Conceptualizing a Pedagogical CAS for Algebraic Manipulation of Expressions

Rein Prank, Marina Lepp

prank@ut.ee,

marina.lepp@ut.ee

University of Tartu (Estonia)

1. Introduction

We assume that using pedagogical CAS (PeCAS)

- the student solves tasks step by step,
- the system provides hints,
- the system checks correctness of the student’s steps, provides feedback about errors and requires correction of them,
- the system records the student’s solutions and information about errors

1. Introduction

2) Content

The paper has the following Sections:

1. Introduction
2. Some existing pedagogical CAS
3. Working modes for the student and solution interface
4. Possible features of error checking
5. Possible applications, outputs and inside of automated Solver.
6. Recording and visualization of information about the results of students’ work.
7. Some ideas for automated assessment
8. Random generation of tasks.
9. Summary of our most important recommendations

1. Introduction

3) **Input-based** and **Rule-based** dialog

When the students solve an expression manipulation task, they should at each solution step

1) Choose a certain **operation** in the algorithm (or some simplification or calculation rule known earlier),
2) Select the **operands** for this operation,
3) Replace them with the **result** of the operation.

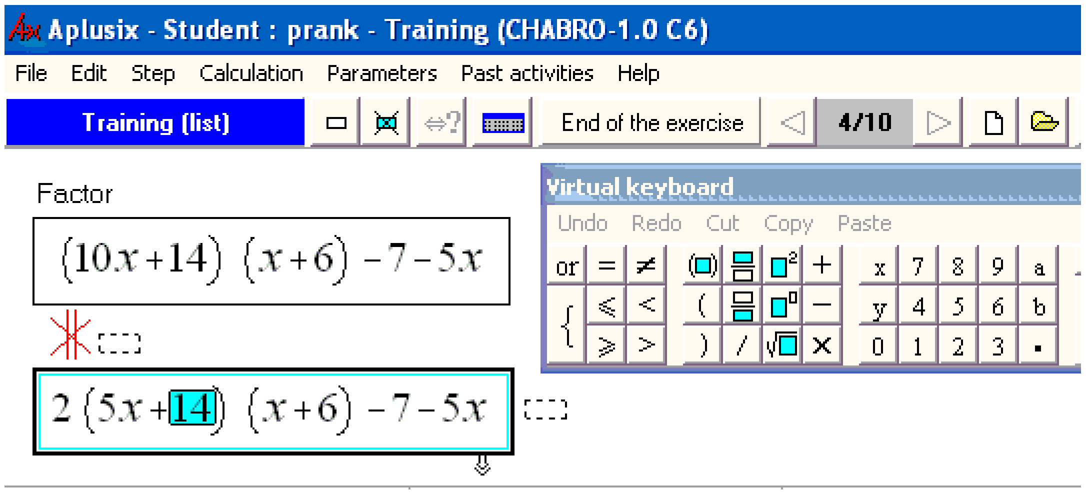
Part of the information created during substeps 1-3 can be considered as redundant.

- The written solutions usually record only the third substep. The same can be implemented in a PeCAS in form of an **Input-based** solution dialog.
- But it is also possible to implement the completely orthogonal **Rule-based** approach where the student performs only first two substeps and the result of the step is computed automatically.

2. Examples of PeCAS

1) Aplusix

For each step, the student copies the previous expression (equation, …) to the next line and then uses the expression editor to transform it into the result of the step

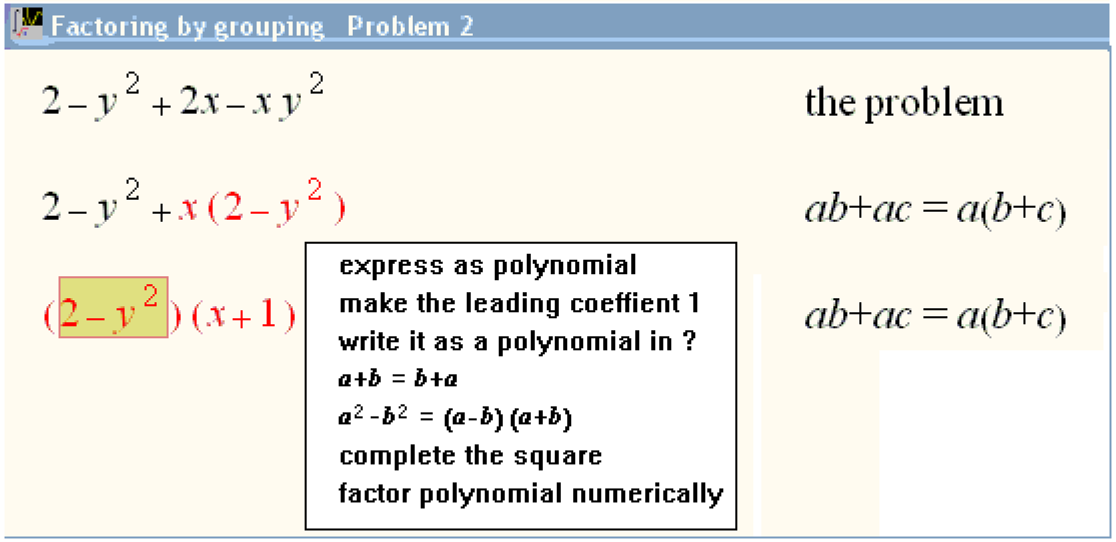

Aplusix - Student : prank - Training (CHABRO-1.0 C6)
File Edit Step Calculation Parameters Past activities Help
Training (list)
End of the exercise
4/10
Virtual keyboard
Undo Redo Cut Copy Paste
Factor

$(10x+14)(x+6)-7-5x$

$2(5x+\boxed{14})(x+6)-7-5x$

2. Examples of PeCAS
2) MathXpert

1) The student marks a **subexpression**
2) The program displays a **menu with rules/commands** applicable to the marked subexpression
3) The student selects an item from the menu and the program applies the corresponding rule/command


