To What Extent Should Computer-Assisted Proofs Be Trusted?

Extended Project

Jack Morrison | 1189

Supervisor: Mr M Dowding | Westcliff High School for Boys| 16635

Table of Contents

Abstract

Computer-assisted proofs have been a controversial topic of mathematics since the use of a computer program to verify the Four Colour theorem in 1976 [1]. Some mathematicians have suggested that computer-assisted proofs are not 'real' proofs, because they do not always involve logical deduction. They are also not always verifiable by humans, and can be prone to error, and this leads to people doubting their validity. It is the intention of this study to consider whether the use of computer programs to prove a mathematical problem's solution are as effective as deductive proofs. It investigates the types of proofs currently accepted in the mathematical community, and compares these to computer assisted proofs. Also, the Four colour theorem and Kepler conjecture are investigated, as both were proved with the assistance of computers. This study concludes that computer assisted proofs are becoming increasingly more reliable, and will soon be regarded as equally mathematically valid as deductive proofs. Currently however, there are still some reliability issues, such as hardware and software reliability, lack of human verification and lack of logical deduction, although these problems are being addressed by using more reliable hardware and software, using proof assistants to verify proofs, and using them for peer reviewing.

Introduction: What is a proof?

One popular type of proof is an Euclidian Proof, which consist of four main elements; definitions, axioms, propositions and mathematical proofs of the propositions. [2]

Firstly, there are definitions. These explain mathematical terms and ideas, for example, the definition of a natural number is:

> *"0" is a natural number.*
>
> *Each natural number has a unique successor, such that:*
>
> > *the successor of a natural number is also a natural number;*
> >
> > *distinct natural numbers have distinct successors;*
> >
> > *no natural number is succeeded by "0".*
>
> *Nothing else is a natural number.*

Proof I [3]

This definition covers all of the conditions of a natural number, and is a special type of definition known as an inductive definition, because it defines a 'natural number' in terms of itself, and so is recursive. Not all definitions are recursive, for example, the definition of isomorphic graphs is 'Graphs that show the same information, but are drawn differently', which is just a statement [4].

The next elements are axioms, which are statements that are assumed to already be true, for example commutative law. This is where the order of the addends does not change the value of the sum in addition, e.g. $a + b = b + a$, whereas in subtraction, the order of the minuend and the subtrahend will result in varying differences, e.g. $a - b \neq b - a$. The axioms in Proof I are known as the Peano axioms [3]. These are the mathematical statements which are used to prove a set is of natural numbers. They only use axioms which are already assumed to be true, for example one axiom states that '*distinct natural numbers have distinct successors'* by stating that:

$$\forall\, x, y \in \mathbb{N}, if\ S(x) = S(y), then\ x = y$$

as these axioms can be assumed. [5]

The third element is a proposition, which is a declaration that can be either true of false, for example, 'The product of two even integers is always even'.

To show if a proposition is true or false, we use mathematical proofs. These use axioms and definitions in order to prove the proposition. The proof for the proposition above could be:

> *Consider two even integers, a and b.*
>
> *Even integers can be written as $a = 2x$ and $b = 2y$, where x and y are integers, because all even numbers have a factor of 2 (this is an axiom).*
>
> *The product of $ab = 4xy = 2(2xy)$ and so as it has 2 as a factor, it is also even.*
>
> *This proves that the product of any two even integers is even.*

This proof can also be written as follows:

$$
\begin{aligned}
& a, b \in \mathbb{Z} \\
& a = 2x \\
& b = 2y \\
\therefore\ & ab = 4xy \\
& = 2(2xy)
\end{aligned}
$$

Proof II

These are the four elements to an Euclidian proof. This is not the only structure of a proof, because it cannot be used to prove every problem; however, it is quite widely used.

Types of proofs

There are many types of proof, however some of the most commonly used are; direct proofs, proofs by mathematical induction, proofs by contraposition, proofs by contradiction, proofs by exhaustion and computer-assisted proofs.

Direct proof

Direct proofs are the simplest type of proof, and they use the Euclidian elements previously discussed. The proof in the section above is an example of a direct proof. These are the most commonly used proofs, because their structure is very simple, as they just use existing axioms and theorems, and so do make any other assumptions.

Proof by mathematical induction

Proof by mathematical induction has a starting case to be proved, usually referred to as the 'base' case. From here, the next case is proved using an induction rule. The first case is usually algebraically represented by n, and the next case $n + 1$. Once the induction rule is proved, the proof is true from a starting case which can be proved, usually 1. An example of this is given as proof I on the previous page. This proof is most commonly used with natural numbers, and so concludes 'this is true for all natural numbers' or '$\therefore\ true\ \forall\ n \in \mathbb{N}$'. It can also be used in reverse, and so can be proved for all negative integers, for example by proving –1 as the base case, and them proving true for n and $n - 1$.

