MathCheck: a tool for checking math solutions in detail

A. Valmari
Professor
Tampere University of Technology, Department of Mathematics
Tampere, Finland
E-mail: antti.valmari@tut.fi

T. Kaarakka
Lecturer
Tampere University of Technology, Department of Mathematics
Tampere, Finland
E-mail: terhi.kaarakka@tut.fi

Conference Key Areas: Online Engineering Education, Mathematics, Engineering Education Research

Keywords: Mathematics, education tools, (in)equation chain checking, feedback

INTRODUCTION

Systems for automatic assessment of exercises, such as Stack [1], only check the final answer and a pre-defined set of intermediate answers. The teacher must give the right answers to the tool. Intermediate answers require that the student solves the problem along the path that the teacher chose.

In this study we discuss the MathCheck tool [2] that supports a different philosophy. It has been designed for providing feedback to the students when they work independently at home, not for assessing the solutions and keeping track of points. It inputs the whole symbolic calculation or reasoning by the student and checks its every step for mathematical correctness. MathCheck does not check that the student followed a particular path, found the right solution, nor even that the student solved the right problem. It only checks that each step in the student's reply is mathematically correct. The student need not tell MathCheck what particular problem is being solved, and the teacher need not tell the right solution. This makes MathCheck flexible and free from bureaucracy (and unable to deliver points). If the delivery of the solutions to the teacher is desired, the student can simply copy the answer from MathCheck and email it to the teacher.

We first motivate MathCheck with a simple example. Then we describe what parts of mathematics MathCheck can deal with. Because of a fundamental phenomenon called undecidability, no computer mathematics system can be perfect. This issue and how it affects MathCheck are discussed next. Although the input language of MathCheck is not much different from the input languages of other computer mathematics systems, big problems arose in our educational experiments. Therefore, we devote a section to the input method issue. The next section reports on our teaching experiments. Finally we discuss ideas for future work and conclude the study.

MathCheck is at http://www.math.tut.fi/mathcheck/. The reader is invited to try the examples in this study and the reader's own examples. No registration or user identification is needed. MathCheck keeps no record of who use it and what they do.

1 MOTIVATION

Consider a student solving a problem at home in the evening. The student has to simplify $\sin\left(\frac{\pi}{2}+x\right)-\sin\left(\frac{\pi}{2}-x\right)$ The student writes the following on paper:

$$\sin\left(\frac{\pi}{2}+x\right)-\sin\left(\frac{\pi}{2}-x\right)=\cos x-\cos(-x)=\cos x+\cos x=2\cos x$$

The next day the student goes to a problem session and learns that this answer is wrong. The right answer is shown, but the student has no time to think about it, because the teacher moves to the next problem.

The next year MathCheck is in use. Another student arrives at the same solution. The student gives it to MathCheck as

```
sin(pi/2+x) - sin(pi/2-x) = cos x - cos(-x) =
cos x + cos x = 2 cos x
```

MathCheck replies as is shown in *Fig. 1*. The student learns that the answer is wrong. The student also learns *the exact place* where the reasoning went wrong.