Factoring by grouping Problem 2

$2-y^2+2x-xy^2$ the problem

$2-y^2+x(2-y^2)$ $ab+ac=a(b+c)$

$(2-y^2)(x+1)$ $ab+ac=a(b+c)$

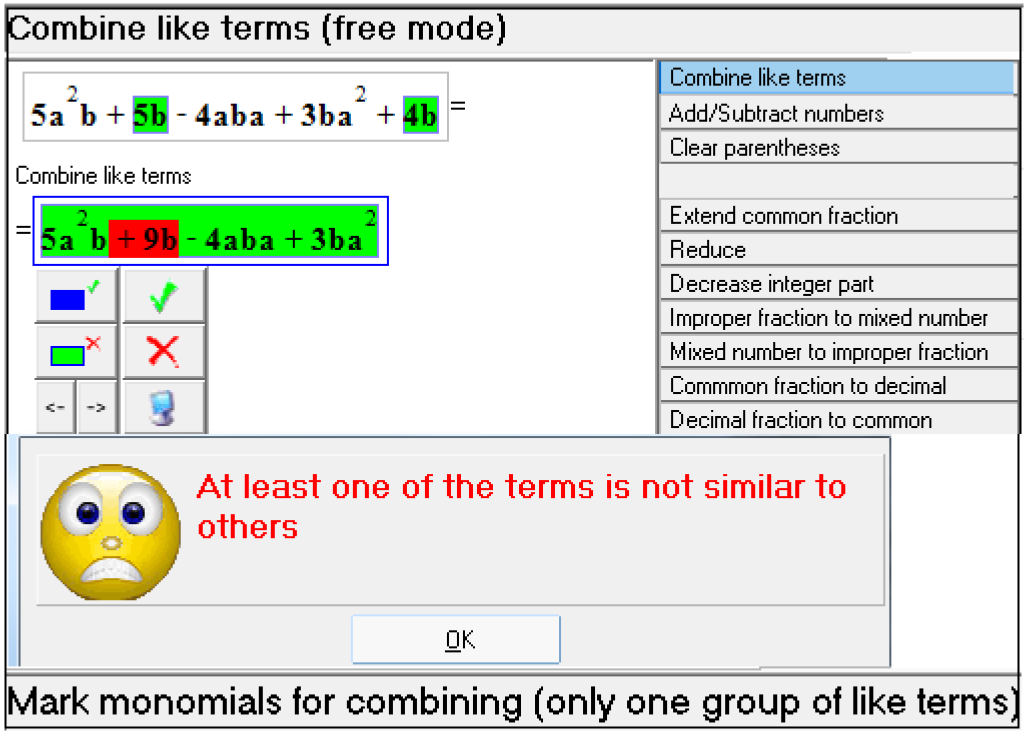
express as polynomial
make the leading coeffient 1
write it as a polynomial in ?
$a+b = b+a$
$a^2-b^2 = (a-b)(a+b)$
complete the square
factor polynomial numerically

2. Examples of PeCAS

3) T-algebra (Tartu, 2004-2008)

1-2) The student selects the rule and marks operands

3) The student enters the expression that replaces the marked part



Combine like terms (free mode)

$5a^2b + 5b - 4aba + 3ba^2 + 4b =$

Combine like terms

$= 5a^2b + 9b - 4aba + 3ba^2$

At least one of the terms is not similar to others

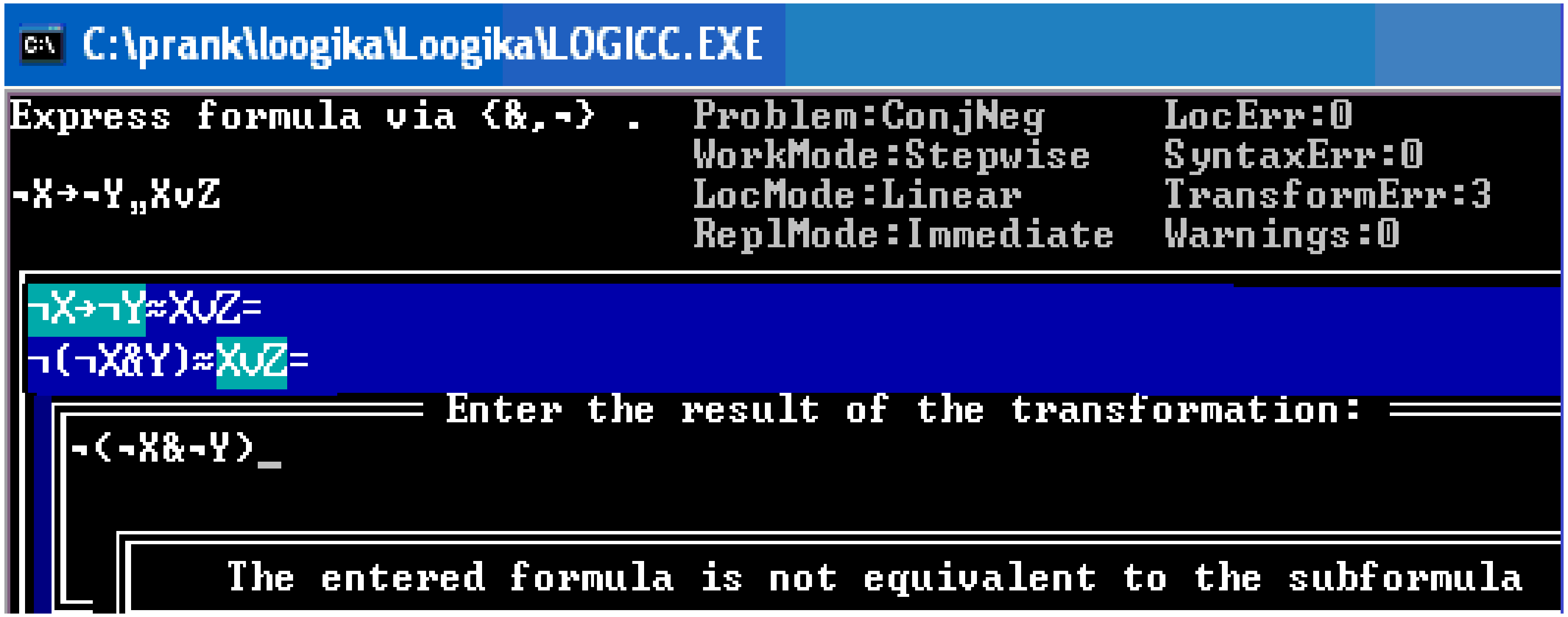
Mark monomials for combining (only one group of like terms)

Combine like terms
Add/Subtract numbers
Clear parentheses
Extend common fraction
Reduce
Decrease integer part
Improper fraction to mixed number
Mixed number to improper fraction
Commmon fraction to decimal
Decimal fraction to common
<-
->
OK

2. Examples of PeCAS

4) Algebraic Manipulation Assistant for Logic (Tartu 1990, …)

1) The student **marks a subformula**
2) The student
   enters a formula that replaced the marked part (**Immediate mode**) or
   selects a rule from the menu (**Rule mode**)



```
C:\prank\loogika\Loogika\LOGICC.EXE

Express formula via {&,¬} .   Problem:ConjNeg      LocErr:0
                              WorkMode:Stepwise    SyntaxErr:0
¬X→¬Y≈XvZ                     LocMode:Linear       TransformErr:3
                              ReplMode:Immediate   Warnings:0

¬X→¬Y≈XvZ=
¬(¬X&Y)≈XvZ=
            Enter the result of the transformation:
¬(¬X&¬Y)_

     The entered formula is not equivalent to the subformula
```

We have modified the program in 2003 and 2013.

3. Working modes and solution interface

3.1 Input mode and Rule mode

1) Both are necessary

WB/BB first stage in learning – Input mode

WB/BB second stage in learning– Rule mode

Learning of algorithms – Rule mode

Solving more creative tasks – Rule mode

Assessment – Input mode

This brings us to recommendation to
implement in a PeCAS both modes where this is possible.

Thereby the teacher should have possibility to fix the working mode for every task

3.1 Input mode and Rule mode
2) Teacher-chosen sets of rules

It is not difficult to design software for **Rule-based** work in a manner that enables the teacher to specify **what set of rules is available**

Then the teacher is responsible for solvability of tasks and could first solve each task himself.

- If the software contains an automatic Solver then the existence of a solution can be approved automatically.

3.1 Input mode and Rule mode

3) Long step problem

Input mode does not regulate the length and content of the steps.

- The next line can be result of **several parallel or consequent conversion steps** made by the student.
- It is possible to enter only the **final answer obtained** using paper and pencil, making calls to some CAS or even stealing an expression from the screen of the neighbor.
- The teacher component of environment should enable at least a **comfortable and quick way to look at the student solutions**
- But the program could also **protest against the student step** when it jumps over too many stages of the solution algorithm or covers too many steps of the Solver.