Proof by contraposition

Proof by contraposition is, simply put;

if P $\implies$ Q, then ¬Q $\implies$ ¬P.

This means that if P implies Q, then not Q implies not P. An example of this is 'If something is a carrot, then it is orange'. The contrapositive of this is 'If something is not orange, then it is not a carrot'. These statements are contrapositives, and the same practice can be used in mathematical proof, for example if a number is two, then it is even, and so if a number is not even, then it is not two. [2]

Proof by contradiction

Proof by contradiction works by assuming the opposite of a statement is true, and then tries to prove that. Inevitably, a contradiction occurs, and so the assumed statement must be false, and so the original statement is true. An example is that there is no 'smallest' rational number greater than zero. The proof is as follows:

Assume there is a smallest rational number, r.

$\frac{r}{2}$ is also a rational number which is greater than zero, but now also smaller than r.

This means there is a contradiction to our original assumption, and so that assumption must be false.

Therefore, there is no smallest rational number which is greater than zero.

Proof III [6]

Proof by exhaustion

Proof by exhaustion is where a proposition is split up into a finite number of cases, and then all of these cases are individually proved. This is known as 'brute force', as every possibility is checked. There can potentially be a large number of cases that need to be proved, and therefore computers are useful for proof by exhaustion, because they can do lots of calculations at a fast speed.

An example of a proof by exhaustion is that every perfect cube is a multiple of 9, one more than a multiple of 9, or one less than a multiple of 9. The proof would consider 3 cases, which are; multiples of 3, one more than a multiple of 3, and one less than a multiple of 3.

Case 1:

$n = 3y$ *This means that n is a multiple of three*

$n^3 = 27y^3 = 9\times 3y^3$ *and therefore n^3 is divisible by nine.*

Case 2:

$n = 3y + 1$ *This means that n is one more than a multiple of three*

$n^3 = 27y^3 + 27y^2 + 9y + 1$
$= 9\times(3y^3 + 3y^2 + y) + 1$ *and therefore n^3 is one more than a multiple of nine.*

Case 3:

$n = 3y - 1$ *This means that n is one less than a multiple of three*

$n^3 = 27 - 27y^2 + 9y - 1$
$= 9\times(3y^3 - 3y^2 + y) - 1$ *and therefore n^3 is one less than a multiple of nine.*

Proof IV [7]

These three cases show all of the possible combinations which could be tested, as every number is either a multiple of three, one more than a multiple of three, or one less than a multiple of three, and therefore every case is represented. This is a common theme in proofs by exhaustion, because obviously not every value can be tested, because there are an infinite amount of numbers, and therefore if a set of cases can be created which represent every possible combination, then a proof by exhaustion is feasible.

Saying this, there may be a case, such as the Four Colour Theorem, where there are hundreds or even thousands of cases which need to be proved, and to find these cases there are hundreds of constraints which need to be considered. This means that a proof by exhaustion is feasible, but only with the assistance of a computer, as to prove something of this magnitude by hand would be next to impossible due to the time and processing power required.

What is a computer-assisted proof?

Computer-assisted proofs started to be introduced during the 20th century, as computers started to advance, and allowed calculations to be computed by programs a lot faster that a human was able to. They are most commonly used with proofs by exhaustion, due to the fact that they can complete lots more calculations per second than a human can, and therefore can prove lots of cases in a short amount of time, which are the bases of proofs by exhaustion.

An example for a structure of a computer-assisted proof may be:

1. The problem is condensed into a finite number of cases, which can all be individually proved.
2. An algorithm is created which will analyse the cases individually and will return if they satisfy the conditions of the original problem.
3. The algorithm is executed on all of the cases to prove that they are true.

This shows that the computer is mainly used to compute the large number of cases, because if only a small group of cases needed to be proved this way, then a human could do it fairly quickly, and so the use of a computer would be redundant. [8]

What is considered a 'complete' proof?

Some mathematicians believe that proofs by exhaustion should not be considered as complete proofs because they are not elegant, and there is the possibility that if there are a large number of cases, then there may not be a mathematical connection between them, and it may just be coincidence. They prefer other methods of proof, including some which are explained above, like proof by mathematical induction, because they are more elegant, and show the reason behind a theorem which is trying to be proved. [7]

Mathematicians therefore extend this opinion to computer assisted proofs, because they are essentially proofs by exhaustion with many more cases, and so require the assistance of a computer to process these cases. They gave the opinion that it extended the use of empirical evidence, which is used in many physical sciences, into mathematics, where it should not be, as it gives the potential for the proof to be wrong. [2]

However, the majority, if not all mathematicians would agree that a proof with a small amount of cases to be proved is undoubtedly true. For example, the following proof has just two parts; proofs relating to odd numbers and to even numbers. It is still a proof by exhaustion, as it proves a collection of cases, but because the cases can be proved by a human simply then they are a valid proof.

