CHECKING THE EQUIVALENCE OF EXPRESSIONS IN COMPUTER ALGEBRA SYSTEMS – APPLICATION POSSIBILITIES IN MATHEMATICS EDUCATION

Eno TONISSON
Institute of Computer Science
University of Tartu
Liivi 2, Tartu 50409, Estonia
e-mail: eno@ut.ee

ABSTRACT

In connection with the spread of computer algebra systems (and algebraic calculators), the natural question arises: how to change the requirements and emphases of mathematics syllabuses? One possible domain that might be given more consideration in the future is checking the equivalence of expressions. It plays an important role in solving equations, manipulating expressions, and other domains (be it performed on a computer or by hand). In this paper, we examine some possibilities of integrating checking the equivalence expression in computer algebra systems more fully into the educational process. We present different schemes that describe the teacher's and student's activities in different situations, considering the particular goal, problem setup, student's preparedness, the specifics of the computer algebra system, access to computer algebra systems, etc. The schemes are based on problems of manipulating expressions selected from various areas of college algebra. They include step-by-step (line-by-line) solutions as well as solutions in which computers are used for solving larger blocks in one step. The described variants are titled:

-The student writes on paper,
-Computer algebra system as a text editor,
-Computer algebra system as an assistant in discovering errors,
-Computer algebra system as the maker of the next step,
-Computer algebra system as a solver,
-Computer algebra system as a component of an intelligent tutoring system,
-The student as the evaluator of computer algebra system.

Such a description of schemes may be beneficial not only to teachers, syllabus developers, textbook authors and the like but also to developers of computer algebra systems. The features expected from computer algebra systems for the realization of these schemes are described. The schemes are primarily designed for the current computer algebra systems (Derive, Maple, Mathematica and MuPAD); however, apart from the available features, mention is made of those that do not (yet?!) exist directly. For certain schemes, user interface is important.

Finally, some potential trends of research for the future are pointed out.

Keywords: Mathematics Education, Computer Algebra Systems

1. Introduction

In connection with the spread of computer algebra systems (and algebraic calculators), the natural question arises: how to change the requirements and emphases of mathematics syllabuses? How should we respond to the fact that the computer is capable of solving many problems of school or college mathematics? Is it possible or necessary to exclude some topics from the syllabus? What are the topics that can and should be emphasized? (See Herget, Heugl, Kutzler & Lehmann 2000.) To what extent should future students learn the step-by-step solving of algebraic problems, for instance? This article proceeds from the premise that step-by-step solving methods will remain on the syllabuses, at least in the nearest future; however, there may and should be changes in the approaches. One possible domain that might be given more consideration in the future is **checking the equivalence of expressions**. It plays an important role in solving equations, manipulating expressions, and other domains (be it performed on a computer or by hand). In this article, we examine some possibilities of integrating expression equivalence checking more fully into the educational process. With regard to problems, the article focuses on the ones dealing with expression manipulation (they are described in more detail in Part 2).

Putting it simply, it is possible to use or not to use computer algebra systems for solving problems. Between these two extremes, there can be plenty of other variants. The subject matter of this article (which is described in Part 3) is schemes that describe the teacher's and student's activities in different situations, considering the particular goal, problem setup, student's preparedness, the specifics of the computer algebra system, access to computer algebra systems, etc. All schemes are usable in practice. To what extent they will actually be used, however, depends on a number of factors. Such a description of schemes may be beneficial not only to teachers, syllabus developers, textbook authors and the like but also to developers of computer algebra systems, if they want to keep abreast with the educational market.

The features expected from computer algebra systems for the realization of these schemes are described in Part 4. There, a brief analysis is also provided of the capabilities of the currently widespread systems (Derive, Maple, Mathematica and MuPAD). In many respects, the computer algebra systems can cope well with checking the equivalence. Nevertheless, they may encounter some challenges as well. For certain schemes, user interface is important.