3.2 Marking of operands

1)

Some programs
(MathXpert) use marking of operand(s) substantially but
some others (Aplusix) do not use it at all.

3.2 Marking of operands

2) One example

In 1989 we tried first time Algebraic Manipulation Assistant with students.

At each step **the student entered the next line** and **the program checked the equivalence** with previous line.

The students who misinterpreted the order of operations complained that the program gives buggy error messages for example after replacing $X \sim Y$ in $X \sim Y \lor Z$ by $X \& Y \lor \neg X \& \neg Y$.

To be able to diagnose misunderstanding of the order of operations, we decided

to **add a substep of marking of the subformula** for modification before entering the result .

3.2 Marking of operands

3) Precise marking?

- MathXpert does not require precise marking of operand(s) . If the user marks a subexpression then MathXpert proposes a list of conversion rules that can be applied to the marked subexpression or to some its subexpression(s).
- Precise marking **enables to apply the conversion precisely to selected operands**
- Requiring of precise marking enables **diagnosing of some errors**

Technically the precise marking requires implementation of **multi-marking** i.e. marking of several separate subexpressions

4. Error checking: content and timing

4. Error checking: content and timing 1)

- Necessary and possible/programmable content of error checking **depends on the working mode**
- Some of the **messages are understandable only when the student has entered their matter explicitly**
- The main problem is that
  many concrete checks are desirable
  but in actual working mode the program does not get **enough information for understanding the roots of the error**

4. Error checking: content and timing

2) Two alternatives

If we want to diagnose better than „not equivalent“ then we should **choose between two alternatives**:

a) Require from the student entering not only the result but also **some additional information about the meaning of the step** or

b) Design and implement
much more intelligent analyse algorithms

4.1 Correctness of marking

- In case of PeCAS we can have didactical goals that contain training of punctuality or testing of correct understanding of the order of operations
- For such purposes **we need the design where the editor enables the student to mark also incorrect pieces of the expression** but displays then corresponding error message
- We have seen that the students need such training at least when they work with new algebraic operations (Propositional Logic)
- Training of perfect handling of syntax of expressions is also useful in Basic school.

4.2 Syntax of entered expressions

- In ‘exercise modes’ the programs usually require correction of syntax errors before checking of equivalence and further issues of correctness
- In general the ‘test modes’ try to mimic the paper and pencil tests. We recommend **obligatory correction of syntax errors** also for the ‘test modes’.
- Otherwise the program diagnoses incidental slips with syntax but not the more important issues

4.3 Equivalence checking

1) Limits for equivalence checking (1)

D. Richardson (1968) + Y. Matiyasevich (1970):

Let F denote the class of functions in one real variable that can be defined by expressions constructed from
the variable x, the integers and the number π,
combined through addition, subtraction, multiplication, sin and abs (absolute value).
There is no algorithm for deciding for an arbitrary given expression $E(x)$ from the class F whether the equality

$$E(x) = 0$$

holds identically for all values of x.

- **Equivalence is also undecidable** because

$A(x) \equiv B(x)$ iff $A(x) - B(x) \equiv 0$

- **Absolute value can be replaced** by functions that are necessary for trigonometry: $|x| = sqrt(x^2)$

4.3 Equivalence checking

2) Limits for equivalence checking (2)

A.Church (1936):

There is no algorithm that could decide the Entscheidungsproblem (whether a formula of predicate logic is a consequence of a finite set of given formulas).

This implies that **there is no algorithm for checking the equivalence of formulas of predicate logic**.

Because of these two theorems

MathXpert and

Predicate part of Algebraic Manipulation Assitant

use only **Rule mode**

4.3 Equivalence checking

3) Limits for equivalence checking (3)

The strongest known **positive result** for real numbers, Macintyre and Wilkie (1996):

If Schanuel's Conjecture (for R) is true then ***first order theory*** ***$\langle$R, 0, 1, +, -, $\times$, exp,<$\rangle$ is decidable***.

- Schanuel's Conjecture is:

If $z_1,\ldots,z_n$ are real numbers linearly independent over ***Q****, then the extension field* ***Q****$(z_1,\ldots, z_n, \exp(z_1),\ldots,\exp(z_n))$ has transendence degree of at least n (over* ***Q****).*

- Only the proof and not the decision algorithm depend on Schanuel conjecture.

4.3 Equivalence checking

4) Equivalence checking mechanisms

PeCAS use different mechanisms for equivalence checking.

1) Many systems contain **automated Solvers** and often the final answer can be considered as canonical form of the expression.
   For checking of equivalence the program can then **compare the answer calculated from the previous and from the new line**
2) Some other systems make
   calls to corresponding commands of general-purpose CAS or **include equivalence checking or simplification modules from computer algebra libraries**
3) Some programs use **evaluation of variables by random values.**
   With high probability **the mistake causes essentially changed value already by first random evaluation**. This can give results even in theoretically unsolvable cases. But unfortunately the evaluation approach does not have good modifications for the case of equations/inequalities etc.

4.3 Equivalence checking

5) Equivalence with hidden assumptions

Result of testing of equivalence by a CAS or random evaluation can be not satisfying. These methods **ignore differences on the sets of measure zero**.

- They accept for example reduction of x^2/x to x although the expressions are really not equivalent
- For some parts of the syllabus such conversions are **acceptable and even the main objects of learning**
- The same time they can cause problems in other tasks, for example to generate **extraneous solutions of equations**
- Chuaqui and Suppes recommend **explicit recording of assumptions** (in our case: $x \neq 0$) that are necessary for equivalence. MathXpert has adopted such approach.

4.4 Checking of solution economy and conformity with algorithm

1) Motivation

- Usually the **PeCAS check that the student steps are 'correct'** but do not evaluate their expediency
- But the students tend to think that **if the program accepted the step then it is reasonable**
- When we started to use **Algebraic Manipulation Assistant with first semester students** we saw that the solution files were several times longer than expected.
  In final test of autumn term 2011 the average number of steps was 44.0 instead of the normal 15-25 steps at most
- In some didactical situations **we need checking of expediency of the steps**

4.4 Economy and conformity with algorithm

2) Absolute economy (of the step)

Consider an ‘absolute’ measure of economy that can be used in any working mode.

- If we have an optimizing Solver that uses some fixed set of conversion rules then **we can find and compare shortest solutions before and after the step**: how much the solution has decreased/increased
- Unfortunately **an optimizing Solver would work slowly**
- But if we replace the optimizing Solver by the Solver that implements the standard algorithm for the task then the loss is not so big

4.4 Economy and conformity with algorithm

3) Comparing a step with the algorithm

This is **quite easy in Rule mode**. The program can use the Solver and check whether the actually used rule corresponds to the algorithm (or is an appropriate simplification rule).

- **First author used such approach in disjunctive normal form exercises with first-term students**. The analyzer was implemented as a supplementary program that evaluated every step in student solutions.
- After making the analyzer available for the students in autumn 2012 we saw that in comparison with 2011 the **average number of algorithmic mistakes** in the normal form task at final test decreased from 9.1 to 3.0 and the **average number of steps** from 44.0 to 27.3 (the optimal length of solutions was about 20 steps).