The conjecture is that if a and b are integers, the product ab is odd if and only if a and b are both odd. The proof is as follows:

Part 1:

If a and b are both odd, then ab is odd.

$$
\begin{aligned}
a, b, n, m &\in \mathbb{Z} \\
a &= 2n + 1 \\
b &= 2m + 1 \\
ab &= (2n + 1)(2m + 1) \\
&= 4nm + 2n + 2m + 1 \\
&= 2(2nm + n + m) + 1 \\
&\therefore ab \; is \; odd
\end{aligned}
$$

Part 2:

If ab is odd, then a and b are both odd. This part is required as the conjecture stated 'if and only if', which means that both conditions must be true for the conjecture to be proved, so the proof needs to be done both ways around.

$$
\begin{aligned}
&ab\ is\ odd\\
&assume\ a\ and\ b\ are\ not\ both\ odd\\
&\boldsymbol{Case\ 1}:\\
&a = 2n\\
&b = 2m + 1\\
ab &= 2n(2m + 1)\\
&= 4nm + 2n\\
&= 2(2nm + n)\\
&\therefore ab\ is\ even\ \therefore contradiction
\end{aligned}
$$

$$
\begin{aligned}
&\boldsymbol{Case\ 2}:\\
&a = 2n\\
&b = 2m\\
ab &= (2n)(2m)\\
&= 2(2nm)\\
&\therefore ab\ is\ even\ \ \therefore contradiction
\end{aligned}
$$

Both cases are contradictions, so a and b must both be odd.

By part 1 and part 2, the original conjecture has been proven.

Proof V [9]

This proof is considered complete by mathematicians because it can be split into two cases; a and b are even, or a is even and b is odd, and these cases can be proved by contradiction in a few lines. This means that the proof can be easily checked for errors, and therefore they can be sure that it is correct. The use of proof by exhaustion here is not questioned by mathematicians, presumably because the number of cases can be counted and proved easily by a human.

However, many argue that increasing the number of cases makes the proof invalid, because there is a higher chance of an error occurring in the proof. While this may be true, it does not mean that it is not possible to use computers to help prove conjectures which would take too long to be proved by humans alone. This is because if the computer which is being used can be thoroughly checked to make sure that it does not have any errors, then there is no difference to whether a human solves the cases in a proof by exhaustion than there is if a computer does it.

Case Studies

Four Colour Theorem

An example of a computer-assisted proof is the Four Colour Theorem. Some people regard this as the first theorem to be proved by a computer, as in 1976 Kenneth Appel and Wolfgang Haken of the University of Illinois managed to reduce the problem down so that it could be proved on a computer. I will look at what this theorem is, how it was originally proved by Appel and Haken, and how since then, others have found more efficient and more reliable ways of proving the theorem.

The Four Colour Theorem was originally proposed in 1852 by South African mathematician Francis Guthrie [10]. He mentioned it to his brother, Frederick Guthrie, as he said 'the greatest number of colours to be used in colouring a map so as to avoid identity of colour in lineally contiguous districts is four' [11]. Frederick was studying at University College London, and so asked his brother is he could show this proof to one of his professors, Augustus De Morgan. After De Morgan saw Francis' 'proof', he confirmed that is was new to him, and when showing it to subsequent students referenced his former student Francis Guthrie as the person with the original idea.

The nature to Guthrie's proof is unknown, however it is most likely incorrect because currently the theorem's proof requires a computer, and in 1852 this would not have been possible. Despite this, the theorem's origin can still be traced back to then, as De Morgan describes the problem in a letter to Sir William Rowan Hamilton dated 23rd October 1852. He stated in this letter:

> *A student of mine asked me to day to give him a reason for a fact which I did not know was a fact – and do not yet. He says that if a figure can be any how divided and the compartments differently coloured so that figures with any portion of common boundary line are differently coloured – four colours may be wanted, but no more.* [11]

This is where the Four Colour Theorem was born, and for the next century, people would still be trying to find a proof for this.

A simplified version of the Four Colour Theorem is:

Can every map be coloured with at most four colours in such a say that neighbouring countries are coloured differently? [11]

A **map** consists of many **regions**. The boundary of these regions are made of **boundary lines**. These boundary lines meet at places called **meeting points**. For countries to be neighbouring, they need to share a boundary line. If countries meet at a point only, then they are not classed as neighbouring. [11]

After a century's worth of mathematicians working on the problem, there was lots of progress made, however, nobody found a formal proof for the theorem. One of the methods attempted included proof by induction, mentioned earlier, however, there were counter-examples to the $(n+1)^{th}$ term. After this, most mathematicians decided the most logical way to prove the theorem was to find a set of reducible configurations so that these could be proved.

