted by the author of the thesis to validate user interface, to evaluate created interactive learning environment (the part of solving linear equations) and to investigate mistakes made by school and university students during linear equation solving in T-algebra and to compare them with mistakes made on paper.

The thesis also contains five Appendices. Appendix A presents Backus-Naur Form full description of expressions in T-algebra. Appendix B gives a brief introduction to the use of T-algebra (quick start). Appendix C describes all problem types realized in the domain of linear equations, linear inequalities and system of linear equations. Appendix D represents a user questionnaire filled out by students during one of the experiments. Appendix E lists the problems used for practice in one of the experiments.

2 T-ALGEBRA INTERACTIVE LEARNING ENVIRONMENT

The goal of the thesis is to create an interactive learning environment for solving linear equations, linear inequalities and systems of linear equations, which would be mathematically and didactically adequate and interesting for students. This goal was realized as a part of T-algebra interactive learning environment, which enables step-by-step solving of algebra problems in four areas of school mathematics:

- calculation of the values of numerical expressions;
- operations with fractions;
- solving of linear equations, inequalities and linear equation systems (my contribution to this environment);
- simplification of polynomials.

T-algebra is developed not only for schools, but also as research tool for compilation of research material. The environment should enable investigation of the following main aspects: speed of work with computer in comparison with work on paper, and mistakes made by the students when creating solutions in the interactive learning environment in comparison with mistakes made on paper.

The environment is being developed from 2004 by the Master's and Doctoral students of the Institute of Computer Science at the University of Tartu (Marina Issakova – solving of linear equations, inequalities and linear equation systems; Dmitri Lepp – simplification of polynomials; Vahur Vaiksaar – operations with fractions and calculation of the values of numerical expressions) and under the supervision of their instructors (Rein Prank – project manager, Eno Tõnisson). The consultants for the content of the program are mathematics teachers (great contribution was made by Mart and Maire Oja) and the authors of textbooks for schools (Tiit Lepmann, Anu Palu). This version is developed as a project financed by the 'Tiger Leap' computerization programme for Estonian schools (Tiger Leap Foundation website).

T-algebra consists of two programs, one is for students and the other is for the teachers. In this section I will thoroughly describe the design and general dialogue scheme of the students' program and give a brief introduction to the teachers' program on the basis of published articles (Issakova and Lepp, 2004; Issakova et al., 2005; Issakova et al., 2006; Issakova, 2006a; Prank et al., 2006a; Prank et al., 2006b; Prank et al., 2007).

2.1 Expressions in T-algebra

The main object for the program to work with is the algebraic expression. In this section, I will describe, which expressions are allowed in the program, i.e., which expressions are treated as correct.

An algebraic expression in T-algebra is defined in the following way: elementary expressions (or basis) are integers, decimals (separator in decimal fractions used in Estonia is "," (comma)) and variables (small letters $a...z$). Expressions are composed by recursively applying different operations (unary $+$, $-$, binary $+$, $-$, $:$, $:$ (division sign used in Estonia is ":" (colon)), exponentiation (\square^{\square} and \square^2), common fraction ($\frac{\square}{\square}$ and $\frac{\square}{\square}$), grouping symbols (parentheses $()$ and brackets $[\]$). In addition, more complicated expressions are realized in the program as well: linear equations (sign $=$), linear inequalities (signs $<$ \leq \geq $>$) and systems of linear equations (sign $\{$).

Expressions in the program must be mathematically correct and involve various combinations of abovementioned symbols. Here are some examples of correct expressions:

- $\frac{1}{2} - 2\frac{2}{3}$ (mixed numbers, e.g., $2\frac{2}{3}$ are widely used in Estonia);
- $x^2(1+x)^2$;
- $2x - 2 \leq 3 - x$.

The following expressions are treated by the program as incorrect:

- $3\frac{x}{3}$, because a variable is not allowed in mixed numbers;
- $a2b + 2bc3av$, because a multiplication sign is required in monomial multiplication, and constants are not permitted between variables in monomial multiplication.

Backus-Naur Form full description of expressions in T-algebra is presented in Appendix A.

There are no quantitative constraints placed on expressions. The program also supports several-storied fractions and exponentiations, etc. However, some constraints are placed on expressions by problem types. For example, when solving linear equations all expressions have to be linear equations – they have to contain an equality sign.

2.2 Description of the problem-solution window

Figure 2.1 shows the problem solution window of the T-algebra program.

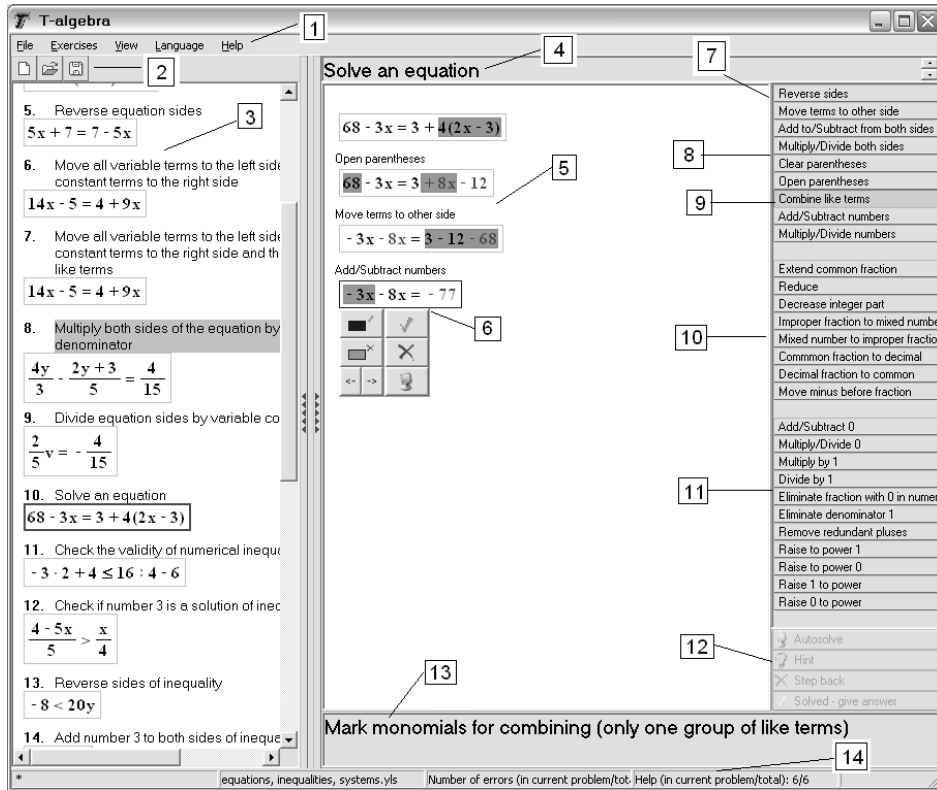


Figure 2.1. The problem-solution window of the T-algebra program

The problem-solution window has been divided into two logical parts. The left-hand part contains a field displaying a list of problems. The list contains expressions and formulations of problems with the number of problem in this file. In addition, information on the problem resolution is displayed – if a problem has been solved, this is indicated in the list by green background color as shown in the figure. A problem, which is currently being resolved, is displayed in a red box. The right-part of the window contains solution steps and a virtual keyboard, the menu of possible actions and instructions for the student in this particular situation.

The main components of the window are:

1. The program menu bar, which enables to manipulate with files (New, Open, Save, Save as, Close) and problems (Previous, Next), open additional windows (for example, view the error counters – categorized

- by the types of errors, or list of all errors), or choose the language of the program.
2. Buttons, which duplicate some items from the File menu (New, Open, Save (as)).
 3. The list of problems to be resolved, which also shows the results of problem solving. It is also possible to hide the list of problems and to use the whole window for viewing the solution.
 4. The text of the problem.
 5. The resolution process for the selected problem – the sequence of steps.
 6. The last expression with virtual keyboard for selecting the operands.
 7. The grouped list of rules.
 8. Rules for steps of linear equation solving algorithm.
 9. The selected rule (differs from other rules by background).
 10. The rules for manipulation with fractions.
 11. The rules for simplification of expressions with 1 , 0 and redundant pluses.
 12. The buttons for asking help, rollbacking the solution steps and giving the answer.
 13. Instructions to aid the problem resolution process (indicating what the student should do next: choose the rule to apply next, mark some parts of expression, enter something, etc.).
 14. The status bar, which shows information about the user and the open set of problems.

2.3 General dialogue scheme in T-algebra

Each solution step consists of three stages:

1. selecting a transformation rule (action);
2. marking the parts of the expression (object);
3. entering the result of the application of the selected rule (input).

Hereafter we will refer to this scheme as the Action-Object-Input scheme, after its three stages.

Part (5) of the sample window in Figure 2.1 shows the resolution process for solving linear equation $6x - 3x = 3 + 4(2x - 3)$. The solution is not yet complete, but some steps have already been taken. At the first step, *Open parentheses* was selected as the operation, and product of number and expression in parentheses was marked. The result of multiplication was entered in the result. At the second step, $6x$ and $8x$ were moved to other side in the same way. During the third step, the numbers on the right side of equation were added/subtracted. The solution path is similar to how the student would solve this problem using pencil and paper, because T-algebra follows the solution

algorithm taught at school. In T-algebra only the final and correct solution path is displayed. If the student performed a wrong or unnecessary step (for example, moved $-3x$ during second step to the right side) and then took this step back, the teacher would not see this deleted step in the solution path.

When solving this problem further on paper, the student would at first examine the expression. She/he should decide to combine like terms. Then she/he would underline the like terms she/he wants to combine and write the resulting equation on the next line. When applying the *Combine like terms* rule the program follows principally the pencil and paper scheme of actions. The corresponding solution step consists of the following three stages (the first is already completed in Figure 2.1):

1. Selecting a transformation rule – the student selects from the rule list the rule of combining like terms – the program allows selecting any rule without checking whether it is possible to apply such a transformation at that stage or not.
2. Marking the parts of the expression – the student marks the terms similar to x , using the mouse (the first like term $-3x$ is already marked) – the program checks whether the selected parts of the expression are actually like terms, and it also checks whether these terms can be combined (i.e., whether they belong to the same sum). The students do not have to select all suitable terms at one time – the minimum selection needed is two similar terms.
3. Entering the result of the application of the selected rule – the program copies unchanged parts of the expression onto the next line and asks the student to enter the resulting term or its parts depending on the solution mode. The third stage has the greatest potential for mistakes, because the student must apply the rule for the marked parts and enter the result. Three different input modes were designed for each rule to achieve better diagnosis of errors (Issakova et al., 2005). Different input modes are described in more detail in the next section.

This example should provide an idea of the connection between the actions of the student and the program – what is checked by the program and when. If an error message is displayed at any checking stage during solution of a problem, the student must first correct the error himself or let the program correct the error in order to proceed to the next stage. For example, the student cannot proceed further if she/he has entered a wrong coefficient of term – the program will diagnose this standard mistake and display an appropriate error message. For each action of the student, the program gives specific instructions (‘Choose the rule to apply next’, ‘Select terms to combine’, etc.). The student can cancel the step at any time. It is also possible at any stage of the step to ask the program for help and let the program complete certain stages automatically. During the input of the result the student can press the special button with

computer image and the program will put the right answers into the boxes. The same help button is also available when marking the operands – the program will select appropriate operands itself (Figure 2.1). 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* (Figure 2.1). The same button will indicate if the problem is solved.






You can make a brief introduction to using T-algebra with the quick start to T-algebra student's program presented in Appendix B.

2.3.1 Selection of the rule and marking the parts of expression

The order of the first two stages is not fixed by the program. The student can mark the operands before, after, or even before and after selection of the rule. Only after confirmation of these two stages the program controls whether the selected parts are suitable for the selected rule. The reason for that is explained in Chapter 4 Section 4.2 (as a result of second experiment).

The set of rules displayed in the menu depends on problem type. Selections of the rule and designed set of rules for the domain of linear equation, linear inequalities and systems of linear equations are thoroughly described in Chapter 3 Section 3.2.

Unlike many other programs, T-algebra requires precise marking of operands for diagnostic purposes. For example, for the operation *Combine like terms* the student should mark only those terms that will be actually combined. Accordingly, the editor of T-algebra enables to mark more than one piece of the expression. The program allows the preceding pluses and minuses to be marked or not, while the program will always consider that the sign was marked. The program also allows the marking of parts as one large item if they stand next to each other, while the program itself will divide it into parts for further processing.

In addition to entering expressions, the expression editor also allows marking the operands. In order to mark a part of an expression, with the expression editor in the marking mode (see Figure 2.2), this part should be selected (either with mouse or keyboard) just like in a regular text editor and then the user should press the  button. To remove marking, the student would have to select the same part and press the  button. Buttons   are for moving between the marked parts. When the user has finished marking and selected operation, she/he should confirm the first two stages by pressing the  button.

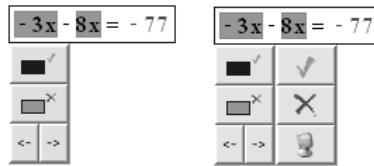


Figure 2.2. Expression editor of T-algebra program in the marking mode (left part – the rule is not selected yet, right part – the rule is already selected)

2.3.2 Entering the result of the application of the rule

The Action-Object-Input scheme was first developed in the Master's Thesis of D. Lepp in 2003 (Lepp, 2003), which serves as a prototype of T-algebra. The author designed the input forms separately for each conversion rule, trying to minimize the input and requiring entering only critical information for any particular operation. The form and number of parts that could be entered became too varied for different rules and the user interface of the program became too confusing. In T-algebra we try to design three fairly uniform and standard input modes for all rules. The three input modes are named *free* input, *structured* input and *partial* input. Free input mode is easily comprehensible (it is similar to working on paper) and it can be designed for each rule. Structured and partial input modes are more specific. The program helps the user in a certain way, whether by indicating the structure of the result or even filling out a part of the result.

At the third stage (Input) of each step, the student should enter some parts of the expression that result from the previously selected operation. The program generates the expression in the next line based on the selected rule and marked parts, and leaves blank certain important parts of the resulting expression. When working with paper and pencil, the students themselves have to write the whole resulting expression. Consequently, they try to reduce the amount of routine rewriting by making several transformations at once. The program makes the work easier for the students by copying the parts of the expression that remain unchanged so that the students would have to enter only the parts that were modified. Only one transformation can be made in each step. This makes it easier for the program to check the solutions and gives the teacher a better overview of the student's solution. The results can be entered on the keyboard or on the virtual keyboard (see Figure 2.3).

$$5x^2 + 12xy + 4y^2x - 2x^2 - 6xy - 2yxy - 4xx =$$

Combine like terms

$$= 5x^2 + 12xy \boxed{} - 2x^2 - 6xy - 4xx$$

The image shows a screenshot of a mathematical software interface. At the top, there is an algebraic expression: $5x^2 + 12xy + 4y^2x - 2x^2 - 6xy - 2yxy - 4xx =$. The terms $4y^2x$ and $2yxy$ are highlighted with yellow boxes. Below this, the text "Combine like terms" is displayed. Underneath, the expression is shown again: $= 5x^2 + 12xy \boxed{} - 2x^2 - 6xy - 4xx$. The box $\boxed{}$ is highlighted with a yellow box. Below the expression is a keyboard input panel with various symbols and numbers.

Figure 2.3. Input of the result (in free input mode)

The parts of the expression that the student has to enter are highlighted with yellow boxes. The form and the number of user-definable parts depend on the selected rule, marked parts and mode. While entering the results, the program protects other parts of the expression from modification – only the highlighted locations of the expression can be modified. This makes it easier for the program to check the solution and, in addition to checking the equivalence between the new expression and the previous one it also enables the correctness of separately entered parts to be checked, thus improving the overall responsiveness of the program to errors. The input mode is selected by the teacher during problem composition.

2.3.2.1 Three input modes

In *free* input mode, the program generates one input box (or two boxes in the case of some rules with fractions and equations) inside the expression on the next line instead of marked parts (see Figure 2.3). The student should enter in the box one expression replacing the whole marked part from the previous line. Even though the name of the mode is ‘free input’, the input is still restricted to some extent. The editor gives the student freedom in entering, but after the input, the program checks not only the syntactical correctness of the expression and equivalence to the previous expression, as also occurs in Aplusix (Nicaud et al., 2004), but also the correctness of applying the rule.

For example, in Figure 2.3 the rule *Combine like terms* was selected and two like terms were marked. After the input is confirmed, the program first checks whether the entered part is syntactically correct, monomial and equivalent to the marked parts. Finally the program checks the equivalence of the complete new line with the previous line. If the student enters $2y^2x$ without the leading addition sign then the entered part is equivalent to the marked parts but the whole expression is not. In some other rules the student should type brackets around the entered sub-expression.

In *structured* input mode, the program uses the information about the actual rule and operands, and itself predicts the structure of the required input using

different input boxes for signs, coefficients, variables, exponents, etc. (see Figure 2.4).

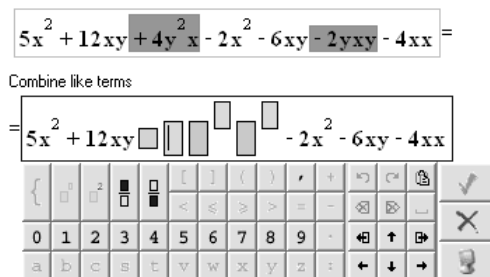


Figure 2.4. Structured input

The size and position of the boxes should immediately indicate to the user what should be entered. In this mode, input into the boxes is restricted. If the cursor is in some input box, the buttons with unavailable symbols on the virtual keyboard are inactive and corresponding keys on the regular keyboard do not work. For example, in Figure 2.4, where the rule *Combine like terms* was selected and two like terms were marked, the program offers a structure of monomial with six boxes in the next line. The first box is the sign input box, the next is the coefficient input box (active in Figure 2.4) followed by boxes for input of variables with exponents.

The program generally offers the same number of boxes for the variables of one monomial as the number of variables in the marked parts. Variables can be entered in arbitrary order inside one monomial, but the program requests the user to standardize the result to some extent (for example xyx to xy^2 or y^2x in Figure 2.4). It is possible to leave some boxes empty. For example, if the power of the variable is 1, then the exponent box can be left empty.

When the user has finished entering, the program checks whether the new expression is equivalent to the previous one and whether the entered parts are equivalent to the parts computed by the computer. If the expressions are not equivalent, the program checks the correctness of each entered part to produce a more specific diagnosis.

Structured input mode is rule-specific (each rule requires its own input pattern of the resulting expression) and it turned out that this mode is useless for some rules. For example, it would be pointless to offer a structure for the result if the applied rule was *Clear parentheses*, because only signs change.

The third mode (*partial input*) is a simplified form of the second mode, where the program fills some boxes by itself. For example, Figure 2.5 shows the same example as Figure 2.4, but using partial input. The program itself writes the variables with exponents. The user should enter only the sign and coefficient of the monomial. The program also simplifies the work of the user by

converting the monomial into normal form. After the input the program checks the correctness of the expression and its equivalence to the previous one as in other modes.

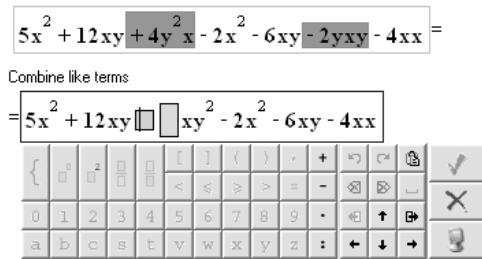


Figure 2.5. Partial input

It turned out that there are some rules for which it is vary hard (and unnecessary) to design boxes for both (structured and partial) input modes (see examples in Chapter 3 Section 3.2, rules *Add to/Subtract from both sides* and *Substitute variable*).

2.3.2.2 Additional input

While designing the rules for T-algebra we found that it is difficult to express some rules purely in terms of Action-Object-Input dialogue (Issakova and Lepp, 2004). In order to decide which features we need to add to the dialogue, we studied written work of the students – how and which steps they make while solving problems on paper. We also reviewed school textbooks to find all the rules used in the solution steps and the algorithms used for solving the problems. In virtually every topic we found some rules where adequate expression required modification of the dialogue. We extended the input stage of the dialogue by adding three new features (Lepp, 2005). Each rule may use one or several of these features at once, depending on the mode running:

- input of the rule-specific additional information;
- input of intermediate result;
- adding terms to the result.

In some rules the result of the application is not uniquely defined by the operands but depends on some additional decision of the student. For example, Estonian textbooks suggest writing addition of fractions with different

denominators as follows: $\frac{1}{6} \overset{4}{\parallel} + \frac{3}{8} \overset{3}{\parallel} = \frac{4+9}{24}$. Here the students first calculate the

common denominator of the fractions being added and write it in the resulting fraction after the equality sign. Then the students find so-called extenders (“extenders” are numbers by which you need to multiply both numerator and denominator of the fraction to convert the denominators to the common denominator (this term is used in Estonian schools and textbooks (Nurk et al.,

2000))) and write them to each addend. Even if this information (common denominator) is included in the final input, it could be very difficult to guess the case of error if the input is inconsistent. This information is also needed for checking the extenders in intermediate input stage.

As we still want to check the students' skills and identify the cause of errors, this specific information has to be entered separately. A separate input window was created for each such rule that needs additional information. When adding fractions, the student has to input one number – the common denominator of the selected fractions (see Figure 2.6). Similar input was used in the MathPert system (Beeson, 1998).

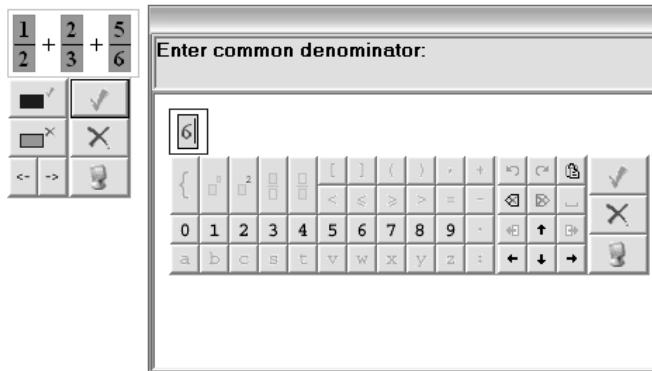


Figure 2.6. Input of the rule-specific additional information for addition of fractions with different denominators

This added window is the first new feature that can be followed by other options or the usual input of the resulting expression. When the objects of the rule have been selected, the program checks whether the rule is applicable to them and after that displays this input window to the user. After the student has made the input in this window, the program checks whether the entered information is correct. If no errors are diagnosed, the student may proceed to the next stage.

Looking at pencil and paper solutions of the students, we found that some rules are applied using two input stages: first, some intermediate result is found (for example, extenders for each fraction are found when adding fractions with different denominators) and then the final result is written. We tried to follow the same pattern while extending the dialogue that is used when working with pencil and paper: at first, the common denominator is entered in an additional window, then the extenders of the fractions are entered (Figure 2.7) and after that the members of the final result are entered. As we wanted to keep the initial expression unchanged with the objects selected in it, the program copies the expression to a new line after entering the common denominator and provides

requires exact application of the multiplication rule only; combining similar terms is not allowed.

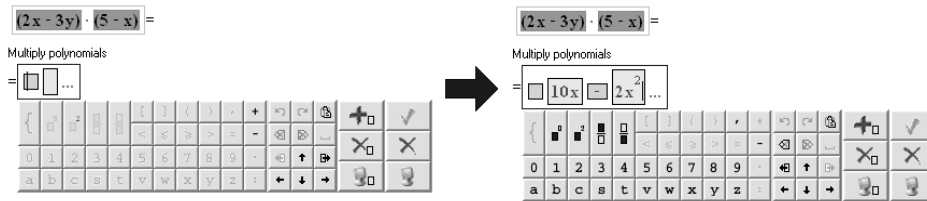


Figure 2.8. Adding terms to the result in the rule of multiplying two polynomials

2.4 Domain expert module in T-algebra environment

According to Wenger (Wenger, 1987), an Intelligent Tutoring System consists of four 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 a computer algebra system instead of the Domain knowledge) (see (Ravaglia et al., 1998; Sangwin, 2005)). 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" (Ravaglia et al., 1998, 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" (Beeson, 1998, 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 (Anderson, 1988), 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' (Buchberger, 1990)) 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, including generation of questions, explanations and responses, and on the 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 of expressions with 0 , 1 and redundant pluses, which are not school 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 manipulation with fractions 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. In the domain of linear equation, the expert module works according to the following algorithm (list of rules) for linear equation solving (this is simplified version of realized algorithm; for the full version see Chapter 3 Section 3.3):

- rules for simplification of expressions with 0 , 1 and redundant pluses (not school algorithm steps: *Add/Subtract 0*, *Multiply/Divide by 1*, etc.);
- rule for arithmetic operations and manipulation with fractions;
- rules *Open parentheses* and *Clear parentheses*;
- rule *Multiply/Divide both sides* for removing fractions (multiplication);
- rule *Add/Subtract numbers*;
- rule *Combine like terms*;
- rule *Move terms to other side*;
- rule *Multiply/Divide both sides* for isolating variable (division).

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

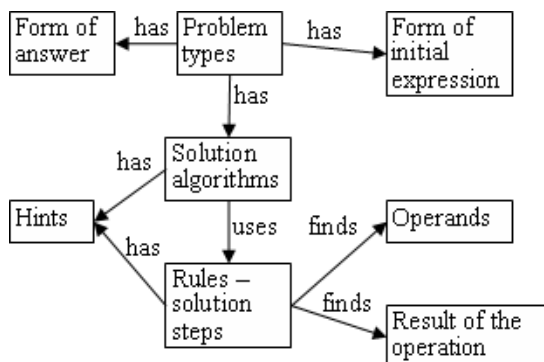


Figure 2.9. Semantic network of domain knowledge

2.4.1 Solution engine

The order in algorithm is very important, because the expert module examines the 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 composes a solution path and gets an answer to the problem.

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 necessary parts of the result depending on the input mode, puts them together with unchanged parts and gets the new expression/equation.

2.4.2 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 section describes the applications of our domain expert module in T-algebra.

2.4.2.1 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* (Figure 2.1). The expert module will check the current expression and problem type, find which rule should be applied and display appropriate help message (Figure 2.10). The same button will tell if the problem is solved.

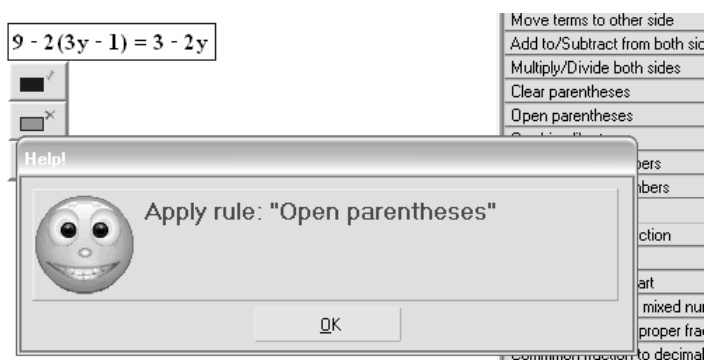


Figure 2.10. Help message for choosing the rule

During the marking of the operands the student can press the special button with computer image (Figure 2.1) and the program will select the appropriate operands itself. If the student selects an impossible rule and asks for help for marking the operands then T-algebra responds that the application of the selected rule is impossible. The same help button is also available when entering the result – the program will put the right answers into the boxes.

T-algebra can also generate and display the whole solution from the current expression (the button *Autosolve*, see Figure 2.1), but we suggest teachers disable this possibility in the problem file.

2.4.2.2 Checking the stages of the step

In existing input-based interactive learning systems the student can err and the program diagnoses only the non-equivalence (like current version of Aplusix). In rule-based systems the student can either select unsuitable rule or the student cannot make mistakes at all. For example, in Mathpert the student cannot select a syntactically incorrect part and even cannot select an unsuitable rule. In this

environment the student first selects some part of the expression (the program selects only syntactically correct parts) and only then the program offers "... those operations that make sense for what you've selected" (Walden, 1997, p. 35). In the last version of MathXpert the systems sometimes offers unsuitable rules too.

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. In some cases where the application of a rule is possible but leads to a completely wrong direction (for example, in the problem on adding a given number to inequality sides, the student is choosing the addition rule for the second time), then the program will show appropriate error message and will not proceed to the next stage. If the application of this rule is simply unreasonable – leads to the 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 checks 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 product or sum in parentheses (Figure 2.11)). 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.

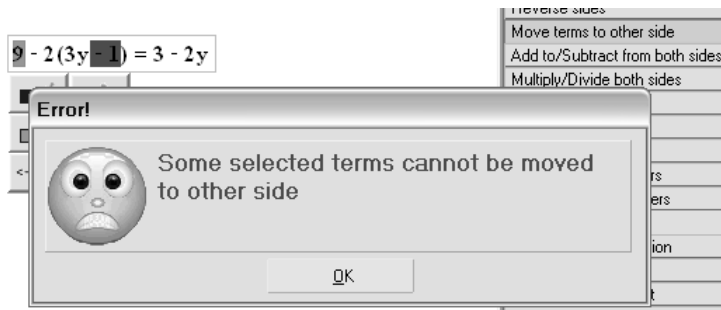


Figure 2.11. Error message displayed when marking the operands

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 confirms 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 the right answer for the 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 implemented in T-algebra (some studies have been conducted to collect the students' mistakes made on paper (Hall, 2002; Issakova, 2005; Sleeman, 1984)), then T-algebra is able to diagnose it and offer an appropriate error message (Figure 2.12). If the mistake is not in the composed set of standard mistakes, then T-algebra tells about the non-equivalence of expressions/equations.



Figure 2.12. Error message displayed when entering the result

2.4.2.3 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 the student believes that she/he has solved the problem she/he should push the button *Solved – give answer* (Figure 2.1). The program checks whether the expression 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 (Figure 2.13).

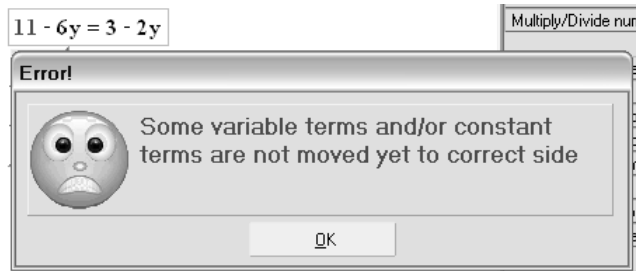


Figure 2.13. Error message displayed when giving an answer

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 recognize 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 multiplying 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 $\frac{5}{6} = x$.

2.4.2.4 Checking the answer

If the student pushed the button *Solved – give answer* 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.

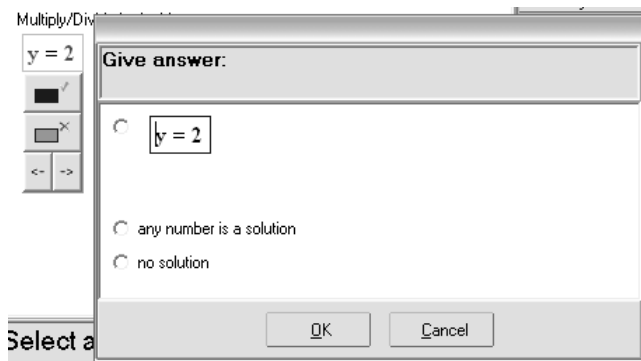


Figure 2.14. Giving an answer

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 *Solved – give answer*, a separate window with three options (Figure 2.14) 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.

2.4.2.5 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 can display the solution path and answer to the teacher if the appropriate button is pressed (see Section 2.6).

2.4.2.6 Checking the equivalence of two expressions

The program uses the solution engine from domain expert module to check equivalence of two expressions. A special algorithm is composed for that purpose, which consists of rules that simplify the expression, for example, the rule *Combine like terms* or rules for simplification of expressions with 0 , 1 and redundant pluses. 23 rules in total were selected for that algorithm. In order to check equivalence, the program composes difference of two expressions (which should be checked) and simplifies or solves it with the composed algorithm in the same way as the solution engine. If this difference is zero, then two expressions are equivalent.

This algorithm cannot be used for checking equivalence of two equations (or inequalities or systems of equations). In T-algebra the left side and the right side of an equation/inequality are usually checked separately. These are expressions and their equivalence can be checked by the composed algorithm. Therefore, the algorithm for checking the equivalence of two equations was not implemented.

The composed checking algorithm is suitable for expressions allowed in T-algebra. As T-algebra does not work with trigonometry and absolute value, we could implement this simple algorithm.

2.5 Student statistics

T-algebra calculates different statistics during the solving process. This statistics is also saved to the solution file (.lah). Calculated statistics can be viewed from the *View* menu in the student's program.

First, it is possible to view *Error counters* and *Error list*. We have designed 20 different categories and divided all diagnosed error types between them. The categories include, for example, selection of objects of wrong form, selection of incompatible objects, errors in the form of entered subexpression, calculation errors, errors in calculating the sign of entered subexpression, etc. For the full list of categories, see Figure 2.15. It is possible to review statistics on all errors made (numbers of errors of each category) (Figure 2.15) as well as error situations themselves (the right lower part of Figure 2.15).