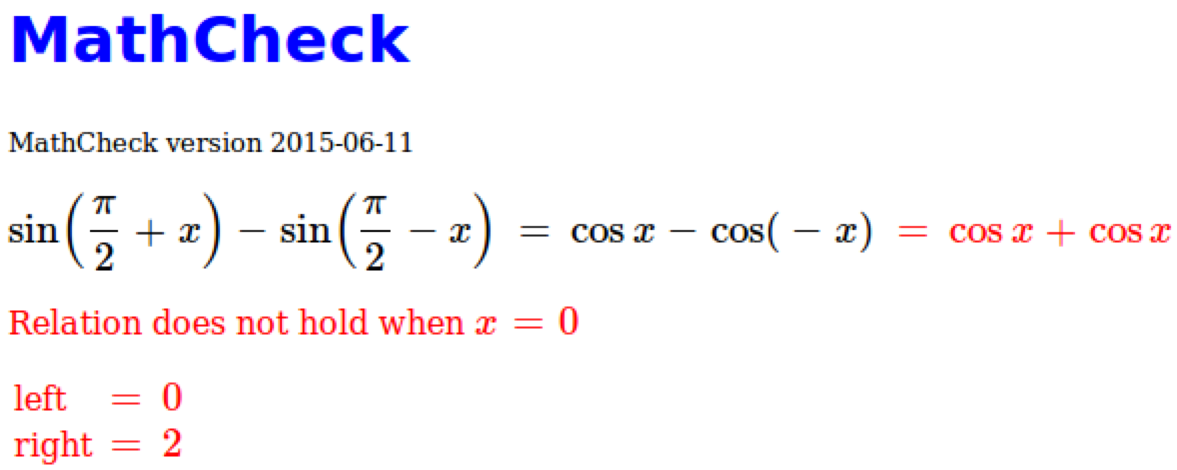

Fig. 1. An example output by MathCheck

By looking at the red expression and the preceding expression, the student realizes that $\cos(-x)$ is not $-\cos x$. Perhaps it is $\cos x$? Checking from a book is old-fashioned, so the student opens another MathCheck window and writes `cos(-x) = cos x` . MathCheck accepts it.

Then the student clicks the back button of the web browser window that is still showing *Fig. 1*. The web form that contains the original answer re-appears. The student replaces the second line of the original answer by `cos x - cos x = 0` and clicks submit. MathCheck accepts this improved solution.

This example illustrates how MathCheck helps the student to improve the solution while still at home until it is free from simple errors. That is, the student can progress further at home. This often means progressing further in general, because students seldom continue with an exercise after its solution has been presented in a problem session.

Many students on our courses seem to behave as if the important thing with a mathematical exercise were to find the correct answer, instead of to learn a particular mathematical skill. They guess the answer or get it from, say, Wolfram Alpha [3], instead of reasoning it themselves. Systems that only check the final answer do not discourage this kind of behaviour. If the exercise cannot be solved without reasoning, the performance of students drops dramatically. Bad performance from hard exercises has caused a tendency among the teachers to present many easy and few hard exercises, and let students pass without solving any of the hard ones.

If students do their homework by pen and paper and check the answers with modern applications, the learning process is constructive, and therefore beneficial for the students. Unfortunately, if students do not immediately know how to approach the problem, they might be tempted to obtain the solution by using these modern symbolic applications. Students themselves might think they learn something, but it is very hard to learn by reading the calculations only. This kind of method is more behavioristic. MathCheck is a tool with very constructive ideology and students really construct their own ideas and own formulas and MathCheck gives them the feedback immediately. Behaviorism and constructivism were compared in [4].

2 CURRENT CAPABILITIES OF MATHCHECK

MathCheck is in an early stage. It has been developed by one person as a low-priority project during January–June 2015 and May 2016.

From the point of view of users, MathCheck is a web page with an input box. The user types or pastes the solution to the box and clicks the submit button. MathCheck gives a reply of the kind in *Fig. 1*. To type a new solution or modify the previous one, the user may click the back button of the web browser. More than one solution can be processed in the same batch by separating them with `#newproblem`.

Currently MathCheck can deal with addition, subtraction, multiplication (both invisible and ·), division, absolute value, square root, power, logarithm, sin, cos, tan, and derivatives. Numbers can be integers, fractions, mixed numbers, decimal numbers, and π and e . Simultaneously there can be at most three variable names, chosen from small-case and capital letters (excluding `e`). So MathCheck can be used, for instance, to check whether the power function is associative, by typing `a^(b^c) = (a^b)^c`. Variables from i to n and I to N assume integer values, while the rest assume real values.

MathCheck checks chains of relations, where the relation operators may be <, ≤, =, ≥, or >. MathCheck checks each relation by first trying to prove it using a proof engine that is currently rather simple. If this succeeds, MathCheck paints the relation symbol green. If it fails, then MathCheck tries to find a counter-example by trying many combinations of values of variables. With other relations than =, MathCheck also uses a numeric hill-climbing algorithm to find a counter-example. If no counter-example is found, then MathCheck paints the relation symbol black.