In July 1976, 124 years after the problem was first posed, Appel and Haken published the paper showing how they used a computer to prove the 1482 reducible configurations which allowed them to prove the four colour theorem. Later, they managed to reduce this further to 1405 combinations, however, reducing the number of combinations was not their main priority because it if the smaller number of priorities all took a longer time individually, then the overall processing time would be

increased. This is why Appel said *"If one configuration replaces twelve, but one configuration takes two hours and the twelve take five minutes, that doesn't make sense"*. [11]

The reactions to publication of the paper was mixed; some people greeted it with scepticism, saying that there had been many other 'proofs' in the last century which turned out to be flawed. Others said that they were disappointed that a theorem which was over a century old had been proved by a method with no mathematical deduction, and some people even went as far as to say that this meant that it could not be classed as a proof.

Donald Albers, who reported on the joint American Mathematical Society and Mathematical Association of America summer meeting at the University of Toronto in August 1976, commented on Haken's presentation of his solution to the theorem. He concluded that many of the mathematicians who attended the meeting did not want to accept the proof, and were cautious that there may be an error due to the length of the proof and the amount of processing time that was required. However, some agreed that the proof was valid, and therefore this caused the community to become divided on what the definition of a proof was. This was one of the first instances of this question appearing, and the controversy over computer-assisted proofs has become more widespread ever since. [11]

Kepler Conjecture

Some mathematicians argue that because computational proofs are often extremely long, that this means people will just assume that they are correct, because people are less willing to look over them and check that they are correct. An example of this is the Kepler Conjecture, which is described below.

The Kepler Conjecture is a mathematical conjecture about packing spheres in three-dimensional Euclidian space, originally suggested by Johannes Kepler in 1611. According to New Scientist, it can be simplified to: "What is the best way to stack a collection of spherical objects?" [12].

In 1998, Thomas Hales, from the university of Pittsburgh, presented a proof to Kepler's original theory, that the most efficient way to stack spheres was in a pyramid arrangement. The proof worked because although there were infinite possibilities to arrange the spheres, they all followed one of thousands of themes, and therefore Hales created a linear programming problem, and using around 100,000 linear programming problems, it allowed him to create a problem which could be solved with the help of a computer.

The proof which Hales produced was over 300 pages long, and therefore some people did not accept that it was a formal proof, because it took 12 people around 4 years to check for errors, and even after this, the reviewers were only 99% certain that the proof was correct. [13]

Even though the proof was in theory correct, the length caused many people to doubt whether it was completely correct, because the extended length increased the possibility of an error. This caused Hales to start the Flyspeck project in 2003, with which he aimed to verify his proof so that it was not questioned by the mathematical community. Using Isabelle and HOL Light, which are formal proof software assistants, Hales and his team managed to prove that their original proof had been verified by these programs, and so it was in fact correct. This revelation came on Sunday 10th August 2014, when Hales and the rest of the Flyspeck team said "This technology cuts the mathematical referees out of the verification process. Their opinion about the correctness of the proof no longer matters.". [12] This shows that in Hales' opinion, the proof is 100% certain, because programs which are designed to check proofs, and are carefully examined to make sure that they themselves do not contain errors, had verified his proof, and despite it not being able to be proved by a person, it is still valid.

Why is there controversy over computer-assisted proofs?

There are three main reasons that some mathematicians have concerns over computer assisted proofs. These are: the lack of logical deduction used to get to a solution, the limits of human verification on proofs due to length and complexity of proofs, and errors in computer hardware and software which may cause proofs to be invalid.

Lack of logical deduction

Some people are not satisfied with computer-assisted proofs because they do not always involve logical deduction. This is because computers are mainly used with proofs by exhaustion, which, as stated previously, is where a theory is split up into a finite number of cases, and then all of these cases are individually proved. Some mathematicians also say that computer-assisted proofs lack mathematical elegance. They say that it turns mathematics into a quasi-empirical science, which is where mathematicians focus more on getting to the solution of a problem, or finding a proof, than on the mathematical methods and reasoning used to get there [14]. This means that while they still may agree that the proof is in fact a proof, they won't regard it as such because it does not have a mathematical foundation, and this means that there is not necessarily an explanation as to why it is true just that the evidence shows it is.