4.4 Economy and conformity with algorithm

4) Comparing with the algorithm in Input mode

The situation is more complicated in Input mode. But

- The Solver can calculate **what rule(s) is (are) recommended by the algorithm**
- For evaluation of student’s step it is possible to check **whether the goal of recommended step is reached**

Such checks **require complicated modeling of all conversion rules** and are much more complex to program than simple comparison of the rules.

But this is the place where **analyze of goals of any new conversion improves the quality of feedback**

4.5 Error classification

Big programs have hundreds of different error messages

- Classification of errors into more general categories enables
  - to get **overview of strong and weak sides** of the student
  - automated **assessment using penalties** for each category
- The student and teacher programs could contain **means for display of error tables in both views**:
  - categories
  - concrete error messages (ordered by number of occurrences)

4.6 Moment of checking

Most **PeCAS's check the input when the student completes the (sub)step and pushes corresponding button**. By pushing this button the student takes some responsibility

- It is possible to **reduce the moment of responsibility** in exercise classes. Aplusix verifies syntax and equivalence continuously. But such dialog does not create space for feedback about the error
- Some programs implement so called **'Test mode'** where correctness is checked only after finishing the task (or test). We recommend for tests also **checking the correctness always before next (sub)step**

5. Features and role of an automated solver

- “… 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”

Michael Beeson

5. Features and role of an automated solver

- The automated solver of a PeCAS
  - should follow the ‘white-box’ principle
  - should produce step-by-step solutions similar to pencil-and-paper ones
  - should be cognitive faithful
    - should solve the problem in the same way as the student should
  - should accept all solution paths (calculated according to the algorithm and all other as well)

5. Features and role of an automated solver

- The automated solver of PeCAS should implement the solution algorithm for every problem type
- For example, algorithms in T-algebra are implemented as an ordered list of rules
  - algorithm for linear equation solving
    - rules for simplification (not algorithm steps: *Add/Subtract 0*, etc.)
    - rules for arithmetic operations and manipulation with numbers (*Extend*, *Reduce*, etc.)
    - rules, which correspond to pencil-and-paper algorithm steps
      - rule *Multiply/Divide both sides* for multiplying (removing fractions)
      - rule *Open parentheses*
      - rule *Add/Subtract numbers*
      - rule *Combine like terms*
      - rule *Move terms to other side*
      - rule *Multiply/Divide both sides* for division

5. Features and role of an automated solver

T-algebra automated solver

- The automated solver
  - examines this list of rules from the beginning
  - finds the first rule that it can apply
  - applies this rule
  - then examines this list again from the beginning
  - finds new (or the same) rule
  - applies this 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 automated solver gets an answer to the problem

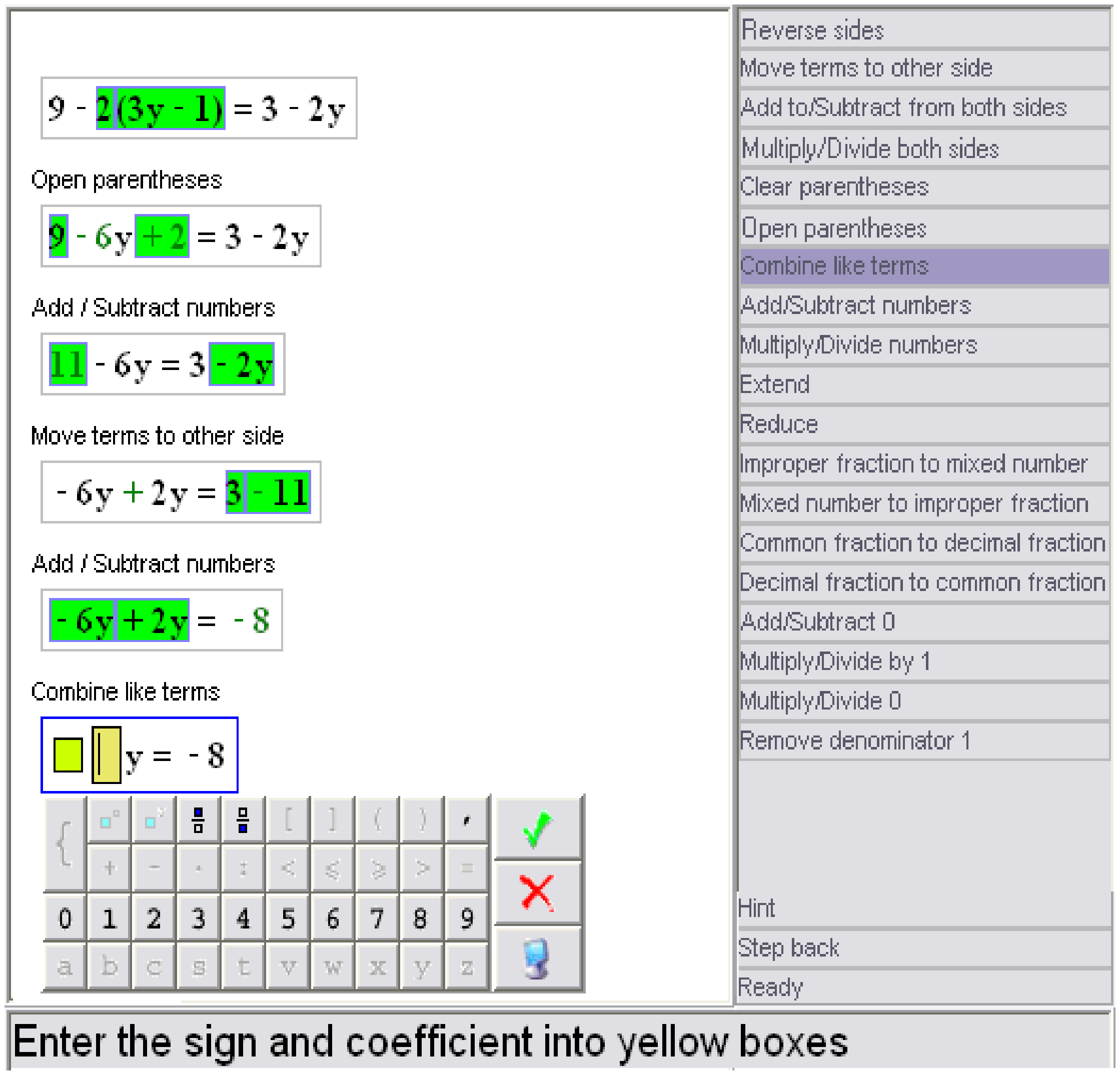
5. Features and role of an automated solver

T-algebra automated solver

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

5. Features and role of an automated solver

T-algebra automated solver



$9 - 2(3y - 1) = 3 - 2y$

Open parentheses

$9 - 6y + 2 = 3 - 2y$

Add / Subtract numbers

$11 - 6y = 3 - 2y$

Move terms to other side

$- 6y + 2y = 3 - 11$

Add / Subtract numbers

$- 6y + 2y = - 8$

Combine like terms

$\square\,\square\,y = - 8$

Enter the sign and coefficient into yellow boxes

5. Features and role of an automated solver

The use of results of the work of the automated solver

1. Testing of solvability of the task by available conversions.
2. Evaluation of suitability of the task (length of the solution, necessary conversions, form and values of intermediate results) by the teacher.
3. Evaluation of appropriateness of the final answer by the teacher.
4. Automated selection of tasks by random generation of initial expressions (ensuring given length of solution, inclusion of necessary conversions, exclusion of undesired values, etc.).
5. Display of step hints or execution of (a part of) the next step at the student's request.
6. Demonstration of the whole solution (from the begin or from the point reached by the student).
7. Checking of equivalence of expressions/equations/….