One of the strengths of MathCheck is that it is very careful with undefined expressions. For instance, while Wolfram Alpha [3] claims that $\frac{x^2}{x}=x$ and $(\sqrt{x})^2=x$ are true, MathCheck reports of the former that if $x=0$, then the left hand side is undefined while the right hand side yields 0 , and of the latter that if $x=-1$, then the left hand side is undefined while the right hand side yields -1 . (The formulae can be given to both tools as `x^2/x = x` and `(sqrt x)^2 = x`.) This is important in mathematics. The advertisement image [5] of the Viope Practice Tool [6] shows no awareness of this issue. In our opinion it is dangerous. If the student starts solving $\frac{x^2+x-2}{x+2}=1$ by multiplying both sides with $x+2$ without recording that x must not be -2 , then, in addition to the correct solution 2 , also an incorrect solution -2 is obtained.

In MathCheck, value combinations can be excluded from the domains of expressions with `#assume … #enda`. MathCheck rejects `(x^2+x-2)/(x+2) = x-1` but accepts `#assume x!=-2 #enda (x^2+x-2)/(x+2) = x-1`. This kind of an assumption can be any logical expression composed of relations with the ¬, ∧, and ∨ operators that is never undefined.

The complicated rules for exponentiation when the base is negative (or zero) have been implemented. While Wolfram Alpha deems `(-8)^(2/3)=4` as false, MathCheck accepts it. (To `(-8)^(2/3)=` Wolfram Alpha replies $4(-1)^{2/3}$ and approximates it as $-2+3.46i$).

The experiments reported in this study were made with the June 2015 version of MathCheck. It computes with precise rational numbers until the numerator or denominator grows too big, and then moves to double precision floating point arithmetic. Great care had been taken to avoid false alarms caused by rounding errors, but their possibility had not been fully ruled out. There are no reports on students having encountered false alarms in the experiments. However, teachers that were preparing exercises have met them once or twice.

In May 2016, double precision arithmetic was replaced by interval arithmetic that maintains two double precision values such that the correct value is between them. This is believed to remove false alarms altogether. Unfortunately, interval arithmetic was exceptionally tricky to implement, so the possibility of false alarms caused by programming bugs cannot be ignored until there is more experience with the new version.

3 UNDECIDABILITY, COMPUTER MATHEMATICS, AND MATHCHECK

It is a fundamental result in theoretical computer science that there is no algorithm that always replies correctly to any mathematical question in a finite time. This phenomenon, called undecidability, already arises with the mathematical concepts known to MathCheck. For instance, if $f(x)$ can be written in the input language of MathCheck, then MathCheck can be challenged to prove or disprove the existence of solutions to $f(x)=0$ by writing $\frac{1}{(f(x))^2}>0$. If there is no solution, then MathCheck should accept it as correct, but if there is a solution, then ideally MathCheck should report that the left hand side is undefined with such a value of x . As another example, the absolute value function causes trouble to finding counter-examples to inequations, but this cannot be solved by leaving the absolute value function out, because it can be implemented as $\sqrt{x^2}$, and leaving either square or square root out would seriously damage the purpose of MathCheck.

Because of undecidability, all non-trivial computer mathematics systems are imperfect in one way or another. The best that can be aimed at is that the system either gives the correct answer or replies “I don't know”. By implementing more advanced heuristics, the “I don't know” answer can be made rarer. There is no end to how far the system designer can go in this direction, so the designer must just stop somewhere. The designers of Wolfram Alpha chose to stop before implementing the full set of rules for exponentiation with a negative base, the designer of MathCheck chose to go that step further. Whether or not some feature is worth implementing depends on the purpose of the system and the difficulty of implementing the feature.

The purpose of MathCheck is to give pedagogically useful feedback to the students. As a consequence, wide range of mathematical concepts was considered more important than the ability to always give an answer. MathCheck gives no answer by painting the relation symbol black (instead of red or green). The goal is that MathCheck must give no false alarms (or must give them so infrequently that it is not a problem), and it must give true alarms “reasonably often”, whatever that means.

In practice, unless a strongly restrictive `#assume` … `#enda` is used, MathCheck is very good in giving true alarms for the following reasons. Equalities of the form $f(x)=g(x)$ typically only have a small number of solutions, when $f(x)$ and $g(x)$ are composed of functions known to MathCheck and are not the same function. Therefore, it is highly likely that at least one of the value combinations that MathCheck tries is not a solution and thus qualifies as a counter-example. If the user knows the value combinations that MathCheck tries, then it is possible to construct $f(x)$ and $g(x)$ such that MathCheck fails to see that they are different. However, the possibility that a student accidentally hits such $f(x)$ and $g(x)$ is small.