Limits of human verification

Computer-assisted proofs are also not always verifiable by humans because humans cannot compute thousands of calculations per second, and so to verify some of the proofs by hand would take many years, which is just not feasible. This means that peer review, where someone with a knowledge of the field will review the proof to make sure that it is accurate, valid and of high quality, is difficult to implement because without reviewing every case, the reviewer cannot be sure that there are no errors. [15] Also, errors can be so subtle that they can be incredibly hard to find by peer reviewers, and therefore if even if every element of the proof is checked, if just one small detail is missed, the proof may be incorrectly verified.

Another reason that there is controversy is that humans cannot always understand computer assisted proofs, because many times too many calculations need to happen which a person would not be able to do because it is not possible to remember this large an amount of information. [16] Some people suggest that getting an answer to a proof is not the part which is ultimately desired, it is understanding how that answer is reached which is what allows a person to feel as if they have proved a theorem. An example of this is shown in 'On Proof and Progress in Mathematics', where the author, William Thurston, said;

> *On a more everyday level, it is common for people first starting to grapple with computers to make large-scale computations of things they might have done on a smaller scale by hand. They might print out a table of the first 10,000 primes, only to find that their printout isn't something they really wanted after all. They discover by this kind of experience that what they really want is usually not some collection of "answers"—what they want is understanding.* [16]

This quote suggests that Thurston accepts computer-assisted proofs, however does not believe that they allow humans to understand how the proof is valid. He suggests that getting the 'answer' is not what the aim of proving a theorem is; the aim is to understand how the theorem is proved, and therefore be able to explain it without the assistance of a computer.

Errors in computer hardware and software

Alongside the other reasons for controversy, computer programs are prone to errors, which means that if a program claims to have proved something then the output of the program may look like it agrees, however a logical error in the program may mean that it has, for example, not proved every case, and so the proof may not be valid. An example of when this type of error has been found was in 1994 when the Pentium bug was discovered. It was a bug which occurred in a group of Intel processors named Pentium in which the processors gave the wrong decimal results when numbers were divided. The problem occurred because the processor had a 'division table', which was stored with values that were used when the processor was dividing numbers, and five of the thousand entries in this division table had been incorrectly entered when the processor was being made, and so caused results to be output incorrectly. A well-known case was discovered by Tim Coe, where if you entered 4195835÷3145727, the result given would be 1.33374 (6sf). However, the actual answer is 1.33382 (6sf), meaning there was a relative error of 0.006%. This would have been a problem if a mathematical problem had been proved using a computer with this processor because even though it only happened in a small amount of cases, some computer-assisted proofs need thousands of cases to be proved, and if even one of these had been incorrectly 'proved', then the whole proof would be flawed [17]. Bugs are usually first searched for when a result is returned and is not what is expected. This is followed by a long process of debugging the program for any instructions which cannot be justified, and then from this the proof has to be re-calculated so that it does not rely on the old faulty software. [8]

As shown above, sometimes the hardware may contain problems, however sometimes the software may contain bugs, which could cause a computer-assisted proof to be falsely proved. Computer programs are created by humans, and therefore are prone to errors. This means that unless every part of a program is checked, there is always the chance of an error. However, to check every part of a program, you would need to do all of the calculations yourself, which defeats the purpose of having the computer do them in the first place. This means that you have to rely on the program not to have any errors, but some mathematicians are not happy with doing this.

There are some examples of why they wouldn't trust computer software, for example some computers such as the Univac 1108, which was used until 1980, calculated 16777216-16777215 = 2, where $16777216 = 2^{24}$. This was because the accumulator in the processor only had a length of 24 bits, and therefore using binary, the larger number could not be represented using one string of binary, and so when the combination of more than one is used, an error in the program meant that the subtraction was off by 1. [18] This is caused by the software not being able to combine the longer and shorter binary strings correctly, as the software was programmed wrong when it was created. This problem could have potentially invalidated many proofs if they used this computer, and that is a reason why people do not trust any computers, as errors like this may not have been discovered when the proof was computed, and so the proof may be invalid.

Another example of why computer programs may not be reliable is due to conversions between data types. If a program is using multiple data types, then some of them may be able to store more information than others, because they have a longer length. This means that is data is converted from one data type to another, then this may cause data to be lost, or an error to occur. This happened in 1996 with the Ariane 5 rocket causing it to crash, because a value relating to the alignment of the rocket was not converted properly. The value was converted from 64-bit floating to point to 16-bit signed integer value, but on the day, the horizontal bias, which relates to the internal alignment of the rocket, was larger than anticipated, and so when it was converted to a 16-bit signed integer value, an operand error occurred. This therefore, per the failure report, caused the rocket to not be aligned

properly, and to therefore veer off course and crash about 40 seconds into its flight. [19] Although the effects of this example may seem much more significant than if the same had happened with regards to a computer assisted proofs, it shows that even after the extremely thorough checking this software had to go through to be allowed on a rocket, there was still an error which ended up to be extremely dangerous and expensive. This demonstrates that errors in computer software can sometimes be so difficult to find, and therefore are a reason why there is controversy over computer assisted proofs.