5. Features and role of an automated solver

The use of results of the work of the automated solver

8. Localization of the erroneous components in the entered result of the step.
9. Recognizing the operation that was executed with mistake by the student (in the Input mode).
10. Evaluation of expediency of the step (comparing the lengths of standard solutions before and after the step).
11. Comparison of executed step (rule) with the step of the standard algorithm.
12. Comparison of the entered result of the step with the result of application of the rule that the student claimed to have applied.
13. Global evaluation of solution economy by comparing the length of the student’s solution with the standard solution (mainly in the Rule mode).
14. Checking whether the required final form is reached.
15. Generating tasks that require application of given conversion rules.

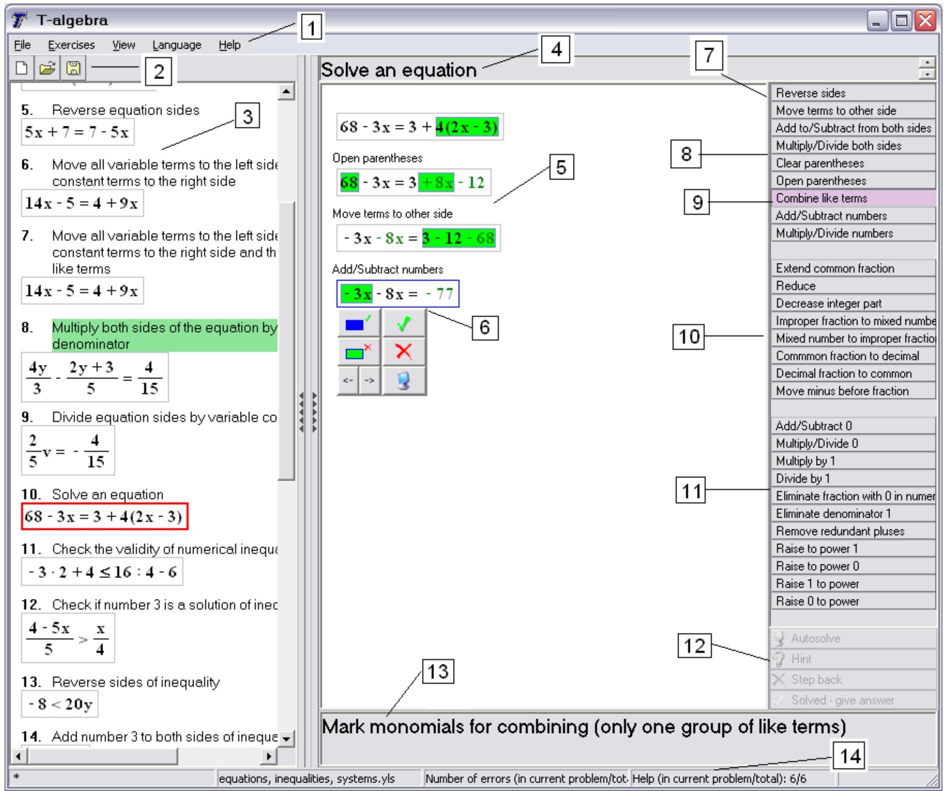
6. Recording and presenting information about students' work

- In PeCAS the checking of solutions and preparing of data for decisions should be done in real time
- A PeCAS should save the solutions and detected errors for further analysis by the teacher
- It is also possible to record the time characteristics of the work

6. Recording and presenting information about students’ work

Created solutions

- Usually the programs enable to restore the picture for subsequent analysis by the student and the teacher



5. Reverse equation sides
$5x + 7 = 7 - 5x$

6. Move all variable terms to the left side
constant terms to the right side
$14x - 5 = 4 + 9x$

7. Move all variable terms to the left side
constant terms to the right side and th
like terms
$14x - 5 = 4 + 9x$

8. Multiply both sides of the equation by
denominator
$\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15}$

9. Divide equation sides by variable co
$\frac{2}{5}v = -\frac{4}{15}$

10. Solve an equation
$68 - 3x = 3 + 4(2x - 3)$

11. Check the validity of numerical inequa
$-3 \cdot 2 + 4 \le 16 : 4 - 6$

12. Check if number 3 is a solution of inec
$\frac{4-5x}{5} > \frac{x}{4}$

13. Reverse sides of inequality
$-8 < 20y$

14. Add number 3 to both sides of inequa

Solve an equation

$68 - 3x = 3 + 4(2x - 3)$

Open parentheses

$68 - 3x = 3 + 8x - 12$

Move terms to other side

$-3x - 8x = 3 - 12 - 68$

Add/Subtract numbers

$-3x - 8x = -77$

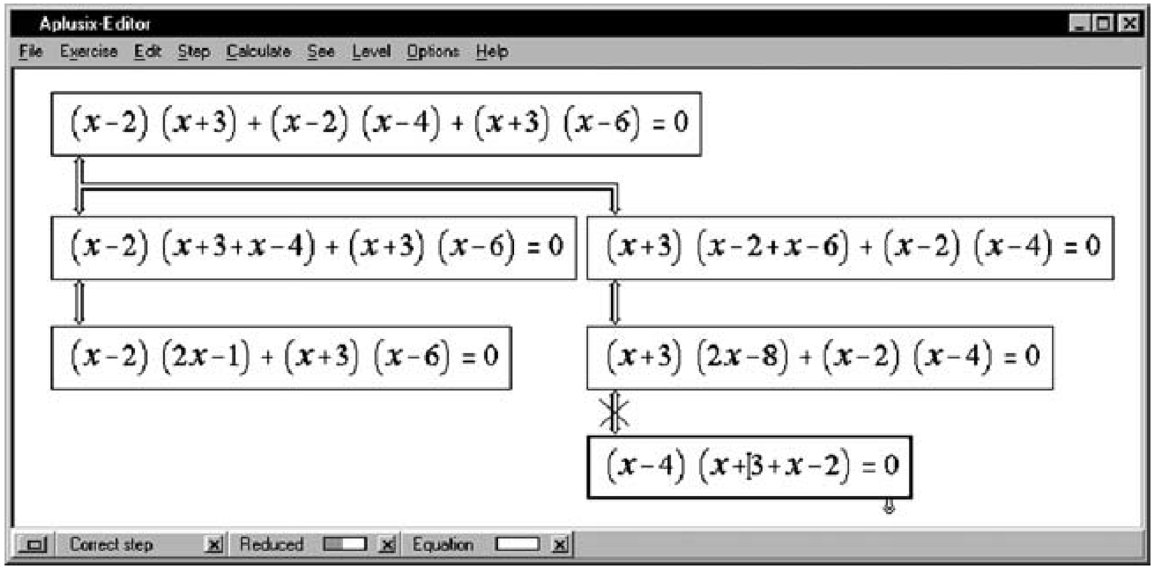
Mark monomials for combining (only one group of like terms)

* | equations, inequalities, systems.yls | Number of errors (in current problem/tot | Help (in current problem/total): 6/6