With an inequation such as $f(x)\leq g(x)$, MathCheck uses several starting points and tries to walk from each into a direction that makes $f(x)-g(x)$ grow. Here, too, in typical situations where $f(x)\leq g(x)$ does not hold, it is highly probable although not certain that at least one of the walks eventually makes $f(x)-g(x)$ so much positive that the result is certainly positive according to the interval arithmetic that MathCheck uses. With both equations and inequations, MathCheck can be blinded by choosing $f(x)$ and $g(x)$ so that evaluating any counter-example leads to outside the range of double-precision arithmetic. Again, it is possible but rare, because it requires, for instance, the exploitation of the 16th decimal or the use of intermediate results that are bigger than 10^{308}.

All this means that MathCheck never gives false alarms (assuming that there are no programming bugs), but sometimes, not very often, fails to give a true alarm. This suffices for giving students feedback. This also means that the current version of MathCheck must not be used for automatic assessment of examinations. Automatic assessment is possible if either the questions are restricted so that a decision algorithm for checking the answers exists (as is trivially the case if the correct answer is unique), or the tool can give a definite verdict in most cases and delegates the rest to a human. The current version of MathCheck gives a definite verdict in most cases where it is “incorrect” (that is, paints the relation symbol red), but often does not give a verdict when it is “correct” (that is, paints the relation symbol black instead of green).

Improving the proof engine of MathCheck is one reasonable direction for future development. So it may be that one day MathCheck becomes usable for automatic assessment. However, it is a low-priority goal. The high-priority goal is to make it possible for students to get automatic feedback that helps them study on their own. In particular, MathCheck should be able to give feedback on any step of a long calculation, instead of just checking the eventual result.

Because of this philosophy, no user identification has been implemented to MathCheck, and there is no button “send my answer to the teacher”. Leaving user identification out saved us from a lot of time-consuming but scientifically uninteresting programming, and made MathCheck simpler to use.

If sending answers to the teacher is desired, the student can easily copy the answer and send it via email, or the features offered by a learning environment can be used. Sending via email requires two mouse button clicks (to select the answer box of MathCheck and select the email window) and three key pressings (<CTRL>-A, <CTRL>-C, and <CTRL>-V), which is so easy that spending time on programming an alternative solution was not justified. The teacher can equally easily copy the answer to MathCheck, click submit, and inspect the answer similarly to inspecting an answer that has been hand-written on paper, with the exception that the teacher need not spend effort on checking the detailed correctness of each step, because MathCheck has done that. More than one solution can be processed in this way in a single batch, by separating them with `#newproblem`.

4 THE INPUT SYNTAX

As is obvious from the examples in this study, the input language of MathCheck is textual. This is similar to many other computer mathematics systems, including MatLab [7] and Wolfram Alpha [3], and different from the formula editors in Microsoft Word and LibreOffice. It is also different from modern handheld CAS (computer algebra system) calculators used at schools.

In the latter input method, mathematical symbols can be selected from menus. After selecting the summation symbol ∑, the window shows it with boxes above, below, and to the right of it. The user navigates between the boxes by clicking the mouse or with arrow keys and fills in the boxes in a similar manner, choosing symbols from menus and so on. This user interface ensures that right ingredients go to right places (for instance, what should be above the symbol ∑, goes there). It may also be possible to feed in the formula or parts of it as text.

The biggest differences between the two input methods are that with the latter, the user sees the formula as it is printed in mathematics already when feeding it in, and it is not necessary to use special syntax such as `{…} over {…}` to control the structure of the formula. For these reasons newcomers and those who write formulae only seldom tend to find it easier. Experienced mathematics authors find it slow and clumsy. For an example of a discussion on the two input methods and attempts to combine their advantages, please see [8].

Originally we took it for granted that the input language might be a problem but not a major problem. Every engineering student in Tampere University of Technology must pass a programming course and thus become familiar with some programming language. They also use MatLab in at least one course. Unofficial information obtained from the students tells that many use Wolfram Alpha when solving homework exercises in mathematics courses (or even use Wolfram Alpha to solve the problems on their behalf). So we expected that everyone is familiar with textual feeding in of mathematics.