To conclude this section, many mathematicians have problems with computer assisted proofs because they do not believe that they can be checked thoroughly, and still believe that there is the possibility of errors. The main reason for this are errors in software and hardware, as something which may seem trivial can jeopardise a whole proof.

How can problems with computer assisted proofs be overcome?

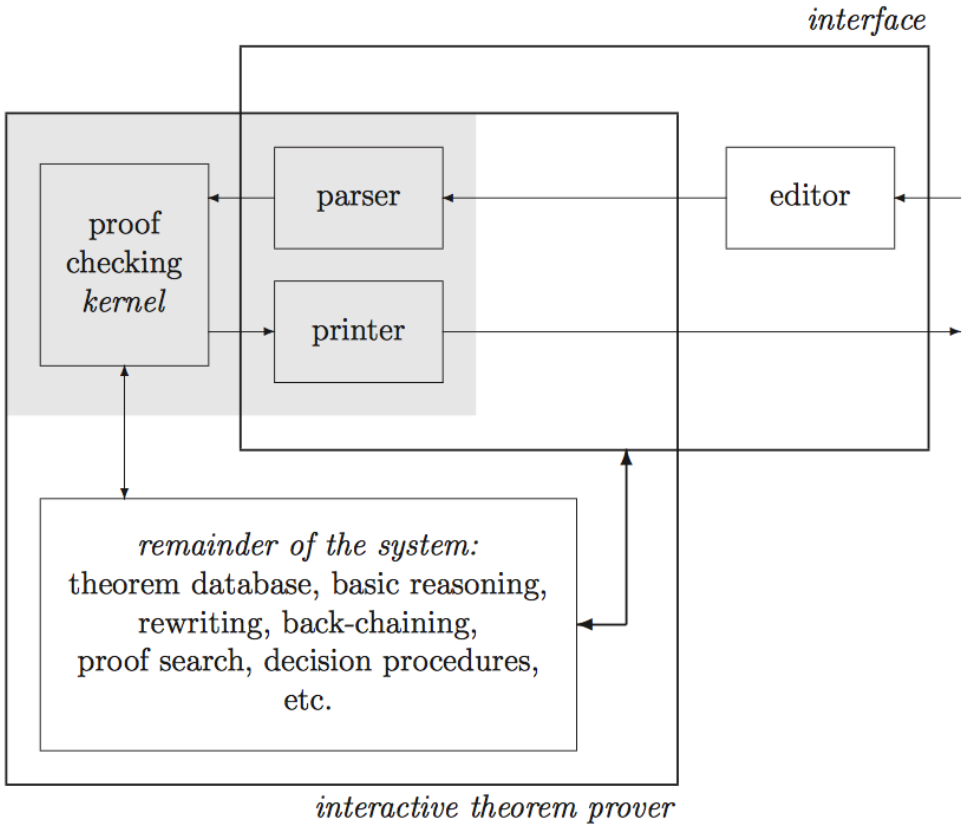
As has been seen, there are many reasons why mathematicians are wary about using computer assisted proofs. However, there are many ways in which people are trying to make them more reliable and easier to be verified.

Proof Assistants

The main way computer-assisted proofs are verified is by the use of proof assistants. They require the user to input the conjecture which needs to be formalised, and from this they use high order logic (HOL) [20] and libraries which contain pre-proved theorems to verify if the proof is correct. [21] Some examples of proof assistants include; Isabelle and HOL Light, which were used by Hales to verify the proof of the Kepler Conjecture, as well as Coq, which was used to verify the proof of the Four Colour Theorem in 2005 by Georges Gonthier. [22] [23]

The problem with proof assistants is that they need to be tested themselves to make sure that they do not contain errors, as this would mean any conjecture that they had tested would not be proved. One way of doing this is by using the de Brujin Criterion, which states that only a small part of the program, called the **proof checking kernel**, is used to check that the logic and mathematics behind the proof is correct. This means that this is the only part of the code of the proof assistant which needs to be checked, as this is the only part which will check the conjecture which is being input. [24]

As shown in the image below, the proof checking kernel is separate from the majority of the system, meaning that any errors in the main program do not affect it.



[24]

One way of checking the kernel is by using the Pollack Inconsistency. This is done by checking that when the assistant outputs something to the user, by **printing** it, it can understand what it has output,

by **parsing** the output back into the assistant. This is then compared to the original output, and if they are the same, then the assistant is Pollack-consistent.

An example of how this done is as follows:

$$parse(print(t)) = t \quad [24]$$

If this returns true, then the proof assistant is Pollack-consistent. This checks that the kernel works correctly because if it can understand and interpret an input to be the same thing that it output, then the program must be outputting the correct information.