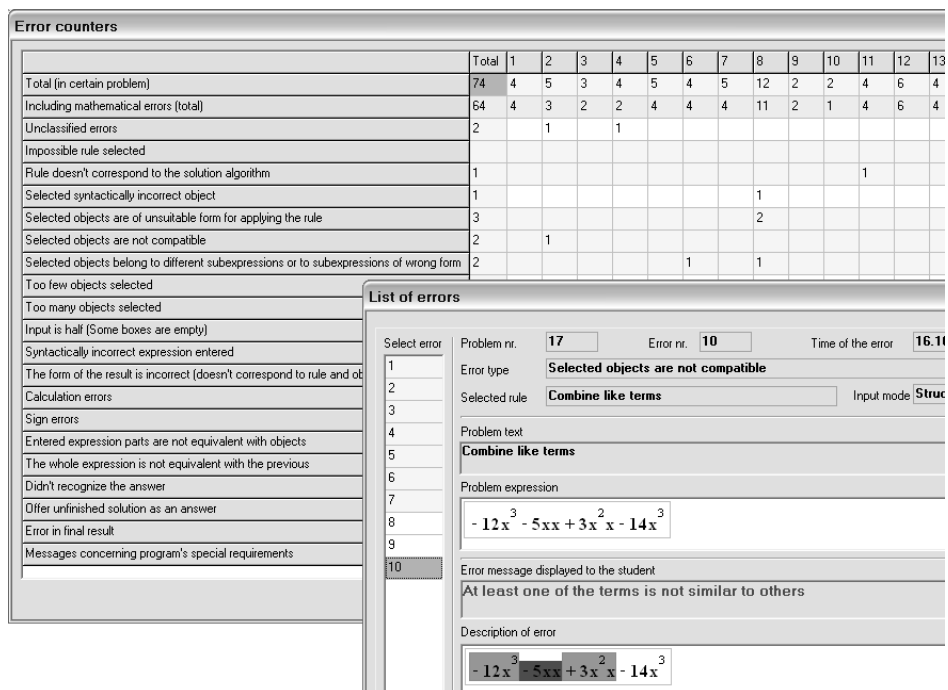


Figure 2.15. Error counters and description of a particular mistake

The next two items in the *View* menu are *Counters of help usage* and *List of help usage*. T-algebra saves all situations when the student asks for help. We grouped help usage into 7 categories depending on the place where help was asked, and it is possible to review statistics on help usage in general (the number of times that help was used in each category) (Figure 2.16) as well as the actual situations of help usage (the right lower part of Figure 2.16).

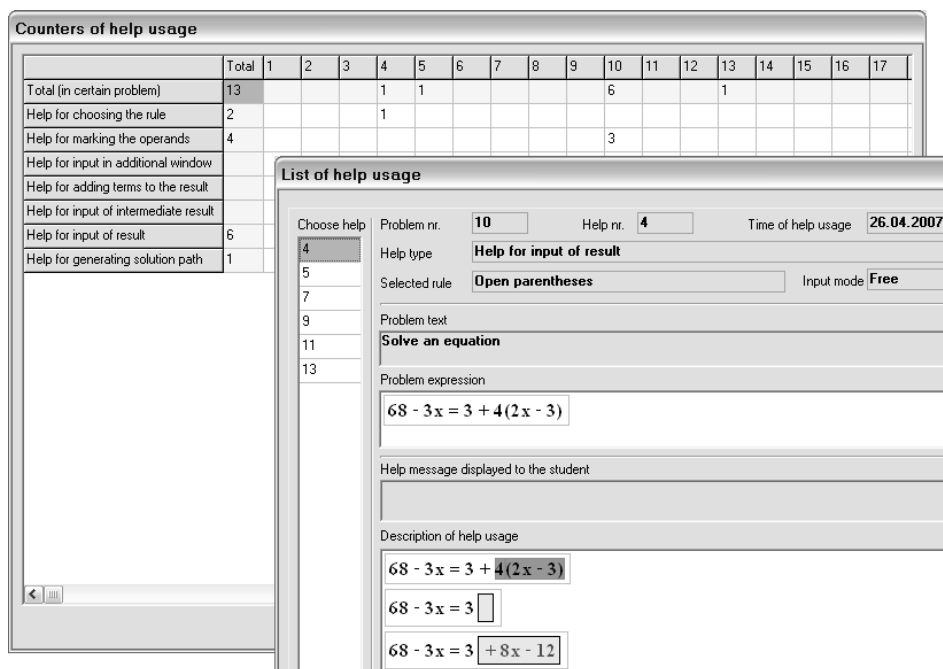


Figure 2.16. Counters of help usage and description of particular help

The last item in the *View* menu is *Statistics of solving*. From this table it is possible to review general statistics of solving, like how many problems are solved and how many errors were made. For the full list of calculated items, see Figure 2.17.

	Total	1	2	3	4	5	6	7	8
Problem is solved	3 / 22	No	No	No	No	Yes	Yes	No	No
Number of made errors	3	0	0	0	0	0	0	0	0
Including mathematical errors	2	0	0	0	0	0	0	0	0
Number of help usage	13	0	0	0	1	1	0	0	0
Uses the button Autosolve	1 / 22	No	No	No	No	Yes	No	No	No
Number of steps	21	0	0	0	1	3	3	0	0
Begginig of solving	26.04.2007				26.04.2007	26.04.2007	8.05.2007		
Time of begginig of solving	11:54:21				11:54:21	11:54:25	14:18:10		
End of solving	8.05.2007					26.04.2007	8.05.2007		
Time of end of solving	14:19:31					11:54:26	14:18:27		
Spent time	133:59	0:00	0:00	0:00	0:04	0:01	0:16	0:00	0:00

Figure 2.17. Student statistics

2.6 The program for teachers

The students solve the problems from problem files. A problem file can contain material for some topic, one lesson/test or even only one problem. Problem files can be created and edited in a teacher's version of T-algebra.

The author composing a problem file chooses for each problem the following:

1. field and type;
2. text (each type has also some default text, for instance *Solve an equation*);
3. initial expression (equation, inequality, equation system) and values of other obligatory parameters (if needed for certain problem type);
4. input mode (free, structured or partial);
5. whether hints are available for the selection of rule, marking of operands, input of result and for demonstrating the entire solution.

Figure 2.18 demonstrates the problem composition window of the teacher's program, where the teacher has entered an equation.

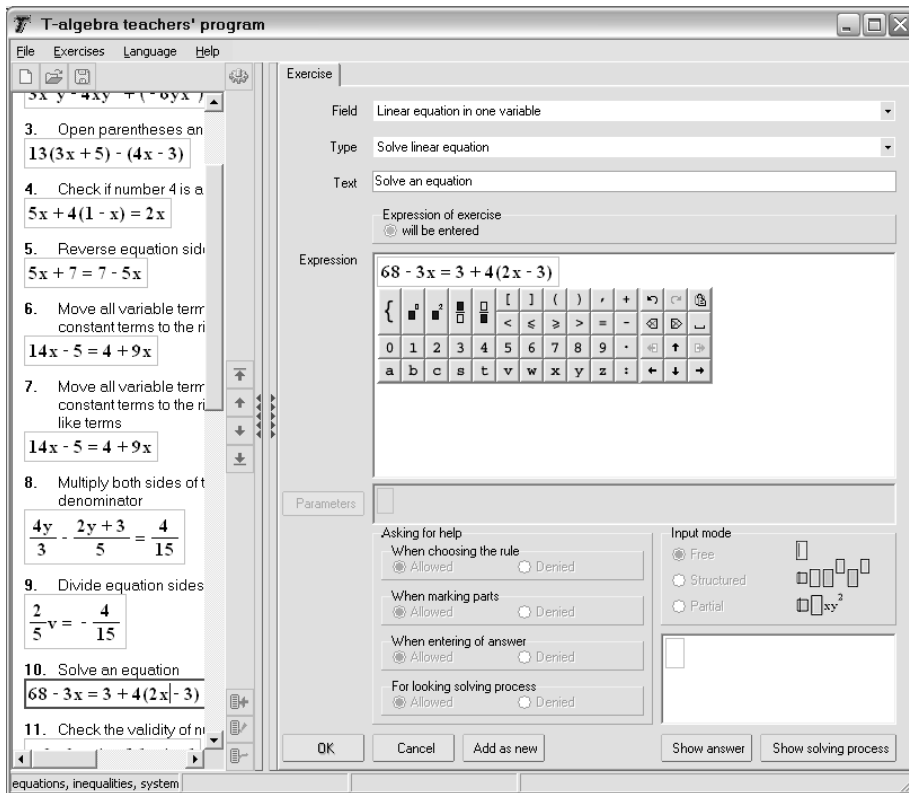


Figure 2.18. Problem composition window of the teacher's program

When composing a problem, the teacher has a possibility to get the answer calculated by the automated solution module of T-algebra (right lower box in Figure 2.18) and also to see the entire demo solution in a separate tab.

It is possible also to select default values of items 4 and 5 (input mode and hints) for the entire file and enable or disable the possibility to modify them in the composition of concrete exercises (Figure 2.19). The author can also assign a password to the problem file.

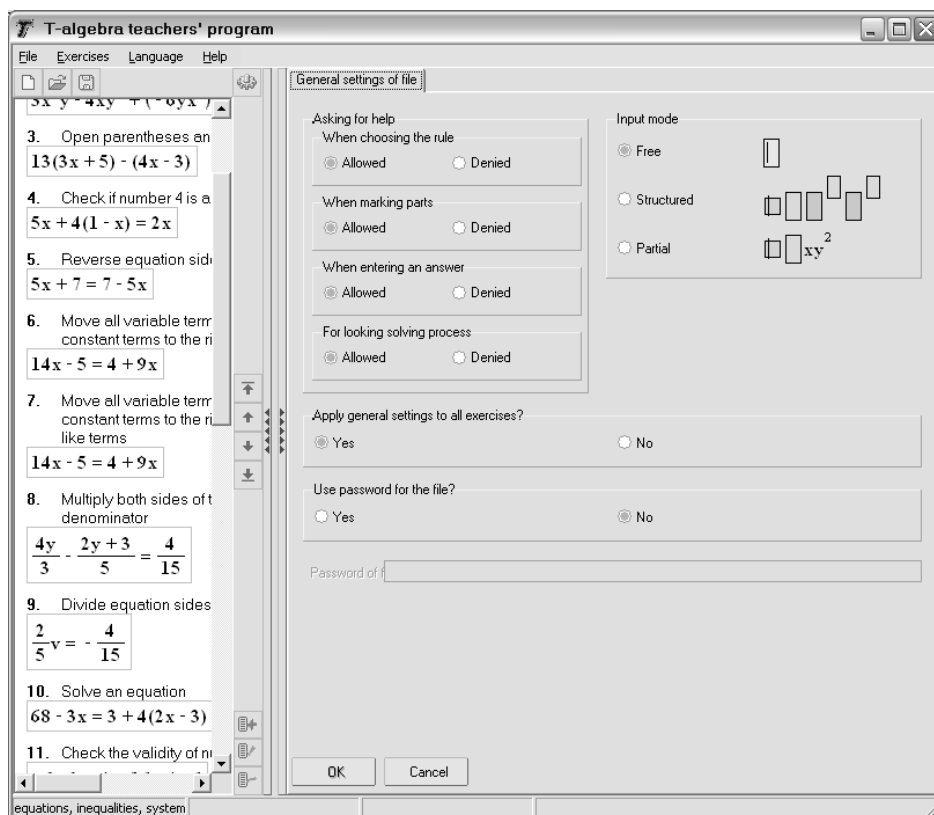


Figure 2.19. General settings for the problem file

2.7 Implementation

T-algebra is written in Delphi programming language (further development of Object Pascal programming language) using Delphi integrated development environment, also known as Borland Delphi. At the time of selecting the programming language, Java language also was under consideration. Delphi language was chosen because it enables to create GUI (graphical user interfaces) more easily and quickly than Java. Furthermore, a program written in Java language would need additional installations (Java runtime

environment) by the user, which could cause some difficulties for the teachers and students. The version of Delphi that was used is 5 (Delphi 5).

The size of the entire program at the moment is more than one hundred thousands lines. It is hard to specify the exact number of lines written by any one of the three authors of the code, but the authors estimate that approximately equal parts were coded by each author. It means that the author of the thesis wrote more than thirty thousands lines. The program consists of two separate programs, one for students and one for teachers. The size of the compiled students' program (.exe file) is slightly more than two megabytes; the size of the compiled teacher program is slightly less than two megabytes. The compiled program can be accompanied by a text-file (.mes file) containing all user interface messages (error messages, help messages, texts on program windows, etc.) in a certain language. If there are several .mes files, then there is a possibility to change the language in the program. If there are no .mes files, then all texts are in Estonian language. Another file (.ini file) appears when the program is used. This file contains all configuration settings (location and size of the program window, the language used, etc.).

The most important classes in implementation are *rule-classes* and *type-classes*. Every rule and every type are written as separate class. Rules are not written as rewrite-rules, but as modules coded directly in the implementation language.

Every *rule-class* has the following methods:

- function that returns objects, for which this rule can be applied, or nil, if this rule can not be applied to this expression;
- procedure that checks whether the student has selected something and whether the selected objects are suitable for application of this rule; in cases of error it throws an exception and shows error message;
- procedure that applies the rule to objects selected by the student and proposes the next line expression with boxes for input depending on input mode;
- procedure that analyzes the parts of the expression entered by the student into the boxes and compares with the parts calculated by previous procedure; in cases of error it throws an exception and shows error message.

If a rule requires some kind of additional input (additional input was described in Section 2.3.2.2), the *rule-class* should have corresponding methods and procedures similar to those mentioned above; for example, a function that proposes help in additional window (rule-specific additional information) or a function that checks the input in additional window and in cases of error throws an exception and shows error message, etc.

Different methods of *rule-classes* are used by the domain expert module for its applications, for example, if the student asks help for marking objects, then the domain expert module calls the function described above (which returns objects) and marks these objects; the procedures that check marking and input are used during confirmation of each stage. The solution engine also uses the function that returns objects and the procedure that applies the rule for composing a solution path.

Naturally, every *rule-class* can contain any additional help functions and procedures.

The average size of *rule-classes* (written by the author of the thesis; for the full list see Section 3.2) is 943 lines, the largest having 2396 lines (rule *Multiply/Divide both sides*, Section 3.2.1) and the smallest having 252 lines (rule *Reverse sides*, Section 3.2.3).

Every *type-class* contains the following functions and procedures:

- procedure that checks whether the expression entered in the teacher program is suitable for selected problem type; in cases of error it throws an exception and shows error message;
- function that returns all rules that can be used by the student in this problem type;
- function that returns the rule that can be applied next, or -1 if the problem is solved;
- procedure that checks whether the current expression is already in solved form; if not, it throws an exception and shows error message.

If *type* includes parameters or answer should be specified (answer differs from solved form), then *type-class* contains corresponding methods and procedures; for example, a procedure that checks whether the parameter entered in the teacher program is suitable for the chosen problem type (in cases of error throws an exception and shows error message) or a function that checks the answer selected in the additional window (in cases of error throws an exception and shows error message), etc.

The function that returns the rule actually realizes the algorithm for this problem type. This function is also used by the solution engine for composing a solution path. Some other methods are used by the domain expert module for its applications, for example, the procedure that checks whether the current expression is already in solved form is used for checking completion.

Any additional help function and procedure can be written in every *type-class*.

The average size of *type-classes* (written by author of the thesis, for full list see Section 3.3 and Appendix C) is 330 lines, with the largest being 584 lines (type *Solve linear equation*, Section 3.3.2) and the smallest 186 lines (type *Move terms to correct side of equation*, Appendix C).

3 PROBLEMS AND ALGORITHMS IN THE DOMAIN OF LINEAR EQUATIONS, LINEAR INEQUALITIES AND SYSTEMS OF LINEAR EQUATIONS IN SCHOOL TEXTBOOKS AND IN T-ALGEBRA

I have chosen the domain of linear equations, linear inequalities and systems of linear equations for exploration and programming in T-algebra. First of all, school textbooks in mathematics were explored, then the rules were programmed, and finally problem types were composed. I will describe problems and algorithms in the domain of linear equations, linear inequalities and systems of linear equations in school textbooks and in T-algebra in this Section on the basis of published articles (Issakova, 2006c; Issakova, 2006d).

3.1 Problems and algorithms in school textbooks

Estonian mathematics school textbooks were explored to clear up operations, algorithms and problem types that are taught at Estonian schools. As different teachers use different textbooks, I explored the best-known books in Estonia (Nurk et al., 2000; Pais, 1998; Tõnso, 2002; Lepik et al., 2000; Pais, 1999; Veelmaa, 2000; Kasemaa and Lind, 1997). Some English textbooks (Barnett and Kearns, 1990; Barnett and Ziegler, 1989; McKeague, 1979; Zuckerman, 1976) were inspected as well, but Estonian textbooks were followed if some differences were found.

As we had already decided that T-algebra will enable the student to solve under the control of the program only expression manipulation (technical) problems where the original expression/equation/inequality/system of equations is given in the text of the problem (entered by the composer), we skipped all other problem types (for example, problems stated in words) in our exploration of textbooks. We also skip all graphing problems and will not describe such problems here.

3.1.1 Linear equations

Most of Estonian textbooks (Pais, 1998; Tõnso, 2002) begin the topic of linear equation with description of combining like terms and opening parentheses. The following definition of like terms is given: Terms, which are the same or differ only by numerical coefficient, are called like terms. The process of combining is described as follows: Like terms are combined by adding their numerical coefficients. In some English textbooks (Barnett and Kearns, 1990), combining like terms is also described in this topic. The English textbooks present a

slightly different definition of like terms: Two terms are called like terms if they have exactly the same variable factors to the same powers. The technique of opening parentheses is also described in Estonian textbooks: if rational number should be multiplied with sum of terms in parentheses, then each term is multiplied with this number and parentheses are left out.

After this subtopic is explained, the student practices the following problem types: *Combine like terms*; *Open parentheses*; *Open parentheses and combine like terms*.

After such introduction almost all textbooks explain the equation and its root or solution: An equality involving at least one variable is called equation; A solution or root of an equation in a single variable is a number that, when substituted in the equation, makes the left side of the equation equal to the right side.

In order to clarify these definitions, the following problems are solved: *Is the statement a first-degree equation in a single variable*; *Check whether the number is a root (solution) of the given equation*.

After that, Estonian mathematics textbooks (Nurk et al., 2000; Pais, 1998; Tõnso, 2002) describe the basic properties of equation:

1. the sides of equation can be reversed;
2. the same quantity can be added to both sides of equation (subtracted from both sides of equation);
3. both sides of equation can be multiplied or divided by the same nonzero number.

From the second property it is derived that it is possible to move terms from one side to other side of equation changing the signs of terms.

In English textbooks (Barnett and Kearns, 1990; Barnett and Ziegler, 1989; McKeague, 1979; Zuckerman, 1976) there are no properties of equation, but there are given the properties (axioms) of equality. There are 4 basic properties of equality: reflexive, symmetric, transitive properties and substitution principle, and 4 further properties of equality: addition, subtraction, multiplication and division properties.

If a , b , and c are names of objects, then:

1. $a = a$ – reflexive property;
2. If $a = b$, then $b = a$ – symmetric property;
3. If $a = b$ and $b = c$, then $a = c$ – transitive property;
4. If $a = b$, then either may replace the other in any statement without changing the truth or falsity of the statement – substitution principle.

For a , b and c any real numbers:

1. If $a = b$, then $a + c = b + c$ – addition property;
2. If $a = b$, then $a - c = b - c$ – subtraction property;
3. If $a = b$, then $ca = cb$, $c \neq 0$ – multiplication property;
4. If $a = b$, then $a : c = b : c$, $c \neq 0$ – division property.

We found the following problem types (requiring application of one or two properties) after that theme in the textbooks: *Reverse equation sides; Move all variable terms to the left side and all constant terms to the right side and then combine like terms; Divide equation sides by variable coefficient; Divide equation sides by common divider of all terms; Multiply both sides of the equation by common denominator.*

After equality/equation properties are given, the following linear equation solving algorithm is presented in all (Estonian and English) textbooks:

1. Use the multiplication property to remove fractions if present.
2. Simplify the left and right sides of the equation by removing grouping symbols and combining like terms.
3. Use the equality properties to get all variable terms on one side (usually the left) and all constant terms on the other side (usually the right).
4. Combine like terms.
5. Isolate the variable (with a coefficient of 1), using the division or multiplication property of equality.

Examples that equation can have exactly one solution, no solution or infinitely many solutions (any number is solution) are presented.

At the end of this topic the solving of linear equation is practiced.

3.1.2 Linear inequalities

The scheme of learning linear inequalities is very similar to learning linear equations. First of all, the definitions of inequality and its solutions are given. Then the properties of inequality are listed. And finally the inequality solving strategy is presented.

The definitions of inequality and its solutions presented in the textbooks are the following: A statement between two expressions (numbers) separated by a sign of inequality ($<$, $>$, \leq , \geq) is called inequality; A solution of an inequality (in a single variable) is a number that, when substituted in the inequality, yields the true statement. The definition of numerical inequality is presented as well: If inequality sides consist of numbers only, then the inequality is called numerical inequality. For each numerical inequality we can say whether it is true or false.

The problems for these definitions are: *Fill in $<$ or $>$; Indicate whether numerical inequality true or false; Check whether the number is a solution of the given inequality.*

The Estonian textbooks describe inequality properties using words, not symbols:

1. Adding the same quantity to both sides of an inequality or subtracting the same quantity from both sides of an inequality, the inequality sign is preserved.
2. Multiplying or dividing both sides of an inequality by the same positive number, the inequality sign is preserved.
3. Multiplying or dividing both sides of an inequality by the same negative number, the inequality sign is reversed.
4. Reversing the sides of an inequality, the inequality sign is reversed.

English textbooks use symbols for inequality properties (the properties are stated using “less than” ($<$) symbol, but they also hold for the other three inequality symbols, i.e., if each inequality sign is reversed or if $<$ is replaced with \leq and $>$ is replaced with \geq):

For a , b , and c any real numbers:

1. If $a < b$, then $a + c < b + c$ – addition property;
2. If $a < b$, then $a - c < b - c$ – subtraction property;
3. If $a < b$ and c is positive, then $ca < cb$ – multiplication property;
4. If $a < b$ and c is negative, then $ca > cb$ – multiplication property;
5. If $a < b$ and c is positive, then $a : c < b : c$ – division property;
6. If $a < b$ and c is negative, then $a : c > b : c$ – division property.

From properties 3 – 6 it is derived that the order of the inequality reverses if we multiply or divide both sides of an inequality statement by a negative number.

The properties are not very much practiced, but we found some problems for them: *Add number to sides of inequality*; *Multiply both sides of inequality by given number*; *Divide both sides of inequality by given number*.

After properties are explained, the solving of inequality takes place. The strategy for solving inequality is not written out entirely; only the following instructions were found: Follow the same steps used to solve a first-degree equation – using, of course, the addition and multiplication properties for inequalities; Inequality solving is very similar to equation solving, only with one essential exception: Multiplying or dividing both sides of an inequality by the same negative number, the inequality sign should be reversed. In other textbook two exceptions are listed, the first one is the same and the second states: Reversing the sides of an inequality, the inequality sign should be reversed.

At the end of this topic the problems *Solve inequality* are practiced.

3.1.3 Systems of linear equations

The theme *System of linear equations* in Estonian textbooks (Lepik et al., 2000; Pais, 1999; Veelmaa, 2000) begins with a definition of linear equations in two variables: Any equation that can be written in the form $ax + by = c$, where x and y are variables and a , b , and c are constants (a and b are not both 0), is called a linear equation in two variables. In order to solve this type of equations, a technique of expression of variable through other variable is explained: move other variable to the other side of the equal sign away from the variable you are solving for, so that the one variable you are solving for stands alone. Exercises *Express variable* are solved in this subtopic.

Further, the definition of a system of two linear equations in two variables or linear system (standard form) is given:

$$\begin{cases} ax + by = m \\ cx + dy = n \end{cases}$$

where a , b , c , d , m , and n are constants, x and y are variables, a , b , c , d are not all 0.

The English textbooks do not use the system sign ($\{$) and the system is expressed as follows:

$$\begin{aligned} ax + by &= m \\ cx + dy &= n. \end{aligned}$$

The definition of solution is also presented: The ordered pair (x, y) is called a solution of the system if (x, y) is a solution of (or satisfies) both equations of the system.

Next, the textbooks describe how to solve systems: solving a linear system means finding all the ordered pairs of real numbers that satisfy both equations at the same time. There are several methods of solving systems of this type: solution by graphing (skipped), solution by substitution and solution by elimination using addition.

Method *Solution by substitution* is described as follows:

1. Choose one of the two equations in a system and solve for one variable in terms of the other. (Choose an equation that avoids getting involved with fractions, if possible.)
2. Then substitute the result into the other equation and solve the resulting linear equation in one variable.
3. Now substitute this result back into the expression found in step 1 (or into one of the original equations) to find the second variable.

The method *Solution by elimination using addition* can be applied in systems of equations where the coefficients of terms containing the same variable are opposites. An extension of the elimination method is to multiply one or both of the equations in a system by some number so that adding eliminates a variable.

The English textbooks describe the method *Solution by elimination using addition* as the replacement of systems of equations with simpler equivalent systems (by performing appropriate operations) until a system with an obvious solution is obtained.

A system of linear equations is transformed into an equivalent system if:

1. Two equations are interchanged.
2. An equation is multiplied by a nonzero constant.
3. A constant multiple of another equation is added to a given equation.

After each method is presented, problems suitable for practicing this method are solved: *Solve by substitution*; *Solve by elimination using addition*; *Solve using any method*.

Most of the Estonian textbooks present only systems that have exactly one solution; this is why T-algebra also accepts only such systems.

Almost all mentioned problem types from the domain of linear equations, linear inequalities and systems of linear equations are realized in the program. In addition, some problem types were added after consulting with Estonian mathematics teachers (for example, types *Reverse sides of inequality*, *Subtract number from sides of inequality*, *Multiply both sides of inequality by common denominator*). For a full list of problem types realized in T-algebra, see Section 3.3 and Appendix C.

3.2 Designed rules in T-algebra

At the first stage of each solution step in T-algebra, the student has to choose the rule that she/he is going to apply. She/he would make the same decision also when using paper and pencil, but in this case she/he would usually not write this decision in the solution. When the teacher checks the solution, she/he has to understand, which rule the student wanted to apply. It is difficult for the program to understand what the student wanted to do if it does not have some additional information. When the program has the information on which rule is applied, it is able to check a number of different attributes. Firstly, such information enables the program to estimate, whether the student knows the algorithm used for solving this type of problems, i.e., to determine the student's skill of choosing the correct rule. The second advantage given to the program by this information is that it can check more efficiently, whether the student's actions on the next stages of the step are correct: e.g., did the student mark the parts of the expression that are suitable for the selected rule, did she/he enter correct parts in the resulting expression, etc.

The textbook algorithms were followed as closely as possible in the design of the rules. Much support in constructing the rules was provided by school math teachers and authors of textbooks. We have tried to make the student's approach to solving the problems within the program parallel to the approach

the student would take solving the problem on paper. We hope that such work in the program will help the student to develop skills, which will carry over to the work on paper.

The designed set of rules is complete, i.e., all exercises in these particular fields are solvable with these rules. The rules in the program correspond to the steps of school solution algorithms. The set of rules for every problem type consists of the new rules for the algorithm, which is being learned, and of the simplification and computation rules learned before. Let us consider the set of rules for linear equation/inequality solving. First of all, the student can use specific rules for equations/inequalities (these rules are used only in the exercises of that field): *Reverse sides*, *Move terms to other side*, *Add to/Subtract from both sides*, *Multiply/Divide both sides*. These spring from the three properties of equation described in the Estonian schoolbooks and one derivation from these properties (see previous section). In addition, it is possible to use rules, which are algorithm steps, but are not specific to linear equations: *Combine like terms*, *Clear parentheses*, *Open parentheses*. Next, some rules for arithmetic operations and manipulation with fractions were added: *Add/Subtract numbers*, *Multiply/Divide numbers*, *Extend*, *Reduce*, *Improper fraction to mixed number*, *Mixed number to improper fraction*, *Common fraction to decimal fraction* and *Decimal fraction to common fraction*. Finally, the student can use rules for simplification, such as *Add/Subtract 0*, *Multiply/Divide 0*, *Multiply by 1*, *Divide by 1*, *Remove denominator 1*, etc.

The designed set of rules is small enough to be displayed in the menu of possible operations at all times (see Figure 2.1), and it gives the possibility to diagnose whether the student knows which step of algorithm to perform at the moment, which rule to select for this step, which rule is applicable to this equation (part of equation). In other environments the set of rules, which allows solving the same equations as in T-algebra, is much larger. For example, MathXpert (Beeson, 1998) has 11 specific rules for linear equations to accomplish the same tasks as our 4 rules. These 11 rules can be easily reduced to four by combining them together. For example, the MathXpert rules *multiply both sides by ?*, *divide both sides by ?* and *change signs of both sides* (which means multiplication or division of both sides by -1) are included in one T-algebra rule *Multiply/Divide both sides*. On the one hand, with more rules the student can indicate more precisely what she/he wants to do, for example, move terms only from left to right or from right to left (MathXpert rules: *transfer ? left to right* and *transfer ? right to left*) or change the signs of both sides (it is a question, whether the student knows that she/he should multiply both sides by -1 to change the signs), but on the other hand, with so many precise rules she/he cannot act as solving on paper, for example, move terms from both sides simultaneously as she/he would do on paper. The large number of rules in MathXpert is necessary due to the opportunity to mark only one part, so the student can not mark one term on the left and one term on the right side for

moving to other side. T-algebra rule *Move terms to other side* allows moving terms only from left to right, only from right to left, but also from both sides simultaneously, because in T-algebra it is possible to mark an unlimited number of terms of an expression and these terms can be located far from each other (and be separated by other terms). In addition, with so many rules in MathXpert the student cannot see all the rules at all times and cannot select an unsuitable rule, because first the student selects some part of the expression and only then the program offers "... those operations that make sense for what you've selected" (Walden, 1997, p. 35).

Some environments (like AlgeBrain (Alpert et al., 1999), Cognitive Tutor: Algebra 1 (Cognitive Tutor by Carnegie Learning, Inc)) offer operations only for transforming the equation using properties of equality: addition, subtraction, multiplication and division. There are no operations for changing the sides and moving terms from one side to other.

We have attempted to make the rules polymorphic so that one and the same rule could be applied to several kinds of objects. For example, the rule *Substitute variable* can be used for checking the solution of linear equation (substituting variable by given number) or for solving a system of linear equations by substitution (substituting variable with expression expressed from other equation). This gives the student the opportunity to apply a once learnt rule in different expressions and even in different kinds of exercises.

Designing the rules, we have taken into account the results of researches on students' mistakes made on paper (Hall, 2002; Sleeman, 1984; Issakova, 2005), and have attempted to leave an opportunity for the student to make the same mistakes in T-algebra, but to provide the program with information about the intentions of the student for the purposes of error diagnosis. We have also tried to make the rules interface as transparent as possible to be sure that mistakes made by the student are caused by misconceptions, not by poor interface design.

After a stage of a solution step is confirmed, T-algebra performs different checks. Possibilities to make mistakes and checking principles were described in Chapter 2 Section 2.4. Let us reiterate the checking principles for the second and third stages of a solution step and distinguish common checks that do not depend on the rule. After the second stage the program checks:

- performance of marking (whether some parts are marked if needed);
- syntactical correctness of marked parts;
- number of marked parts (only one needed, at least two needed, *etc.*, described for every rule separately);
- form of marked parts (may differ depending on the rule, described for every rule separately);
- in some cases, the position of marked parts (described for every rule separately).

After the third stage of a step, the checks depend on the input mode and on the rule. Let me describe common checks for every input mode. I will describe other checks, which depend on the rule, when we come to the description of rules. Checks are performed in the order they are described here and each subsequent check takes place only if all previous checks are successfully passed.

In free input mode T-algebra checks:

- performance of input of the result (the boxes are not empty);
- the syntactical correctness of entered parts;
- inequality mark (in the case of inequality);
- rule specific checks, which are described for every rule separately;
- the equivalence of the entered parts to the parts calculated by the computer (the current program version only alerts about non-equivalence and does not perform more checks in the free input mode).

In structured input mode T-algebra checks:

- performance of input of the result (the boxes are not empty);
- syntactical correctness of entered parts;
- inequality mark (in the case of inequality);
- form of every part (may differ depending on rule, described for every rule separately);
- the equivalence of the entered parts to the parts calculated by the computer;
- every component of term (mark, coefficient, variable).

In partial input mode T-algebra checks:

- performance of input of the result (the boxes are not empty);
- syntactical correctness of entered parts;
- inequality mark (in the case of inequality);
- the equivalence of the entered parts to the parts calculated by the computer.

In partial input mode, the form of every part is not checked because it is correct by design, i.e., the student can enter into the box only constrained parts, for example signs or numbers, which have the correct form. Every component of term in partial input mode is checked by the equivalence of the entered parts to the parts calculated by the computer, because entered parts are components of term, not the whole term.

Let us take a closer look at the specific rules for linear equation/inequality and system of linear equations in T-algebra. For every rule I give an overview of its applications (where and for what purpose this rule can be applied) and expressions this rule is applicable to (and constraints for expression, if any). Then I describe what parts of the expression the student has to mark in order to

apply this rule. I also present the input of additional information and input of intermediate result if these stages are included in the application of the rule. Finally, I write out the input of the result in three different input modes describing only rule specific checks (common checks are described above). I also write out instructions for every stage, which the student sees on the program window. For one rule (first rule *Multiply/Divide both sides*) I also present all error messages that the program shows to the student in case of a mistake.

3.2.1 Rule *Multiply/Divide both sides*

Applications:

- the first step in linear equation/inequality solving algorithm: to remove fractions from equation;
- the last step in linear equation/inequality solving algorithm: to isolate the variable (with a coefficient of 1);
- multiply both sides of inequality by given number (problem type *Multiply both sides of inequality by given number*);
- divide both sides of inequality by given number (problem type *Divide both sides of inequality by given number*);
- multiply both sides of one equation from system of linear equations for solving by elimination using addition.

Expression: equation, inequality, equation from system of equations.

Constraints for expression:

- parentheses should be opened/cleared before multiplication/division;
- fractions, if present, should be with integer denominator;
- multiplication sign should not be presented (products should be calculated);
- mixed numbers should be transformed to improper fractions (mixed numbers can be presented only if multiplication/division with -1 (or 1) takes place) (the reason see in description of input of intermediate result).