Promising is the harnessing of a computer algebra system, with all its mathematical capabilities, to an intelligent tutoring system as an expert module. In this case, the user cannot see the computer algebra system; instead, he communicates with the “shell” created specifically for teaching and learning, which exchanges mathematical information with the computer algebra system. The described schemes are useful for such tutoring system.

Finally, Part 5 points out some potential trends of research for the future.

2. Expression Manipulation Problems

The schemes discussed in this article are applicable to expression manipulation problems. In many school and college algebra problems, the texts are: *Remove parentheses and simplify, Combine into a single fraction and simplify, Combine like terms, simplify, Factor out factors common to all terms, factor by grouping terms, Simplify, and write answers using positive exponents, simplify, write in simplest radical form, etc.* (The topics are Polynomials, Exponents, Radicals, Logarithms, for instance). As a rule, the student needs to solve these problems step by step (line by line). In expression manipulation, all lines need to be equivalent to one another (as it is, the equality sign is put between them). Consequently, the checking of expression equivalence is

very important. However, it is usual that textbooks and teachers do not pay much attention to it, confining themselves to the performance of certain steps. For instance, the formula $\frac{x^2-2x}{2x-4}=\frac{x(x-2)}{2(x-2)}=\frac{x}{2}$ is varied, overlooking the fact that x = 2 would render the initial expression undefined.

Analogously, the schemes discussed in this article are more or less applicable to checking the equivalence of equations and inequalities as well. However, the solution of equations and inequalities is a slightly different matter in that in some types of problems (for instance, equations involving radicals), steps are deliberately made that may change the set of solutions.

Apart from equivalence, another important issue regarding a new line in expression manipulation is rationality, of course. We may find a large number of lines that are equivalent to the previous line; however, they may not take us any closer to the solution. In this article, the issue of rationality is not tackled.

3. Schemes for Role Distribution

The use of computer algebra systems (and checking the expression equivalence) in solving mathematical problems may be sectioned into different schemes based on the roles of the student, the teacher and the computer algebra system. The boundaries of the schemes presented herein are fairly subjective, and different role distribution schemes are undoubtedly possible. The use of the *black* and the *white* (also called *glass*) *box* methods for a computer algebra system has been discussed for years already (for instance, by Buchberger 1990). In simplified terms, it means that when using the black box method, one is only able to "see" the problem and the solution whereas the white box method allows one to follow the entire solution procedure (regardless of whether it is presented by a human or a computer algebra system). The distribution given in this article represents an attempt at creating a "grayscale" specifically in terms of equivalence checking. Not all schemes are rational to be applied to all types of problems. Their rationality is contingent on various factors.

The student writes on paper

For a full scheme system, let us start from the conventional and common variant (marked with "A" in tables in this article). Under this variant, the student writes the solution procedure on paper without being aided by a computer algebra system, and the teacher checks it, also without a computer algebra system. However, if the teacher is able to use a computer algebra system in correcting the papers (B), he or she can simplify his work by checking the equivalence of the lines of manipulation using a certain strategy (for instance, binary search) for locating the error(s). This variant requires no access to a computer algebra system on the part of the student, and this is important in view of the fact that the teacher's access to a computer algebra system is easier to organize.

Computer algebra system as a text editor

Although the use of a computer algebra system as just a text editor is clearly an underutilization of its capabilities, this variant should still be given some consideration. As it is, attempts have been made to create reasonable capabilities for entering mathematical text on computer algebra systems (using buttons, palettes and key combinations, etc). Computer algebra systems are appropriate for entering the solution procedure. Of course, this makes checking the tests much easier for the teacher (D), since they have no need to type in the solution (in full or in part) themselves. In

addition, it allows the student and the teacher to communicate over the Internet, for instance, which enables distance training. If the student has access to a computer algebra system and the teacher has not (C), it is possible, in principle, to check the printout of the solution. In this case, the only practical benefit for the teacher is the better readability of the script.