6. Recording and presenting information about students' work

Created solutions - branches

- The designers of PeCAS should decide whether to maintain only one branch of solution or more during the solution dialog
- PeCAS should save the cancelled steps too and display all branches of the solution tree for analysis



$(x-2)(x+3)+(x-2)(x-4)+(x+3)(x-6)=0$

$(x-2)(x+3+x-4)+(x+3)(x-6)=0$

$(x+3)(x-2+x-6)+(x-2)(x-4)=0$

$(x-2)(2x-1)+(x+3)(x-6)=0$

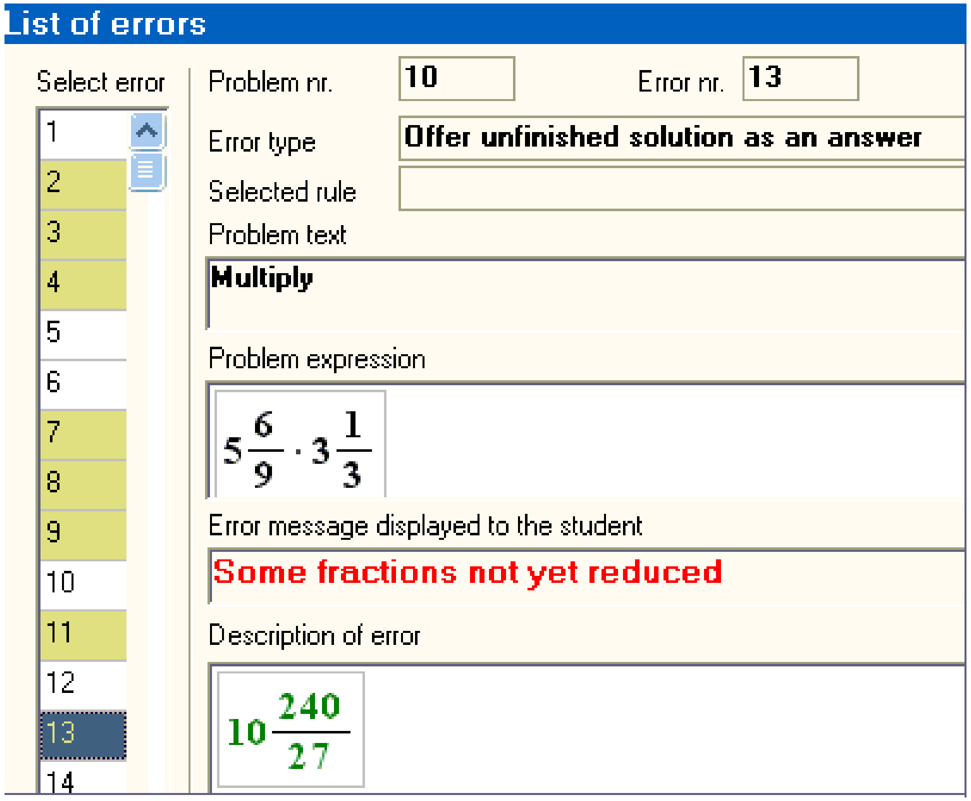
$(x+3)(2x-8)+(x-2)(x-4)=0$

$(x-4)(x+3+x-2)=0$

6. Recording and presenting information about students' work

Error situations

- Next important facility is recording and demonstration of error situations and help requests


List of errors

Select error

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Problem nr. 10

Error nr. 13

Error type: Offer unfinished solution as an answer

Selected rule:

Problem text: **Multiply**

Problem expression:

$$5\frac{6}{9}\cdot 3\frac{1}{3}$$

Error message displayed to the student: Some fractions not yet reduced

Description of error:

$$10\frac{240}{27}$$

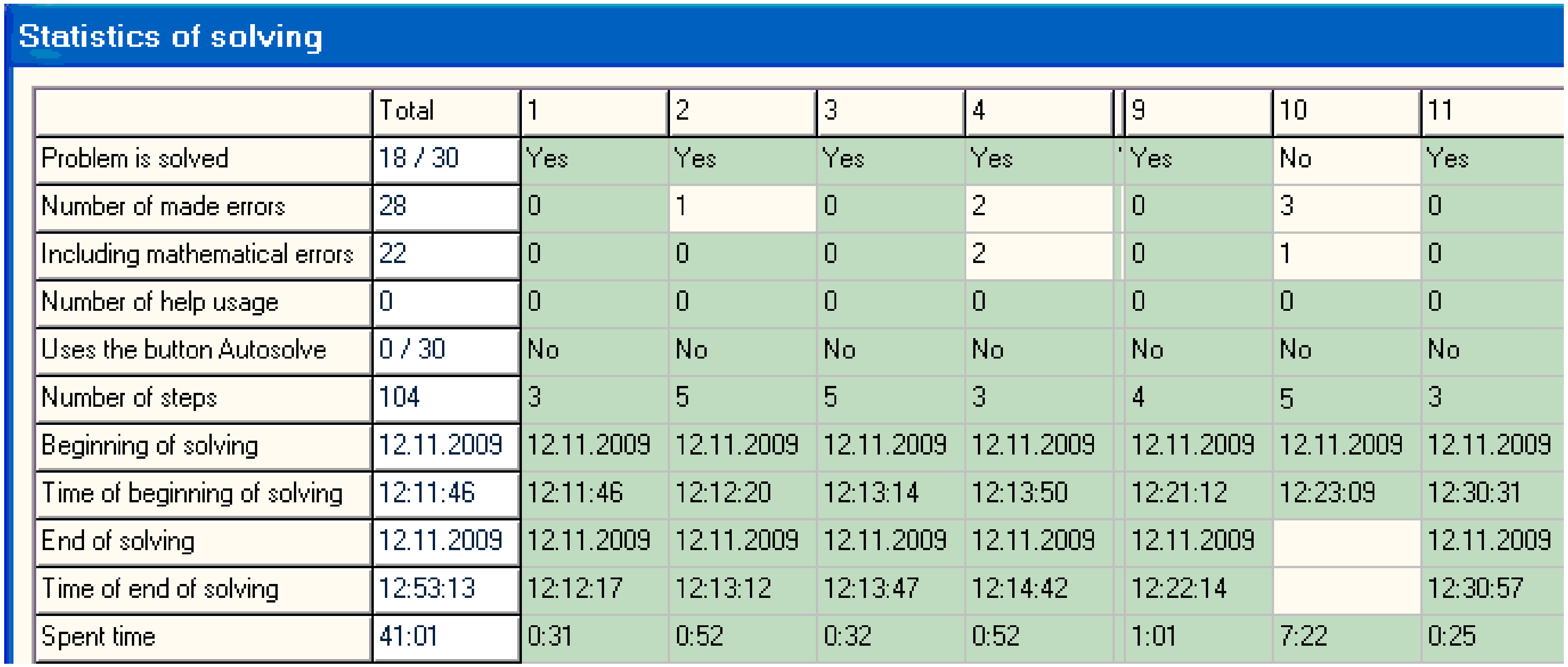
6. Recording and presenting information about students' work

Summarized statistical information

- The statistics should include a variety of information:
  - number and designation of problems that were solved by the student
  - number of errors (total and separately for each task)
  - number of instances of help use
  - number of solution steps created
  - time characteristics (begin, end, duration) of each task
- The statistics enables to evaluate
  - the student's general performance
    - speed of work
    - success at different kinds of tasks
    - number of errors
  - working habits
    - using or ignoring help functions
    - completing all tasks or giving up in difficult cases
    - pauses between tasks