Instruction for marking: Mark an equation/inequality for multiplying/dividing the sides.

Marking: In the case of solving one equation/inequality, it is not necessary to mark anything in T-algebra; this rule is applicable to the whole equation/inequality (if expression is suitable). But the possibility to mark is preserved. If something is marked, the program checks whether the whole equation is selected. In the case of a system of linear equations, the one equation should be marked.

Error messages after marking:

- Selected rule cannot be applied to this expression;
- Exactly one equation or inequality should be selected;
- Before multiplication/division, parentheses should be opened;
- This rule cannot be applied to selected term(s);
- Transform all mixed numbers to improper fractions before multiplication/division.

Instruction for input of additional information: Enter action sign and number.

Input of additional information: The program asks to enter multiplication or division sign and a number (common denominator, common factor, coefficient of variable term) in a separate window (Figure 3.1). At the first step of the algorithm the program allows multiplying the equation/inequality only by common denominator of all fractions (left part of Figure 3.1). The common denominator should not be the least, but it should be positive. If equation does not contain fractions, the program allows multiplication with an arbitrary number. If the student wants to divide the equation/inequality, which contains only one variable term on the left side of the equation/inequality and one constant term on the right, then the program allows dividing only by the coefficient of variable term or its factor (right part of Figure 3.1). If equation/inequality contains more terms, the program allows division only by common factor of all terms (common factor should not be the greatest). In the case of problem types *Multiply (Divide) both sides of inequality by given number* the program allows to multiply/divide only by given number.

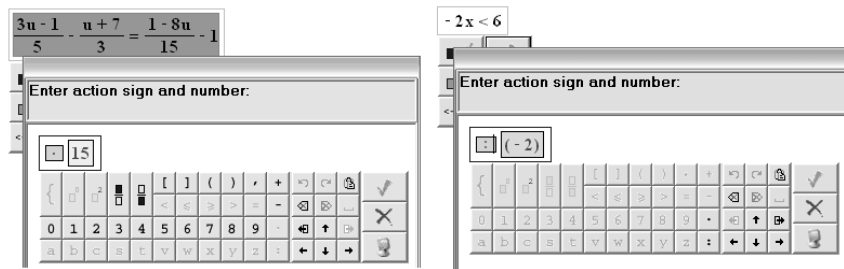


Figure 3.1. Input of additional information applying rule *Multiply/Divide both sides*

Error messages after input of additional information:

- Action sign and/or number are not entered;
- Multiplication/division sign is missing;
- Incorrect sign;
- Parentheses are missed;
- Error in number;
- It is possible to multiply/divide only by number;

- It is not possible to divide by zero;
- Multiplication by zero can produce additional solutions;
- Equation/inequality, which contains mixed numbers, can be multiplied only by -1;
- Number does not match the text of problem;
- Sign does not match the text of problem;
- Only multiplication/division by -1 is possible in this problem;
- Only division by -1 is possible because of appearance of fractions;
- Common denominator cannot be negative;
- Common denominator cannot be fractional;
- The number for multiplication should be common denominator;
- It is possible to divide by coefficient of variable, not multiply;
- Factor cannot be fraction;
- Divider should be coefficient of unknown or its factor;
- It is possible to multiply by common denominator, not divide;
- Divider should be mixed number (before variable);
- Divider should be fraction (before variable) or factor of its numerator;
- Common factor cannot be fractional;
- Divider should be common factor of all members.

Instruction for input of intermediate result: Enter extenders.

Input of intermediate result: If the student wants to multiply an equation that contains fractions, then input of ‘extenders’ is required (Figure 3.2). Extenders are numbers by which both numerator and denominator of the fraction should be multiplied to convert the denominator to the common denominator (used in Estonian schoolbooks). Mixed numbers should be transformed to improper fractions beforehand, because otherwise the extenders should be written separately for integer part and for fraction part of mixed number, which is not used at schools.

$$\frac{3u - 1}{5} - \frac{u + 7}{3} = \frac{1 - 8u}{15} - 1$$

Multiply/Divide both sides

$$\frac{3u - 1}{5} \cdot \boxed{3} - \frac{u + 7}{3} \cdot \boxed{5} = \frac{1 - 8u}{15} \cdot \boxed{1} - 1 \cdot \boxed{15} \quad | \cdot 15$$

The calculator interface below the equation includes a grid of buttons for mathematical symbols and operations:

{	□	□ ²	□	[]	()	·	+	←	→	⊗	✓
<	≤	≥	>	=	-	⊗	⊗	⊗	⊗	⊗	⊗
0	1	2	3	4	5	6	7	8	9	←	↑
a	b	c	s	t	v	w	x	y	z	:	←
											→

Figure 3.2. Input of intermediate result applying rule *Multiply/Divide both sides*

Error messages after input of intermediate result:

- Extender is not entered;
- Error in entered expression;
- Error in calculation of extender.

Instruction for input of result (free input mode): Enter equation/inequality with multiplied/divided sides.

Input of result (free input mode): In the case of equation, only equality sign and two boxes appear in the next line. The student should enter the whole multiplied left side into the first box and the right side into the second box (left part of Figure 3.3). In the case of inequality, three boxes are given: one for the left side, one for inequality sign and one for the right side (right part of Figure 3.3). After the input is confirmed, the program performs common checks described above. Before last control (the equivalence of the entered parts to the parts calculated by the computer) the program checks whether the entered parts are equivalent to the parts calculated by the computer with opposite marks (the marks are changed during multiplying/dividing with positive number or the marks are not changed during multiplying/dividing with negative number). From the controls to be performed it is clear that in free input mode the student can multiply/divide the sides, but she/he can also just write out the right multiplication/division like on Figure 3.3. If the student asks for help, the program will put the multiplied sides (and right inequality sign) to the boxes.

The figure illustrates the input process for two mathematical problems. On the left, an equation $\frac{3u-1}{5} - \frac{u+7}{3} = \frac{1-8u}{15} - 1$ is shown. Below it, the instruction 'Multiply/Divide both sides' is followed by the equation with boxes for input: $\frac{3u-1}{5} - \frac{u+7}{3} = \frac{1-8u}{15} - 1 \cdot 15$. A second 'Multiply/Divide both sides' instruction leads to the final input: $3(3u-1) - 5(u+7) = 1-8u-15$. On the right, an inequality $-2x < 6$ is shown. Below it, the instruction 'Multiply/Divide both sides' is followed by the inequality with boxes: $-2x < 6 \quad | : (-2)$. A second 'Multiply/Divide both sides' instruction leads to the final input: $x > -3$. At the bottom, a calculator interface is shown with various mathematical symbols and numbers.

Figure 3.3. Input of result (free input mode) applying rule *Multiply/Divide both sides*

Error messages after input of result (free input mode):

- Result cannot be empty;
- One side of equation/inequality is missed;
- Inequality sign is missed;
- Error in expression;
- Error during entering inequality sign;
- Inequality sign changes by multiplication/division with negative number;

- Inequality sign does not change by multiplication/division with positive number;
- Incorrect inequality sign;
- Signs of terms change by multiplication/division with negative number;
- Signs of terms do not change by multiplication/division with positive number;
- Error in multiplication/division of sides.

Instruction for input of result (structured input mode): Enter equation/inequality with multiplied/divided sides.

Input of result (structured input mode): In the case of structured input, equality sign (or box for inequality sign like in the case of free input mode) and a number of boxes appear in the next line (Figure 3.4). There are two kinds of boxes: small boxes are for input of signs + and −, larger boxes are for entering numbers and variables. The number of boxes corresponds to the number of terms in the result. The terms can be placed in the boxes in arbitrary order within one side of equation (left part of Figure 3.4). The program performs common checks. The form of every entered part should be monomial and this is controlled by performing a rule specific check (form of every part). If the student asks for help, the program will put the correct multiplied terms (and correct inequality sign) in the boxes.

Figure 3.4. Input of result (structured input mode) applying rule *Multiply/Divide both sides*

Error messages after input of result (structured input mode):

- Inequality sign is missed;
- Action sign is not entered;
- Empty left side;
- Empty right side;
- Error in term;
- Error in sign;

- Error during entering inequality sign;
- Inequality sign changes by multiplication/division with negative number;
- Inequality sign does not change by multiplication/division with positive number;
- Incorrect inequality sign;
- Error in form of term;
- Incorrect sign;
- Error in calculation;
- Incorrect term;
- Missing members on left side;
- Missing members on right side.

Instruction for input of result (partial input mode): Enter equation/inequality with multiplied/divided sides.

Input of result (partial input mode): In the case of partial input, the same number of boxes is given like in the case of structured input mode, but less data should be entered. Variables are already displayed and the user should enter only the signs and numbers (Figure 3.5). The orders of terms can be changed only if variable part is the same. The program performs the same controls like in the case of structured input mode.

Figure 3.5. Input of result (partial input mode) applying rule *Multiply/Divide both sides*

Error messages after input of result (partial input mode):

- Inequality sign is missed;
- Action sign is not entered;
- Empty left side;
- Empty right side;
- Error in term;
- Error in sign;
- Error during entering inequality sign;

- Inequality sign changes by multiplication/division with negative number;
- Inequality sign does not change by multiplication/division with positive number;
- Incorrect inequality sign;
- Error in form of term;
- Incorrect sign;
- Error in calculation;
- Incorrect term;
- Missing members on left side;
- Missing members on right side.

3.2.2 Rule *Move terms to other side*

Applications: move the parts containing a variable to one side of the equation/inequality (usually the left according to the school algorithm) and constant parts to the other side by reversing the signs of all moved parts (additive moving).

Expression: equation, inequality, equation from system of equations.

Constraints for expression: none.

Instruction for marking: Mark in one equation/inequality the terms for moving to other side.

Marking: In order to apply this rule, the student has to mark the terms that she/he wants to move to the other side (Figure 3.6). Moving can take place from both sides simultaneously. Not all terms that should be moved according to the algorithm should be moved in one step. The program checks at the confirmation of marking, whether at least one part is selected. After that the program checks whether the form of selected parts is suitable for application of this rule (must be monomials) and whether the location of selected parts is suitable (must be member of sum, which is not part of product, parentheses etc.). In the case of system of linear equations the program also checks whether all the selected parts belong to one equation. When inappropriate parts have been marked, the program displays a respective error message.

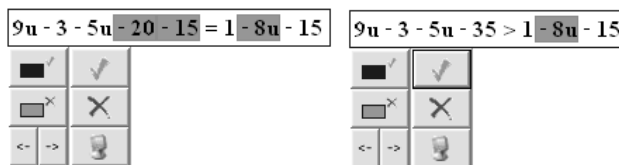


Figure 3.6. Marking stage applying rule *Move terms to other side*

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter the moved terms.

Input of result (free input mode): If the marking was correct, the equation/inequality with boxes is written onto the next line. In the case of free input, there is now one box on each side of the equation/inequality (Figure 3.7). The student should enter the moved terms in the appropriate box. Two boxes are displayed to leave an opportunity for the student to err. In the case of inequality the third box for inequality sign is given in addition to two boxes. Free input mode is similar to working on paper: the program does not prompt anything. The program does not prompt on which side the moved term should be and how many terms are moved to the other side. After the input is confirmed, the program performs common checks. Additionally, T-algebra checks whether the entered part begins with right sign (in some cases the position of the operands causes additional requirements and check of equivalence of entered part is not sufficient, for example, in the case on Figure 3.7 the student should certainly enter $+8u$, not $8u$). If the student asks for help, the program will put the moved terms (and right inequality sign) in the boxes.

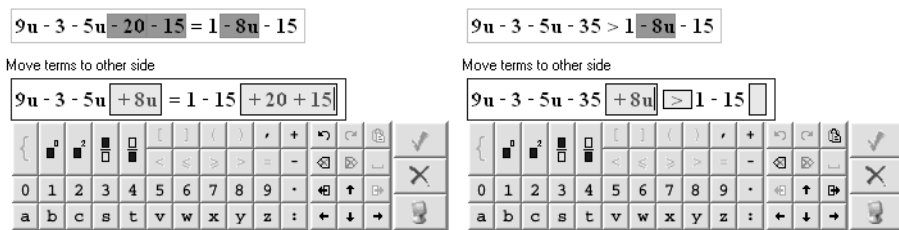


Figure 3.7. Input of result (free input mode) applying rule *Move terms to other side*

Instruction for input of result (structured input mode): Enter the moved terms.

Input of result (structured input mode): In the case of structured input, the unchanged part and a number of boxes appear in the next line (Figure 3.8). In the case of equation there are two kinds of boxes: small boxes are for input of signs $+$ and $-$, larger boxes are for entering numbers and variables. In the case on inequality the box for inequality sign is also given (right part of Figure 3.8). The boxes appear on the other side of selected parts. The terms can be entered in the boxes in arbitrary order within one side of equation, but terms should be exactly the same, only with changed sign. Structured mode is rule specific: the program helps the user, indicating the structure of the result. The program prompts how many terms are moved and to which side and it is very difficult for the student to forget some terms, because the number of boxes corresponds to the number of moved terms. The program also reminds in a certain way about the sign of the term, which is very important in this operation: a separate box is

given for entering the sign. First, the common checks are executed (every part should be monomial like in the rule *Multiply/Divide both sides*). The program diagnoses the common mistakes, which the student would do on paper: forget some moved term and not change the sign of a moved term, and can give appropriate error message and show the exact position (box) of the incorrect part.

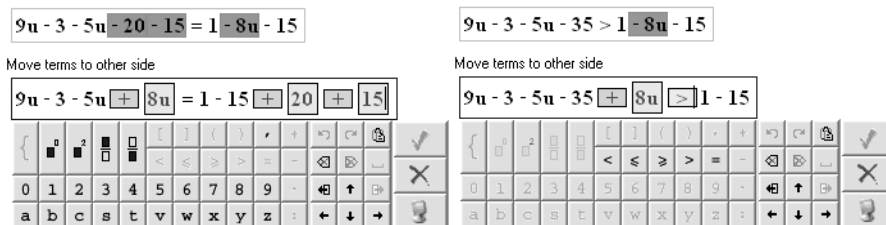


Figure 3.8. Input of result (structured input mode) applying rule *Move terms to other side*

Instruction for input of result (partial input mode): Enter signs.

Input of result (partial input mode): In the case of partial input, the equation/inequality is written onto the next line so that the selected terms have been already moved and only the small boxes have been left blank (Figure 3.9). The student should enter + or – signs in these boxes. The correctness of these signs is checked when the correctness of the step is evaluated. The only mistake that can be made in this mode is not changing the sign. The sign is most important part in applying this rule, and in this mode the program leaves for the student only this part of entering the result. In case of inequality the box for inequality sign is given and the student should enter it as well.

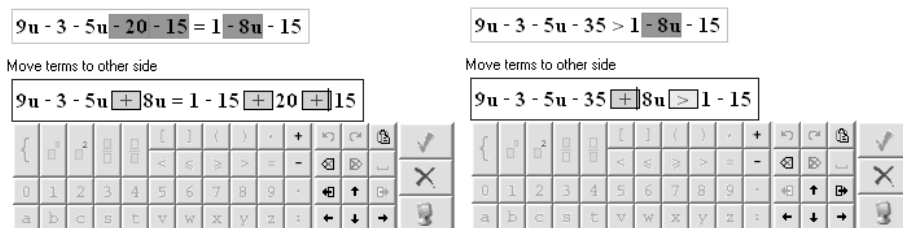


Figure 3.9. Input of result (partial input mode) applying rule *Move terms to other side*

3.2.3 Rule Reverse sides

Applications: reverse equation sides (usually for getting an answer in the following form: *variable = constant*, but also in all other cases); reverse inequality sides.

Expression: equation, inequality, equation from system of equations.

Constraints for expression: none.

Instruction for marking: Mark an equation/inequality for reversing sides.

Marking: To apply this rule, the student should either not mark anything or should mark the whole equation/inequality in the same way as for the rule *Multiply/Divide both sides*. In the case of a system of equations, one equation should be marked.

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter the equation/inequality with reversed sides.

Input of result (free input mode): In free input mode, only one box is given and the student should enter the whole equation/inequality with reversed sides here (Figure 3.10). Of course, she/he can always use Copy-Paste to simplify her/his work. Besides common checks T-algebra controls whether the signs of the terms are the same (changing all signs is a very common mistake made by Estonian students).

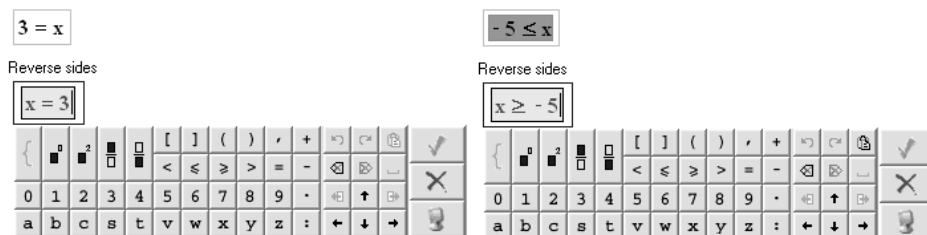


Figure 3.10. Input of result (free input mode) applying rule *Reverse sides*

Instruction for input of result (structured input mode): Enter the equation/inequality with reversed sides.

Input of result (structured input mode): In the case of structured input mode, three boxes are given: one for the left side, one for equation/inequality sign and one for the right side (Figure 3.11). T-algebra performs all common checks and also checks the sign of the terms like during free input mode.

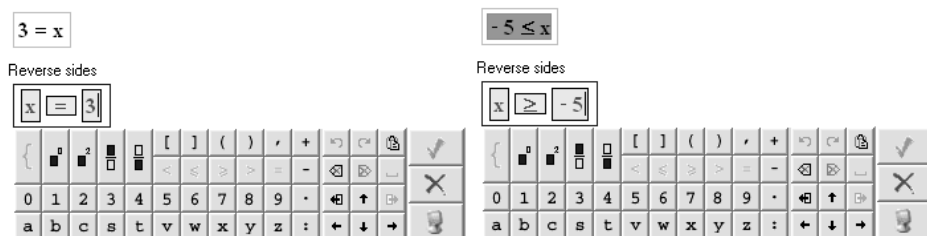


Figure 3.11. Input of result (structured input mode) applying rule *Reverse sides*

Instruction for input of result (partial input mode): Enter inequality sign.

Input of result (partial input mode): If solving an equation, partial input mode is done automatically; the student does not enter anything. In the case of inequality, the inequality sign should be entered by the student (Figure 3.12).

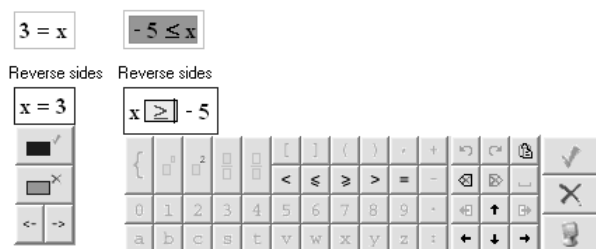


Figure 3.12. Input of result (partial input mode) applying rule *Reverse sides*

3.2.4 Rule *Add to/Subtract from both sides*

Applications: get all variable terms on one side (usually the left) and all constant terms on the other side (usually the right); add given number to both sides of inequality (problem type *Add number to sides of inequality*); subtract given number from both sides of inequality (problem type *Subtract number from sides of inequality*).

Expression: equation, inequality, equation from system of equations.

Constraints for expression: none.

Instruction for marking: Mark equation/inequality for adding/subtracting a term.

Marking: The marking is accomplished in the same way as in the rules *Multiply/Divide both sides* and *Reverse sides*.

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter term for addition/subtraction.

Input of result (free input mode): In the case of equation, the program displays equation with two boxes, one box on each side (left part of Figure 3.13). The student should enter the term with the sign (addition or subtraction). In the case of inequality, the box for inequality sign is added (right part of Figure 3.13). At the end of the input, the program executes common checks and inspects whether the entered parts are equivalent to each other. In the case of problem types *Add number to sides of inequality* and *Subtract number from sides of inequality* the program allows to add/subtract only for the given number. If the student asks for help, in the case of given number addition

to/subtraction from both sides of inequality, the program will put this number with the correct sign (addition or subtraction) to the boxes. In other cases the program will suggest to add/subtract all terms that should be moved to other side (terms with variable to the left and constant terms to the right side) with opposite signs (right part of Figure 3.13).

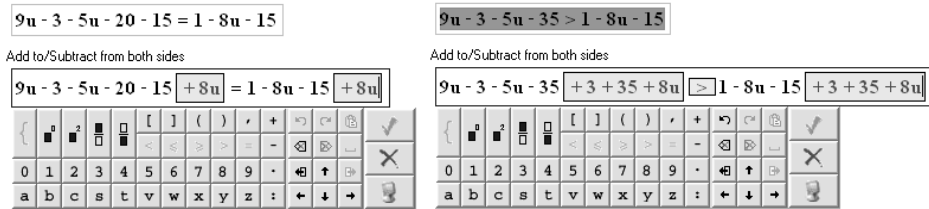


Figure 3.13. Input of result (free input mode) applying rule *Add to/Subtract from both sides*

Instruction for input of result (structured and partial input modes): Enter term for addition/subtraction.

Input of result (structured and partial input modes): The structured input coincides with partial input mode (in essence structured input mode). In these modes the program gives two boxes on each side of equation: one for the sign and second for the term, and a box for inequality sign in the case of inequality (Figure 3.14). The same checks are performed as in free input mode. The help button works in the same way as in the case of free input mode.

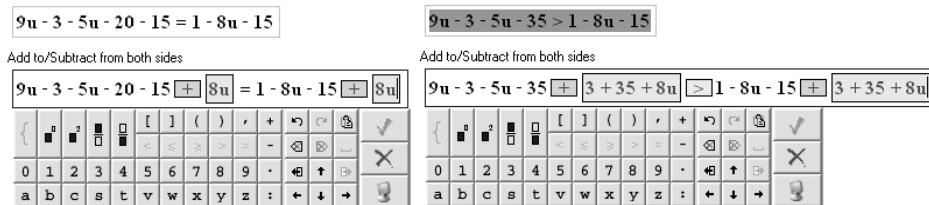


Figure 3.14. Input of result (structured and partial input modes) applying rule *Add to/Subtract from both sides*

3.2.5 Rule *Express variable*

Applications: express variable from equation (usually for solving literal equations, but also used for problem type *Express variable from equation* and for solving a system of equations by substitution).

Expression: equation, equation from system of equations.

Constraints for expression:

- left side of equation (or equation from system of equations) should be one product, where all numbers should be integer and product should not be in the form *mark*variable*;

- right side of equation should not contain the same variable (which will be expressed);
- right side of equation should be simplified, i.e., parentheses should be opened/cleared, like terms should be combined;
- in the case of problem type *Express variable* left side of equation should contain variable from the text of problem (variable to express).

Instruction for marking: Mark an equation to express variable.

Marking: The marking proceeds in the same way as in rules *Multiply/Divide both sides*, *Reverse sides* and *Add to/Subtract from both sides*, i.e., in the case of equation the student can skip marking or can mark, but in the case of system of equations the student has to mark one equation (Figure 3.15).

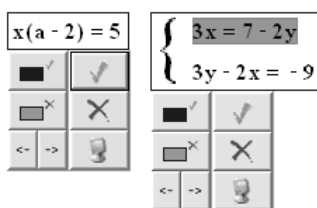


Figure 3.15. Marking stage applying rule *Express variable*

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter equation with expressed variable (Right side should be fraction).

Input of result (free input mode): In the case of free input mode, two boxes and equality sign are given on the next line (Figure 3.16). The first box (on the left) is for input of variable and only the variable should remain on the left side. The second box is designed for input of fraction, which is the result of expression of variable. After input is confirmed, besides common checks T-algebra checks whether the left side consists of just variable and whether the right side consists of correct fraction. The right side of equation must be in the form of fraction even if left side of equation contained a product in the form *number*variable*, because this rule is *Express*, not *Divide*. If the student wants to divide then she/he should select the rule *Multiply/Divide both sides*, i.e., tell the program the correct name of operation. This rule does not allow reducing fraction; for that purpose, T-algebra controls the equivalence of numerator and denominator of fraction to parts calculated by the program separately.

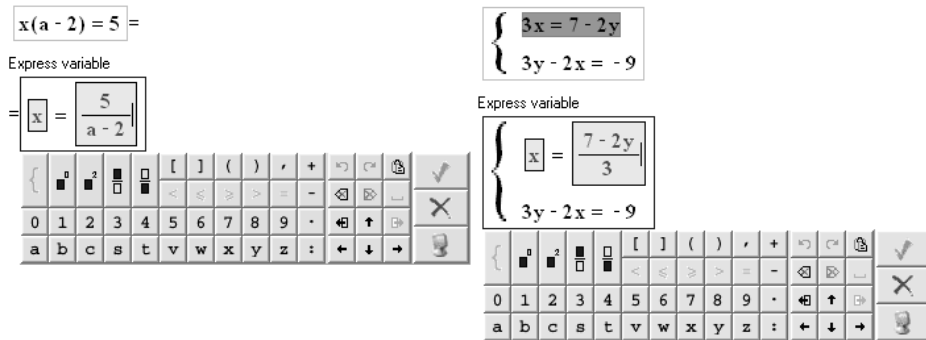


Figure 3.16. Input of result (free input mode) applying rule *Express variable*

Instruction for input of result (structured input mode): Enter equation with expressed variable.

Input of result (structured input mode): In the case of structured input mode, three boxes are given: one for input of the left side of equation (i.e., variable), the second for input of numerator and the last one for input of denominator (Figure 3.17). The program tells the student that she/he should only express, not divide giving the pattern of fraction on the right side. The same checks are performed as in the case of free input mode.

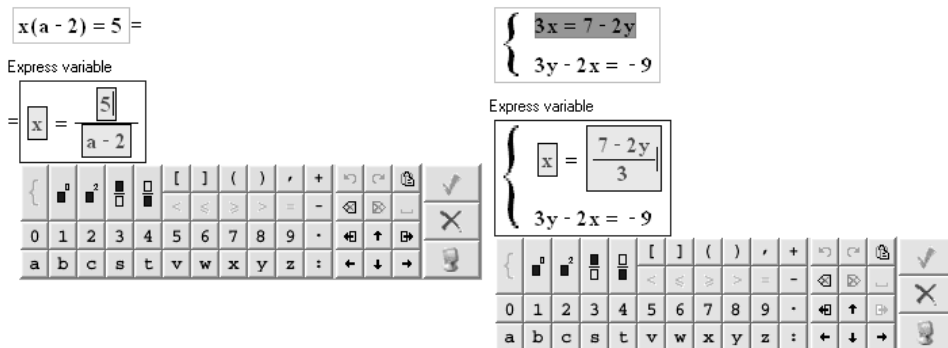


Figure 3.17. Input of result (structured input mode) applying rule *Express variable*

Instruction for input of result (partial input mode): Enter the right side of equation.

Input of result (partial input mode): The partial input mode is a simplified form of the structured input mode, where the variable on the left side is already displayed and the student should enter only the numerator and denominator of fraction on the right side of equation (Figure 3.18).

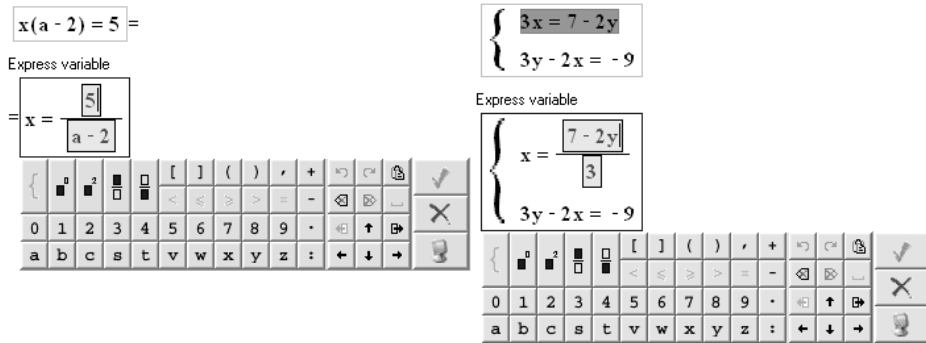


Figure 3.18. Input of result (partial input mode) applying rule *Express variable*

3.2.6 Rule *Substitute variable*

Applications: calculate the value of an expression containing variables (problem type *Calculate the value of literal expressions, if all variable values are given*); check the solutions of a equation; check the validity of inequality; solve a equation system in two variables by substitution; solve a equation system in two variables by elimination using addition after one variable is solved.

Expression: equation, inequality, system of equations, expression with variables.

Constraints for expression: if expression is equation/inequality, then it should be in one variable.

Instruction for marking: Mark the variable to be replaced in this equation/inequality/expression.

Marking: In order to carry out the operation, the student has to mark exactly one occurrence of the variable that is to be replaced. In order to solve an equation system, the variable should be marked in the equation in which it is to be replaced (thus, the replacement will not take place in the other equation) (Figure 3.19). System of equations should contain at least one other equation where the selected variable is expressed. If checking the solution of equation or validity of inequality in one variable, then the student can skip marking, because the expression has only one variable. At the confirmation of the marking, the program checks, whether the selected part is a variable (a respective message is displayed if it is not).

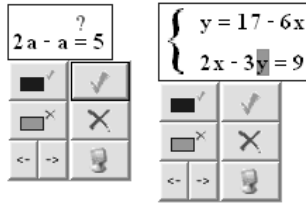


Figure 3.19. Marking stage applying rule *Substitute variable*

Instruction for input of additional information: Enter a number or expression to substitute the variable.

Input of additional information: If the selected part is suitable for the application of the rule, then in the case of structured and partial input modes a separate window will be opened to enter the number or expression that should replace the variable (Figure 3.20). If an expression is entered, the program checks, whether it is a number given in the text of the problem, or in the case of an equation system, whether the entered expression is equivalent to the right side of the other equation, where this variable was expressed through other variables.

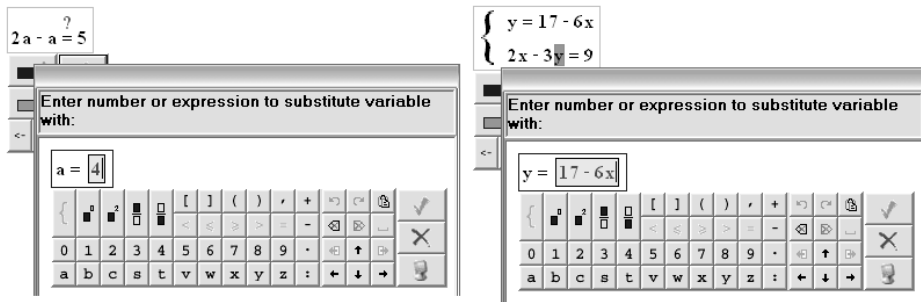


Figure 3.20. Input of additional information applying rule *Substitute variable*

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter a number or expression to substitute the variable and multiplication sign and/or parentheses if needed.

Input of result (free input mode): Input of additional information is skipped in case of free input mode. If the marked part is suitable for the application of this rule, then in the case of free input mode the program copies the expression onto the next line of the main window. All occurrences of the variable (in the case of an equation system, all its occurrences in the equation in which it was marked) are replaced with empty boxes and the student has to enter a number or expression to substitute variable (Figure 3.21). The program allows substituting variable only by given number (in case of equation,

inequality and expression with variables) or by the right side of the other equation, where this variable was expressed (in case of a system of equations). Besides common checks T-algebra also controls that multiplication sign or parentheses are entered if needed, for example, when substituting a by number 4 in expression $2a$ it is necessary to insert a multiplication sign between the numbers 2 and 4 to produce multiplication and not simply the number 24 (left part of Figure 3.21).

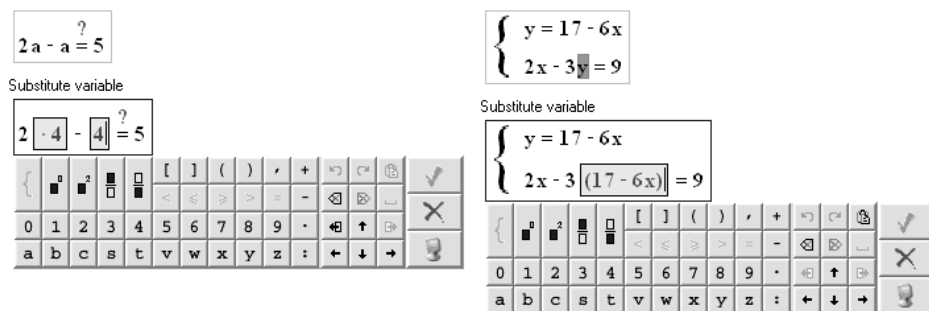


Figure 3.21. Input of result (free input mode) applying rule *Substitute variable*

Instruction for input of result (structured and partial input modes): Enter multiplication sign and/or parentheses if needed.

Input of result (structured and partial input modes): Structured and partial input modes are identical for this rule (in essence partial input mode), because we did not find how to design boxes for both of them for this rule. After the input of additional information has been confirmed, the program copies the expression onto the next line of the main window. All occurrences of the variable (in the case of an equation system, all its occurrences in the equation in which it was marked) are replaced with the entered expression and some boxes are added (Figure 3.22). There are two kinds of boxes: small boxes are for input of multiplication sign and larger boxes are for input of parentheses. Signs and parentheses are checked on the confirmation of input.