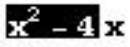
Computer algebra system as an assistant in discovering errors

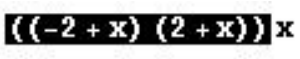
Next, it is natural to consider the variant (E) where the students, after entering the line, uses computer algebra system for checking its equivalence to the previous line himself, and corrects his own work, if necessary. This is a variant that is perhaps the most promising (see Kutzler 1996). Of course, the correct establishment of equivalence does not necessarily guarantee the rationality of a particular step. We may also consider a variant where the student checks not his own answer but a prescribed solution (F). This creates various possibilities, from simply checking a work made by a classmate to checking a solution procedure with a "subtle" error hidden in it.

Unlike the previous schemes, this variant expects a lesser role from the teacher and a greater activity from the student.

Computer algebra system as the maker of the next step

Delving deeper into the capabilities of a computer algebra system, it is natural to desire that the computer algebra system do the next step itself. Computer algebra systems have various commands which could be applied to a part of or to the entire expression (equation), and which could take the student to the next line (G, H). (However, the length of a step made by these commands would often differ from that made by a human.) Several computer algebra systems are equipped with commands like *Factor, Expand, Simplify*, which herein may be called *step operations*. (In principle, the systems may provide the teacher with the possibility of programming such commands of different levels himself). The student selects the operation and the computer algebra system performs it. In this case, it may seem that the student can trust the computer's work and skip the equivalence check. However, let us present a somewhat surprising example here. This is an issue that pertains more to the user interface to enable the selection of a sub-expression. Let us assume that in Mathematica it is necessary to perform the following factorization. There is the possibility of applying a command (for instance, "Factor", "Expand", etc.) to only a part of an expression. It is a good possibility. Unfortunately, erroneous results are possible even there, if, for instance, the user chooses a wrong sub-expression and applies factorisation in response to the expression x^2 - 4x





If we now check it, we find no equivalence.

Computer algebra system as a solver

If we do not want to follow the steps, we may have the system solve the problem as a black box. This is perhaps the most widespread application of computer algebra systems in educational setting today. Here we can distinguish between a variant where the option of having the computer algebra system solve the problem as a black box is selected right from the beginning (I) and one where the student has already solved part of the problem (J). Indeed, there are special commands available in computer algebra systems, such as *Solve*, *Simplify*, *Factor*, *Expand*, etc. (which herein

may be called *final operations*). Depending on the system and the situation, the same commands may also be executed for a single step.

As a rule, computer algebra systems are able to solve the problems of school and college algebra. In this case, expression equivalence check is not important either, considering the reliability of computer algebra systems.

Mention may also be made of the variant where the computer algebra system provides a textbook-like solution (K). In this case, the intermediate steps are also observable. Currently, this option is not directly available in computer algebra systems; in principle, however, it is programmable. Such a variant would provide the student with the opportunity to familiarize himself with the steps of the solution. The teacher would be able to obtain sample solutions and examine the suitability of problems for students. The full-solution approach would take us to the next variant.

Computer algebra system as a component of an intelligent tutoring system

If we had a feature that would fully present the steps of a solution for certain types of problems, the next variant conceivable would be checking the student's steps of solution against those presented by a computer algebra system. Of course, a problem can be solved in several different ways that are all correct, and to require the student to strictly adhere to a prescribed set of steps would be too one-sided. However, expression equivalence check would still play a major role in checking both the "prescribed" and the "innovative" steps of solution performed by the student.

Considering the fact that there are other capabilities suitable for an intelligent tutoring system (student module, tutor model, etc.), we can speak about intelligent tutoring systems. Since computer algebra systems already possess a number of the required features, they may have a future as expert modules of intelligent tutoring systems (Prank & Tonisson 2001). This means that they will be accessed for performing expression equivalence check, for instance.

The student as the evaluator of computer algebra system

The last variant (M) is one where advantage is taken of the fact that a computer algebra system is never perfect. Thus, the student can be assigned the task of checking whether there really is equivalence between expressions as shown by the computer algebra system. Emotionally, it is a fairly interesting variant. It seems to be more suitable for stronger students. However, this variant tends to be short-lived as the computer algebra systems are being steadily improved.