Peer Review

Peer review is usually thought of as humans checking a proof to make sure that it is legitimate and does not contain any errors. However, for computer assisted proofs, this may not be able to be the case, because as discussed earlier, there are too many cases which can be of too high complexity for humans to thoroughly verify them all individually. This makes sense, because if the proof was able to be solved by humans in the first case, then the peer review could also be done by humans. This means that logically, the only way to verify a computer assisted proof is with a computer, as that is the only thing that can check every case. [15]

Some mathematicians may argue that this poses the same problems, however, if computers using different software and different hardware were to check the same proof, then the chances that the same error occurs in both is minimal. This means that in a sense, a peer review is possible, because the proof is being reviewed by another system. If this is repeated with multiple systems, then the chance of there being any error basically disappears.

However, a benefit of using humans as peer reviews is that they may spot a more efficient method of solving a problem, or they may be able to give other subtle improvements which will improve the program, whereas using a computer would not allow this to happen. This is one part of a peer review which would therefore still need to be done by humans, however, the checking of cases could be completed by computers, as this does not require any imagination. [25]

Reliable hardware and software

Hardware problems in mainstream computers are becoming increasingly less common. This is because of the greater number of users using mainstream computers compared to the 20^{th} century, when theorems like the Four Colour Theorem were proved, and this means errors are much more likely to be picked up on. [26]

The same is to be said about software, as the increase in users means that they are more likely to pick up on any subtle errors in the operating system because there are so many people who are likely to find it. However, there are still more ways in which software errors can be prevented. Using lower level languages is one of these. Low-level languages are closer to machine code, which means each instruction is nearly equivalent to one instruction in machine code. [27] This means that there is less chance of an error occurring when the code is being interpreted, because the programmer is telling the computer exactly how it wants it to act, as it is possible to even decide which memory location are to be used. This means that the possibility that there is an error in the software is greatly reduced because the program can be seen to compute exactly as intended.

Conclusion: The future of proofs

In the future, computer assisted proof are going to become much more important in the mathematical community. This is because of the ease with which they can be proved, and the need for some of these proofs by computer scientists. Mathematicians like elegant proofs because they like to be able to see why a problem can be proved, but if a proof is needed by a computer scientist to create a more powerful system, then they may not care how it is proved, only that the answer is true, and that they can utilise this. [28] Some mathematicians even say that it is a necessity, as some theorems are too complex to be proved by humans, and therefore it is just evolution. [29]

Additionally, proof checking processes are only going to become increasingly more thorough, because as more proofs are found, the amount of mathematical knowledge known by mankind will increase, and there will be too much for one person to know everything. This means that computers will need to be used in order to store everything we know about mathematics, so that when a new conjecture needs to be proved, there is a central place where all knowledge can be found. [30]

Some people suggest that automated theorem provers are the future of mathematical proofs. These are programs which are given mathematical rules and some axioms, and from these are able to generate proofs of mathematical results. [28] These have a much wider variety of uses, for example they can be used to mathematically model situations so that you do not need to test every possibility. This means that they can save a lot of time for businesses who need to test something but do not have the time and money to do so, as they can mathematically verify that a system will always perform they way it is supposed to. One disadvantage to these is that they still require lots of user input, as they need to know a great deal of axioms in order to create a proof, but once these are entered, then they can be extremely efficient. Also, if a centralised database of all known mathematical knowledge was created, then this could be shared and edited by people worldwide, which would allow new proofs to be generated much more easily. [28]

However, one thing which is certain is that problems which exist today relating to computer-assisted proofs will be become less important as time progresses. This is because the advancement of software and hardware will mean that proof checking can become much more accurate, reliable and efficient. However, there is always the chance of a small error, and this therefore means it is unlikely that it can ever be guaranteed that every proof is completely valid, but the likeliness of errors will become so miniscule that eventually computer assisted proofs will be trusted.

Works Cited

[1] L. Rogers, "The Four Colour Theorem," 2011 February 2011. [Online]. Available: http://nrich.maths.org/6291. [Accessed 28 August 2016].

[2] J. Brikaite, N. Megens, O. Somova, S. Kerem Ünal and T. Sørensen Felby, "Computer Assisted Proofs and Their Effects on Pure Mathematics: A case study of the four colour theorem," Roskilde University.

[3] G. Mints, "Peano Axioms," [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Peano_axioms&oldid=36502. [Accessed 15 August 2016].

[4] S. Jameson, Decision Mathematics 1, Pearson Education Limited, 2010, p. 33.

[5] R. Pelayo, *The Peano Axioms,* University of Hawaii, p. 3.