6. Recording and presenting information about students' work

Summarized statistical information

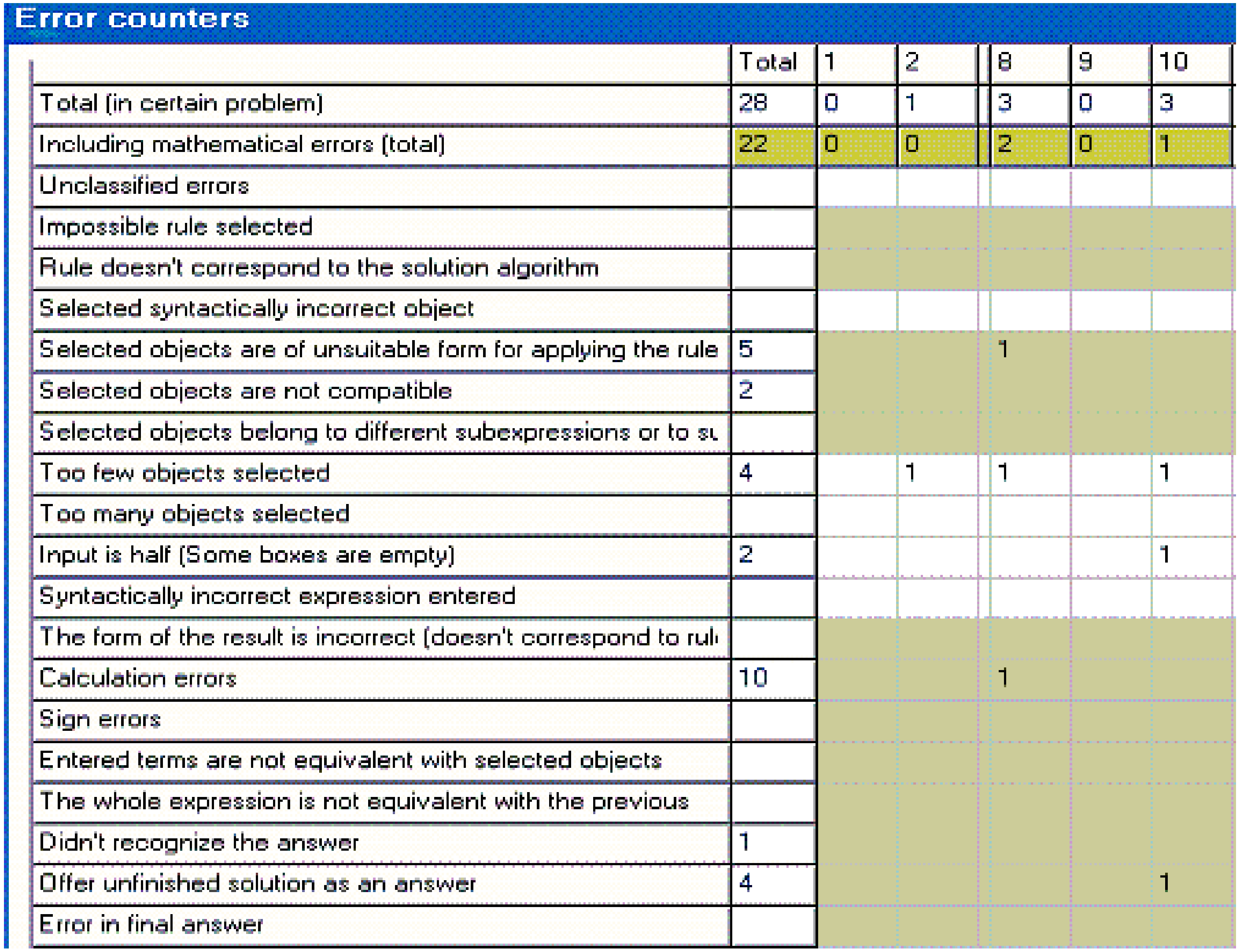


Statistics of solving

	Total	1	2	3	4	9	10	11
Problem is solved	18 / 30	Yes	Yes	Yes	Yes	Yes	No	Yes
Number of made errors	28	0	1	0	2	0	3	0
Including mathematical errors	22	0	0	0	2	0	1	0
Number of help usage	0	0	0	0	0	0	0	0
Uses the button Autosolve	0 / 30	No	No	No	No	No	No	No
Number of steps	104	3	5	5	3	4	5	3
Beginning of solving	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009
Time of beginning of solving	12:11:46	12:11:46	12:12:20	12:13:14	12:13:50	12:21:12	12:23:09	12:30:31
End of solving	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009	12.11.2009		12.11.2009
Time of end of solving	12:53:13	12:12:17	12:13:12	12:13:47	12:14:42	12:22:14		12:30:57
Spent time	41:01	0:31	0:52	0:32	0:52	1:01	7:22	0:25

6. Recording and presenting information about students’ work

Error/help usage counters by categories and tasks



Error counters

	Total	1	2	8	9	10
Total (in certain problem)	28	0	1	3	0	3
Including mathematical errors (total)	22	0	0	2	0	1
Unclassified errors						
Impossible rule selected						
Rule doesn't correspond to the solution algorithm						
Selected syntactically incorrect object						
Selected objects are of unsuitable form for applying the rule	5			1		
Selected objects are not compatible	2					
Selected objects belong to different subexpressions or to su						
Too few objects selected	4		1	1		1
Too many objects selected						
Input is half (Some boxes are empty)	2					1
Syntactically incorrect expression entered						
The form of the result is incorrect (doesn't correspond to rul						
Calculation errors	10			1		
Sign errors						
Entered terms are not equivalent with selected objects						
The whole expression is not equivalent with the previous						
Didn't recognize the answer	1					
Offer unfinished solution as an answer	4					1
Error in final answer						

6. Recording and presenting information about students’ work

Overview of the work of group of students

- table of solved/unsolved tasks
  - rows for each student and columns for each task
- table of step counts
  - numbers of steps in each (finished or unfinished) solution
- table of solution times
  - time for each task (finished or unfinished)
- table of error counts
  - numbers of errors made in each task
- two overview tables of error types in tasks
  - the rows display the categories of errors in the first table and particular error messages in the second table, the columns correspond to tasks

7. Assessment facilities

1) What aspects?

At least the following **three aspects** should be taken into account in automated assessment:

1) What part of the task is solved (if the solution is incomplete),
2) Penalties for errors,
3) Solution economy/conformity with the algorithm.

7. Assessment facilities

2) How to count ?

1) What part of the task is solved (if the solution is incomplete),
2) Penalties for errors,
3) Solution economy/conformity with the algorithm.

1) **numerical weights of the stages of the algorithm** and a measure inside of some stages

2) **basic penalties for each error category** and **penalty calculation formulas**

3a) Rule-based modes - **actual number of steps / information from the teacher or Solver**

3b) Evaluation of each step & **penalties for inexpedient conversions**

8. Random generation of tasks

1) What do we need?

The teacher tools should allow specifying the generating parameters so that the tasks have **desired pedagogical properties**:

- **use of certain conversion rules**,
- **opportunities for making certain errors**,
- have reasonable **probabilities for choices** to be made by the student,
- require approximately **equal amount of work from different students**

8. Random generation of tasks

2) How to generate?