	Student	**Computer Algebra System (CAS)**	**Expression Equivalence Check**	**Teacher**
A	writes on paper			checks
B	writes on paper	assists teacher	teacher searches for errors	enters and checks expressions using CAS
C	writes in CAS	is a text editor		checks (without CAS)
D	writes in CAS	is a text editor and teacher's assistant	teacher searches for errors	checks, doesn't need to enter expressions himself
E	writes in CAS and checks the equivalence between a line and the previous line	searches for errors	student searches for errors	performs different operations depending on the approach
F	searches for errors in another person's solution	searches for errors	student searches for errors	performs different operations depending on the approach
G	writes and chooses the next step (e.g. Factor, Expand, etc.) for a part of expression	performs the operation	checks the correctness of the operation	performs different operations depending on the approach

H	writes and chooses the next step (e.g. Factor, Expand, etc.) for the entire expression	performs the operation	checks the correctness of the operation	performs different operations depending on the approach
I	writes and selects the final operation (e.g. Simplify, Solve, sometimes also Factor) at the beginning of the solution process	solves (from beginning to end), shows only the final result	checks the correctness of the operation	performs different operations depending on the approach
J	writes and selects the final operation (Simplify, Solve, sometimes also Factor) in the middle of the solution process	solves a certain part starting from the middle, shows only the final result	checks the correctness of the operation	performs different operations depending on the approach
K	examines the solution procedure	solves the problem, shows individual steps	(dependent on the structure of the solution procedure)	obtains problems and solutions
L	uses an intelligent tutoring system	is the expert module of an intelligent tutoring system	is important in checking expressions entered by student	obtains data on each student's errors, progress, etc.
M	checks CAS		may contain errors	is passive

4. What should a computer algebra system offer?

The following table presents evaluations of the necessity of one or another feature of a computer algebra system for the use of a particular scheme. Some evaluations are unambiguous (if a particular feature is unavailable then a particular scheme is inapplicable) while others are ambiguous. The necessity of the availability of a computer algebra system is expressed by the columns *CAS To Student* and *CAS To Teacher*. Availability here means the presence of both the possibility of and the skills for using a computer algebra system.

A computer algebra system's features may be listed with different degrees of detail. The list presented here represents only one possible way of doing it. It enables the user (teacher) to determine what schemes can be implemented using the computer algebra system at his disposal. Likewise, the computer algebra system developers can obtain ideas for improving their computer algebra systems.

The capabilities of expression equivalence check are described in the following columns. The column *Expression Equivalence Check* evaluates the necessity of a particular checking means in general, without imposing particular requirements on it (except that correctness should perhaps be assumed). Variant M is the only one to assume that we do not trust the expression equivalence check performed by a computer algebra system. For the implementation of the other schemes mentioned above, it is necessary that computer algebra systems correctly cope with checking the equivalence of expressions in practice. All the computer algebra systems under study (DERIVE, Maple, Mathematica and MuPAD) allow for the possibility of checking equivalence *Simplify(expression1 – expression2)=0* (or *Simplify(expression1/expression 2)=1* in checking the equivalence of expression1 to expression2. It appears that many school or college algebra problems are readily surmountable by computer algebra systems; however, there are also those that pose difficulties for them (Tonisson 2002).

Some computer algebra systems have special commands for checking the equivalence of expression equivalence (for instance, *testeq* in Maple) that perform a probabilistic check. (Tonisson 2002). Such commands can also be programmed using the programming tools available

in the computer algebra systems. The column *Equivalence Check Command* presents an evaluation of the necessity of such a separate command.

Considering the needs for the above-mentioned schemes, it is important, particularly for the student, that equivalence check be easily performable in terms of the user interface as well. If much effort is needed for entering expressions, there is not much hope for efficient use. Palettes offer some advantages in this respect (Fuchs & Dominik 1999). The column *Comfortable Equivalence Check* is dedicated to the very availability of a button, a palette or some other handy option (for example tool of selecting two expressions).