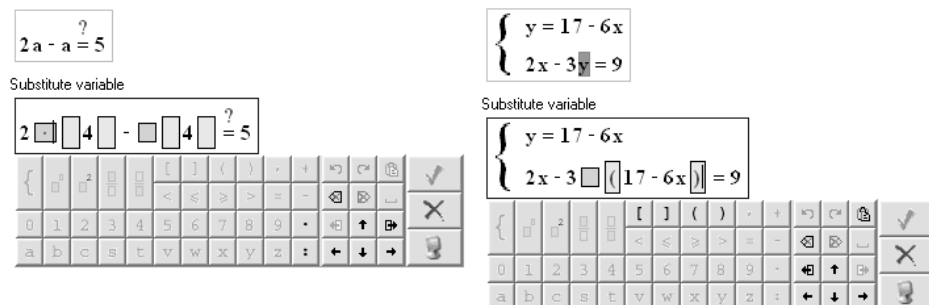


Figure 3.22. Input of result (structured and partial input modes) applying rule *Substitute variable*

3.2.7 Rule *Add equations*

Applications: add two equations in the system of linear equations in two variables for eliminating one variable.

Expression: system of linear equations.

Constraints for expression:

- system of linear equations should be in two variables and consist of exactly two equations;
- linear equations should be simplified, i.e., parentheses should be opened/cleared, fractions should be removed, like terms should be combined, etc.;
- linear equations should be converted into normal form, i.e., linear equation should be in the form: *variable term* \pm *variable term* = *constant term* or in some simplified form (one variable term can be absent);
- one variable should eliminate by addition, i.e., linear equation(s) should be multiplied/divided for getting suitable coefficients;
- system of linear equations where both variables eliminate by one addition does not suit.

Instruction for marking: Mark the equation that will be substituted by the equation resulting from addition.

Marking: In order to apply this rule the student has to mark the equation that will be substituted by the equation resulting from addition (the second equation will be the same).

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter the result of addition.

Input of result (free input mode): In free input mode, two boxes and equality sign are given instead of the selected equation (unselected equation is added to make the system) (Figure 3.23). The first box is designed for input of the left side of equation and the second one for right side. Besides common checks T-algebra checks whether only one monomial is entered into the first box and only one number into the second. T-algebra does not allow just writing out the addition; the program wants a simplified result of addition. Like in case of other rules where the result is exactly one monomial, T-algebra also checks the sign, coefficient and variable separately to produce more precise diagnosis. The same checks are done with the second box.

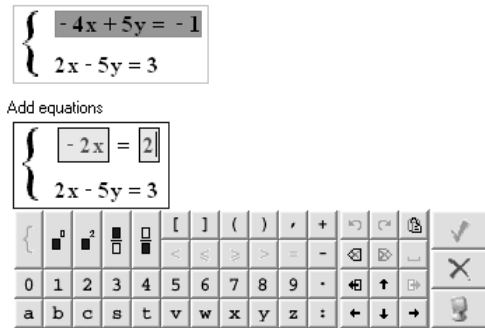


Figure 3.23. Input of result (free input mode) applying rule *Add equations*

Instruction for input of result (structured input mode): Enter the result of addition.

Input of result (structured input mode): In the case of structured input mode, four boxes are given instead of the selected equation (Figure 3.24). Small boxes are displayed for input of signs + or -. The second box on the left side is designed for input of one monomial (the result of addition of the left sides of equations). The second box on the right side is designed for input of one number (the result of addition of right sides of equations). The same checks are performed as in the case of free input mode.

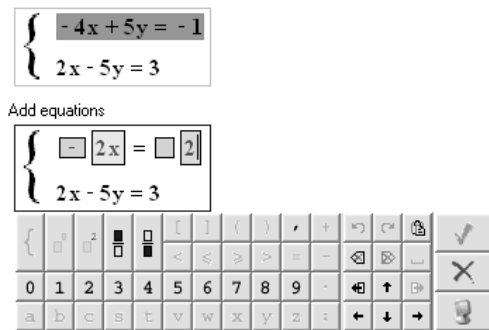


Figure 3.24. Input of result (structured input mode) applying rule *Add equations*

Instruction for input of result (partial input mode): Enter the result of addition.

Input of result (partial input mode): In the case of partial input mode, four boxes are given (Figure 3.25). There are two kinds of boxes: small boxes are for input of sign + or -, the larger boxes are for input of number. The variable is already displayed by the program. The same checks are performed as in the case of free or structured input modes.

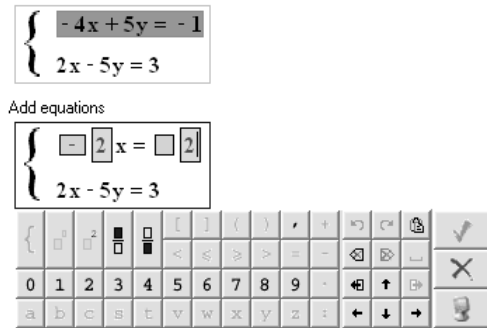


Figure 3.25. Input of result (partial input mode) applying rule *Add equations*

3.2.8 Rule *Multiply fraction with variable by number*

Applications: multiply fraction with variable by number (usually such product appears after substitution of variable in a system of linear equations).

Expression: any expression (including linear equation, linear inequality and system of linear equations) that contains a product of number and fraction with variable.

Constraints for expression (for product):

- denominator of fraction should be integer number;
- numerator of fraction should not contain parentheses;
- numerator of fraction should be monomial or sum of monomials, i.e., should not contain multiplication and division signs;
- fraction should contain variable, other way the student should use the rule *Multiply/Divide numbers*;
- product should consist of exactly one number and exactly one fraction.

Instruction for marking: Mark number and fraction with variable for multiplication.

Marking: The student should mark the number and fraction from one product either separately (right part of Figure 3.26) or together (left part of Figure 3.26). Only one number and one fraction can be multiplied in one step (one product can be simplified). On the confirmation of marking the program checks whether selected parts are appropriate: whether exactly one number and exactly one fraction with variable from one product are selected. Then the program checks whether the selected product and especially the included fraction are suitable for multiplication (see constraints above). If the fraction is placed in parentheses, then the fraction with parentheses should be marked.

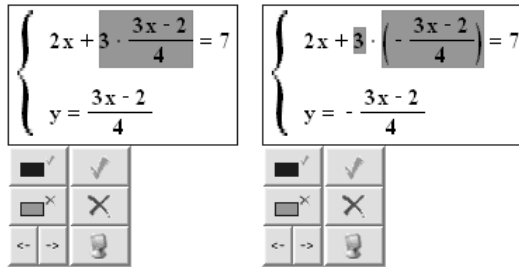


Figure 3.26. Marking stage applying rule *Multiply fraction with variable by number*

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter result of multiplication (fraction).

Input of result (free input mode): In the case of free input mode, the one box for inputting the whole result is given (Figure 3.27). Besides common checks T-algebra also checks whether the result of multiplication begins with the correct mark, if required. In free input mode T-algebra does not insist on the multiplied result, but allows just writing out the multiplication (right part of Figure 3.27).

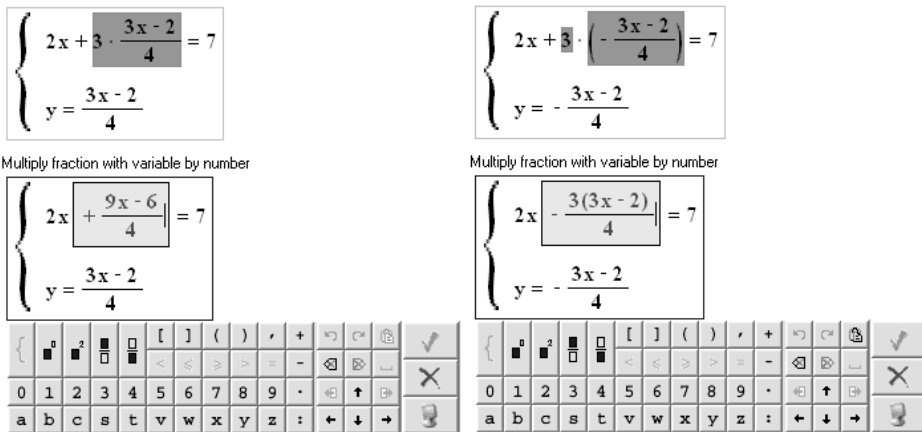


Figure 3.27. Input of result (free input mode) applying rule *Multiply fraction with variable by number*

Instruction for input of result (structured input mode): Enter numerator and denominator of fraction.

Input of result (structured input mode): In structured input mode the program proposes the structure of the result with the number of boxes on the

next line (Figure 3.28). First, the small box for input of sign (+ or -) before fraction is given. Then the pattern of fraction is given: one box for denominator and a number of boxes for terms of numerator. The number of boxes in the numerator corresponds to the number of terms in the result. In place of the numerator there are two kinds of boxes: small boxes are for input of signs + and -, larger boxes are for entering numbers and variables. The terms can be entered in the boxes in arbitrary order within numerator. T-algebra executes common checks and also controls the form of the terms of numerator (they should be monomials). T-algebra allows all variants for signs of terms of numerator, of course depending on the sign entered before fraction.

Multiply fraction with variable by number

$$\begin{cases} 2x + 3 \cdot \frac{3x-2}{4} = 7 \\ y = \frac{3x-2}{4} \end{cases}$$

Multiply fraction with variable by number

$$\begin{cases} 2x + 3 \cdot \left(-\frac{3x-2}{4} \right) = 7 \\ y = -\frac{3x-2}{4} \end{cases}$$

Multiply fraction with variable by number

$$\begin{cases} 2x \left[+ \right] \frac{\left[9x \right] \left[- \right] \left[6 \right]}{\left[4 \right]} = 7 \\ y = \frac{3x-2}{4} \end{cases}$$

Multiply fraction with variable by number

$$\begin{cases} 2x \left[+ \right] \frac{\left[6 \right] \left[- \right] \left[9x \right]}{\left[4 \right]} = 7 \\ y = -\frac{3x-2}{4} \end{cases}$$

Figure 3.28. Input of result (structured input mode) applying rule *Multiply fraction with variable by number*

Instruction for input of result (partial input mode): Enter numerator of fraction.

Input of result (partial input mode): Partial input mode is similar with structured input mode. The difference is that denominator is already displayed and variables in the terms of numerator are given (Figure 3.29). The student should enter the sign before fraction and signs and coefficients of terms of numerator. The orders of terms can be changed only if variable part is the same. The program performs the same controls as in the case of structured input mode.

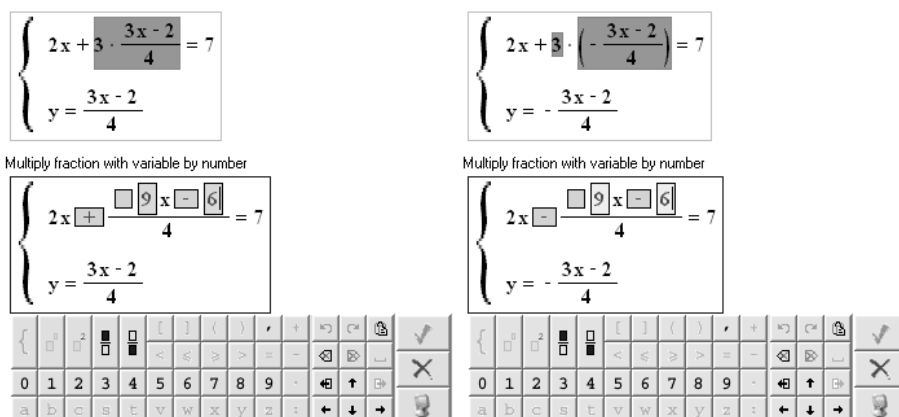


Figure 3.29. Input of result (partial input mode) applying rule *Multiply fraction with variable by number*

3.2.9 Rule *Open parentheses*

This rule is a renamed and modified copy of Dmitri Lepp’s rule *Multiply/Divide polynomial by monomial*. Copying and modification was needed because the students solving linear equations and inequalities in the 7th grade do not know yet about monomials and polynomials; they name this operation *Open parentheses*. The modifications concerned all texts about mistakes (words monomial and polynomial were excluded), structured input mode (addition of terms was skipped, program gives the whole structure of the result), partial input mode (the program does not give the box for input of power of variable) and some small details.

Applications: open parentheses, i.e., multiply number and expression in parentheses.

Expression: any expression (including linear equation, linear inequality and system of linear equations) that contains product of number and expression in parentheses.

Constraints for expression: none.

Instruction for marking: Mark a number and term in parentheses to be multiplied.

Marking: In order to apply this rule the student has to mark the number(s) and expression in parentheses (Figure 3.30). Only one product can be simplified in one step, i.e., the number and expression in parentheses should be from the same product – T-algebra does not allow parallel applications of the rule. Only one expression in parentheses can be opened in one step. If the product contains more than one such expression, the student must select only one. Expression in parentheses should be marked together with parentheses. All this is checked by the program when the marking is confirmed.

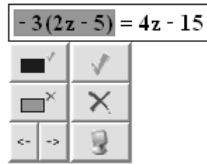


Figure 3.30. Marking stage applying rule *Open parentheses*

Instruction for input of additional information: none.

Input of additional information: none.

Instruction for input of intermediate result: none.

Input of intermediate result: none.

Instruction for input of result (free input mode): Enter result of multiplication.

Input of result (free input mode): In free input mode, the whole result should be entered into a single yellow box (Figure 3.31). As in other rules, where the result of multiplication is entered in one box and multiplied result is the sum, T-algebra allows partial application of the rule. All common checks are performed when the input is confirmed. If the student asks the program for help, T-algebra will put the multiplied result in the box.

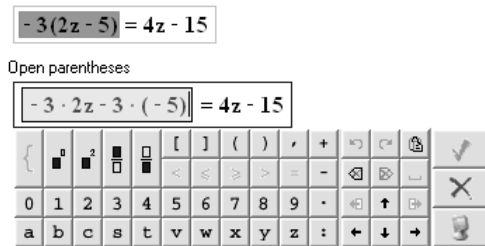


Figure 3.31. Input of result (free input mode) applying rule *Open parentheses*

Instruction for input of result (structured input mode): Enter result of multiplication.

Input of result (structured input mode): In structured input mode the structure of the result is given (Figure 3.32). The student has to fill it with the signs and terms. The order of terms can be changed like in other rules. T-algebra requires from the student exact application of this rule only – no combining of like terms or other simplifications is allowed at this step. Besides common checks the program tries to identify errors only in coefficient, sign or in variable.

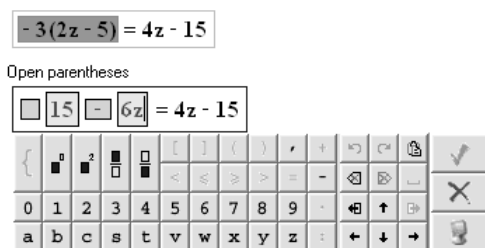


Figure 3.32. Input of result (structured input mode) applying rule *Open parentheses*

Instruction for input of result (partial input mode): Enter missing parts of result.

Input of result (partial input mode): In partial input mode, the student has to fill only gaps – coefficients and signs of the resulting terms (Figure 3.33). In partial input mode checks are the same as in structured mode.

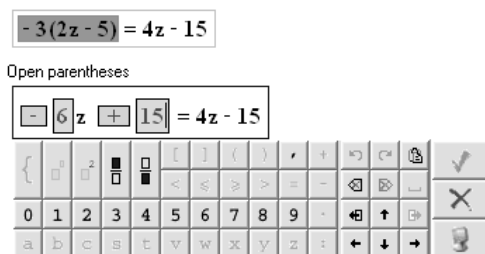


Figure 3.33. Input of result (partial input mode) applying rule *Open parentheses*

3.3 Designed problem types in T-algebra

T-algebra enables the student to solve under the control of the program almost any problem in the given fields (incl. linear equations, linear inequalities and systems of linear equations), provided that the original expression/equation has been given in the problem (entered by the composer). Each type of problems is linked with a specific solving algorithm. T-algebra is able to solve the problem by itself and help the student during solving (give advice) according to this algorithm.

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, in addition to linear equation solving, the section of linear equations in the program enables to practice separate steps of equation solving algorithm: combine like terms; open parentheses; reverse equation sides; 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 common denominator of all terms. The algorithms for solving these problems are very simple; almost all of them

consist only of applying one rule. However, these problem types allow practicing the properties of equality/equation – single steps of equation solving algorithm.

The domain of linear equations, linear inequalities and systems of linear equations contains the following types:

1. problem type *Combine like terms*;
2. problem type *Open parentheses and combine*;
3. problem type *Check the solution of equation*;
4. problem type *Reverse sides of equation*;
5. problem type *Move terms to correct side of equation*;
6. problem type *Move terms to correct side of equation and combine*;
7. problem type *Multiply both sides of equation by common denominator*;
8. problem type *Divide equation sides by variable coefficient or by common divider*;
9. problem type *Solve linear equation*;
10. problem type *Check numerical inequality*;
11. problem type *Check the solution of inequality*;
12. problem type *Reverse sides of inequality*;
13. problem type *Add number to sides of inequality*;
14. problem type *Subtract number from sides of inequality*;
15. problem type *Multiply both sides of inequality by given number*;
16. problem type *Multiply both sides of inequality by common denominator*;
17. problem type *Divide both sides of inequality by given number*;
18. problem type *Solve linear inequality*;
19. problem type *Express variable from equation*;
20. problem type *Solve by substitution*;
21. problem type *Solve by elimination using addition*.

In this section I present some examples of the designed problem types, describing:

- typical texts for this type (one problem type can allow slightly different texts and expressions);
- expression that the program allows for selected problem type;
- constraints for expression, if any;
- what parameters should the composer enter for the program;
- rules that the student can use during solving problems of this type;
- algorithm – ordered list of rules used by T-algebra to solve problems of this type;
- one example of generated solution;
- the solved form of problem what the program will accept;

- possible answers (in some problem types the student should specify some additional properties of the answer after the program checked that the expression is in appropriate solved form; in such problem types I specify what possibilities the program proposes for answer).

All problem types are described in Appendix C using the same structure.

The list of rules for some problem types can be quite long; therefore, the rules are grouped for user convenience. First, the student can use the rules derived from this problem type – steps of solution algorithm and rules for arithmetic operations. Then the rules for manipulation with fractions and simplification rules are given. The last two groups are the same for most problem types and I present them here.

Group of rules for manipulation with fractions:

- Extend common fraction;
- Reduce;
- Decrease integer part;
- Improper fraction to mixed number;
- Mixed number to improper fraction;
- Common fraction to decimal fraction;
- Decimal fraction to common fraction;
- Move minus before fraction.

Group of rules for simplification of expressions with 0 , 1 and redundant pluses:

- Add/Subtract 0 ;
- Multiply/Divide 0 ;
- Multiply by 1 ;
- Divide by 1 ;
- Eliminate fraction with 0 in numerator;
- Eliminate denominator 1 ;
- Remove redundant pluses;
- Raise to power 1 ;
- Raise to power 0 ;
- Raise 1 to power;
- Raise 0 to power.

In most algorithms we use two sub-algorithms for solving “standard sub-problems”. The first one is for trying to apply simplification rules, all 11 rules mentioned above in sequence. The second algorithm is for combining like terms. It was created to simplify the student’s understanding of the solution path. If the student applies the rule *Combine like terms*, the program allows combining different like terms, for example, terms with common fractions and terms with decimal fractions (left part of Figure 3.34) or terms with unlike

fractions, if she/he can add these numbers mentally. But if such solution step will be generated or proposed by the program for the student, then it might be difficult for the student to understand how the result was obtained (how these numbers were added). Therefore, our program generates a longer solution and transforms common fraction to decimal if possible or decimal fraction to common and unlike fractions to similar before combining (right part of Figure 3.34). This longer way was described in a separate algorithm that can be used in any algorithm of any problem type.

$$\begin{array}{l} \boxed{\frac{2}{3}x + 0.3x} = \\ \text{Combine like terms} \\ = \boxed{\frac{29}{30}x} \end{array} \qquad \begin{array}{l} \boxed{\frac{2}{3}x + 0.3x} = \\ \text{Decimal fraction to common} \\ = \boxed{\frac{2}{3}x + \frac{3}{10}x} = \\ \text{Extend common fraction} \\ = \boxed{\frac{2}{3}x + \frac{3}{10}x} = \\ \text{Extend fractions} \\ = \boxed{\frac{20}{30}x + \frac{3}{10}x} = \\ \text{Combine like terms} \\ = \boxed{\frac{29}{30}x} \end{array}$$

Figure 3.34. Combining like terms

This algorithm consists of the following rules:

- Decrease integer part (to get rid of the mixed number with negative numerator, for example $1\frac{-1}{5}$, or to avoid a mixed number where numerator will be negative after evaluation, for example $1\frac{2-3}{5}$);
- Add/Subtract numbers;
- Common fraction to decimal fraction (if coefficients of some like terms are decimal fractions and coefficients of all other like terms, which are common fractions or mixed numbers, can be transformed to terminating decimal fractions);
- Decimal fraction to common fraction (if at least one coefficient of like term, which is common fraction or mixed number, can not be transformed to terminating decimal fraction and coefficients of some like terms are decimal fractions);
- Extend common fraction (for like terms with unlike fractions);
- Combine like terms;
- Reduce;
- Improper fraction to mixed number.

One more remark should be made before proceeding to the description of problem types. Several algorithms present two rules: *Clear parentheses* and *Open parentheses*. Why do we have two rules for removing parentheses? The rule *Open parentheses* was described in the previous section and it can be used for product of number and expression in parentheses. However, in some cases we have just expression in parentheses (for example $(x+3)$) or sign previous to expression in parentheses ($-(x+3)$). The rule *Open parentheses* cannot be used for this type of expression. In the Estonian textbooks, such operations are described with name *Clear parentheses*, the same was done in T-algebra. So the rule *Open parentheses* can be used for product, but the rule *Clear parentheses* can be used for parentheses only or the sign and parentheses. For an example of applications of these rules, see the description of type *Open parentheses and combine* in Appendix C.

3.3.1 Problem type *Move terms to correct side of equation and combine*

Typical texts: move all variable terms to the left side and all constant terms to the right side and then combine like terms.

Expression: equation.

Constraints (for expression):

- equation should contain variable terms on the right side or constant terms on the left side (student should move some terms);
- left side and right side of equation should be monomial or polynomial without multiplication and division signs and without parentheses and brackets;
- if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Move terms to other side;
- Combine like terms;
- Add/Subtract numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Move terms to other side*;
3. Algorithm for combining.

Example of generated solution:

Text of problem: Move all variable terms to the left side and all constant terms to the right side and then combine like terms.

$$14x - 5 = 4 + 9x$$

Move terms to other side

$$14x - 9x = 4 + 5$$

Add/subtract numbers

$$14x - 9x = 9$$

Combine like terms

$$5x = 9$$

Answer:

$$5x = 9$$

Solved form: equation, where all variable terms are moved to the left side and all constant terms to the right side and after that all like terms are combined.

Answer: solved form.

3.3.2 Problem type *Solve linear equation*

Typical texts: solve linear equation.

Expression: equation.

Constraints (for expression): equation should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;

2. Rule *Reverse sides* (if linear equation is in the form: $number = variable$);
3. Rules *Open parentheses* and *Clear parentheses*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Rule *Multiply/Divide both sides* for removing fractions (multiplication);
6. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
7. Algorithm for combining;
8. Rule *Move terms to other side*;
9. Rule *Multiply/Divide both sides* for isolating variable (division).

Example of generated solution:

Text of problem: Solve an equation.

$68 - 3x = 3 + 4(2x - 3)$ <p>Open parentheses</p> $68 - 3x = 3 + 8x - 12$ <p>Add/subtract numbers</p> $68 - 3x = -9 + 8x$ <p>Move terms to other side</p> $-3x - 8x = -9 - 68$ <p>Add/subtract numbers</p> $-3x - 8x = -77$	<p>Combine like terms</p> $-11x = -77$ <p>Multiply/Divide both sides</p> $-11x = -77 \quad : (-11)$ <p>Multiply/Divide both sides</p> $x = 7$ <p>Answer:</p> $x = 7$
---	--

Solved form: $variable = number$ or $number = number$, where

- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);
- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer:

- solved form (i.e., $variable = number$ or $number = number$; for example, $x = 7$);
- there is no solution;
- any number is solution.

3.3.3 Problem type *Check the solution of inequality*

Typical texts: check if number ... is a solution of inequality.

Expression: inequality.

Constraints (for expression): inequality should be in one variable.

Parameters: value (one number) of variable to be checked.

Rules:

- Substitute variable;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Raise number to a power;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Substitute variable*;
3. Rule *Raise number to a power*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Algorithm for combining;
6. Rules *Open parentheses* and *Clear parentheses*;
7. Rule *Mixed number to improper fraction* (if equation is not yet in form *number = number*).

Example of generated solution:

Text of problem: Check if number 3 is a solution of inequality.

$\frac{5x - 4}{5} > \frac{3x}{4}$	Extend common fraction	Improper fraction to mixed number
Substitute variable	$\frac{11}{5} > \frac{9}{4}$	$2\frac{4}{20} > \frac{45}{20}$
$\frac{5 \cdot 3 - 4}{5} > \frac{3 \cdot 3}{4}$	Extend fractions	Improper fraction to mixed number
Multiply/Divide numbers	$\frac{44}{20} > \frac{9}{4}$	$2\frac{4}{20} > 2\frac{5}{20}$
$\frac{15 - 4}{5} > \frac{3 \cdot 3}{4}$	Extend common fraction	Answer:
Multiply/Divide numbers	$\frac{44}{20} > \frac{9}{4}$	<div style="border: 1px solid black; padding: 2px; display: inline-block;">is not a solution</div>
Add/Subtract numbers	Extend fractions	
$\frac{11}{5} > \frac{9}{4}$	$\frac{44}{20} > \frac{45}{20}$	

Solved form: *number* \otimes *number*, where

- \otimes is one of inequality signs $<$, $>$, \leq , \geq ;
- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);

- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer: Number ...

- is a solution;
- is not a solution.

3.3.4 Problem type *Solve by elimination using addition*

Typical texts: solve by elimination using addition.

Expression: system of linear equations.

Constraints (for expression):

- system of linear equations should be in two variables and consist of exactly two equations;
- system should have exactly one solution;
- system should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Substitute variable;
- Add equations;
- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions and rule *Common fraction to division*;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rules *Open parentheses* and *Clear parentheses*;
3. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
4. Rule *Move terms to other side*;
5. Algorithm for combining;
6. Rule *Multiply/Divide both sides*;

7. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
8. Rule *Reverse sides*;
9. Rule *Substitute variable*;
10. Rule *Add equations*;
11. Rule *Multiply/Divide both sides* for getting suitable coefficients (for eliminating one variable by addition);
12. Rule *Common fraction to division*.

Example of generated solution:

Text of problem: Solve by elimination using addition.

$\begin{cases} 3x + 4y = -2 \\ 5x - 6y = -16 \end{cases}$	<p>Add equations</p> $\begin{cases} -38y = -38 \\ 15x - 18y = -48 \end{cases}$	<p>Move terms to other side</p> $\begin{cases} y = 1 \\ 15x = -48 + 18 \end{cases}$
<p>Multiply/Divide both sides</p> $\begin{cases} 3x + 4y = -2 & \cdot (-5) \\ 5x - 6y = -16 \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} -38y = -38 & : (-38) \\ 15x - 18y = -48 \end{cases}$	<p>Add/Subtract numbers</p> $\begin{cases} y = 1 \\ 15x = -30 \end{cases}$
<p>Multiply/Divide both sides</p> $\begin{cases} -15x - 20y = 10 \\ 5x - 6y = -16 \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} y = 1 \\ 15x - 18y = -48 \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} y = 1 \\ 15x = -30 & : 15 \end{cases}$
<p>Multiply/Divide both sides</p> $\begin{cases} -15x - 20y = 10 \\ 5x - 6y = -16 & \cdot 3 \end{cases}$	<p>Substitute variable</p> $\begin{cases} y = 1 \\ 15x - 18 \cdot 1 = -48 \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} y = 1 \\ x = -2 \end{cases}$
<p>Multiply/Divide both sides</p> $\begin{cases} -15x - 20y = 10 \\ 15x - 18y = -48 \end{cases}$	<p>Multiply by 1</p> $\begin{cases} y = 1 \\ 15x - 18 = -48 \end{cases}$	<p>Answer:</p> $\begin{cases} y = 1 \\ x = -2 \end{cases}$

Solved form: $\begin{cases} \text{first variable} = \text{number} \\ \text{second variable} = \text{number} \end{cases}$

Answer: solved form.

4 CONDUCTED EXPERIMENTS

We have conducted different experiments for different purposes while designing and programming T-algebra. I have participated in five of them. These experiments were conducted

1. to identify the 7th grade student mistakes (during solving linear equations) made when working with pencil and paper and their possibility in T-algebra;
2. to validate user interface;
3. to find out the 7th grade student errors made during linear equation solving in T-algebra and compare them with mistakes made on paper (from the first experiment);
4. to learn about the errors made in the solution of linear equations by the first year university students and to compare them with the errors made by the 7th grade students in the same subject (from the third experiment);
5. to answer the question: How does an interactive learning environment affect the students' learning? (to evaluate the created interactive learning environment).

I will describe all five experiments and their results in this Chapter on the basis of published articles (Issakova, 2005; Issakova et al., 2006; Issakova, 2006b; Issakova, 2007a; Issakova, 2007b).

4.1 First experiment

The research took place in Estonian schools in the winter of 2005. For this research mathematics teachers composed seven tests for different topics (numerical expressions, fractions, linear equations, inequalities, systems of linear equations, monomials, polynomials) and for different grades (6th, 7th, and 8th) in two variants. 93 students aged between 13 and 15 years (7th grade) participated in the linear equations test. They had covered the topic of linear equations in autumn 2004 and the material of the test was not new. The students did not know about the test beforehand and had 45 minutes for solving the test.

The same types of problems were chosen for the test as realized in T-algebra. Linear equations test contained 16 problems (composed by mathematics teachers Mart Oja and Maire Oja). There were five types of problems. The problems of variant A were the following:

- reverse equation sides: 1) $3m - 7 = 5 + 2m$; 2) $5x = 8x - 5$;
- divide equation sides by variable coefficient: 1) $7x = 21$; 2) $-0,3y = -1,2$;
3) $-5n = 25$;

- multiply both sides of the equation by common denominator: 1) $\frac{x}{3} = \frac{2}{5}$;
 2) $\frac{y-2}{4} = \frac{2}{3}$; 3) $2 - \frac{n}{3} = \frac{1}{2}$; 4) $\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15}$;
 5) $\frac{x+2}{2} - \frac{2x-3}{3} = \frac{x-1}{14} - \frac{3x+1}{6}$;
- move all variable terms to the left side and all constant terms to the right side and then combine like terms: 1) $3x-4=7x$; 2) $9-2y=5y+3$;
 3) $2m-3+5=2-5m+1+3m$;
- solve an equation: 1) $9x-15=2-8x$; 2) $9-2(3y-1)=3-2y$;
 3) $\frac{3m-1}{2} - \frac{m+3}{3} = \frac{m+5}{4}$.

The problems of variant B were the following:

- reverse equation sides: 1) $19 = 4y + 6$; 2) $5u + 3 = 2u - 9$;
- divide equation sides by variable coefficient: 1) $3y = 36$; 2) $1,4n = -4,2$;
 3) $-7y = -35$;
- multiply both sides of the equation by common denominator:
 1) $\frac{2y}{9} = -\frac{1}{6}$; 2) $\frac{m+2}{6} = \frac{3}{4}$; 3) $\frac{x}{5} + 3 = \frac{7}{10}$; 4) $\frac{2u}{5} - \frac{u-3}{4} = \frac{2u+3}{10}$;
 5) $\frac{n-3}{6} - \frac{2n+1}{2} = \frac{3n-1}{8} + \frac{n+3}{12}$;
- move all variable terms to the left side and all constant terms to the right side and then combine like terms: 1) $9m+12=4$; 2) $7s+12=9s-2$;
 3) $2-5x+3-x=2x-6+x+10$;
- solve an equation: 1) $6-5n=3n+22$; 2) $2x-3(2x+3)=3-8x$;
 3) $\frac{3u-1}{5} - \frac{u+7}{3} = \frac{1-8u}{15} - 1$.

The result of the test confirmed our assumptions that the students make both, specific mistakes, which occur only during linear equation solving (for example, in moving terms to other side), as well as mistakes related to previously studied material (for example, in adding numbers). Let us take a closer look at specific mistakes at two steps of linear equation solution: multiplication of equation sides and moving terms to the other side of equation. The last column shows the number of students who made this mistake. These tables do not reflect whether the student made this mistake more than once. We included only the most common mistakes in these tables. The other mistakes and mistakes at other steps of linear equation solution are described in the third experiment in Section 4.3.

Table 4.1. Mistakes in multiplying the equation sides

No	Nature of mistake	Example of mistake	Number of students
1	Minus sign before fraction is taken into account only at first term	$\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15} \quad \cdot 15$ $20y - 6y + 9 = 4$	51
2	Arithmetic mistake	$\frac{2y}{9} = -\frac{1}{6} \quad \cdot 18 \Rightarrow 4y = -2$	15
3	Wrong extender	$\frac{n-3}{6} - \frac{2n+1}{2} = \frac{3n-1}{8} + \frac{n+3}{12} \quad \cdot 24$	13
4	Extender is multiplied only with first term of numerator	$\frac{y-2}{4} = \frac{2}{3} \quad \cdot 12 \Rightarrow 3y - 2 = 4 \cdot 2$	10
5	Whole number is not multiplied	$\frac{x}{5} + 3 = \frac{7}{10} \quad \cdot 10 \Rightarrow 2x + 3 = 7$	9
6	One term is forgotten (which is not fraction)	$\frac{3u-1}{5} - \frac{u+7}{3} = \frac{1-8u}{15} - \frac{1}{15} \quad \cdot 15$ $9u - 3 - 5u - 35 = 1 - 8u$	6

Table 4.2. Mistakes in moving terms to the other side of equation

No	Nature of mistake	Example of mistake	Number of students
7	Sign is not changed	$8u - 5u + 15 = 4u + 6$ $8u - 5u - 4u = 15 + 6$	24
8	Mistake in passive rewriting (changing sign, forgetting term)	$2m - 3 + 5 = 2 - 5m + 1 + 3m$ $2m + 5m - 3m = 2 - 1 + 3 - 5$	18
9	One removed term is lost	$9 - 6y + 2 = 3 - 2y$ $-6y + 2y = 3 - 9$	3

Let us consider which mistakes can be made in T-algebra. If a mistake can be made, it means that T-algebra can respond to it as well. If the mistake is in the set of standard mistakes, then T-algebra is able to diagnose it and offer advice. If not, then T-algebra tells about the non-equivalence of equations.