What we did not think about in advance was that the students in our first experiment (please see Section 5) had just arrived to the university, and thus had not yet learnt any programming language, MatLab, or Wolfram Alpha. They did have serious problems with the input language. This affected negatively the results of the experiment, as discussed in Section 5. It was quite hard for the students to start using MathCheck. Several students told us that the interface was difficult to use. Fortunately, after a couple of exercises, they began to succeed well.

The lesson is that until the engineering world switches completely to what-you-see-is-what-you-get user interfaces, the ability to use a textual user interface for feeding in structured information must be treated as a learning goal in itself that is explicitly addressed in some course early on in the curriculum.

5 EXPERIENCE

We have tested MathCheck in two different courses: Engineering Mathematics 1 in autumn 2015 (a first-year course) and Algorithm Mathematics in spring 2016 (a second-year course). More than 150 students used MathCheck in the former and 120 students in the latter. The idea was that the students would use MathCheck in one or two homework assignments every week in the first case, and a few times with very special exercises in the second case. Students used it to check their solutions to homework assignments. At the end of the first course, the students gave us feedback on using MathCheck.

In Engineering Mathematics 1, students used MathCheck when studying limits, continuity, and derivatives, in particular when they manipulated expressions. For example, one of the assignments was “Prove that $\sinh^2(x)=\frac{\cosh(2x)-1}{2}$. Check your calculation using MathCheck.” The experiment taught us teachers that if the exercises are too easy, then there is no point in using MathCheck. If no intermediate steps are needed to solve an exercise, then the ability of MathCheck of checking intermediate steps is of no advantage. It may be that too easy exercises were given, because giving rather easy exercises has been the tradition, because too many students have been unable to solve more difficult exercises at home. In any event, the experiment did not appropriately test the ability of MathCheck to help students solve more difficult problems at home.

As feedback from Engineering Mathematics 1, we had 120 answers. Of them, 53 considered MathCheck useful, 48 were of the opposite opinion, and 19 replied that they had not used MathCheck. Inconsistencies in the replies revealed that many students who reported MathCheck as useless had actually not tried it, although they claimed that they had. For instance, many students said to our research assistant in group interviews that the input language of MathCheck is a stumbling block; instead of what it is like, it should be like the input language of MatLab. In reality, the part of the input language that the students had to use is virtually the same as the corresponding part of the input language of MatLab. We guess that in the interview situation, when one student had given this answer, many others found it as a nice excuse for not trying MathCheck. So they joined the opinion without much thought.

It also seems that many students rejected or failed to understand the idea of checking the solution in full, instead of finding or checking the final answer. Their previous experience has been with symbolic calculators and other tools that find the answer, and we failed to communicate them that the purpose of MathCheck is different.

Because the exercises were too easy in the Engineering Mathematics 1 experiment, more complicated exercises were given to the students in the Algorithm Mathematics experiment. For example, one of the assignments was “Simplify the expression $f(x)=\frac{\ln((x^2+4x-12)^2)}{\ln(100)}-\frac{\ln(x+6)}{\ln 10}$, and give the answer in terms of the log-function“. In Engineering Mathematics 1 we asked only “check your answer using MathCheck”, but in Algorithm Mathematics we demanded students to send a PDF file of their calculations via the Moodle course page.

The author TK has extensive experience of many years of teaching mathematics to first- and second-year students. She has seen that it is very hard for students to approximate expressions. At school they are used to calculating strict values, but in the real word it is more valuable to estimate limits. In the Algorithm Mathematics course the students must approximate to calculate time complexity. TK observed that the students who used MathCheck for studying how to approximate functions upwards or downwards, understood the concept of time complexity quicker. Nevertheless, we have not done any research on the actual learning process, we only studied the students’ opinion in using MathCheck. The next step is to do more research on the actual learning process.

In addition to the previously mentioned MathCheck exercise, there was another exercise in the same week: Approximate the expression $\log(n^4+n^3-5)$ upwards to find $c \in R$ and $n_0 \in N$ such that $\log(n^4+n^3-5) \leq c \log(n)$ when $n \geq n_0$. A week later the concept of time complexity was considered. There were two exercises, which we asked the students to check with MathCheck and deliver the output of MathCheck as a PDF file via Moodle: Prove using the definition that (a) $2n^3-n^2+5n=O(n^3)$ and (b) $2n^2-10n+3=\Omega(n^2)$. By completing the exercises at both weeks, students earned two exercise bonus points.