Automated Solver should be used much more

- The properties of **exercises with short solutions** can easily be checked immediately or be built in using reverse engineering.
- For the properties of **exercises with long solutions**, the most natural way seems to be: to generate an initial expression, apply the automatic Solver and check whether the solution has the desired properties.
- The **properties are usually quite trivial** conditions about existence or number of some elements in the solution.

Many PeCAS have Solver, but

Solver is used locally for step hints and does not record the solution in a form that can be used for analysis

9. Recommendations

MathXpert and Aplusix are classic examples of solution environments that implemented the Rule-based and the Input-based dialog in a very natural way.

MathXpert enables to experiment with solution ideas without performing the calculations by hand. Aplusix gives the student feedback about technical correctness of the solution steps.

Designing new environments today we should think **what could be added** to these pioneering environments.

9. Main recommendations

1. Solution environments should implement **both the Input and the Rule mode**.
2. It is time to **give feedback about the expediency of the step** and not be restricted to applicability of the rule or equivalence to previous line.
3. The designers should decide whether to use a **step dialog that embodies intention-related components** or to **implement intelligent techniques** for guessing the ideas behind the steps
4. The automated **Solver should produce step-by-step solutions similar to pencil-and-paper solutions.** The expert module should be 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.
5. Teachers need reasonably generalized and visualized **information about the performance of individual students and whole class**.
6. Teachers need **Computer-Aided Assessment**. The PeCAS should contain tools for appointing task weights, error penalties, penalty counting formulas, etc.
7. Teachers need **random generation of tasks with necessary pedagogical properties**, including the properties of solutions

References

- Anderson, J.R. (1988). The expert module. In: Polson, M., Richardson, J. (eds.): Handbook of Intelligent Training Systems. Hillsdale, NJ: Erlbaum, 1988 21-53.
- M.Beeson. Design Principles of Mathpert: Software to support education in algebra and calculus, in: Kajler, N. (ed.) Computer-Human Interaction in Symbolic Computation, pp. 89-115, Springer-Verlag, Berlin/ Heidelberg/ New York (1998).
- Brna, P., Warr, K., Chiam, S-T. and Pain, H. Learning to Diagnose Algebra Errors. In *Proceedings of the Seventh International PEG Conference*, Volume 1, pp40-47, Edinburgh: Moray House. 1993.
- B.Buchberger. Should Students Learn Integration Rules? ACM SIGSAM Bulletin, 24, 1, 10-17, (1990)
- S. Burris, K.Yeats, The Saga of the High School Identities, Algebra Universalis, 52, No.2–3, (2004), pp.325–342.
- R. Chuaqui, P. Suppes. An equational deductive system for the differential and integral calculus. In: P. Martin-Löf, G.Mints (Eds) Proceedings of International Conference on Computer Logic, Tallinn,1988, LNCS 417, 1990, 25-49
- A.Church. A note on Entscheidungsproblem. Journal of Symbolic Logic, 1, 1936, 40-41.
- R. Dedekind. Was sind und was sollen die Zahlen?", 1888
- T.Fisher. Probabilistic Checks for the Equivalence of Mathematical. A Senior Thesis by. Travis Fisher. 1999 www.cse.unl.edu/~sscott/students/tfisher.pdf
- G. H. Gonnet. Determining Equivalence of Expressions in Random Polynomial Time. Proceedings of the 16th ACM Symposium on the Theory of Computing, 1984, 334-341.
- R. Gurevič. Equational Theory of Positive Numbers with Exponentiation is Not Finitely Axiomatizable. Annals of Pure and Applied Logic, 49, 1, 1990, 1-30.
- H.U.Hoppe. Deductive error diagnosis and inductive error generalisation for intelligent tutoring systems. *Journal of Artificial Intelligence in Education*, 5, 1994, 27-49.
- Issakova, M., Lepp, D., Prank, R.: T-algebra: Adding Input Stage to Rule-Based Interface for Expression Manipulation. The International Journal for Technology in Mathematics Education. 13, 89--96 (2006)
- Macintyre, A.J.Wilkie. On the decidability of the real exponential field. P. Odifreddi (ed.) Kreiseliana: about and around Georg Kreisel. A.K.Peters, 1996. 441-467.
- MathSpace. https://mathspace.co/
- Nicaud, J., Bouhineau, D., Chaachoua, H.: Mixing microworld and cas features in building computer systems that help students learn algebra. International Journal of Computers for Mathematical Learning. 5, 169--211 (2004)

- R. Prank, H.Viira. Algebraic Manipulation Assistant for Propositional Logic. Computerised Logic Teaching Bulletin, Vol.4, No.1, St Andrews Univ, 1991, 13-18.
- R. Prank. Using Computerised Exercises on Mathematical Logic. Informatik-Fachberichte, Vol. 292, Springer-Verlag, 1991, pp. 34-38.
- R.Prank, V.Vaiksaar. Expression manipulation environment for exercises and assessment. 6th International Conference on Technology in Mathematics Teaching. Volos-Greece, October 2003. 342-348.
- Prank, R., Issakova, M., Lepp, D., Tõnisson, E., Vaiksaar, V.: Integrating Rule-based and Input-based Approaches for Better Error Diagnosis in Expression Manipulation Tasks. In : Shangzhi Li, Dongming Wang, Jing-Zhong Zhang (eds.): Symbolic Computation and Education. 174--191. World Scientific, Singapore (2007)
- Prank, R., Lepp, D. Tools for Acquiring Data about Student Work in Interactive Learning Environment T-Algebra. In: Aleven, Vincent; Kay, Judy; Mostow, Jack (Eds.), Intelligent Tutoring Systems 2010. LNCS 6095, pp. 396-398.
- R.Prank. Software for Evaluating Relevance of Steps in Algebraic Transformations In: Intelligent Computer Mathematics: Conferences in Intelligent Mathematics, Bath, July 8-12, 2013. (Toim.) Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. Lecture Notes in Artificial Intelligence; 7961, Springer, 2013, 374 - 378.
- R. Prank. A tool for evaluating solution economy of algebraic transformations. Journal of Symbolic Computation, 61-62, 2014,100 - 115.
- Ravaglia, R., Alper, T., Rozenfeld, M., and Suppes, P. Successful pedagogical applications of symbolic computation. In: Kajler, N. (ed.): Computer-Human Interaction in Symbolic Computation. Berlin Heidelberg New York: Springer-Verlag, 1998, 61-88.
- D.Richardson. Some unsolvable problems involving elementary functions of a real variable. J.Symbolic Logic 33, 1968, 514-520.
- D.Richardson. Solution of the Identity Problem for Integral Exponential Functions. Zeitschrift für mathematical Logik und Grundlagen der Mathematik. Vol. 15, 1969, 333-340.
- Sangwin, C.J. (2005). Making Mathematical Distinctions In CAA With Computer Algebra. Proceedings of the 7th International Conference on Technology in Mathematics Teaching, Vol. 1. Bristol, UK 292-299.
- D. Sleeman, J. S. Brown (Eds). Intelligent Tutoring Systems. Academic Press, 1982.
- Walden, J. (1997). Mathpert Calculus Assistant: User’s Guide. Santa Clara, USA: Mathpert Systems, an Imprint of Recognix.