In applying the rule *Multiply/Divide both sides* for multiplication the student can make a lot of mistakes, e.g., in finding common denominator and extenders, or in evaluating the result of multiplication. All the mistakes described in Table 4.1 can be made in T-algebra even in the most constrained input mode. However, it is possible that the rule dialogue helps to avoid some mistakes. For example, in the case of input of the result in some modes, the program proposes boxes for every term of equation and the student could not to forget to write some term (mistake No. 6). Maybe this structure of the rule would help to avoid the mistake No. 4 too, because the program offers boxes for every term of numerator.

As a result of the research we composed the following set of error messages in T-algebra (in applying of the rule *Multiply/Divide both sides*):

- Wrong extender (mistake No. 3).
- In multiplying the second term of fraction numerator minus sign before fraction is not taken into account (or Wrong sign) (mistake No. 1).
- Extender is not multiplied with second term of numerator (or Error in term) (mistake No. 4).
- Whole number is not multiplied with extender (or Error in calculation) (mistake No. 5).
- Number of equation terms is changed (term is missed) (mistake No. 6).
- Mistake in evaluating the result (mistake No. 2).

Of course, programming this rule we extended this set (see Section 3.2.1).

In applying the rule *Move terms to other side* the student may err in marking the parts and entering the result. In partial input the student could not forget a term or change the sign of the term, which is not moved to the other side. In this mode it is possible to make only mistake No. 7 from Table 4.2. In free and structured input it is possible to make mistakes No. 7 and No. 9. It turned out that mistakes that occur when the student does not know how to move term to other side can be made, but mistakes that happen in the process of rewriting and are not substantial cannot be made.

Applying the rule *Move terms to other side* it is possible to get the following error messages if abovementioned mistakes are made:

- In moving terms to the other side the sign is not changed (mistake No. 7).
- Number of equation terms is changed (term is missed) (mistake No. 9).

4.2 Second experiment

In the spring of 2005 the same students from the first experiment participated in the trial of T-algebra. T-algebra was in the development phase at that time and, therefore, the objective of this trial was to validate only the user interface of the program from the point of view of its usability. Two topics were chosen for that purpose: operations with fractions and simplification of polynomials (the same topics were covered in paper tests). In this trial, the students were given exactly the same problems as in previously completed tests on paper. In addition, the problem set contained some demonstration examples from other chapters. The trial was conducted in two different classes. A 6th grade class was chosen for the topic of operations with fractions and an 8th grade class for the topic of simplification of polynomials. The students already had sufficient experience with computers (using the keyboard, mouse, Windows), but it was the first time they had seen T-algebra. The students could choose whether they wanted to sit at the computer alone or in pairs. For operations with fractions we had 25

computers occupied by the students and for simplification of polynomials 21 computers were occupied.

The sessions lasted for one hour. During the first five minutes we demonstrated T-algebra and the solution processes in T-algebra and wrote our general dialogue scheme on the blackboard. In the first ten minutes, the students asked questions concerning the use of the computer (keyboard), the use of T-algebra tools (how to mark the objects and what to enter in the boxes), and mathematical questions about the solution steps. After that, questions concerning the use of software disappeared. Questions about mathematics (on operations with fractions and polynomials) continued after the first ten minutes. Questions about which rule to select in the menu continued throughout the trial. At that time the particular problem types were not yet implemented in our program and the menu contained all the rules needed for the actual topic. In most cases, the students even knew how they wanted to change the expression but they were often unable to find the name of the necessary operation. It is clear that we should pay attention to this issue when preparing the teachers for using rule-based software.

We collected the records of this trial – files with data about errors made by the students – for further study. The collected data included initial expression, current expression, selected rule, marked objects, entered parts (in the case of an error at the input stage) and any error messages shown to the student. We also had some notes taken by the observers during the trial (two mathematics teachers and the four authors of T-algebra). When reviewing the files containing the students' mistakes, initially we noticed almost all the students had made mistakes in marking the objects for applying the rule. The reason was probably that the students did not understand how to use the software – how and which parts of the expressions had to be marked for applying the rules. The mistakes of this type occurred two or three times in the beginning and then disappeared. Almost all subsequent mistakes were due to a lack of mathematical knowledge (how to calculate the result of applying the rule, arithmetic errors, etc.) – the students made the same mistakes as they made in paper tests.

When reviewing the trial, we noticed that many students preferred to mark the objects of the rule before selecting the rule itself (despite the “Select the rule” instructions on the screen and the instruction “1. Select the rule. 2. Mark the operands. 3. Enter the result” on the blackboard). At that time our program gave no opportunity for marking more than one part in the expression before the rule was selected – it confused some of the students and they asked questions about that. After the trial we added the possibility to select objects for applying the rule before the rule itself is selected. Yet the hints on selection of objects become available only after selection of the rule. We are also planning a study to determine which order of actions will be used the most and what could affect the order (actual rule, location of menu on the screen, etc).

Summarizing the results of the first trial, we can say that the time required for learning the dialogue stages is quite short. In the first hour with T-algebra, most of the students had solved the same number of problems that were given to them in paper sessions. But unlike in the paper tests, the students corrected all the mistakes they made. Error messages shown by the program were clear enough for the students to correct the mistakes. Different input modes of different rules were tested during the trial – all input modes were found useful. When solving the problems, no questions were asked on why all three stages of the dialogue are needed; the idea of the first two stages was clear to the students. All the students (even the weakest in mathematics) were using the program with great interest.

4.3 Third experiment

When designing the program, we have taken into account the results of known studies of student mistakes made when working with pencil and paper (Hall, 2002; Sleeman, 1984) and have conducted our own study to identify the mistakes made by the Estonian students (first experiment, see Section 4.1).

In the spring of 2006, I conducted a study on errors made by the students who solved linear equations in the T-algebra environment. In this trial, the students were given exactly the same problems as in the tests on paper (see Section 4.1). The trial was conducted in three different classes and 83 students of the 7th grade participated in the test. These students were different from the participants in the first experiment, but they were from the same school and had the same mathematics teachers. Like the students from the second experiment, they already had sufficient experience with computers, but it was the first time they had seen T-algebra. The sessions lasted for one hour. A scheme similar to the second experiment was used. A demonstration of T-algebra and the solution processes in T-algebra took place during the first five minutes; we also wrote our general dialogue scheme on the blackboard. Then the students had ten minutes to try T-algebra by themselves. Analogous questions like in second experiment were asked (questions concerning the use of the computer, the use of T-algebra tools, and mathematical questions about the solution steps). After that, questions concerning the use of software disappeared. In the last 45 minutes the students solved the test. The structured input mode was chosen for the test and the help in the program was denied during the test. Like in the second experiment I collected the files with data about errors made by the students and had some notes taken by the observers during the trial.

At the end of the session, the students filled out a questionnaire (composing this questionnaire, the questionnaire presented in (Mitrovic and Ohlsson, 1999) was used; see Appendix D). I evaluated usability of T-algebra analyzing these questionnaires. The students were comfortable with the program. The majority (60%) reported that they needed 5 to 20 minutes to start using the system; 33%

needed less than 5 minutes and only 7% of the students needed more than 20 minutes to learn to use T-algebra. When asked to rate ease of use on a scale: *Not at all*; *Rather not*; *Hard to say*; *Rather yes*; *Yes, very much*, half the students (48%) chose the alternative *Rather yes* and 33% the alternative *Yes, very much*. When asked whether they enjoyed working with T-algebra, almost equal percentages of students chose alternative *Rather yes* (38%) and *Yes, very much* (37%). When asked whether they would like to use T-algebra more, 65% answered *Yes* and when asked whether they would recommend T-algebra to other students, 70% said *Yes*. In short, a majority of the students found T-algebra easy to learn, easy to use and enjoyable and would like to use it more.

I also asked the students did they learn something new about mathematics from using T-algebra. 44% of the students chose the alternative *Rather not* on the same scale and 24% chose *Hard to say*. This is because the students had already covered the topic of linear equations in the class. I also asked them about error messages shown in the case of mistakes. The majority found them understandable: 44% chose the alternative *Rather yes* and 44% the alternative *Yes, very much*. To the question, did error messages help to correct mistakes, 53% of students answered *Yes, very much* and 38% chose *Rather yes*. In short, the students did not learn much using T-algebra, because the material was not new to them, but the majority of the students found error messages understandable and helpful.

The following Table 4.3 shows the results of the tests: the percentages of problems solved correctly, wrongly, etc.

Table 4.3. Results of the tests

Test	Correct answer	Wrong answer	Half solution	Blank	Wrong solution, right answer (cribbed?)
Paper	45.38 %	32.95 %	5.16 %	16.3 %	0.2 %
T-algebra	96.69 %	impossible	0.98 %	2.33 %	impossible

If the student gave an answer in T-algebra and the program checked that the equation is in appropriate form, it means that this is the correct answer because it is impossible to produce incorrect solution steps in T-algebra. If an error message was displayed at any checking stage during problem solving, the student had to correct the error in order to proceed to the next stage. Additionally, it is impossible in T-algebra to proceed with the wrong solution and then give the right answer.

Although the program requires the student to enter more information (to choose the rule and to mark the parts for the rule) than she/he would write during solving on paper, the study showed that students solved more problems with computer than on paper (the percentage of half-solutions and blank exercises was lower) in the same time period. We have some explanations for that. First, the program generates the expression in the next line based on the

selected rule and marked parts, and leaves blank certain important parts of the new expression. When solving problems at school with paper and pencil, the students always have to write an expression of the same length to the next line. The program makes the work easier for the students by copying the parts of the expression that remain unchanged so that the students would have to enter only the parts that were modified. The other reason is that the teachers request checking the solution of linear equations when solving on paper (fifth type of problems – solve an equation). This checking stage is omitted in T-algebra, because the program does not permit incorrect solutions as stated above. Still, the test included only three problems of this type. Another very important point is that the students managed to correct all their mistakes by themselves during this time period.

Now let us take a look at the results of the tests grouped by the types of problems.

Table 4.4. Results of the tests on paper grouped by the types of problems

Type	Correct answer	Wrong answer	Half solution	Blank	Wrong solution, right answer (cribbed?)
Type <i>Reverse</i> (paper)	19.02 %	79.89 %	0 %	1.09 %	0 %
Type <i>Divide</i> (paper)	70.29 %	17.75 %	2.54 %	9.42 %	0 %
Type <i>Multiply</i> (paper)	32.61 %	30.33 %	1.09 %	36.3 %	0 %
Type <i>Move</i> (paper)	55.8 %	23.91 %	14.13 %	6.16 %	0 %
Type <i>Solve</i> (paper)	46.38 %	35.87 %	5.8 %	10.87 %	1.09 %

Table 4.5. Results of the tests in T-algebra grouped by the types of problems

Type	Correct answer	Wrong answer	Half solution	Blank	Wrong solution, right answer (cribbed?)
Type <i>Reverse</i> (T-algebra)	96.99 %	impossible	1.81 %	1.2 %	impossible
Type <i>Divide</i> (T-algebra)	96.39 %	impossible	1.61 %	1.61 %	impossible
Type <i>Multiply</i> (T-algebra)	98.55 %	impossible	0.24 %	1.2 %	impossible
Type <i>Move</i> (T-algebra)	97.19 %	impossible	0.4 %	2.41 %	impossible
Type <i>Solve</i> (T-algebra)	92.77 %	impossible	1.61 %	5.62 %	impossible

As we can see, the most difficult problem type on paper is *Reverse sides* (Table 4.4). This is probably because the theme of linear equations was already learned and the students remember the best and practice the most the equation-solving algorithm and this step is omitted in the algorithm. Students act as if solving the equation to the end and move all variable terms to the left side and all constant terms to the right side; or they might reverse the equation sides, but change the signs too, as during moving.

The most successful type on paper is *Divide equation sides by variable coefficient* (Table 4.4), because the most attention is paid to solving the equation to the end and this operation is the last step in the equation-solving algorithm and students remember it very well.

In T-algebra all types were solved successfully (Table 4.5), only some students did not manage to solve the test to the end. The percentage of the blank exercises of the type *Solve an equation* is the highest, because these exercises were in the end of the test.

4.3.1 Comparison of student mistakes made during solving on paper and in T-algebra

The result of the tests confirmed our assumptions that students make both, specific mistakes, which occur only during linear equation solving (for example, in moving terms to other side), as well as mistakes related to previously studied material (for example, in adding numbers). Let us take a closer look at the mistakes made during solving on paper and compare them with the mistakes made during solving in T-algebra. The errors statistics is presented in the tables grouped by steps during which they were made. Columns named *% paper* and *% T-algebra* show the percentage of students who made this mistake. Columns named *No. mistakes* show the average number of mistakes of this kind per student who made this mistake. The numbers in the column *No. mistakes paper* should be compared only with numbers in the column *No. mistakes T-algebra*. The numbers within a column can not be compared between themselves, because these tables do not reflect how many possibilities for making some kind of mistake were in the test, for example, if there was only one possibility to make the mistake number 6 from Table 4.6, the average number of these mistakes is also one, but if there were four possibilities to make the mistake number 1 from Table 4.6, the average number of these mistakes is 2.57. The fat line in the tables isolates the mistakes that were diagnosed only in the T-algebra environment.

4.3.1.1 Mistakes made during multiplying both sides of equation

The first thing that can be noticed in Table 4.6 is that the average number of mistakes in T-algebra is smaller than the average number of mistakes on paper. We think that showing the error message immediately is the cause of that. For example, let us consider the mistake number 1 from Table 4.6. The student errs with the sign and immediately gets the respective error message. Next time, the student errs with the sign and gets the error message again. Then the third time the student already pays more attention in the similar situation (when the minus sign is before fraction and two terms are in the numerator of fraction) and does not make the mistake. This was also noticed by observers during the test.

Let us take a look at the mistake number 2 from Table 4.6. The percentage of students who made this mistake is much higher in T-algebra than on paper. In T-algebra, the students must enter the extender ('extenders' are numbers by which you need to multiply both the numerator and denominator of the fraction to convert the denominators to a common denominator – this term is used in the Estonian schools and textbooks (Nurk et al., 2000)) for every term of equation (for every fraction and also for every number which is not fraction). Reviewing the paper tests, we noticed that the students often do not write the extender for a number, which is not fraction, and this causes the mistake number 5 from Table 4.6. In T-algebra, the student could not proceed if some extender is not entered; this increases the number of students who made the mistake number 2 (reviewing the records, we found that almost half of wrong extenders are written for whole number, for example $\frac{x^{16}}{5} + 3 \frac{110}{10} = \frac{7}{10} \cdot 30$). However, after finding the right extender for whole number the students do not forget to multiply them and, consequently, nobody made the mistake number 5 from Table 4.6 in T-algebra.

The next mistake from Table 4.6 that is worth examining is the mistake number 7. Nobody made this mistake in T-algebra. We think that this is the result of the design of the rule dialogue (structured input mode) in T-algebra. In order to enter the result of application of the rule *Multiply/Divide both sides*, the program gives an equality sign and a number of boxes to the user (Figure 3.4 from Section 3.2.1). There are two kinds of boxes: small boxes are for input of signs + and –, larger boxes are for entering numbers and variables. The program prompts how many terms are in the result and it is very difficult for the student to forget some terms, because the number of boxes corresponds to the number of terms in the result of multiplication.

The mistake number 9 from Table 4.6 surprised us, but after a precise analysis of paper tests, we found that sometimes students just do not write the common denominator on paper, but act correctly further. In T-algebra the

student must write the common denominator, so we drew the conclusion that some students do not know how to find a common denominator, but somehow they know how to remove fractions from the equation. On paper, they just skip writing the common denominator, but it is not possible to act like that in T-algebra.

The mistakes number 12-17 were diagnosed only in T-algebra. The mistakes number 15, 16, 17 can not be diagnosed on paper at all, because the students do not write/explain on paper what they are doing and often do not mark the parts for operation. The mathematics teachers consider the choice of the rule in T-algebra as a good opportunity to teach students the correct names of the operations, because the students often know how they should solve, but they do not know how to describe what they are doing. The same seems to apply to precise marking of operands for the selected rule.

We still do not know the reasons for mistakes 12, 13 and 14 and why they appear only in T-algebra. These mistakes need to be examined more carefully during the next studies. The mistake 12 is very strange, because the program writes what the student should enter and even offers a separate box for the sign (Figure 3.1 from Section 3.2.1) and we do not know why the students leave it blank.

We assume that mistake number 13 can be caused by the location of extenders (right from the fraction, near the second term of the numerator, see Figure 3.2 from Section 3.2.1), but exactly the same notation is used during solving on paper and in textbooks, and this does not cause mistakes on paper.

Table 4.6. Mistakes in multiplying both equation sides

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	Minus sign before fraction is taken into account only at first term	$\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15}$ ·15 $20y - 6y + 9 = 4$	52.69 %	2.57	63.86 %	1.9
2	Wrong extender	$\frac{n-3}{6} - \frac{2n+1}{2} =$ $= \frac{3n-1}{8} + \frac{n+3}{12}$ ·24	12.9 %	4.75	26.5 %	1.86
3	Extender is multiplied only with first term of numerator	$\frac{y-2}{4} = \frac{2}{3}$ ·12 $3y - 2 = 8$	11.83 %	2.18	4.82 %	1.25
4	Arithmetic mistake	$\frac{3m-1}{2} - \frac{m+3}{3} =$ $= \frac{m+5}{4}$ ·12 $6m - 6 - 4m - 12 =$ $= 3m + 15$	11.83 %	1.18	18.07 %	1.2
5	Whole number is not multiplied	$\frac{x}{5} + 3 = \frac{7}{10}$ ·10 $2x + 3 = 7$	7.53 %	1.29	0 %	0
6	Minus sign before fraction is not taken into account	$\frac{2y}{9} = -\frac{1}{6}$ ·18 $4y = 3$	7.53 %	1	8.43 %	1
7	One term is forgotten (which is not fraction)	$\frac{3u-1}{5} - \frac{u+7}{3} =$ $= \frac{1-8u}{15} - \frac{1}{15}$ ·15 $9u - 3 - 5u - 35 = 1 - 8u$	6.45 %	1	0 %	0
8	The result is written with common denominator	$\frac{x}{3} = \frac{2}{5}$ ·15 $\frac{5x}{15} = \frac{6}{15}$	5.38 %	3.6	1.2 %	1

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
9	Wrong common denominator	$\frac{2u}{5} - \frac{u-3}{4} = \frac{2u+3}{10}$ ·10	4.3 %	4	26.5 %	1.95
10	Variable is omitted	$\frac{x^{\frac{15}{3}}}{3} = \frac{2^{\frac{15}{5}}}{5}$ $\frac{5-6}{15}$ or $5=6$	3.23 %	3	10.84 %	1.11
11	Multiplied, but the sign is division	$\frac{3u-1^{\frac{15}{3}}}{5} - \frac{u+7^{\frac{15}{5}}}{3} =$ $= \frac{1-8u^{\frac{15}{3}}}{15} - 1^{\frac{15}{5}}$ ·15 $9u-3-5u-35 =$ $= 1-8u-15$	2.15 %	1	13.25 %	1.64
12	Multiplication sign is omitted	$\frac{y-2}{4} = \frac{2}{3}$ 12	0 %	0	16.87 %	1.07
13	Extender is multiplied only with second term of numerator	$\frac{m+2^{\frac{12}{6}}}{6} = \frac{3^{\frac{12}{4}}}{4}$ ·12 $\underline{m} + 4 = 9$	0 %	0	13.25 %	1.73
14	The sign of second term of numerator is changed	$\frac{n-3^{\frac{24}{6}}}{6} - \frac{2n+1^{\frac{24}{2}}}{2} =$ $= \frac{3n-1^{\frac{24}{6}}}{8} + \frac{n+3^{\frac{24}{2}}}{12}$ ·48 $8n+24-48n-24 =$ $= 18n+6+4n+12$	0 %	0	12.05 %	1.8
15	Selected the rule <i>Multiply/Divide numbers</i>		0 %	0	9.64 %	1
16	Selected the rule <i>Extend</i>		0 %	0	4.82 %	1
17	Only one side of equation is selected for applying the rule		0 %	0	4.82 %	1

4.3.1.2 Mistakes made during dividing both sides of equation

Like in the case of mistake number 9 from Table 4.6, the increase in students who made the mistake number 2 from Table 4.7 and the increase in the average number of these mistakes are caused by the required input of the divider (on paper, the students often skip this part). In addition, this can be caused by the number of boxes that are given for entering the divider. The student gets two boxes (Figure 3.1 from Section 3.2.1). Maybe the students do not understand that they write two signs side by side, because they are in different boxes (the separate box is given for entering division sign). We made the conclusion that during the next study we should try giving only one box for entering the sign and number, then we could say whether this increase was caused by the number of boxes or not.

The increase in the average number of mistakes number 1 and 3 can be explained by the fact that T-algebra requires correction of the mistake right after the mistake was made. Reviewing the record files, we noticed that some students do not know how to correct the mistake and just try different possibilities and make not one mistake like on paper, but two or three mistakes before they discover the right answer. This can be noticed particularly in the mistake number 3 from Table 4.7. The students have problems in dividing the decimal numbers. When they write 0.4 for the result and get the error message, they try to give for answer the number 0.3 or 0.04 and different other numbers and only after that offer the number 4 for the answer.

The mistake number 4 from Table 4.7 probably has the same explanation as mistake number 12 from Table 4.6, but we did not prove it yet.

Mistake number 5 from Table 4.7 is impossible in T-algebra. The student could not proceed further with the wrong divider and give the right answer. Therefore, all the attempts to write a wrong divider were counted as mistake number 6 from Table 4.7 in T-algebra. This can explain the increased percentage of students who made the mistake number 6 in T-algebra.

The mistakes number 8–10 from Table 4.7 were and can be diagnosed only in T-algebra like mistakes 15–17 from Table 4.6. Almost all number 9 mistakes were made by selecting the right side of the equation, which was just a number, for dividing. Apparently, the students thought that they should just divide the right side of the equation with the coefficient of variable, and they did not understand that the left side should be divided as well. They probably thought that they do not have to divide the left side if they simply do not write the coefficient before the variable.

Table 4.7. Mistakes in dividing both equation sides

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	Mistake in sign	$-4y = -8 \quad :(-4)$ $y = \underline{-2}$	26.88 %	1.16	24.1 %	1.45
2	The parentheses (in divider) are omitted	$-5n = 25 \quad : -5$ $n = -5$	23.66 %	1.32	45.78 %	1.68
3	Arithmetic mistake	$-0.3y = -1.2 \quad :(-0.3)$ $y = \underline{0.4}$	19.35 %	1	36.14 %	1.4
4	Division sign is omitted	$-8n = 16 \quad (-8)$ $n = -2$	5.38 %	2.4	14.46 %	1.17
5	Wrong divider (right answer)	$-4y = -8 \quad :(-2)$ $y = 2$	3.23 %	1	impossible	impossible
6	Wrong divider	$7x = 21 \quad : \underline{3}$	2.15 %	2	8.43 %	1.29
7	Variable is omitted	$7x = 21 \quad : 7$ $\underline{1} = 3$	1.08 %	1	7.23 %	1.67
8	Selected the rule <i>Multiply/Divide numbers</i>		0 %	0	31.33 %	1.38
9	Only one side of equation is selected for applying the rule		0 %	0	9.64 %	1.38
10	Selected the rule <i>Reduce</i>		0 %	0	3.61 %	1

4.3.1.3 Mistakes made during moving terms to the other side of equation

Like in the case of mistake number 1 from Table 4.6, we may notice that the average number of mistakes number 1 from Table 4.8 in T-algebra is lower than the average number of mistakes on paper. We believe that this is because of the displayed error message and directing attention to the mistake and its location (box). Similarly, in the case of mistake number 1 from Table 4.7, we may notice that the percentage of the students who made this mistake in T-algebra is lower, because the program reminds in a certain way about the sign of the term, which is very important in this operation: a separate box is given for entering the sign (see Figure 3.8 from Section 3.2.2).

The mistake number 2 is not possible in T-algebra. The program copies the unchanged parts automatically and the student could not err in passive rewriting. We assume that the loss of this mistake is not essential.

The mistakes number 3 and 4 from Table 4.8 were not made in T-algebra probably because of the offered structure for the result. The program shows the unchanged part and a number of boxes in the next line (Figure 3.8). The boxes appear on the opposite side to the selected parts. The program prompts how many terms are moved and to which side and it is very difficult for the student to forget some terms and it is impossible to write terms to both sides, because the number of boxes corresponds to the number of moved terms.

Mistakes 6, 7 and 8 were diagnosed only in T-algebra. The mistake 8 is actually not a mistake in moving terms to other side, but a mistake in opening parentheses. Some students understand $3(2x+3)$ like addition between 3, 2x and 3, as if it would be written $3+(2x+3)$. This mistake was found during checking the paper tests and it is presented in Table 4.12 (mistake number 3). However, in T-algebra this mistake was discovered during the selection of the parts for the rule *Move terms to other side* and presented here, in Table 4.8.

Table 4.8. Mistakes in moving terms to the other side of equation

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	Sign is not changed	$8u - 5u + 15 = 4u + 6$ $8u - 5u - 4u = \underline{15} + 6$	29.03 %	2.11	14.46 %	1.5
2	Mistake in passive rewriting (changing sign, forgetting term)	$2m - 3 + 5 = 2 - 5m + 1 + 3m$ $2m + 5m - 3m = 2 - 1 + 3 - 5$	26.88 %	1.96	impossible	impossible
3	One removed term is lost	$9 - 6y + \underline{2} = 3 - 2y$ $-6y + 2y = 3 - 9$	5.38 %	1	0 %	0
4	The terms are moved and left on the same side	$9u - 3 + 35 = 1 - 8u - 15$ $\underline{9u + 8u} =$ $= 1 - 15 + 3 - 35 - \underline{9u + 8u}$	2.15 %	2	impossible	impossible
5	Removed term is changed	$9m + 12 = 4$ $9m = 4 - \underline{15}$	1.08 %	1	14.46 %	1.25
6	Selected the rule <i>Reverse sides</i>		0 %	0	10.84 %	1.11
7	All terms are selected for moving, but only part of them are moved		0 %	0	8.43 %	1.29
8	All selected terms are not suitable for moving	$2x - \underline{3(2x+3)} = 3 - \underline{8x}$	0 %	0	2.41 %	1

4.3.1.4 Mistakes made during reversing the equation sides

In Table 4.9, we may see that the percentage of the students who made the mistakes 1-3 in T-algebra is much lower than that of the students who made these mistakes on paper. Some students tried to reverse sides during the first 15 minutes (introduction to T-algebra), made mistakes and discovered how to reverse sides correctly during this time. Therefore, during the test they solved these problems correctly without mistakes. We can also see that the average number of mistakes is smaller in T-algebra than on paper. The students made mistakes during solving the first problem of this type. They understood how to reverse sides and almost did not err during solving the second problem of this type. We think that this is a sign of the good influence of T-algebra on the students.

The mistake number 5 from Table 4.9 arose from the dissatisfaction of the program with the answer. When the students thought that they should move all variable terms to the left side and all constant terms to the right side to reverse sides, they had the opportunity for that. The rule *Move terms to other side* was available and the students used it successfully. Then they gave an answer, but the program said that the sides are not reversed completely. They did not understand why and acted like during solving linear equation to the end (tried to combine like terms). Yet the program did not accept that and then students tried one more time to combine and gave the combined answer but with different coefficients or different signs, so the average number of these mistakes is higher than one. Observing the files, we noticed that some students were stuck at this moment and they left the solution as it was (did not solve to the end).

Table 4.9. Mistakes in reversing the equation sides

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	All variable terms are moved to the left side and all constant terms to the right side	$3m - 7 = 5 + 2m$ $\underline{3m} - 2m = \underline{5} + 7$	58.06 %	1.8	26.5 %	1.09
2	Signs are partially changed	$3m - 7 = 5 + 2m$ $5 - 2m = 3m + 7$	17.2 %	1.31	8.43 %	1
3	Signs are changed	$3m - 7 = 5 + 2m$ $\underline{-5} - 2m = 7 - 3m$	15.05 %	2	4.82 %	1
4	Only one side of equation is selected for applying the rule		0 %	0	12.05 %	1.1
5	All terms are combined	$5u - 2u = -9 - 3$ $-12 = 3u$	0 %	0	10.84 %	1.78

4.3.1.5 Mistakes made during combining like terms and during adding/subtracting numbers

The mistakes in combining like terms and adding/subtracting numbers are presented in Tables 4.10 and 4.11. As we can see, the percentage of students who made these mistakes in T-algebra is higher than that of the students who made these mistakes on paper. We do not know whether it happens because of the software use or simply because the students who participated in the second test (the students participating in T-algebra and paper tests were different) did not know how to add/subtract numbers.

As in above cases, the increase in the average number of mistakes is caused by the program requirement to correct the mistake.

Mistake number 4 from Table 4.10 shows that the students are confused by terminology. They could combine numbers (numbers are similar terms), but they could not add/subtract numbers from similar terms – monomials.

Table 4.10. Mistakes in combining like terms

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	Wrong coefficient	$3x - 7x = 4$ $-3x = 4$	11.83 %	1	28.92 %	1.41
2	Wrong sign	$9u - 5u + 8u = 1 - 15 + 35 + 3$ $-12u = 24$	9.68 %	1.11	10.84 %	1.22
3	Variable is omitted	$3x - 7x = 4$ $-4 = 4$	2.15 %	1	0 %	0
4	Selected the rule <i>Add/Subtract numbers</i>		0 %	0	9.64 %	1

Table 4.11. Mistakes in adding/subtracting numbers

No	Nature of mistake	Example of mistake	% paper	no mistakes paper	% T-algebra	no mistakes T-algebra
1	Wrong sign	$-2y - 5y = -9 + 3$ $-7y = \underline{6}$	11.83 %	1.18	14.46 %	1
2	Arithmetic mistake	$7s - 9s = 2 - 12$ $-2s = \underline{-24}$	7.53 %	1.29	32.53 %	1.37

4.3.1.6 Mistakes made during opening the parentheses

The mistakes 1 and 2 from Table 4.12 were made by fewer students in T-algebra than on paper. We believe this is because the program always gives boxes for the parts that are changing (Figure 3.32 from Section 3.2.9), so the students do not forget to multiply the second term from the parentheses and the program also reminds about the sign by giving a separate box for it.

Mistake number 3 was diagnosed in applying the rule *Move terms to other side* and displayed in Table 4.8 (mistake number 8).

It is difficult to say anything about the average numbers of mistakes from Table 4.12, because in the test there was only one place, where the rule *Open parentheses* could be applied, so the average number of mistakes is always 1.

Table 4.12. Mistakes in opening parentheses

No	Nature of mistake	Example of mistake	% paper	No. mistakes paper	% T-algebra	No. mistakes T-algebra
1	Minus sign before parentheses is taken into account only at first term	$9 - 2(3y - 1) = 3 - 2y$ $9 - 6y - 2 = 3 - 2y$	29.03 %	1	19.28 %	1
2	Number before parentheses is multiplied only with first term from parentheses	$2x - 3(2x + 3) = 3 - 8x$ $2x - 6x + 3 = 3 - 8x$	6.45 %	1	1.2 %	1
3	Not multiplied, but added	$9 - 2(3y - 1) = 3 - 2y$ $3y + 2y = -9 + 2 + 1 + 3$	5.38 %	1	impossible	impossible
4	Arithmetic mistake	$2x - 3(2x + 3) = 3 - 8x$ $2x - 6x - 6 = 3 - 8x$	2.15 %	1	1.2 %	1

4.3.2 Conclusions

The results of questionnaire showed that T-algebra is easy to learn, easy to use and enjoyable and the students want to use it more. The students also judged error messages as understandable and helpful.

Although the program requires the student to enter more information than she/he would write during solving on paper, the study showed that the students solved more problems with computer than on paper in the same time period. Very important result is that the students managed to correct all their mistakes by themselves during this time.

As we have seen, in most cases the average number of mistakes per student who made this mistake in T-algebra is smaller than on paper. Immediate display of the error message and indication of the error location make the students think and be more careful. The increase in the average number of mistakes in some places can be explained by the fact that T-algebra requires correction of mistakes. The student does not know how to correct the mistake and just tries different possibilities and makes not one mistake like on paper, but two or three mistakes before she/he discovers the right answer.

T-algebra helps to diagnose mistakes in places where some intermediate result, which can be skipped on paper, needs to be entered in T-algebra. On paper, the students take advantage of this opportunity and leave the result blank if they are not sure in it. The need to enter the intermediate result causes the increase in the number of students to whom this mistake was diagnosed, but helps to prevent other mistakes, which are caused by this misunderstanding.

Mistakes in selecting the rule and marking the parts for the selected rule can be diagnosed in T-algebra explicitly without any efforts, but on paper the diagnosis of these mistakes is very labor-intensive, because the students do not write/explain on paper what they are doing and often do not mark the parts for the operation. The mathematics teachers consider the choice of the rule in T-algebra as a good opportunity to teach the students the correct names of the operations, because the students often know how they should solve, but do not know how to describe what they are doing. The same seems to apply to precise marking of operands for the selected rule.

During the study, we found some kinds of mistakes, which were made in T-algebra, but never exist on paper. Unfortunately, we still do not know the reason for these mistakes and they will be the objects of careful study next time.