Also provided are three more columns that are directly related not so much to expression equivalence check as to the schemes: *The Possibility Of Stepwise Operations*, *The Possibility Of Final Operations* and *The Presentation Of Full Solution*.

The evaluation was performed on a five-point scale, and the meanings of the grades are as follows:

2 – availability inevitable, urgently needed

1 – availability recommended

0 – no difference, 0? – depends on the approach

-1– availability not recommended

-2 – availability unacceptable (or the feature must be inaccessible for the moment)

(The possibility of making one or another feature of a computer algebra system inaccessible for educational purposes would be important in several instances.)

(* – for teacher, ** – for intelligent tutoring system)

	CAS To Student	CAS To Teacher	Expression Equivalence Check	Equivalence Check Command	Comfortable Equivalence Check	The Possibility Of Step Operations	The Possibility Of Final Operations	The Presentation Of Full Solution
A	-2	0	0	0	0	0	0	-2
B	-2	2	2	1	1	0	0	1
C	2	0	0	0	0	0	0	-2
D	2	2	2	1	1	0	0	-2 (1*)
E	2	1	2	1	1	-1 (0*)	-1 (0*)	-2 (0*)
F	2	1	2	1	1	0	0	0
G	2	0?	2	1	1	2	0	0
H	2	0?	2	1	1	2	0	0
I	2	0?	2	1	1	0	2	0
J	2	0?	2	1	1	0	2	0
K	1	2	0	0	0	0	0	2
L			2**	1**	0	1**	1**	1**
M	2	0?	2	1	1	0	0	0
	dependent on the country, school etc.		exists somehow in every CAS	exists in some CASs	insufficient	partially implemented	partially implemented	not implemented

The very brief comments about the existence of the features in present computer algebra systems are placed in the last row.

5. Future trends

The scheme system presented in this article is just a sketch. Naturally, all the schemes presented can be described in more detail, and their efficiency can be investigated by conducting further experiments. The schemes can be expanded and adapted to be applicable to other mathematical topics. Much can be made for the improvement of computer algebra systems, both in terms of their user interfaces and their mathematical capabilities. Quite a few items are already programmable in the existing computer algebra systems. However, a promising trend seems to be the creation of a separate interface where the mathematical capabilities of computer algebra systems (including equivalence check) could be realized more efficiently.

REFERENCES

-Buchberger, B., 1990, "Should Students Learn Integration Rules?" *SIGSAM Bulletin*, Vol 24-1.

-Fuchs, K., Dominik, A., 1999, "MATHEMATICA palettes – a methodical way to provoke students into using mathematical strategies." *Proceedings of the Fourth International Conference on Technology in Mathematics Teaching (CD)*. Plymouth.

-Herget, W., Heugl, H.., Kutzler, B., Lehmann, E., 2000, "Indispensable Manual Calculation Skills in a CAS Environment." Austria. http://www.kutzler.com/.

-Kutzler, B., 1996, *Improving Mathematics Teaching with DERIVE*. Chartwell-Bratt.

-Prank, R., Tonisson, E. , 2001, "Is the Domain Expert Module for expression manipulation exercises ready?" *Proceedings of the Tenth International PEG Conference*. Tampere, pp. 51-56.

-Tonisson, E., 1999, "Step-by-step Solution Possibilities in Different Computer Algebra Systems." *Proceedings of ACDCA Summer Academy: Recent Research on DERIVE/TI-92-Supported Mathematics Education*. Gösing, Austria. http://www.acdca.ac.at/kongress/goesing/g_toniss.htm.

-Tonisson, E., 2002, "Expression Equivalence Checking in Computer Algebra Systems." *Technology in Mathematics Teaching. Proceedings of ICTMT5 in Klagenfurt 2001. Schriftenreihe Didaktik der Mathematik vol 25*. Vienna, pp 329-332.