[6] "Proof by contradiction," [Online]. Available: https://en.wikipedia.org/wiki/Proof_by_contradiction#No_least_positive_rational_number. [Accessed 15 August 2016].

[7] "Undergraduate Mathematics/Proof by exhaustion," 28 November 2014. [Online]. Available: https://en.wikibooks.org/wiki/Undergraduate_Mathematics/Proof_by_exhaustion. [Accessed 10 October 2016].

[8] A. Neumaier, *Computer-assisted proofs,* Wien Univeristy, 2006, pp. 2-5.

[9] M. McQuain, "Virginia Tech Math 2534 Lecture: Proof," [Online]. Available: http://www.math.vt.edu/people/mcquain/answers_proofs.pdf. [Accessed 10 December 2016].

[10] W. Knight, "Computer generates verifiable mathematics proof," New Scientist, 19 April 2005. [Online]. Available: https://www.newscientist.com/article/dn7286-computer-generates-verifiable-mathematics-proof/. [Accessed 9 October 2016].

[11] R. Wilson, Four Colors Suffice, Oxford: Princeton University Press, 2014.

[12] J. Aron, "Proof confirmed of 400-year-old fruit-stacking problem," New Scientist, 12 August 2014. [Online]. Available: https://www.newscientist.com/article/dn26041-proof-confirmed-of-400-year-old-fruit-stacking-problem/. [Accessed 28 September 2016].

[13] R. Khamsi, "Mathematical proofs getting harder to verify," New Scientist, 19 February 2006. [Online]. Available: https://www.newscientist.com/article/dn8743-mathematical-proofs-getting-harder-to-verify/. [Accessed 28 September 2016].

[14] "Quasi-empiricism in mathematics," [Online]. Available: http://encyclopedia.kids.net.au/page/qu/Quasi-empiricism_in_mathematics. [Accessed 28 August 2016].

[15] A. Perry, "Evaluating Information Sources: What Is A Peer-Reviewed Article?," Lloyd Sealy Library, 14 September 2016. [Online]. Available: http://guides.lib.jjay.cuny.edu/c.php?g=288333&p=1922599. [Accessed 11 December 2016].

[16] W. P. Thurston, "On Proof and Progress in Mathematics," *Bulletin of the American Mathematical Society,* vol. 30, no. 2, pp. 161-177, 1 April 1994.

[17] M. Janeba, "The Pentium Problem," 20 April 2011. [Online]. Available: http://www.willamette.edu/~mjaneba/pentprob.html. [Accessed 02 September 2016].

[18] S. M. Rump, *Computer-assisted Proofs and Self-validating Methods,* Linköping University, 2005, pp. 195-240.

[19] P. J. L. Lions, *Ariane 5 Flight 501 Failure Report,* Paris: ESA and CNES independent Inquiry Board, 1996.

[20] V. Voevodsky, "Computer Proof Assistants - the future of mathematics," Institute for Advanced Study in Princeton, NJ, 27 August 2014. [Online]. Available: http://www.math.ias.edu/vladimir/files/2014_08_27_NUS.pdf. [Accessed 14 December 2016].

[21] J. P. Bridge, "Machine learning and automated theorem proving," University of Cambridge, Cambridge, 2010.

[22] P. Schnider, *An Introduction to Proof Assistants - Student Seminar in Combinatorics: Mathematical Software,* ETH Zurich, 2014.

[23] L. C. Paulson, *The Future of Formalised Mathematics,* Cambridge: University of Cambridge, 2016.

[24] F. Wiedijk, "Pollack-inconsistency," Electronic Notes in Theoretical Computer Science, Radboud University Nijmegen Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands, 2010.

[25] D. J. Benos, K. L. Kirk and J. E. Hall, "How to Review a Paper," *Advances in Physiology Education,* vol. 27, no. 2, pp. 47-52, 2003.

[26] J. P. Eckmann, H. Kock and P. Wittwer, "A computer-assisted proof for universality for area-preserving maps," *Memoirs of the American Mathematcial Society,* vol. 47, no. 289, January 1984.

[27] P. Heathcote and R. Heathcote, "OCR AS and A Level Computer Science," Dorset, PG Online, 2016, p. 44.

[28] M. Freiberger and R. Thomas, "The future of proof," 10 April 2015. [Online]. Available: https://plus.maths.org/content/future-proof. [Accessed 11 December 2016].

[29] M. Harris, "Mathematicians of the Future?," 23 March 2015. [Online]. Available: http://www.slate.com/articles/health_and_science/science/2015/03/computers_proving_mathematical_theorems_how_artificial_intelligence_could.html. [Accessed 14 December 2016].

[30] H. Geuvers, *Proof Assistants: history, ideas and future,* The Netherlands: Radboud University Nijmegen, 2009.