4.4 Fourth experiment

Since 1998, admission to Estonian universities takes place on the basis of the results of state examinations, which are held at the schools during the last school year. The Faculty of Mathematics and Computer Science at the University of Tartu in Estonia selects students according to the summary of the results of three exams: mathematics, essay in mother tongue and foreign language. The universities themselves have no admission examinations. During last year's lectures in the Faculty of Mathematics and Computer Science several lecturers noticed that the ability of many students to pass the first year courses (like Mathematical Analysis, Algebra and Geometry, Programming) has decreased. I decided to organize a research on the first year students' knowledge of elementary algebra (like linear equation solving) using T-algebra. This research can help our lecturers to get to know the gaps in student knowledge to adjust their courses.

Exactly the same tests like in the first and third experiments (see Sections 4.1 and 4.3) and the scheme of experiment like in the third experiment (one hour sessions: five minutes for demonstration of T-algebra, ten minutes for trying T-algebra, 45 minutes for test; see Section 4.3) were used in the study on the first year students' mistakes in autumn 2006. The trial was conducted in three groups of different curricula: Computer Science, Mathematics, and Mathematical Statistics, and 46 students participated in it. The trial was carried out during one lesson of course *Introduction to Computer Applications*, the students were all 1st year students of these curricula who came to this lesson at this day (i.e., the student were not specifically selected).

4.4.1 Results of experiment

The following Table 4.13 shows the results of the tests: the percentage of problems solved correctly, the percentage of unfinished solutions, etc.

Table 4.13. Results of the tests

Participants	Correct answer	Half solution	Blank
7th grade students	96.69 %	0.98 %	2.33 %
1st year students	99.21 %	0.14 %	0.27 %

Only one student from the Statistics curriculum did not solve the test to the end. She/he did not manage to solve all problems of type *Divide equation sides by coefficient of variable*. She/he tried five times to apply the rule *Multiply/Divide numbers* instead of rule *Multiply/Divide both sides* and did not succeed. It is interesting that when solving the equation to the end she/he applied the rule *Multiply/Divide both sides* for dividing correctly. The difficulty was probably caused by the text of the exercise (Divide equation sides by coefficient of variable); it is likely that the student did not understand what the coefficient of variable is and how to divide by it.

Now consider mistakes made during solving linear equations. Table 4.14 shows the average number of mistakes per student.

Table 4.14. The average number of mistakes per student

	1st year	Statistics	Mathematics	Informatics	7th grade
Average number of mistakes per student	4.93	7.38	3.78	5	10.69

The assumption that the first year students make mistakes caused by the material learned after the 7th grade was not confirmed. The only mistake caused by the new material was made by one student who did not multiply the numerator with the extender, but raised numerator to a power of extender

$$\left(\frac{x^{15}}{3} = \frac{2^{13}}{5} \mid \cdot 15 \rightarrow x^5 = 8\right).$$

Many mistakes of different kinds were made, but the following Table 4.15 presents only the mistakes that were made by more than ten percent of the first year students to demonstrate the common gaps in student knowledge.

Table 4.15. Mistakes made during solving linear equations

No	Nature of mistake	Example of mistake	1st year	Statistics	Mathematics	Informatics	7th grade
1	The parentheses (in divider) are omitted	$-5n = 25 \quad : -5$ $n = -5$	58.7 %	75 %	66.67 %	45 %	45.78 %
2	During multiplication minus sign before fraction is taken into account only at first term	$\frac{4y}{3} \frac{5}{5} - \frac{2y+3}{5} = \frac{4}{15} \quad \cdot 15$ $20y - 6y + 9 = 4$	32.61 %	25 %	22.22 %	45 %	63.86 %
3	Selected the rule <i>Multiply/Divide numbers</i> for dividing both sides of equation		23.91 %	50 %	27.78 %	10 %	31.33 %
4	Selected the rule <i>Reverse sides</i> for moving terms to other side		21.74 %	25 %	22.23 %	20 %	10.84 %
5	Arithmetic mistake in multiplying	$\frac{3m-1}{2} - \frac{m+3}{3} = \frac{m+5}{4} \quad \cdot 12$ $6m - 6 - 4m - 12 = 3m + 15$	17.39 %	0 %	11.11 %	30 %	18.07 %
6	Division sign is omitted	$-8n = 16 \quad (-8)$ $n = -2$	15.21 %	25 %	22.22 %	5 %	14.46 %
7	Sign is not changed during moving to other side	$8u - 5u + 15 = 4u + 6$ $8u - 5u - 4u = \underline{15} + 6$	13.04 %	12.5 %	5.56 %	20 %	14.46 %
8	Removed term is changed	$9m + 12 = 4$ $9m = 4 - \underline{15}$	13.04 %	50 %	5.56 %	5 %	14.46 %
9	Wrong extender in multiplying	$\frac{n-3}{6} - \frac{2n+1}{2} =$ $= \frac{3n-1}{8} + \frac{n+3}{12} \quad \cdot 24$	10.87 %	12.5 %	11.11 %	10 %	26.5 %
10	Arithmetic mistake in dividing	$-0.3y = -1.2 \quad : (-0.3)$ $y = \underline{0.4}$	10.87 %	12.5 %	11.11 %	10 %	36.14 %
11	Arithmetic mistake in combining like terms and in adding / subtracting numbers	$7s - 9s = 2 - 12$ $-2s = \underline{-24}$	10.87 %	25 %	5.56 %	10 %	32.53 %

The mistakes number 1 and 6 are caused by the required input of the divider in T-algebra. We saw the same during the experiment with the 7th grade students. On paper, they just skip writing the divider, but it is not possible to act like that in T-algebra.

The mistakes number 3 and 4 show that the students do not know the correct names of the operations. At school, the students do not write/explain on paper what they are doing. The 7th grade and 1st year students often know how they should solve, but they do not know how to describe what they are doing. The mathematics teachers consider the choice of the rule in T-algebra as a good opportunity to teach the students to explain what they are doing.

The experiment showed that the students made many arithmetic mistakes (mistakes number 5, 10, 11) and some mistakes in sign (mistakes number 2 and 7). The rate of such mistakes was lower among the 1st year students than among the 7th grade students, but it was still too high. Mistake 8 shows that students made the same amount mistakes in rewriting, which is probably caused by low attention level.

There are also some mistakes that were made by the 7th grade students, but were not made by the 1st year students. Very few students made mistakes in sign (except mistakes 2 and 7 from Table 4.15):

- mistake in sign in dividing (2.17% of 1st year students, 24.1% of 7th grade students);
- wrong sign in combining (2.17% of 1st year students, 10.84% of 7th grade students);
- wrong sign in evaluating (4.35% of 1st year students, 14.46% of 7th grade students);
- in opening parentheses minus sign before parentheses is taken into account only at first term (2.17% of 1st year students, 19.28% of 7th grade students).

4.4.2 Conclusions

T-algebra helped to identify the gaps in the knowledge of elementary algebra among the students. We saw that the students have some gaps in the knowledge of correct names of operations and have some problems with things that they could skip on paper (like divider). The experiment also showed that the students make a fairly large number of arithmetic mistakes and some mistakes in sign. But the first year students do not make mistakes caused by the material learned later (after the 7th grade) and also do not err in sign in most cases.

4.5 Fifth experiment

Before distribution of T-algebra to all Estonian schools, I organized an experiment to clarify how the program affects the learning results. The study was carried out in the winter 2007. Seven classes (126 students) of 7th grade (13 years old) from four different Estonian schools participated in the experiment. Classes of two schools, where there was more than one 7th grade class, were divided into experimental classes and control classes. The remaining two schools participated as experimental classes. After division we had 2 control classes and 5 experimental classes. Classes from schools with more than one 7th grade class were taught by the same teacher.

The topic of linear equations was chosen for experiment and the experiment began when the topic was explained and practiced in the schools. The experiment consisted of four 45-minute sessions. In the first session, the students solved a pre-test on paper. In the next two sessions, the students practiced solving the problems of the same topic (linear equations). The experimental groups practiced solving these problems with T-algebra, while the control groups practiced solving exactly the same problems using traditional instruction technology – paper and pencil. In the last session, the students solved a post-test on paper. Teachers had exact instructions what, when and how to do and the same materials (pre-test, problems for practicing and post-test) for all teachers were prepared.

Pre-test was solved in both groups using paper and pencil. Pre-test was organized right after linear equation solving was learned, before other themes. During the pre-test the students could not use any assistance materials. A test in two variants was composed by one of mathematics teachers (Janika Kaljula), who participated in the experiment. The other teacher (Sirje Pihlap) advised assigning points to each problem. Test contained 17 problems (6 types of problem) and it was possible to earn 39 points in total. The problems (with maximum points) of variant A were the following:

- check if number 5 is solution of equation $10 - 2(3y - 1) = 2y - 7(y - 1)$ (3 p.);
- reverse equation sides: 1) $24 = 3y + 7$ (1 p.); 2) $4u + 1 = 7u - 6$ (1 p.);
- divide equation sides by variable coefficient: 1) $4y = 48$ (1 p.); 2) $1,3n = -3,9$ (1 p.); 3) $-9y = -45$ (1 p.);
- multiply both sides of the equation by common denominator: 1) $\frac{3n}{8} = -\frac{5}{6}$ (2 p.); 2) $\frac{k+3}{4} = \frac{5}{8}$ (2 p.); 3) $\frac{n}{6} + 4 = \frac{5}{12}$ (2 p.); 4) $\frac{3m}{10} - \frac{m+2}{5} = \frac{4m-5}{6}$ (2 p.); 5) $\frac{x+3}{9} - \frac{3x+1}{12} = \frac{2x+5}{4} + \frac{x-5}{6}$ (2 p.);

- move all variable terms to the left side and all constant terms to the right side and then combine like terms: 1) $6 + 4b = 2$ (2 p.); 2) $6m + 11 = 5m - 4$ (2 p.); 3) $5 - 7x + 4 - x = 3x - 9 + x + 11$ (2 p.);
- solve an equation: 1) $7 - 4m = 3m + 21$ (4 p.); 2) $5y - 2(3y + 4) = 7 - 4y$ (5 p.); 3) $\frac{3x-5}{7} - 1 = \frac{3x-12}{14}$ (6 p.).

The problems of variant B were the following:

- check if number -4 is solution of equation $2x - 3(x - 1) = 3(x - 2)$;
- reverse equation sides: 1) $6s - 5 = 4 + 3s$; 2) $3y = 7y - 4$;
- divide equation sides by variable coefficient: 1) $5x = 20$; 2) $-0,6y = -2,4$; 3) $-7n = 49$;
- multiply both sides of the equation by common denominator: 1) $\frac{x}{4} = \frac{2}{7}$;
 2) $\frac{y-3}{5} = \frac{3}{4}$; 3) $3 - \frac{m}{4} = \frac{1}{3}$; 4) $\frac{3y}{2} - \frac{3y+5}{8} = \frac{3}{16}$;
 5) $\frac{x+1}{6} - \frac{3x-1}{2} = \frac{x-2}{9} - \frac{2x+5}{3}$;
- move all variable terms to the left side and all constant terms to the right side and then combine like terms: 1) $4x - 7 = 9x$; 2) $8 - 3y = 6y + 2$;
 3) $3m - 4 + 1 = 5 - 4m + 2 + 5m$;
- solve an equation: 1) $7x - 12 = 18 - 8x$; 2) $10 - 3(2y - 1) = 1 - 2y$;
 3) $\frac{2u-3}{5} - \frac{u+2}{3} = -1$.

The students got the mark (score) according to the following scale:

- 5: > 35 points (90% - 100 %);
- 4: > 27 points and \leq 35 points (70% - 90 %);
- 3: > 19 points and \leq 27 points (50% - 70%);
- 2: > 11 points and \leq 19 points (30% - 50%);
- 1: \leq 11 points (0% - 30%).

During the next two mathematics lessons, the experimental groups practiced solving similar problems using T-algebra in computer class. The practice took place immediately after the pre-test in the next mathematics lesson and linear equations were not taught in the ordinary class between pre- and post-tests. The students received the corrected pre-test before practice. The students had seen and tried T-algebra before in learning other topics, so teachers did not have to explain the environment to the students. T-algebra was installed and problem file was copied to computers, so students did not waste time on this. The problem file contained 40 problems (see the full list in Appendix E) chosen from the files composed during T-algebra course by teachers. The structured

input mode was chosen for the problems, because it gives the students more detailed error messages. The button *Autosolve* was disabled in all problems; in some problems (approximately a half), other help functions (for choosing a rule, for marking the operands, for entering the result) were also disabled (the problems with denied help are marked in the list in Appendix E). After the first lesson the students saved the solutions to a solution file and continued solving in the next lesson. After the second lesson the students saved solutions, the teachers gathered them and sent to us for examination.

While the experimental groups practiced solving using T-algebra, the control groups solved exactly the same problems (see Appendix E) using paper and pencil. During lessons the students solved the problems in their notebook and somebody wrote solution to the blackboard. The teacher showed and corrected mistakes in the solutions on the blackboard, but did not explain anything new and did not correct solutions in the notebooks.

During the fourth consecutive lesson, both groups solved a post-test using paper and pencil. Arrangement of the post-test was the same as in pre-test. The tests from the first experiment (see Section 4.1) were used for the post-test, but one problem was added: check if number -3 is solution of equation $3x - 2(x - 1) = 4(x - 2)$ (in variant A); check, if number 4 is solution of equation $12 - 3(2m - 1) = 3m - 7(m - 1)$ (in variant B). The students got the same variant like in the pre-test (if a student had variant A in the pre-test, then she/he had variant A in the post-test as well). Of course the students could not use any assistance materials, least of all the corrected pre-test.

After the experiment I gathered papers of the pre- and post-tests and files with solutions in T-algebra for analysis.

4.5.1 Results of experiment

After analysis of papers and files, the students who had missed at least one lesson were excluded and works of 115 students remained. Further tests were analyzed and students whose result of pre-test was less than 11 points were excluded, because in the preconditions of the experiment we assumed that topic has been taught to the students, i.e., the student can get at least 30% of the points. I wanted to evaluate how T-algebra affects practicing after the topic has been explained by the teacher, not how it influences learning new material. While all other students had some basic knowledge about linear equations, these students (who got less than 30%) did not. After that, works of 106 students remained; 76 of them had participated in the experimental group and 30 in the control. The following Table 4.16 shows the results (average number of points) of pre- and post-test in both (experimental and control) groups. As we can see, the average number of points in pre-test is almost equal in experimental and control groups. This difference is not statistically significant (unpaired t -test $t = 0.0368$, $p = 0.97$) and the groups can be considered as equal.

Table 4.16. Results (average number of points) of the tests

	Experimental	Control
Pre-test	29.4	29.3
Post-test	31.3	29.9

Table 4.16 shows that the knowledge of students from experimental group is statistically significantly improved (paired *t*-test $t = 3.571$, $p < 0.01$), but no statistically significant difference (improvement) can be found in the points earned by the control group (paired *t*-test $t = 1.2024$, $p > 0.05$). This shows that even short use (2 lessons) of T-algebra affects the results of learning.

The next Table 4.17 shows which percent of problems were solved correctly, which percent wrongly, which percent of problems were left with half (correct) solutions and which percent just blank.

Table 4.17. Results of the tests

Test	Correct answer	Wrong answer	Half solution	Blank
Experimental pre-test	59.7 %	32.3 %	5.3 %	2.7 %
Control pre-test	60.1 %	35.6 %	3.8 %	0.5 %
Experimental post-test	68.5 %	27.7 %	2.7 %	1.1 %
Control post-test	65.0 %	28.8 %	5.2 %	1.0 %

The following Table 4.18 shows the percentage of students from experimental group with different scores in pre- and post-tests. As we can see, the percentage of students with highest score has grown. The percentage of students with score 4 remained the same and the percentage of students with low scores (3 and 2) has decreased.

Table 4.18. Division of students (from experimental group) between scores in pre- and post-tests

	% Students with scores 5	% Students with scores 4	% Students with scores 3	% Students with scores 2
Pre-test	25 %	38 %	24 %	13 %
Post-test	39.5 %	38 %	14.5 %	8 %

While checking the post-test of the experimental group, I noticed that one experimental student emulated the writing style of T-algebra. The operation *Multiply/Divide both sides* has a slightly different appearance in Estonian textbooks and in students notebooks. The students perform this operation in two rows using paper and pencil and as a result we can see following solution:

$$\frac{x^{\text{is}}}{3} = \frac{2^{\text{is}}}{5} \quad | \cdot 15$$

$$5x = 6$$

But the application of this rule in T-algebra consists of marking the equation (first row), input of additional information – common denominator (separate window), then input of intermediate result – extenders (second row) and finally input of result – multiplied equation (third row) (see Figure 3.4 in Section 3.2.1). The mentioned student was not able to solve the problems of type *Multiply both sides* in the pre-test. In the post-test he solved all problems of this type successfully, but writing in three rows, so that his result (in all five problems) was as follows:

$$\frac{x}{3} = \frac{2}{5} \quad | \cdot 15$$

$$\frac{x^{15}}{3} = \frac{2^{15}}{5}$$

$$5x = 6$$

This picture was very unusual on paper, so we drew the conclusion that even two hours with T-algebra affect the students. Of course, long-term experiments can confirm or refute this assumption.

Now let us see in detail the mistakes made in pre- and post-tests in experimental (exp in Table 4.19) and control groups (con in Table 4.19). Table 4.19 shows the percentage of students who made this mistake in pre-test, in post-test and the last columns show the percentage of the students who made this mistake in pre-test and then in post-test in both groups. Many different kinds of mistakes were made, but Table 4.19 presents only the mistakes that were made in the pre-test by more than ten percents of the students in both groups (experimental and control). This restriction was introduced to enable comparison of mistakes in pre- and post-tests (if only some students in one or another group made some mistakes than it is very hard to say something about the influence of T-algebra). This Table 4.19 does not reflect whether the student made this mistake more than once (either in pre-test or in post-test).

Table 4.19. Mistakes made in pre- and post-tests in experimental and control groups

No	Nature of mistake	Example of mistake	Pre-test		Post-test		Pre-test in post-test	
			% exp	% con	% exp	% con	% exp	% con
1	Minus sign before fraction is taken into account only at first term	$\frac{4y^{\underline{5}}}{3} - \frac{2y+3^{\underline{3}}}{5} = \frac{4^{\underline{1}}}{15} \quad \cdot 1$ $20y - 6y + 9 = 4$	55	56	29	46	52	82
2	Arithmetic mistake in combining and in evaluating	$7s - 9s = 2 - 12$ $-2s = \underline{-24}$	46	50	21	33	45	67
3	In problem <i>Check if number is a solution</i> the equation is solved		40	30	19	10	48	33
4	In problem <i>Reverse sides</i> all variable terms are moved to the left side and all constant terms to the right side		32	23	8	10	24	43
5	In opening parentheses minus sign before parentheses is taken into account only at first term	$9 - 2(3y - 1) = 3 - 2y$ $9 - 6y - 2 = 3 - 2y$	22	46	10	40	47	85
6	Mistake in sign in dividing	$-4y = -8 \quad :(-4)$ $y = \underline{-2}$	21	20	7	7	31	33
7	Arithmetic mistake in dividing	$-0.3y = -1.2 \quad :(-0.3)$ $y = \underline{0.4}$	17	14	9	7	53	50
8	Sign is not changed during moving to other side	$8u - 5u + 15 = 4u + 6$ $8u - 5u - 4u = \underline{15} + 6$	16	20	8	10	50	50
9	Whole number is not multiplied	$\frac{x^{\underline{12}}}{5} + 3 = \frac{7^{\underline{11}}}{10} \quad \cdot 10$ $2x + \underline{3} = 7$	15	23	3	10	18	43

As we assumed, T-algebra affects some error types. We can see that the students from the experimental group made fewer mistakes in sign of second term (mistakes number 1 and 5) in the post-test. The same was observed in the third experiment and we think that showing the error message immediately and directing attention to the mistake and its location (box) are the cause of that. However, T-algebra does not affect mistakes in sign, which are not connected to second term, like mistakes in sign in dividing and during moving to other side (mistakes 6 and 8). The mistake number 6 was not affected by T-algebra in the third experiment either.

A decrease in the number of students from the experimental group who made the mistake number 9 was also predictable. The same was noticed in the third experiment (see Section 4.3). The students often do not write on paper the extender for a number, which is not fraction, but T-algebra does not allow proceeding if some extender is not entered. However, after finding the right extenders for the whole number the students do not forget to multiply them. This causes the reduction of this mistake in the post-test.

T-algebra can also affect learning of algorithms. The students from the experimental group made the mistake number 4 (in problem *Reverse sides*) less frequently than the students from the control group. Experimental students made the mistake number 3 (in problem *Check if number is a solution*) in the post-test more often than the control students, because the teachers request checking the solution of linear equations when solving on paper (sixth type of problems – solve an equation). Therefore, the students solving equation on paper also practice checking the solution. This checking stage is omitted in T-algebra, because the program does not permit incorrect solutions. Consequently, the percentage of the students from the control group who made this mistake decreased, because they had more practice with this type than the experimental students.

It is hard to say whether T-algebra affects arithmetic mistakes or not. The students from the experimental group made the mistake number 2 less frequently than the students from the control group, but the frequency of the mistake number 7 was equal (and even slightly higher). We assumed that T-algebra will not affect arithmetic mistakes and we hope long-term experiments will explain the decrease in the frequency of the mistake number 2.

4.5.2 Conclusions

As we saw in the post-test, the students from the experimental group did better than the students from the control group (the average number of points in the pre-test was almost equal in experimental and control groups). This shows that even short use (2 lessons) of T-algebra affects the results of learning. We also saw a progress of the students from the experimental group to a higher score in the post-test.

The experiment showed that even two hours with T-algebra affect the students' writing style on paper. However, this observation needs a long-term experiment for confirmation or refutation.

Finally, we saw that T-algebra affects some error types, i.e., the students from experimental group made fewer mistakes of certain types (like mistakes in sign of second term in opening parentheses and in multiplying fractions). However, this short period of use of T-algebra gave strange results for arithmetic mistakes; we hope to find an explanation for the change in these mistakes from the long-term experiment.

CONCLUSIONS

Expression manipulation (incl. solving of linear equations, inequalities and systems of linear equations) is an area in the mathematics curriculum where computers can offer help in learning. The aim of this thesis was to create an interactive learning environment for solving linear equations, linear inequalities and systems of linear equations. This goal was realized as a part of the T-algebra interactive learning environment, which enables step-by-step solving of algebra problems in four areas of school mathematics.

We have succeeded in creating a rule dialogue (Action-Object-Input scheme) in T-algebra that gives the student the possibility to learn both solution algorithms and their steps in a similar manner to solving the problem on paper. The design of the rules and rule dialogue is the most important part in T-algebra that distinguishes it from other environments. The designed three-stage dialogue in T-algebra enables the student to make the same mistakes in T-algebra as on paper and the program to give understandable feedback about mistakes. The dialogue has also several advantages for precise diagnosis of student mistakes. T-algebra can diagnose separately incorrect choice of operation, wrong selection of operands for the chosen operation and erroneous application of the selected operation.

In addition to computing only the answers, the domain expert module of T-algebra is able to produce step-by-step solutions similar to pencil-and-paper solutions. We have succeeded in creating in T-algebra a domain expert module 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.

The author of the thesis conducted several experiments that showed that the students find T-algebra easy to learn, easy to use and enjoyable and want to use it more. The students also judge the error messages as understandable and helpful. The time required for learning the dialogue stages is quite short and the solution dialogue is intuitively understandable for the students.

The conducted experiments showed that the students solve more problems with computer than on paper in the same time period and manage to correct all their mistakes by themselves during this time. The experiments also demonstrated that even a short use (2 lessons) of T-algebra for supplementary practice (when the topic had been learned) affects the results of learning. The knowledge of the students from the experimental group (who used T-algebra) was statistically significantly improved, but no statistically significant difference (improvement) could be found in the score of the control group (who did not use T-algebra and solved the same problems on paper). Furthermore, T-algebra affects some error types, i.e., after using T-algebra the students make fewer mistakes of certain type (e.g., mistakes in sign) on paper as well.

Obviously our decisions and ideas need some years of practical classroom trials before they can be finally confirmed. Starting from the school year 2006-2007, tens of teachers in Estonian schools use T-algebra for practice. The results of the school trials and teacher experiences will contribute to and support further development of T-algebra.

We already have some ideas how we can improve T-algebra in the future. First, in the current free input mode, only the equivalence of the entered parts to the parts calculated by T-algebra is checked. In next versions of T-algebra, the program should perform more detailed diagnosis in the free input mode as well. Second, the mode chosen by the teacher cannot be changed during solving in the current version. In the future the program should automatically change the mode to more detailed one (structured or even partial) if the student is in trouble. Moreover, the current design of some rules and algorithms slightly differs from that used in the school. The future development should minimize this difference. A possibility for automatic assessment of student solutions could be added in the next versions as well. This can be done by assigning weights to each solution step in the algorithm and penalties for each error type and for asking hints from the program.

REFERENCES

1. ActiveMath by ActiveMath Team. Available at <http://www.activemath.org/demo.php>. Visited on 17.05.2007.
2. AiM by AiM Team. Available at <http://maths.york.ac.uk/moodle/aiminfo/>. Visited on 17.05.2007.
3. 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.
4. Anderson, J.R. (1988). The expert module. In *Handbook of Intelligent Training Systems*, pp. 21–53, Erlbaum.
5. Anderson, J.R., Corbett, A.T., Koedinger, K. and Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of Learning Sciences* 4: 167–207.
6. Anderson, J.R., Boyle, C.F., Corbett, A. and Lewis, M.W. (1990). Cognitive modelling and intelligent tutoring. *Artificial Intelligence*, 42: 7–49.
7. Aplusix by IMAG-Leibniz laboratory. Available at <http://aplusix.imag.fr/en/index.html>. Visited on 16.05.07.
8. Barnett, R.A. and Kearns, T.J. (1990). *Intermediate Algebra: Structure and Use*. 4th ed. McGraw-Hill.
9. Barnett, R.A. and Ziegler, M.R. (1989). *College Algebra*. 4th ed. McGraw-Hill.
10. Beeson, M. (2002). MathXpert: un logiciel pour aider les élèves à apprendre les mathématiques par l'action. *Sciences et Techniques Educatives*, 9(1–2). English translation ‘MathXpert: Learning Mathematics in the 21st Century’ available at <http://www.mathcs.sjsu.edu/faculty/beeson/Papers/English-ste/English-ste.html>. Visited on 10.05.2007.
11. Beeson, M. (1998). Design Principles of Mathpert: Software to support education in algebra and calculus. In *Computer-Human Interaction in Symbolic Computation*, pp. 89–115, Springer-Verlag.
12. Beeson, M. (1990). Mathpert: a computerized learning environment for algebra, trigonometry, and calculus. *International Journal of Artificial Intelligence in Education*, 1(4): 65–76.
13. Brown, J.S. (1985). Process versus Product: A perspective on tools for communal and informal electronic learning. *Journal of Educational Computing Research*, 1: 179–201.
14. Brown, J.S. and Burton, R.R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2: 155–192.
15. Buchberger, B. (1990). Should Students Learn Integration Rules? *ACM SIGSAM Bulletin*, 24(1): 10–17.
16. Burton, R.R. (1982). Diagnosing bugs in a simple procedural skill. In *Intelligent Tutoring Systems*, pp. 157–183, Academic Press.
17. Büdenbender, J., Frischauf, A., Gogvadze, G., Melis, E., Libbrecht, P. and Ullrich, C. (2002). Using Computer Algebra Systems as Cognitive Tools. In *ITS 2002*, LNCS 2363, pp. 802–810, Springer-Verlag.

18. Cerulli, M. and Mariotti, M.A. (2002). L'Algebrista: un micromonde pour l'enseignement et l'apprentissage de l'algèbre. *Science et techniques éducatives*, 9: 149–170.
19. Char, B.W., Fee, G.J., Geddes, K.O., Gonnet, G.H. and Monagan, M.B. (1986). A tutorial introduction to MAPLE. *Journal of Symbolic Computation*, 2(2): 179–200.
20. Cognitive Tutor by Carnegie Learning, Inc. Available at <http://www.carnegielearning.com/>. Visited on 02.05.2007.
21. Derive by Texas Instruments. Available at http://education.ti.com/educationportal/sites/US/productDetail/us_derive6.html. Visited on 15.05.2007.
22. Hall, R. (2002). An Analysis of Errors Made in the Solution of Simple Linear Equations. *Philosophy of Mathematics Education Journal*, 15.
23. Heffernan, N.T. and Koedinger, K.R. (2000). Intelligent Tutoring Systems are Missing the Tutor: Building a More Strategic Dialog-Based Tutor. In *Proceedings of the AAAI Fall Symposium on Building Dialogue Systems for Tutorial Applications*, pp. 14–19.
24. Issakova, M. (2007a). Do First Year Students Know how to Solve Simple Linear Equations? An Experiment with T-algebra. In *Proceedings of the 8th International Conference on Technology in Mathematics Teaching (ICTMT8)*, 6 p., Hradec Králové, Czech Republic.
25. Issakova, M. (2007b). How does an interactive learning environment affect the students' learning? In *Proceedings of First Central- and Eastern European Conference on Computer Algebra- and Dynamic Geometry Systems in Mathematics Education (CADGME)*, Pecs, Hungary, (to appear).
26. Issakova, M. (2006a). Domain Expert Module for Step-By-Step Linear Equation Solving. In *Proceedings of The 11th Asian Technology Conference in Mathematics (ATCM 2006)*, pp. 193–202, Hong Kong, China.
27. Issakova, M. (2006b). Comparison of student errors made during linear equation solving on paper and in interactive learning environment. In *Proceedings DES-TIME-2006: Dresden International Symposium on Technology and its Integration into Mathematics Education 2006*, 20 p., Dresden, Germany.
28. Issakova, M. (2006c). Learning Linear Equation Solving Algorithm and Its Steps in Intelligent Learning Environment. In *ITS 2006 Proceedings*, LNCS 4053, pp. 725–727, Springer-Verlag.
29. Issakova, M. (2006d). Intelligent Environment for Learning Linear Equation Solving Algorithm and Its Steps. In *Proceedings of the Student Track ITS 2006*, pp. 8–17, Jhongli, Taiwan.
30. Issakova, M. (2005). Possible Mistakes During Linear Equation Solving On Paper And In T-algebra Environment. In *Proceedings of the 7th International Conference on Technology in Mathematics Teaching (ICTMT7)*, volume 1, pp. 250–258, Bristol, UK.
31. Issakova, M. and Lepp, D. (2004). Rule dialogue in problem solving environment T-algebra. In *Proceedings TIME-2004: Montreal International Symposium on*

- Technology and its Integration into Mathematics Education*, 16 p., Montreal, Canada.
32. Issakova, M., Lepp, D. and Prank, R. (2006). T-algebra: Adding Input Stage To Rule-Based Interface For Expression Manipulation. *International Journal for Technology in Mathematics Education*, 13(2): 89–96.
 33. Issakova, M., Lepp, D. and Prank, R. (2005). Input Design in Interactive Learning Environment T-algebra. In *Proceedings ICALT–2005: The 5th IEEE International Conference on Advanced Learning Technologies*, pp. 489–491, Kaohsiung, Taiwan.
 34. Kasemaa, P. and Lind, A. (1997). *Mathematics textbook for V–IX grades* (in Estonian). Koolibri.
 35. Kutzler, B. (1996). *Introduction to DERIVE for Windows*. Gutenberg-Werbering GmbH.
 36. Lepik, M., Nurk, E., Telgmaa, A. and Undusk, A. (2000). *Mathematics for VIII grade* (in Estonian). Koolibri.
 37. Lepp, D. (2005). Extended Solution Step Dialogue In Problem Solving Environment T-algebra. In *Proceedings of the 7th International Conference on Technology in Mathematics Teaching (ICTMT7)*, volume 1, pp. 267–274, Bristol, UK.
 38. Lepp, D. (2003). Program for exercises on operations with polynomial. In *Proceedings of the 6th International Conference Technology in Mathematics Teaching (ICTMT6)*, pp. 365–369, Volos, Greece.
 39. LiveMath by MathMonkeys. Available at <http://www.livemath.com/>. Visited on 08.05.2007.
 40. Maple by Waterloo Maple Inc. Available at <http://www.maplesoft.com/>. Visited on 15.05.2007.
 41. Mathematica by Wolfram Research. Available at <http://www.wolfram.com/>. Visited on 09.05.2007.
 42. Math-Teacher by MATH-KAL. Available at <http://www.mathkalusa.com>. Visited on 08.05.2007.
 43. MathXpert by Help With Math. Available at <http://www.helpwithmath.com/>. Visited on 10.05.2007.
 44. McArthur, D., Stasz, C. and Hotta, J. (1987). Learning problem-solving skills in algebra. *The Journal of Educational Technology Systems*, 15(3): 303–324.
 45. McKeague, C.P. (1979). *Intermediate Algebra*. Academic Press.
 46. Melis, E., Andrès, E., Büdenbender, J., Frischauf, A., Gogvadze, G., Libbrecht, P., Pollet, M. and Ullrich, C. (2001). ActiveMath: A Generic and Adaptive Web-Based Learning Environment. *International Journal of Artificial Intelligence in Education*, 12: 385–407.
 47. Mitrovic, A. and Ohlsson, S. (1999). Evaluation of a Constraint-Based Tutor for a Database Language. *International Journal of Artificial Intelligence in Education*, 10: 238–256.
 48. Ms. Lindquist by Neil Heffernan. Available at <http://www.algebratutor.org/>. Visited on 11.05.2007.