At the examination, there was the following question about time complexity: Prove that $\log(2n^3-6n^2)=\Omega(\log(n))$, using the definition. *Table 1* shows the dependency between the points earned from this question and the points earned from MathCheck exercises.

Table 1. Numbers of students earning MathCheck and examination points

	0 exam points	1 exam point	2 exam points
0 MathCheck points	49	8	14
1 MathCheck point	9	8	13
2 MathCheck points	4	8	22

The correlation between the exam points and MathCheck points is 0.4845. The p-value is less than 0.001, which means that the correlation is statistically highly significant. Successful use of MathCheck clearly predicts success in the examination. We do not know how much this was because of MathCheck helping learning and how much because the best students tend to succeed in almost everything.

MathCheck was also tried in a high school course with 19 pupils [9]. Because pupils had no idea of how to use a textual input language, the experiment was a failure. 53% of the pupils deemed MathCheck as difficult to use, 16% found it useless, and 31% replied that it is or could be useful.

6 CONCLUSIONS AND FUTURE PLANS

Our pedagogical experiments have been small and the results have been affected by the input language problem. Furthermore, the experiments were designed too much as add-ons to existing courses, and not enough attention was paid to designing the problems where MathCheck was to be used. The ability of MathCheck of checking intermediate steps of a solution is only useful if the problem is so laborious that intermediate steps are needed, and the existing problems were not sufficiently like this.

As a consequence, there is not yet any strong evidence in favour or against MathCheck. It has become evident that MathCheck requires new kind of thinking from both teachers and students. Of course we hope that this is a symptom of MathCheck directing the students towards the kind of mathematical thinking that we wish them to acquire. For instance, we want them to be fluent with the abstract recursive structure of formulae (that is, how formulae consist of sub-formulae). The problems with the input language of MathCheck suggest that they do not have that skill when they arrive at the university.

The next planned major feature is propositional logic. There it is easy to implement a perfect proof engine. This would make it possible for university students to get exercise on logic and simple proofs.

Regarding predicate logic, the development of a sufficient checking algorithm is far too difficult. A limited idea seems possible, where the teacher gives MathCheck a formula such as $1 \leq k \leq n \wedge \forall i ; 1 \leq i < k : \forall j ; i < j \leq n : A[i] < A[j]$ and expresses the same claim to the student in natural language. The student has then to write the claim as a formula. MathCheck compares the two formulas and gives the student feedback. This makes it possible to get exercise on formulating precise claims. This is especially useful for computer science students.

When the proof engine becomes better, MathCheck can also be adapted to abstract algebra, such as groups, rings, and fields. Unlike in more familiar mathematics, the types of proofs that students are expected to learn in abstract algebra are very similar to how computer proof systems work. Furthermore, in small finite algebraic structures perfect checking is possible by trying all possible elements.

REFERENCES

[1] System for teaching and assessment using a computer algebra kernel, http://www.stack.bham.ac.uk/ 2016-05-27

[2] Valmari, A. (2016), MathCheck relation chain checker, *Proceedings of the Finnish Mathematical Days 2016*, pp. 44-46.

[3] Wolfram Alpha, https://www.wolframalpha.com/ 2016-05-26

[4] Weegar, M.A. and Pacis, D. (2012), A Comparison of two theories of learning — behaviorism and constructivism as applied to face-to-face and online learning, *E-Leader Conference, Manila, Philippines*, http://www.g-casa.com/conferences/manila/papers/Weegar.pdf, 2016-05-04

[5] Viope Practice Tool (screen capture) https://www.viope.com/img/layout/products/mathautoasses.png 2016-05-26

[6] Viope Practice Tool https://www.viope.com/#!/products/mathtool/mprac 2016-05-26

[7] MathWorks, MatLab, http://se.mathworks.com/products/matlab/ 2016-05-27

[8] Kastrup, D. (2002), Revisiting WYSIWYG paradigms for authoring LaTeX, *TUGboat*, Vol. 23, No. 1, pp. 57-64.

[9] Rasimus, E. (2016), MathCheck oppimisen apuna, *BSc. Thesis*, Tampere University of Technology