49. Nicaud, J.F., Chaachoua, H. and Bittar, M. (2006). Automatic Calculation of Students' Conceptions in Elementary Algebra from Aplusix Log Files. In *ITS 2006 Proceedings*, LNCS 4053, pp. 433–442, Springer-Verlag.
50. Nicaud, J.F., 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: 169–211.
51. Nicaud, J.F., Bouhineau, D., Varlet, C. and Nguyen-Xuan, A. (1999). Towards a product for teaching formal algebra. In *Proceedings of Artificial Intelligence in Education*, pp. 207–217.
52. Nurk, E., Telgmaa, A. and Undusk, A. (2000). *Mathematics for VII grade* (in Estonian). Koolibri.
53. Oliver, J. and Zukerman, I. (1990). DISSOLVE: A system for the generation of human oriented solutions to algebraic equations. In *Proceedings of AI'88*, LNCS 406, pp. 91–107, Springer-Verlag.
54. Pais, E. (1999). *Mathematics for VIII grade* (in Estonian). Avita.
55. Pais, E. (1998). *Mathematics for VII grade* (in Estonian). Avita.
56. Passier, H. and Jeuring, J. (2006). Feedback in an interactive equation solver. In *WebALT 2006 Proceedings*, pp. 53–68, Eindhoven, Netherlands.
57. Prank, R. (1991). Using Computerised Exercises on Mathematical Logic. *Informatik-Fachberichte*, 292: 34–38, Springer-Verlag.
58. Prank, R. and Viira, H. (1991). Algebraic Manipulation Assistant for Propositional Logic. *Computerised Logic Teaching Bulletin*, 4(1): 13–18.
59. Prank, R., Issakova, M., Lepp, D., Tõnisson, E. and Vaiksaar, V. (2007). Integrating rule-based and input-based approaches for better error diagnosis in expression manipulation tasks. In *Symbolic Computation and Education*, World Scientific Publishing Co., (to appear).
60. Prank, R., Issakova, M., Lepp, D. and Vaiksaar, V. (2006a). Designing Next-Generation Training and Testing Environment for Expression Manipulation. In *International Conference on Computational Science (ICCS 2006)*, Part I, LNCS 3991, pp. 928–931, Springer-Verlag.
61. Prank, R., Issakova, M., Lepp, D., Vaiksaar, V. and Tõnisson, E. (2006b). Problem solving environment T-algebra. In *Proceedings of 7th International Conference Teaching Mathematics: Retrospective and Perspectives*, pp. 190–197, Tartu, Estonia.
62. Quigley, M. (1989). A Simple Algebra Tutor. *Journal of Artificial Intelligence in Education*, 1(1): 41–52.
63. Ravaglia, R., Alper, T., Rozenfeld, M. and Suppes, P. (1998). Successful pedagogical applications of symbolic computation. In *Computer-Human Interaction in Symbolic Computation*, pp. 61–88, Springer-Verlag.
64. Razzaq, L.M. and Heffernan, N.T. (2004). Tutorial Dialog in an Equation Solving Intelligent Tutoring System. In *ITS 2004 Proceedings*, LNCS 3220, pp. 851–853, Springer-Verlag.
65. Sangwin, C.J. (forthcoming). Assessing Elementary Algebra with STACK. *International Journal of Mathematical Education in Science and Technology*.

66. Sangwin, C.J. (2005). Making Mathematical Distinctions In CAA With Computer Algebra. In *Proceedings of the 7th International Conference on Technology in Mathematics Teaching (ICTMT7)*, volume 1, pp. 292–299, Bristol, UK.
67. Sangwin, C.J. (2004). Assessing mathematics automatically using computer algebra and the internet. *Teaching Mathematics and its Applications*, 23(1): 1–14.
68. Sleeman, D. (1984). An Attempt to Understand Students' Understanding of Basic Algebra. *Cognitive Science*, 8: 413–437.
69. Sleeman, D. and Smith, M.J. (1981). Modelling student's problem solving. *Artificial Intelligence*, 16(2): 171–187.
70. Stacey, K., Chick, H. and Kendal, M. (eds.) (2004). *The Future of the Teaching and Learning of Algebra: The 12th ICMI Study*. Kluwer Academic Publishers.
71. STACK by Chris Sangwin. Available at <http://www.stack.bham.ac.uk/>. Visited on 17.05.2007.
72. Strickland, P. and Al-Jumeily, D. (1999). A Computer Algebra System for improving student's manipulation skills in Algebra. *The International Journal of Computer Algebra in Mathematics Education*, 6(1): 17–24.
73. The Learning Equation (TLE) by ITP Nelson. Available at <http://www.thomsonedu.com/tle/>. Visited on 08.05.2007.
74. Thompson, P. and Thompson, A. (1987). Computer presentations of structure in algebra. In *Proceedings of the Eleventh Annual Meeting of the International Group for the Psychology of Mathematics Education*, volume 1, pp. 248–254.
75. Tiger Leap Foundation website. Available at <http://www.tiigrihype.ee/?setlang=eng>. Visited on 02.05.2007.
76. Tonisson, E., Issakova, M., Lepp, D. and Prank, R. (2006). Intelligent problem solving environment T-algebra. In *Proceedings DES-TIME-2006: Dresden International Symposium on Technology and its Integration into Mathematics Education 2006*, 14 p., Dresden, Germany.
77. Tõnso, T. (2002). *Mathematics for VII grade* (in Estonian). Mathema.
78. Veelmaa, A. (2000). *Mathematics for VIII grade* (in Estonian). Mathema.
79. Walden, J. (1997). *Mathpert Calculus Assistant: User's Guide*. Mathpert Systems, an Imprint of Recognix.
80. Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Morgan Kaufmann Publishers.
81. Xambo, S., Eixarch, R. and Marques, D. (2002). WIRIS: An Internet platform for the teaching of mathematics in large educational communities. *Contributions to Science*, 2(2): 269–276.
82. Zuckerman, M.M. (1976). *Intermediate algebra*. W. W. Norton & Company.

SUMMARY IN ESTONIAN

Lineaarvõrrandite, lineaarvõrratuste ja lineaarsete võrrandisüsteemide lahendamine interaktiivses õpikeskkonnas

Kokkuvõte

Algebralised ülesanded (sh. lineaarvõrrandite, lineaarvõrratuste ja lineaarsete võrrandisüsteemide lahendamine) on matemaatika ainekavas teema, mis tekitab paljudele õpilastele raskusi ja mis on küllalt töömahukas õpetajate jaoks. Õpilastel ülesannete lahendamise ajal tekkivad probleemid on küllalt erinevad ja vajavad õpetajalt pikemat süvenemist, et detailidest aru saada. Traditsioonilise õppetehnoloogia korral ei jõua õpetaja anda kogu klassile õigeaegselt nõuandeid ega juhtida tähelepanu tehtud vigadele. Suurte andmemahutude kiire analüüsi vajadus näitab, et kompuuteriseeritud lahenduskeskkondade kasutamine võib parandada algebraliste ülesannete lahendamise oskust ja selle oskuse testimist.

Olemasolevad süsteemid ei vasta kõikidele nõuetele korraga. Mõnesid süsteeme võib kasutada ainult õpetuseks (need süsteemid ei lase teha palju vigu), teisi ainult kontrollimiseks (need süsteemid ei paku abi ja vigade diagnoosi). Praeguseks praktiliselt veel polegi maailmas lahendamise keskkondi, mis vastaksid kõikidele nõuetele täielikult.

Käesoleva väitekirja eesmärk ongi disainida, realiseerida ja testida uut liiki keskkonda, mis aitaks õppida lineaarvõrrandite, lineaarvõrratuste ja lineaarsete võrrandisüsteemide lahendamist ning hinnata ja diagnoosida lünkasid õpilaste teadmistes ja oskustes.

Väitekiri on üks osa suuremast projektist, milles on loodud programm sammukaupa ülesannete lahendamiseks neljas matemaatika valdkonnas: arvuliste avaldiste väärtuste arvutamine, tehted murdudega, lineaarvõrrandite, võrratuste ning lineaarsete võrrandisüsteemide lahendamine, hulkliikmete lihtsustamine. Väitekiri tutvustab uudse disainiga interaktiivset õpisüsteemi T-algebra, mille loomises osales väitekirja autor. Disain on uudne, sest see kombineerib kahte teatud lähenemist: reeglipõhist ja sisestamisepõhist. Ühendamise tulemust on nimetatud Tegevus-Objekt-Sisestamine (Action-Object-Input) skeemiks. Väitekirja autori peamine panus selle projekti jaoks on lineaarvõrrandite, lineaarvõrratuste ja lineaarsete võrrandisüsteemide valdkonna reeglite dialoogide ja ülesannete tüüpide disainimine koos vigade diagnoosiga, programmeerimine ja katsetamine.

T-algebra autoritel õnnestus luua selline skeem, mis lubab õppida nii lahendamise algoritme kui ka erinevate reeglite rakendamist. Reeglite ning reeglite dialoogide disain on kõige tähtsam osa T-algebras, mis eristab teda

teistest keskkondadest. Disainitud kolmeetapiline dialoog T-algebras annab õpilasele võimaluse teha samu vigu nagu ka paberil lahendades ja programmile piisavalt informatsiooni, et diagnoosida tehtud vigu ning pakkuda arusaadavat tagasisidet ja abi. T-algebra valdkonna ekspert oskab mitte ainult näidata vastust, aga ka luua paberil kirjutatud lahendustega sarnaseid sammukaupa lahendusi.

Väitekirja autor korraldas mitu eksperimenti, mis näitasid, et õpilased arvavad, et T-algebrat on lihtne õppida, lihtne ja nauditav kasutada ja et õpilased tahavad seda rohkem kasutada. Samuti õpilased leidsid, et veateated on arusaadavad ja kasulikud. Dialoogi õppimiseks vajalik aeg on üsna lühike ja lahendusdialoog on õpilasele intuiitiivselt arusaadav.

Eksperimendid näitasid, et õpilased lahendavad sama ajaga arvuti taga rohkem ülesandeid kui paberil ja jõuavad selle aja jooksul ise kõik vead ära parandada. Eksperimendid näitasid samuti, et isegi T-algebra lühike kasutamine (2 tundi) täiendõppeks (kui teema on juba ära õpitud) mõjutab õppetulemusi. Katserühma õpilastel (kes kasutasid T-algebrat) suurenesid teadmised katse käigus statistiliselt oluliselt, kuid kontrollrühma õpilastel (kes ei kasutanud T-algebrat) olulist muutust ei toimunud. Samuti T-algebra mõjutab teatud vigade tüüpe, st peale T-algebra kasutamist õpilased teevad teatud sorti vigu (näiteks märgivigu) vähem ka paberil.

ACKNOWLEDGEMENTS

I would like to thank my parents, my brother and my husband Dmitri Lepp who gave me the opportunity to concentrate on my PhD thesis. I would also like to thank all my friends for their support.

I wish to thank my supervisor Rein Prank for pleasant cooperation. I thank my colleague Eno Tõnisson who encouraged me not to give up in difficult moments. I am also grateful to Piret Luik for helping me in statistical analysis.

I would like to thank the mathematics teachers Mart Oja and Maire Oja for their cooperation and their students for solving different tests. Many thanks also to mathematics teachers Sirje Pihlap, Urve Olesk, Eha Kuld and Janika Kaljula and their students for participating in the experiment. I thank Tiina Lasn for giving me permission to use her 1st year students for experiment.

I thank translator Alar Helstein for correcting my English.

I was supported by the ‘Tiger Leap’ computerization programme for Estonian schools as they financed the T-algebra project. I was partially supported by the Estonian Science Foundation under grants No. 5272 and No. 7180. I was furthermore partially supported by the targeted financing project No. SF0182712s06 “The methods, environments, and applications for solving large and complex computational problems”.

I was supported by the Estonian Information Technology Foundation and Estonian Doctoral School in Information and Communication Technologies.

APPENDIX A

Backus-Naur Form full description of expressions

Backus-Naur Form full description of expressions in T-algebra is following:

<digit> ::=	0 1 2 3 4 5 6 7 8 9
<non-zero digit> ::=	1 2 3 4 5 6 7 8 9
<integer> ::=	<non-zero digit> <digit> <non-zero digit> <integer>
<power> ::=	^ <integer> ^ + <integer> ^ - <integer>
<decimal separator> ::=	, .
<zero> ::=	0 0 <zero>
<decimal> ::=	<integer> <decimal separator> <integer> <integer> <decimal separator> <zero> <integer>
<number> ::=	<integer> <integer> <power> <decimal> <decimal> <power> <numerical fraction> <mixed number>
<numerical atom> ::=	<numerical parentheses> <numerical parentheses> <numerical atom> <numerical parentheses> <number> <numerical parentheses> <number> <numerical atom>
<numerical term> ::=	<number> <numerical atom> <number> <numerical atom>
<numerical mul div> ::=	<numerical term> <numerical term> * <numerical mul div> <numerical term> : <numerical mul div>
<numerical sign mul div> ::=	+ <numerical mul div> - <numerical mul div>
<numerical non-sign sum sub> ::=	<numerical mul div> <numerical mul div> + <numerical non-sign sum sub> <numerical mul div> - <numerical non-sign sum sub>
<numerical sum sub> ::=	<numerical non-sign sum sub> <numerical sign mul div> <numerical sign mul div> + <numerical non-sign sum sub> <numerical sign mul div> - <numerical non- sign sum sub>
<numerical parentheses> ::=	[<numerical sum sub>] (<numerical sum sub>) [<numerical sum sub>] <power> (<numerical sum sub>) <power>
<numerical fraction> ::=	<numerical sum sub> / <numerical sum sub>
<mixed number> ::=	<integer> <numerical fraction>
<letter> ::=	a b c d e f g h i j k l m n o p q r s t u v w x y z
<variable> ::=	<letter> <letter> <power>

<atom> ::=	<variable> <parentheses> <variable> <atom> <parentheses> <atom> <parentheses> <number> <parentheses> <number> <atom>
<term> ::=	<number> <atom> <number> <atom>
<mul div> ::=	<term> <term> * <mul div> <term> : <mul div>
<sign mul div> ::=	+ <mul div> - <mul div>
<non-sign sum sub> ::=	<mul div> <mul div> + <non-sign sum sub> <mul div> - <non-sign sum sub>
<sum sub> ::=	<non-sign sum sub> <sign mul div> <sign mul div> + <non-sign sum sub> <sign mul div> - <non-sign sum sub>
<parentheses> ::=	[<sum sub>] (<sum sub>) [<sum sub>] <power> (<sum sub>) <power>
<fraction> ::=	<sum sub> / <sum sub>
<equation inequality signs> ::=	= < > <= >=
<equation inequality> ::=	<sum sub> <equation inequality signs> <sum sub>
<system> ::=	<equation inequality> & <equation inequality> <equation inequality> & <system>
<expression> ::=	<system> <equation inequality> <sum sub>

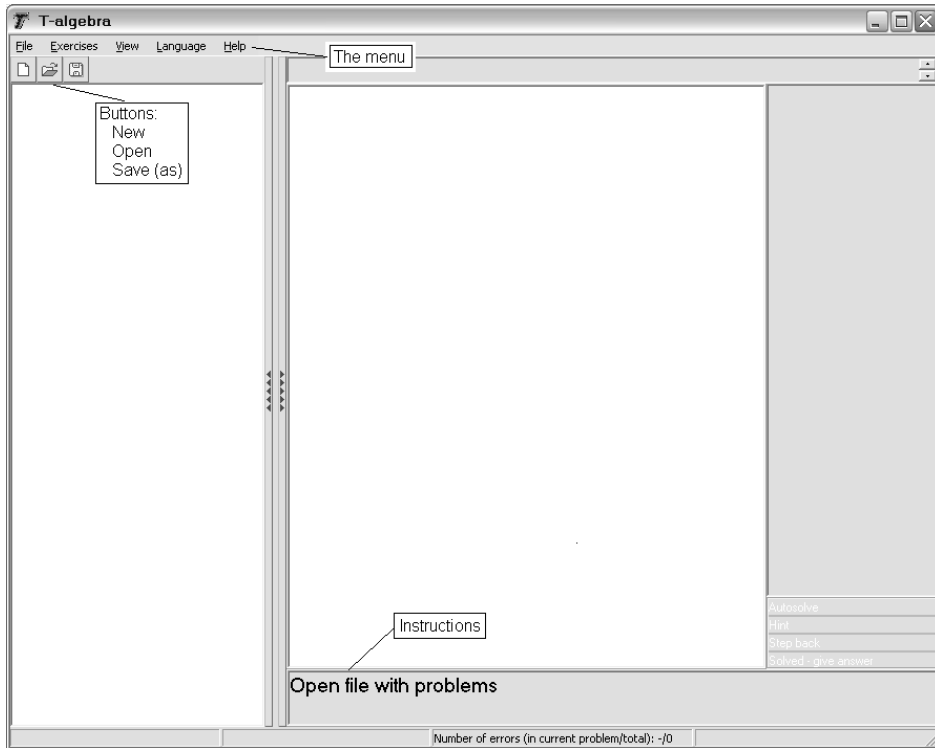
APPENDIX B

Quick start to T-algebra student's program

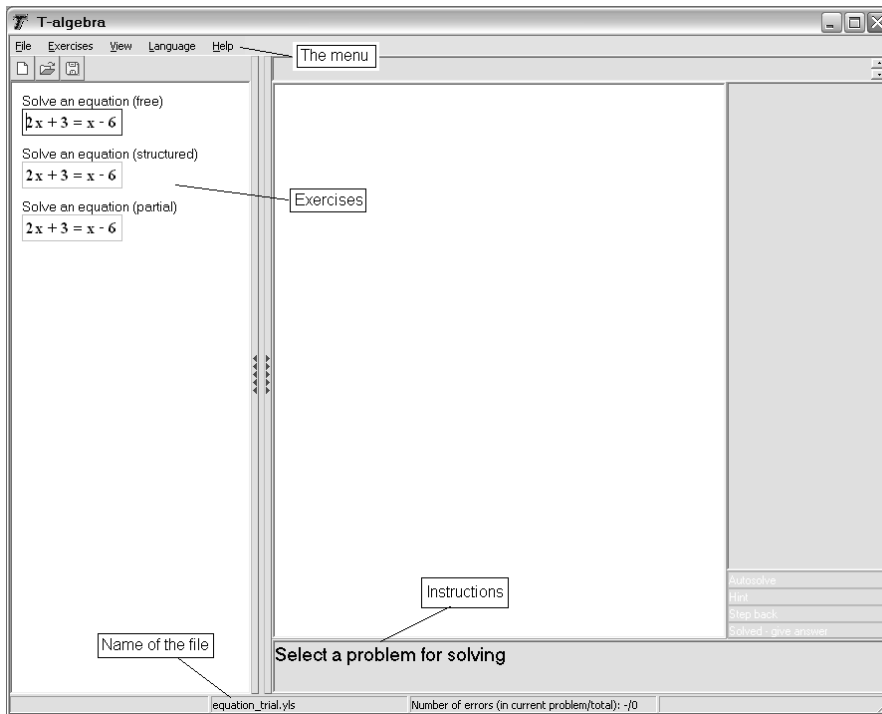
This document was used in T-algebra workshop during DES-TIME conference (Tonisson et al., 2006).

1. Run the program T-algebra (double click the program icon).

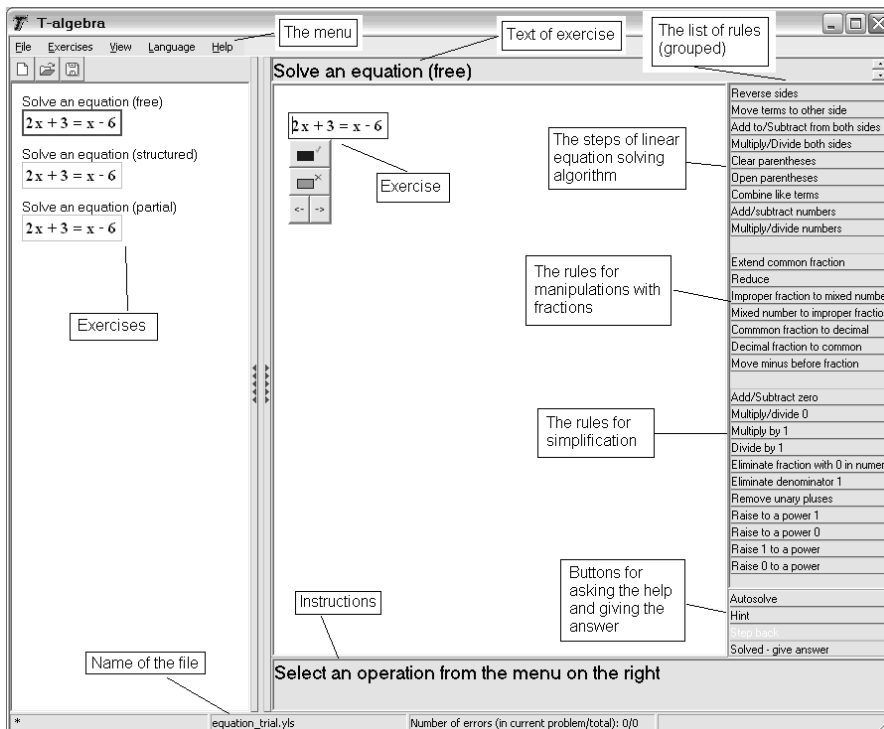
Description of the program window:




2. Open file with exercises:
 - o From the *File* menu choose *Open* or click the button *Open file*.
3. Choose the file *equation_trial.yls*:
 - o On the left side you will see all exercises of this file.




4. Choose the first exercise (Solve an equation (free) $2x+3=x-6$). For that, double click the left mouse button on the problem expression.
 - One click on the exercise makes the line around the exercise blue.
 - Double click on the exercise makes the line around the exercise red and the exercise is opened on the right.




5. Solve an equation:

- The general scheme of solving in T-algebra:
 - 1) Choose the rule (tell the program what you intend to do);
 - 2) Look at the instructions (the program shows what you should do next);
 - 3) Mark the necessary parts (tell the program which parts you will modify);
 - 4) Look at the instructions;
 - 5) Enter the necessary parts (tell the program what is the result).
- Solution process of this equation $2x+3=x-6$:
 - 1) Choose the rule *Move terms to other side*;
 - 2) Mark 3 (or +3) with the mouse (the background should be blue);
 - 3) Press the button , the background of 3 (or +3) becomes green;
 - 4) Mark x with the mouse (the background should be blue);

- 5) Press the button , the background of x becomes green

$$2x + 3 = x - 6$$

- 6) Confirm the choice – press the button 


- 7) The box(es) and virtual keyboard appear on the next line;

$$2x + 3 = x - 6$$

Move terms to other side

$$2x \square = -6 \square$$

{	x^0	x^2	x^3	x^4	<	<<	>>	>	=	-	↶	↷	↵	✓
0	1	2	3	4	5	6	7	8	9	.	↶	↷	↵	✗
a	b	c	s	t	v	w	x	y	z	:	←	↓	→	👤

- 8) If something went wrong (wrong rule, wrong parts) and you want to go back, press the button 


- 9) Enter the result of moving in the boxes;

$$2x + 3 = x - 6$$

Move terms to other side

$$2x - x = -6 - 3$$

{	x^0	x^2	x^3	x^4	<	<<	>>	>	=	-	↶	↷	↵	✓
0	1	2	3	4	5	6	7	8	9	.	↶	↷	↵	✗
a	b	c	s	t	v	w	x	y	z	:	←	↓	→	👤

- 10) Confirm the input – press the button 

$$2x + 3 = x - 6$$

Move terms to other side

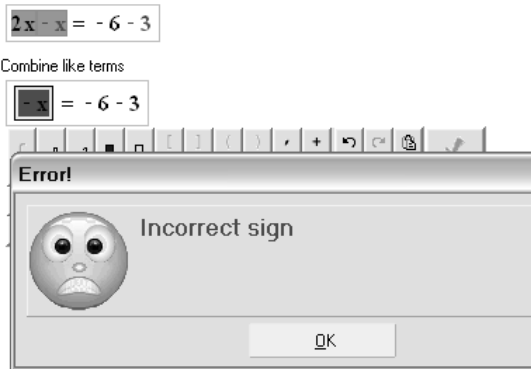
$$2x - x = -6 - 3$$





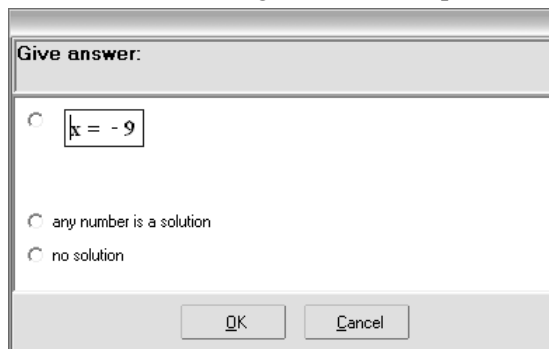
<- ->

- 11) If something went wrong (wrong rule, wrong parts) and you want to go back, press the button *Step back*;
- 12) Choose the next rule (*Combine like terms*);
- 13) Mark like terms, confirm the choice;
- 14) Enter a wrong result in the box: $-x$. The program will display an error message and highlight the wrong part with red background. It is impossible to press the *OK* button for

three seconds; during this time you should read the error message;



- 15) Correct the mistake, confirm the input;
- 16) It is possible to ask the program for help at each step and at each stage of the step (try all the possibilities):
 - i. You can ask the program which rule should be applied next – the button *Hint* (choose the rule *Add/Subtract numbers*);
 - ii. You can ask help for marking the parts – button  (confirm the choice);
 - iii. You can ask help for entering the result – button  (confirm the choice);
- 17) If the exercise is solved, tell it to the program:
 - i. Press the button *Solved – give answer*;
 - ii. Choose the right answer and press the *OK* button;



- 18) The background of the text of this exercise on the left panel becomes green.

6. The input stage in the program can proceed in three different modes:
- 1) **free** input: only one box for the whole result ;
 - 2) **structured** input: several boxes for entering the result, separate boxes for sign, coefficient and/or variable, exponent ;
 - 3) **partial** input: the program fills out some parts of the result and leaves empty boxes only for most important parts: sign, coefficient, exponent x.
7. The first exercise was in free input mode, the second is in structured input mode and the last is in partial input mode. Solve these equations, compare how the solving process is handled, what parts should be entered.

NB! The virtual keyboard shows, what is possible to enter into the selected box.

For example:



Only numbers should and can be entered in this box (coefficient box).

APPENDIX C

Full list of designed problem types

In this appendix I represent the full list of designed problem types in the domain of linear equation, linear inequality and system of linear equations using the structure of description presented in Section 3.3. The examples described in Section 3.3 are repeated here as well (to present a full list in one place).

Problem type *Combine like terms*

Typical texts: combine like terms.

Expression: sum of monomials.

Constraints (for expression):

- sum should contain like terms;
- parentheses can be only around one monomial.

Parameters: none.

Rules:

- Combine like terms;
- Add/Subtract numbers;
- Clear parentheses;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Algorithm for combining;
3. Rule *Clear parentheses*.

Example of generated solution:

Text of problem: Combine like terms.

$$\begin{aligned} & 3x^2y - 4xy^2 + (-6yx^2) - 5 + 5xyy + 2 + 2y^2x + (-2y) = \\ & \text{Add/Subtract numbers} \\ & = 3x^2y - 4xy^2 + (-6yx^2) - 3 + 5xyy + 2y^2x + (-2y) = \\ & \text{Combine like terms} \\ & = -3x^2y - 4xy^2 - 3 + 5xyy + 2y^2x + (-2y) = \\ & \text{Combine like terms} \\ & = -3x^2y + 3xy^2 - 3 + (-2y) = \\ & \text{Clear parentheses} \\ & = -3x^2y + 3xy^2 - 3 - 2y \\ & \text{Answer:} \\ & -3x^2y + 3xy^2 - 3 - 2y \end{aligned}$$

Solved form: simplified monomial or simplified sum of monomials, where all like terms are combined and all parentheses are cleared.

Answer: solved form.

Problem type *Open parentheses and combine*

Typical texts: open parentheses (if like terms will not appear in one sum); open parentheses and combine like terms.

Expression: any expression (including linear equation, linear inequality and system of linear equations).

Constraints (for expression): expression should contain parentheses (or brackets) that can be opened or cleared, i.e., expression should contain either product consisting of number and parentheses or just expression in parentheses.

Parameters: none.

Rules:

- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Open parentheses*;
3. Rule *Clear parentheses*;
4. Rule *Multiply/Divide numbers*;
5. Algorithm for combining.

Example of generated solution:

Text of problem: Open parentheses and combine like terms.

$$13(3x + 5) - (4x - 3) =$$

Open parentheses

$$= 39x + 65 - (4x - 3) =$$

Clear parentheses

$$= 39x + 65 - 4x + 3 =$$

Add/subtract numbers

$$= 39x + 68 - 4x =$$

Combine like terms

$$= 35x + 68$$

Answer:

$$35x + 68$$

Solved form: expression, where all parentheses and brackets are opened or cleared and all like terms are combined.

Answer: solved form.

Problem type *Check the solution of equation*

Typical texts: check if number ... is a solution of equation.

Expression: equation.

Constraints (for expression): linear equation should be in one variable.

Parameters: value (one number) of variable to be checked.

Rules:

- Substitute variable;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Raise number to a power;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Rules for fractions;

- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Substitute variable*;
3. Rule *Raise number to a power*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Algorithm for combining;
6. Rules *Open parentheses* and *Clear parentheses*;
7. Rule *Mixed number to improper fraction* (if equation is not yet in the form $number = number$).

Example of generated solution:

Text of problem: Check if number 4 is a solution of equation.

$5x + 4(1 - x) = 2x$	Add/subtract numbers $20 + 4(-3) = 8$
Substitute variable $5 \cdot 4 + 4(1 - 4) = 2 \cdot 4$	Multiply/divide numbers $20 - 12 = 8$
Multiply/divide numbers $20 + 4(1 - 4) = 2 \cdot 4$	Add/subtract numbers $8 = 8$
Multiply/divide numbers $20 + 4(1 - 4) = 8$	Answer: is a solution

Solved form: $number = number$, where

- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);
- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer: Given number ...

- is a solution;
- is not a solution.

Problem type *Reverse sides of equation*

Typical texts: reverse equation sides.

Expression: equation.

Constraints (for expression): if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Reverse sides;
- Move terms to other side;
- Multiply/Divide both sides;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Reverse sides* (if equation is not changed yet);
3. Rule *Multiply/Divide both sides* (if equation is equivalent to given equation where all signs are changed to opposite);
4. Rule *Move terms to other side* (to get all terms on the correct side if some part of terms was already moved to other side).

Example of generated solution:

Text of problem: Reverse equation sides.

$$5x + 7 = 7 - 5x$$

Reverse sides

$$7 - 5x = 5x + 7$$

Answer:

$$7 - 5x = 5x + 7$$

Solved form: equation with changed sides.

Answer: solved form.

Problem type *Move terms to correct side of equation*

Typical texts: move all variable terms to the left side and all constant terms to the right side.

Expression: equation.

Constraints (for expression):

- equation should contain variable terms on the right side or constant terms on the left side (student should move some terms);
- left side and right side of equation should be monomial or polynomial without multiplication and division signs and without parentheses and brackets;
- if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Move terms to other side;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Move terms to other side*.

Example of generated solution:

Text of problem: Move all variable terms to the left side and all constant terms to the right side.

$$14x - 5 = 4 + 9x$$

Move terms to other side

$$14x - 9x = 4 + 5$$

Answer:

$$14x - 9x = 4 + 5$$

Solved form: equation where all variable terms are moved to the left side and all constant terms to the right side.

Answer: solved form.

Problem type *Move terms to correct side of equation and combine*

Typical texts: move all variable terms to the left side and all constant terms to the right side and then combine like terms.

Expression: equation.

Constraints (for expression):

- equation should contain variable terms on the right side or constant terms on the left side (student should move some terms);
- left side and right side of equation should be monomial or polynomial without multiplication and division signs and without parentheses and brackets;
- if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Move terms to other side;
- Combine like terms;
- Add/Subtract numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Move terms to other side*;
3. Algorithm for combining.

Example of generated solution:

Text of problem: Move all variable terms to the left side and all constant terms to the right side and then combine like terms.

$$14x - 5 = 4 + 9x$$

Move terms to other side

$$14x - 9x = 4 + 5$$

Add/subtract numbers

$$14x - 9x = 9$$

Combine like terms

$$5x = 9$$

Answer:

$$5x = 9$$

Solved form: equation where all variable terms are moved to the left side and all constant terms to the right side and after that all like terms are combined.

Answer: solved form.

Problem type *Multiply both sides of equation by common denominator*

Typical texts: multiply both sides of the equation by common denominator.

Expression: equation.

Constraints (for expression):

- equation should contain common fraction(s) or mixed number(s);
- equation should not contain parentheses or brackets;
- fractions should be with integer denominator;
- equation should not contain multiplication sign;
- if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Multiply/Divide both sides;
- Multiply/Divide numbers;
- Mixed number to improper fraction;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Multiply/Divide numbers*;
3. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides, see the Section 3.2.1);
4. Rule *Multiply/Divide both sides*.

Example of generated solution:

Text of problem: Multiply both sides of the equation by common denominator.

$$\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15}$$

Multiply/Divide both sides

$$\frac{4y}{3} \cdot \frac{5}{5} - \frac{2y+3}{5} \cdot \frac{3}{3} = \frac{4}{15} \cdot \frac{1}{1} \quad | \cdot 15$$

Multiply/Divide both sides

$$20y - 6y - 9 = 4$$

Answer:

$$20y - 6y - 9 = 4$$

Solved form: simplified (using simplifying rules and rule *Multiply/Divide numbers*) multiplied equation without common fractions and mixed numbers.

Answer: solved form.

Problem type *Divide equation sides by variable coefficient or by common divider*

Typical texts: divide equation sides by variable coefficient; divide equation sides by common divider.

Expression: equation.

Constraints (for expression):

- equation should be in the form *variable term = constant term* (where variable term should not be just variable, i.e., should contain coefficient and/or sign) or terms should have common divider;
- equation should not contain parentheses or brackets;
- if equation is numerical, then it should be true.

Parameters: none.

Rules:

- Multiply/Divide both sides;
- Mixed number to improper fraction;
- Reduce;
- Multiply/Divide numbers;
- Improper fraction to mixed number;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Mixed number to improper fraction*;
3. Rule *Multiply/Divide both sides*;
4. Rule *Multiply/Divide numbers*;
5. Rule *Reduce*;
6. Rule *Improper fraction to mixed number*.

Example of generated solution:

Text of problem: Divide equation sides by variable coefficient.

$$\frac{2}{5}v = -\frac{4}{15}$$

Multiply/Divide both sides

$$\frac{2}{5}v = -\frac{4}{15} \quad | \cdot 15$$

Multiply/Divide both sides

$$6v = -4$$

Multiply/Divide both sides

$$6v = -4 \quad | : 6$$

Multiply/Divide both sides

$$v = -\frac{2}{3}$$

Answer:

$$v = -\frac{2}{3}$$

Solved form: equation in the form *variable = constant term* (where constant term is transformed to normal form) or equation where terms do not have a common divider.

Answer: solved form.

Problem type *Solve linear equation*

Typical texts: solve linear equation.

Expression: equation.

Constraints (for expression): equation should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;

- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Reverse sides* (if linear equation is in the form: *number = variable*);
3. Rules *Open parentheses* and *Clear parentheses*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Rule *Multiply/Divide both sides* for removing fractions (multiplication);
6. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
7. Algorithm for combining;
8. Rule *Move terms to other side*;
9. Rule *Multiply/Divide both sides* for isolating variable (division).

Example of generated solution:

Text of problem: Solve an equation.

$68 - 3x = 3 + 4(2x - 3)$ <p>Open parentheses</p> $68 - 3x = 3 + 8x - 12$ <p>Add/subtract numbers</p> $68 - 3x = -9 + 8x$ <p>Move terms to other side</p> $-3x - 8x = -9 - 68$ <p>Add/subtract numbers</p> $-3x - 8x = -77$	<p>Combine like terms</p> $-11x = -77$ <p>Multiply/Divide both sides</p> $-11x = -77 \quad : (-11)$ <p>Multiply/Divide both sides</p> $x = 7$ <p>Answer:</p> $x = 7$
---	--

Solved form: *variable = number* or *number = number*, where

- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);
- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer:

- solved form (i.e., *variable = number* or *number = number*; for example, $x = 7$);
- there is no solution;
- any number is solution.

Problem type *Check numerical inequality*

Typical texts: check the validity of numerical inequality.

Expression: numerical inequality.

Constraints (for expression): none.

Parameters: none.

Rules:

- Add/Subtract numbers;
- Multiply/Divide numbers;
- Raise number to a power;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Raise number to a power*;
3. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
4. Algorithm for combining;
5. Rules *Open parentheses* and *Clear parentheses*;
6. Rule *Mixed number to improper fraction* (if equation is not yet in form *number = number*).

Example of generated solution:

Text of problem: Check the validity of numerical inequality.

$-3 \cdot 2 + 4 \leq 16 : 4 - 6$	Add/subtract numbers
$-6 + 4 \leq 16 : 4 - 6$	$-2 \leq 4 - 6$
Multiply/divide numbers	Add/subtract numbers
$-6 + 4 \leq 16 : 4 - 6$	$-2 \leq -2$
Multiply/divide numbers	Answer:
$-6 + 4 \leq 4 - 6$	true

Solved form: *number* \otimes *number*, where

- \otimes is one of inequality signs $<$, $>$, \leq , \geq ;
- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);

- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer: Given inequality is

- true;
- false.

Problem type *Check the solution of inequality*

Typical texts: check if number ... is a solution of inequality.

Expression: inequality.

Constraints (for expression): inequality should be in one variable.

Parameters: value (one number) of variable to be checked.

Rules:

- Substitute variable;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Raise number to a power;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Substitute variable*;
3. Rule *Raise number to a power*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Algorithm for combining;
6. Rules *Open parentheses* and *Clear parentheses*;
7. Rule *Mixed number to improper fraction* (if equation is not yet in form $number = number$).

Example of generated solution:

Text of problem: Check if number 3 is a solution of inequality.

$$\frac{5x - 4}{5} > \frac{3x}{4}$$

Substitute variable

$$\frac{5 \cdot 3 - 4}{5} > \frac{3 \cdot 3}{4}$$

Multiply/Divide numbers

$$\frac{15 - 4}{5} > \frac{9 - 3}{4}$$

Multiply/Divide numbers

$$\frac{15 - 4}{5} > \frac{9}{4}$$

Add/Subtract numbers

$$\frac{11}{5} > \frac{9}{4}$$

Extend common fraction

$$\frac{11}{5} > \frac{9}{4}$$

Extend fractions

$$\frac{44}{20} > \frac{9}{4}$$

Extend common fraction

$$\frac{44}{20} > \frac{9}{4}$$

Extend fractions

$$\frac{44}{20} > \frac{45}{20}$$

Improper fraction to mixed number

$$2 \frac{4}{20} > \frac{45}{20}$$

Improper fraction to mixed number

$$2 \frac{4}{20} > 2 \frac{5}{20}$$

Answer:

is not a solution

Solved form: $number \otimes number$, where

- \otimes is one of inequality signs $<$, $>$, \leq , \geq ;
- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);
- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer: Number ...

- is a solution;
- is not a solution.

Problem type Reverse sides of inequality

Typical texts: reverse inequality sides.

Expression: inequality.

Constraints (for expression): if inequality is numerical, then it should be true.

Parameters: none.

Rules:

- Reverse sides;
- Move terms to other side;
- Multiply/Divide both sides;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Reverse sides* (if inequality is not changed yet);
3. Rule *Multiply/Divide both sides* (if inequality is equivalent to given inequality where signs of all terms are changed to opposite);
4. Rule *Move terms to other side* (to get all terms on the correct side if some part of terms was already moved to other side).

Example of generated solution:

Text of problem: Reverse sides of inequality.

$$-8 < 20y$$

Reverse sides

$$20y > -8$$

Answer:

$$20y > -8$$

Solved form: inequality with changed sides.

Answer: solved form.

Problem type *Add number to sides of inequality*

Typical texts: add number ... to both sides of inequality.

Expression: inequality.

Constraints (for expression): numerical inequality should be true.

Parameters: number, which has to be added to both sides of inequality.

Rules:

- Add to/Subtract from both sides;
- Add/Subtract numbers;
- Combine like terms;
- Clear parentheses;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Rule *Add to/Subtract from both sides* (one time if inequality is not changed yet);
2. Algorithm for simplification;
3. Algorithm for combining;
4. Rule *Clear parentheses*.

Example of generated solution:

Text of problem: Add number 3 to both sides of inequality.

$$-12 \leq 8$$

Add to/Subtract from both sides

$$-12 + 3 \leq 8 + 3$$

Add/subtract numbers

$$-9 \leq 8 + 3$$

Add/subtract numbers

$$-9 \leq 11$$

Answer:

$$-9 \leq 11$$

Solved form: simplified inequality with number added to both sides.

Answer: solved form.

Problem type *Subtract number from sides of inequality*

Typical texts: subtract number ... from both sides of inequality.

Expression: inequality.

Constraints (for expression): if inequality is numerical, then it should be true.

Parameters: number that has to be subtracted from both sides of inequality.

Rules:

- Add to/Subtract from both sides;
- Add/Subtract numbers;
- Combine like terms;
- Clear parentheses;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Rule *Add to/Subtract from both sides* (one time if inequality is not changed yet);
2. Algorithm for simplification;
3. Algorithm for combining;
4. Rule *Clear parentheses*.

Example of generated solution:

Text of problem: Subtract number 9 from both sides of inequality.

$$8 > 4$$

Add to/Subtract from both sides

$$8 - 9 > 4 - 9$$

Add/subtract numbers

$$-1 > 4 - 9$$

Add/subtract numbers

$$-1 > -5$$

Answer:

$$-1 > -5$$

Solved form: simplified inequality with subtracted number from both sides.

Answer: solved form.

Problem type *Multiply both sides of inequality by given number*

Typical texts: multiply both sides of inequality by number

Expression: inequality.

Constraints (for expression):

- inequality should not contain parentheses or brackets;
- if inequality is numerical, then it should be true.

Parameters: number by which both sides of inequality have to be multiplied (not zero; if fractions are presented in inequality then this number should be positive common denominator).

Rules:

- Multiply/Divide both sides;
- Multiply/Divide numbers;
- Mixed number to improper fraction;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Multiply/Divide numbers*;
3. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
4. Rule *Multiply/Divide both sides*.

Example of generated solution:

Text of problem: Multiply both sides of inequality by number -3.

$$-4x \geq 16$$

Multiply/Divide both sides

$$-4x \geq 16 \quad | \cdot (-3)$$

Multiply/Divide both sides

$$12x \leq -48$$

Answer:

$$12x \leq -48$$

Solved form: simplified multiplied inequality.

Answer: solved form.

Problem type *Multiply both sides of inequality by common denominator*

Typical texts: multiply both sides of inequality by common denominator of all terms.

Expression: inequality.

Constraints (for expression):

- inequality should contain common fraction(s) or mixed number(s);
- inequality should not contain parentheses or brackets;
- if inequality is numerical, then it should be true.

Parameters: none.

Rules:

- Multiply/Divide both sides;
- Multiply/Divide numbers;
- Mixed number to improper fraction;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Multiply/Divide numbers*;
3. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
4. Rule *Multiply/Divide both sides*.

Example of generated solution:

Text of problem: Multiply sides of inequality by common denominator of all terms.

$$\frac{12x - 5}{12} - 1 \leq \frac{3 - 4x}{8} + x$$

Multiply/Divide both sides

$$\frac{12x - 5}{12} \cdot 24 - 1 \cdot 24 \leq \frac{3 - 4x}{8} \cdot 24 + x \cdot 24 \quad | \cdot 24$$

Multiply/Divide both sides

$$24x - 10 - 24 \leq 9 - 12x + 24x$$

Answer:

$$24x - 10 - 24 \leq 9 - 12x + 24x$$

Solved form: simplified (using simplifying rules and rule *Multiply/Divide numbers*) multiplied inequality without common fractions and mixed numbers.

Answer: solved form.

Problem type *Divide both sides of inequality by given number*

Typical texts: divide both sides of inequality by number ...

Expression: inequality.

Constraints (for expression):

- inequality should not contain parentheses and brackets;
- inequality should not contain common fractions, decimal fractions and mixed numbers;
- if inequality is numerical, then it should be true.

Parameters: number by which both sides of inequality have to be divided (not zero).

Rules:

- Multiply/Divide both sides;
- Mixed number to improper fraction;
- Reduce;
- Multiply/Divide numbers;
- Improper fraction to mixed number;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rule *Mixed number to improper fraction*;

3. Rule *Multiply/Divide both sides*;
4. Rule *Multiply/Divide numbers*;
5. Rule *Reduce*;
6. Rule *Improper fraction to mixed number*.

Example of generated solution:

Text of problem: Divide both sides of inequality by number 2.

$$4x + 72 \leq 52 - 6x$$

Multiply/Divide both sides

$$4x + 72 \leq 52 - 6x \quad | : 2$$

Multiply/Divide both sides

$$2x + 36 \leq 26 - 3x$$

Answer:

$$2x + 36 \leq 26 - 3x$$

Solved form: simplified divided inequality.

Answer: solved form.

Problem type *Solve linear inequality*

Typical texts: solve linear inequality.

Expression: inequality.

Constraints (for expression): inequality should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;

2. Rule *Reverse sides* (if inequality is in the form: $number \otimes variable$, where \otimes is one of inequality signs $<, >, \leq, \geq$);
3. Rules *Open parentheses* and *Clear parentheses*;
4. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
5. Rule *Multiply/Divide both sides* for removing fractions (multiplication);
6. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
7. Algorithm for combining;
8. Rule *Move terms to other side*;
9. Rule *Multiply/Divide both sides* for isolating variable (division).

Example of generated solution:

Text of problem: Solve inequality.

$$\frac{2-x}{4} - \frac{x-4}{3} > 3$$

Multiply/Divide both sides

$$\frac{2-x}{4} \cdot \frac{3}{3} - \frac{x-4}{3} \cdot \frac{4}{4} > 3 \cdot \frac{12}{12} \quad | \cdot 12$$

Multiply/Divide both sides

$$6 - 3x - 4x + 16 > 36$$

Add/subtract numbers

$$22 - 3x - 4x > 36$$

Combine like terms

$$22 - 7x > 36$$

Move terms to other side

$$-7x > 36 - 22$$

Add/subtract numbers

$$-7x > 14$$

Multiply/Divide both sides

$$-7x > 14 \quad | : (-7)$$

Multiply/Divide both sides

$$x < -2$$

Answer:

$$x < -2$$

Solved form: $variable \otimes number$ or $number \otimes number$, where

- \otimes is one of inequality signs $<, >, \leq, \geq$;
- numbers should be transformed to normal form (reduced and transformed to mixed numbers if needed);
- fractions/mixed numbers (if both numbers are fractions/mixed numbers with the same sign) should be transformed to similar fractions/mixed numbers.

Answer:

- solved form ($variable \otimes number$ or $number \otimes number$ (where \otimes is one of inequality signs $<, >, \leq, \geq$); for example, $x < -2$);
- there is no solution;
- any number is solution.

Problem type *Express variable from equation*

Typical texts: express variable ... from equation.

Expression: equation.

Constraints (for expression): it should be possible to get solved form from equation by presented rules.

Parameters: variable that has to be expressed from equation.

Rules:

- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Express variable;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rules *Open parentheses* and *Clear parentheses*;
3. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
4. Rule *Multiply/Divide both sides* for multiplication;
5. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
6. Algorithm for combining;
7. Rule *Move terms to other side*;
8. Rule *Express variable*;
9. Rule *Multiply/Divide both sides* for division.

Example of generated solution:

Text of problem: Express variable y from equation.

$$-0,5x + 0,2y = 0,7$$

Move terms to other side

$$0,2y = 0,7 + 0,5x$$

Multiply/Divide both sides

$$0,2y = 0,7 + 0,5x \quad | \cdot 5$$

Multiply/Divide both sides

$$y = 3,5 + 2,5x$$

Answer:

$$y = 3,5 + 2,5x$$

Solved form: *variable from the text = expression without this variable.*

Answer: solved form.

Problem type *Solve by substitution*

Typical texts: solve by substitution.

Expression: system of linear equations.

Constraints (for expression):

- system of linear equations should be in two variables and consist of exactly two equations;
- system should have exactly one solution;
- system should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Substitute variable;
- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Express variable;
- Clear parentheses;
- Open parentheses;
- Multiply fraction with variable by number;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;

- Common fraction to division;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rules *Open parentheses* and *Clear parentheses*;
3. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
4. Rule *Multiply fraction with variable by number*;
5. Rule *Multiply/Divide both sides*;
6. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides of equation);
7. Algorithm for combining;
8. Rule *Move terms to other side*;
9. Rule *Substitute variable*;
10. Rule *Express variable*;
11. Rule *Reverse sides*;
12. Rule *Common fraction to division*.

Example of generated solution:

Text of problem: Solve by substitution.

$\begin{cases} -6t + 3s = 24 \\ -5t + s = 2 \end{cases}$	<p>Move terms to other side</p> $\begin{cases} 9t = 24 - 6 \\ s = 2 + 5t \end{cases}$	<p>Substitute variable</p> $\begin{cases} t = 2 \\ s = 2 + 5 \cdot 2 \end{cases}$
<p>Move terms to other side</p> $\begin{cases} -6t + 3s = 24 \\ s = 2 + 5t \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} 9t = 24 - 6 \quad : 3 \\ s = 2 + 5t \end{cases}$	<p>Multiply/divide numbers</p> $\begin{cases} t = 2 \\ s = 2 + 10 \end{cases}$
<p>Substitute variable</p> $\begin{cases} -6t + 3(2 + 5t) = 24 \\ s = 2 + 5t \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} 3t = 8 - 2 \\ s = 2 + 5t \end{cases}$	<p>Add/subtract numbers</p> $\begin{cases} t = 2 \\ s = 12 \end{cases}$
<p>Open parentheses</p> $\begin{cases} -6t + 6 + 15t = 24 \\ s = 2 + 5t \end{cases}$	<p>Add/subtract numbers</p> $\begin{cases} 3t = 6 \\ s = 2 + 5t \end{cases}$	<p>Answer:</p> $\begin{cases} t = 2 \\ s = 12 \end{cases}$
<p>Combine like terms</p> $\begin{cases} 9t + 6 = 24 \\ s = 2 + 5t \end{cases}$	<p>Multiply/Divide both sides</p> $\begin{cases} 3t = 6 \quad : 3 \\ s = 2 + 5t \end{cases}$	
	<p>Multiply/Divide both sides</p> $\begin{cases} t = 2 \\ s = 2 + 5t \end{cases}$	

Solved form: $\begin{cases} \text{first variable} = \text{number} \\ \text{second variable} = \text{number} \end{cases}$

Answer: solved form.

Problem type *Solve by elimination using addition*

Typical texts: solve by elimination using addition.

Expression: system of linear equations.

Constraints (for expression):

- system of linear equations should be in two variables and consist of exactly two equations;
- system should have exactly one solution;
- system should be solvable (it should be possible to get a solved form) by presented rules.

Parameters: none.

Rules:

- Substitute variable;
- Add equations;
- Reverse sides;
- Move terms to other side;
- Add to/Subtract from both sides;
- Multiply/Divide both sides;
- Clear parentheses;
- Open parentheses;
- Combine like terms;
- Add/Subtract numbers;
- Multiply/Divide numbers;
- Rules for fractions;
- Common fraction to division;
- Simplification rules.

T-algebra algorithm:

1. Algorithm for simplification;
2. Rules *Open parentheses* and *Clear parentheses*;
3. Rules *Multiply/Divide numbers* and *Move minus before fraction*;
4. Rule *Move terms to other side*;
5. Algorithm for combining;
6. Rule *Multiply/Divide both sides*;
7. Rule *Mixed number to improper fraction* (if mixed numbers disturb multiplying/dividing both sides);
8. Rule *Reverse sides*;
9. Rule *Substitute variable*;
10. Rule *Add equations*;

11. Rule *Multiply/Divide both sides* for getting suitable coefficients (for eliminating one variable by addition);
12. Rule *Common fraction to division*.

Example of generated solution:

Text of problem: Solve by elimination using addition.

$$\begin{cases} 3x + 4y = -2 \\ 5x - 6y = -16 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} 3x + 4y = -2 & | \cdot (-5) \\ 5x - 6y = -16 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} -15x - 20y = 10 \\ 5x - 6y = -16 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} -15x - 20y = 10 \\ 5x - 6y = -16 & | \cdot 3 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} -15x - 20y = 10 \\ 15x - 18y = -48 \end{cases}$$

Add equations

$$\begin{cases} -38y = -38 \\ 15x - 18y = -48 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} -38y = -38 & | : (-38) \\ 15x - 18y = -48 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} y = 1 \\ 15x - 18y = -48 \end{cases}$$

Substitute variable

$$\begin{cases} y = 1 \\ 15x - 18 \cdot 1 = -48 \end{cases}$$

Multiply by 1

$$\begin{cases} y = 1 \\ 15x - 18 = -48 \end{cases}$$

Move terms to other side

$$\begin{cases} y = 1 \\ 15x = -48 + 18 \end{cases}$$

Add/Subtract numbers

$$\begin{cases} y = 1 \\ 15x = -30 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} y = 1 \\ 15x = -30 & | : 15 \end{cases}$$

Multiply/Divide both sides

$$\begin{cases} y = 1 \\ x = -2 \end{cases}$$

Answer:

$$\begin{cases} y = 1 \\ x = -2 \end{cases}$$

Solved form: $\begin{cases} \text{first variable} = \text{number} \\ \text{second variable} = \text{number} \end{cases}$

Answer: solved form.

APPENDIX D

User questionnaire

The questionnaire presented in (Mitrovic and Ohlsson, 1999) was used to compose this questionnaire.

1. How much time did you need to learn to use T-algebra?
 - a. Less than 5 minutes
 - b. 5 to 20 minutes
 - c. More than 20 minutes
2. Do you find T-algebra easy to use?
 - a. Not at all
 - b. Rather not
 - c. Hard to say
 - d. Rather yes
 - e. Yes, very much
3. Did you learn something new about mathematics from using T-algebra?
 - a. Not at all
 - b. Rather not
 - c. Hard to say
 - d. Rather yes
 - e. Yes, very much
4. Did you enjoy learning with T-algebra?
 - a. Not at all
 - b. Rather not
 - c. Hard to say
 - d. Rather yes
 - e. Yes, very much
5. Would you like to use T-algebra more?
 - a. No
 - b. Do not know
 - c. Yes
6. Would you recommend T-algebra to other students?
 - a. No
 - b. Do not know
 - c. Yes
7. Do you find error messages understandable?
 - a. Not at all
 - b. Rather not
 - c. Hard to say
 - d. Rather yes
 - e. Yes, very much
8. Did error messages help to correct mistakes?
 - a. Not at all
 - b. Rather not
 - c. Hard to say
 - d. Rather yes
 - e. Yes, very much
9. What did you like in particular about T-algebra?
10. What did you dislike about T-algebra?
11. Did you encounter any software problems or crashes?
 - a. No
 - b. Yes

APPENDIX E

Problem file

Problem file used in the fifth experiment.

1. Check if number 10 is solution of equation
 $2x - 13 = 52 - 4x$
2. Check if number 4 is solution of equation
 $5a + 4(1 - a) = 2a$
3. Check if number -2 is solution of equation (help denied)
 $3(x + 8) - 12 = 5(7 + 3x)$
4. Reverse equation sides
 $3t + 4 = 5t - 12$
5. Reverse equation sides
 $x = 5 + (-3x + 7)$
6. Reverse equation sides (help denied)
 $-7 - 8x = 5 - 6x$
7. Reverse equation sides (help denied)
 $6(y - 1) = 18$
8. Divide equation sides by variable coefficient
 $8x = 64$
9. Divide equation sides by variable coefficient
 $1,5z = -4,5$
10. Divide equation sides by variable coefficient (help denied)
 $12v = -144$
11. Divide equation sides by variable coefficient (help denied)
 $-0,4x = -2,4$
12. Divide equation sides by common divider
 $4x - 16 = 12 + 2x$
13. Multiply equations sides by common denominator of all terms
 $\frac{x}{4} + \frac{1}{2} = -\frac{1}{6}$
14. Multiply equations sides by common denominator
 $\frac{a + 7}{3} = 3$
15. Multiply equations sides by common denominator of all terms
 $\frac{10x - 4}{3} - \frac{9x + 3}{4} = 3 - \frac{2x + 1}{2}$
16. Multiply equations sides by common denominator of all terms
 $\frac{5x - 7}{9} - \frac{2x}{3} + 2 = \frac{4x - 3}{5} + \frac{6 - 3x}{15}$
17. Multiply equations sides by common denominator of all terms (help denied)
 $\frac{5z}{9} - \frac{2}{3} = \frac{z}{6}$
18. Multiply equations sides by common denominator of all terms (help denied)
 $\frac{y + 2}{12} - 2 = y$
19. Multiply equations sides by common denominator of all terms (help denied)
 $\frac{x + 7}{7} - \frac{2x - 5}{4} = \frac{9 - x}{2} + 5$

20. Multiply equations sides by common denominator of all terms (help denied)

$$\frac{3x+1}{8} - 2 = 1 - \frac{6+9x}{6}$$

21. Move all variable terms to the left side and all constant terms to the right side and then combine like terms

$$12 - 3x = 7x + 2$$

22. Move all variable terms to the left side and all constant terms to the right side and then combine like terms

$$8 - 3t + 4 = 3t$$

23. Move all variable terms to the left side and all constant terms to the right side and then combine like terms

$$6b - 3 - 4b - 4 = 2b - 7$$

24. Move all variable terms to the left side and all constant terms to the right side and then combine like terms

$$4,34 + 5,4c = -1,2c + 1,7$$

25. Move all variable terms to the left side and all constant terms to the right side and then combine like terms

$$\frac{5}{9}z - \frac{2}{3} = \frac{1}{6}z$$

26. Move all variable terms to the left side and all constant terms to the right side and then combine like terms (help denied)

$$4a = 3a - 6$$

27. Move all variable terms to the left side and all constant terms to the right side and then

combine like terms (help denied)

$$8s + 13 - 3s = 27 + 2s - 5$$

28. Solve an equation

$$7x + 9 = 4x$$

29. Solve an equation

$$17z - 36 + 15z = 8 - 22z + 64$$

30. Solve an equation

$$3(8v - 1) = 9v - 33$$

31. Solve an equation

$$-6(5x - 14) = 27 - 3(8x + 15)$$

32. Solve an equation

$$\frac{5 - 3x}{7} = 5 + x$$

33. Solve an equation

$$\frac{5x - 3}{6} - \frac{4x + 1}{9} = 2 - \frac{7 - x}{2}$$

34. Solve an equation

$$\frac{5x + 1}{6} + \frac{x + 3}{4} = x + 1 + \frac{x - 3}{12}$$

35. Solve an equation (help denied)

$$4b + 8 = 6b - 2$$

36. Solve an equation (help denied)

$$4x - 3 - 2(2x - 1) = -3$$

37. Solve an equation (help denied)

$$7(t + 6) = 2(15 - 2t) + 1$$

38. Solve an equation (help denied)

$$\frac{1 + 6a}{8} = \frac{4 - 5a}{3}$$

39. Solve an equation (help denied)

$$\frac{x - 2}{4} - 2 = \frac{x + 2}{8} - 1 + \frac{1 - 2x}{2}$$

40. Solve an equation (help denied)

$$\frac{9a - 17}{5} - \frac{12 - 3a}{2} = \frac{14 - 4a}{3} - \frac{a - 2}{6}$$

CURRICULUM VITAE

Marina Issakova

Citizenship: Estonian Republic
Born: July 14, 1980, Tallinn, Estonia
Marital status: married
Address: Loopealse 20, Ülenurme alevik, Tartumaa, Estonia
Contacts: phone: +372 55 650931
e-mail: marina.issakova@ut.ee

Education

1986–1990 Tallinn Secondary School No. 56
1990–1998 Lasnamae Russian High School
1998–2001 University of Tartu, Bachelor in Computer Science (*cum laude*)
2001–2003 University of Tartu, MSc in Computer Science
2003–... University of Tartu, PhD studies in Computer Science

Professional employment

2004–... University of Tartu, part-time assistant lecturer of software systems
2005 University of Tartu, part-time software engineer
2006–... University of Tartu, part-time research fellow

Scientific work

The main fields of interest are interactive learning environments and the use of computers in mathematics education.

CURRICULUM VITAE

Marina Issakova

Kodakondsus: Eesti Vabariik
Sünniaeg ja -koht: 14. juuli 1980, Tallinn, Eesti
Perekonnaseis: abielus
Aadress: Loopealse 20, Ülenurme alevik, Tartumaa, Eesti
Kontakt: tel.: +372 55 650931
e-post: marina.issakova@ut.ee

Haridus

1986–1990 Tallinna 56. Keskkool
1990–1998 Lasnamäe Vene Gümnaasium
1998–2001 Tartu Ülikool, informaatika bakalaureus (*cum laude*)
2001–2003 Tartu Ülikool, informaatika magister (MSc)
2003–... Tartu Ülikool, doktoriõpe informaatika erialal

Erialane teenistuskäik

2004–... Tartu Ülikool, osalise koormusega tarkvarasüsteemide assistent
2005 Tartu Ülikool, osalise koormusega programmeerija
2006–... Tartu Ülikool, osalise koormusega teadur

Teadustegevus

Peamine uurimisvaldkond on interaktiivsed õpisüsteemid ja arvutid matemaatikahariduses.

DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS

1. **Mati Heinloo.** The design of nonhomogeneous spherical vessels, cylindrical tubes and circular discs. Tartu, 1991, 23 p.
2. **Boris Komrakov.** Primitive actions and the Sophus Lie problem. Tartu, 1991, 14 p.
3. **Jaak Heinloo.** Phenomenological (continuum) theory of turbulence. Tartu, 1992, 47 p.
4. **Ants Tauts.** Infinite formulae in intuitionistic logic of higher order. Tartu, 1992, 15 p.
5. **Tarmo Soomere.** Kinetic theory of Rossby waves. Tartu, 1992, 32 p.
6. **Jüri Majak.** Optimization of plastic axisymmetric plates and shells in the case of Von Mises yield condition. Tartu, 1992, 32 p.
7. **Ants Aasma.** Matrix transformations of summability and absolute summability fields of matrix methods. Tartu, 1993, 32 p.
8. **Helle Hein.** Optimization of plastic axisymmetric plates and shells with piece-wise constant thickness. Tartu, 1993, 28 p.
9. **Toomas Kiho.** Study of optimality of iterated Lavrentiev method and its generalizations. Tartu, 1994, 23 p.
10. **Arne Kokk.** Joint spectral theory and extension of non-trivial multiplicative linear functionals. Tartu, 1995, 165 p.
11. **Toomas Lepikult.** Automated calculation of dynamically loaded rigidplastic structures. Tartu, 1995, 93 p. (in Russian).
12. **Sander Hannus.** Parametrical optimization of the plastic cylindrical shells by taking into account geometrical and physical nonlinearities. Tartu, 1995, 74 p.
13. **Sergei Tupailo.** Hilbert's epsilon-symbol in predicative subsystems of analysis. Tartu, 1996, 134 p.
14. **Enno Saks.** Analysis and optimization of elastic-plastic shafts in torsion. Tartu, 1996, 96 p.
15. **Valdis Laan.** Pullbacks and flatness properties of acts. Tartu, 1999, 90 p.
16. **Märt Põldvere.** Subspaces of Banach spaces having Phelps' uniqueness property. Tartu, 1999, 74 p.
17. **Jelena Ausekle.** Compactness of operators in Lorentz and Orlicz sequence spaces. Tartu, 1999, 72 p.
18. **Krista Fischer.** Structural mean models for analyzing the effect of compliance in clinical trials. Tartu, 1999, 124 p.

19. **Helger Lipmaa.** Secure and efficient time-stamping systems. Tartu, 1999, 56 p.
20. **Jüri Lember.** Consistency of empirical k-centres. Tartu, 1999, 148 p.
21. **Ella Puman.** Optimization of plastic conical shells. Tartu, 2000, 102 p.
22. **Kaili Müürisep.** Eesti keele arvutigrammatika: süntaks. Tartu, 2000, 107 lk.
23. **Varmo Vene.** Categorical programming with inductive and coinductive types. Tartu, 2000, 116 p.
24. **Olga Sokratova.** Ω -rings, their flat and projective acts with some applications. Tartu, 2000, 120 p.
25. **Maria Zeltser.** Investigation of double sequence spaces by soft and hard analytical methods. Tartu, 2001, 154 p.
26. **Ernst Tungal.** Optimization of plastic spherical shells. Tartu, 2001, 90 p.
27. **Tiina Puolakainen.** Eesti keele arvutigrammatika: morfoloogiline ühestamine. Tartu, 2001, 138 p.
28. **Rainis Haller.** $M(r,s)$ -inequalities. Tartu, 2002, 78 p.
29. **Jan Villemson.** Size-efficient interval time stamps. Tartu, 2002, 82 p.
30. **Eno Tõnisson.** Solving of expression manipulation exercises in computer algebra systems. Tartu, 2002, 92 p.
31. **Mart Abel.** Structure of Gelfand-Mazur algebras. Tartu, 2003. 94 p.
32. **Vladimir Kuchmei.** Affine completeness of some ockham algebras. Tartu, 2003. 100 p.
33. **Olga Dunajeva.** Asymptotic matrix methods in statistical inference problems. Tartu 2003. 78 p.
34. **Mare Tarang.** Stability of the spline collocation method for volterra integro-differential equations. Tartu 2004. 90 p.
35. **Tatjana Nahtman.** Permutation invariance and reparameterizations in linear models. Tartu 2004. 91 p.
36. **Märt Möls.** Linear mixed models with equivalent predictors. Tartu 2004. 70 p.
37. **Kristiina Hakk.** Approximation methods for weakly singular integral equations with discontinuous coefficients. Tartu 2004, 137 p.
38. **Meelis Käärrik.** Fitting sets to probability distributions. Tartu 2005, 90 p.
39. **Inga Parts.** Piecewise polynomial collocation methods for solving weakly singular integro-differential equations. Tartu 2005, 140 p.
40. **Natalia Saealle.** Convergence and summability with speed of functional series. Tartu 2005, 91 p.
41. **Tanel Kaart.** The reliability of linear mixed models in genetic studies. Tartu 2006, 124 p.

42. **Kadre Torn.** Shear and bending response of inelastic structures to dynamic load. Tartu 2006, 142 p.
43. **Kristel Mikkor.** Uniform factorisation for compact subsets of Banach spaces of operators. Tartu 2006, 72 p.
44. **Darja Saveljeva.** Quadratic and cubic spline collocation for Volterra integral equations. Tartu 2006, 117 p.
45. **Kristo Heero.** Path planning and learning strategies for mobile robots in dynamic partially unknown environments. Tartu 2006, 123 p.
46. **Annely Mürk.** Optimization of inelastic plates with cracks. Tartu 2006. 137 p.
47. **Annemai Raidjõe.** Sequence spaces defined by modulus functions and superposition operators. Tartu 2006, 97 p.
48. **Olga Panova.** Real Gelfand-Mazur algebras. Tartu 2006, 82 p.
49. **Härmel Nestra.** Iteratively defined transfinite trace semantics and program slicing with respect to them. Tartu 2006, 116 p.
50. **Margus Pihlak.** Approximation of multivariate distribution functions. Tartu 2006, 82 p.
51. **Ene Käärrik.** Handling dropouts in repeated measurements using copulas. Tartu 2006, 99 p.
52. **Artur Sepp.** Affine models in mathematical finance: an analytical approach. Tartu 2006, 147 p.