A Common Type of Rigorous Proof that Resists Hilbert's Programme

A Common Type of Rigorous Proof that Resists Hilbert's Programme *

Alan Bundy
University of Edinburgh
A.Bundy@ed.ac.uk

Mateja Jamnik
University of Cambridge
mateja.jamnik@cl.cam.ac.uk

October 15, 2018

Abstract

Following Hilbert, there seems to be a simple and clear definition of mathematical proof: it is a sequence of formulae each of which is either an axiom or follows from earlier formulae by a rule of inference. Automated theorem provers are based on this Hilbertian concept of proof, in which the formulae and rules of inference are represented in a formal logic. These logic-based proofs are typically an order of magnitude longer than the rigorous proofs produced by human mathematicians. There is a consensus, however, that rigorous proofs could, in principle, be unpacked into logical proofs, but this programme is rarely carried out because it would be tedious and uninformative. We argue that, for at least one class of rigorous proofs, which we will call *schematic proofs*, such a simple unpacking is not available. We will illustrate schematic proofs by analysing Cauchy's faulty proof of Euler's Theorem V-E+F = 2, as reported in [Lakatos, 1976] and giving further examples from [Nelsen, 1993]. We will then give a logic-based account of schematic proofs, distinguishing them from Hilbertian proofs, and showing why they are error prone.

1 Introduction

Hilbert's Programme is defined in [Zach, 2009, p9] as:

> "The main goal of Hilbert's program was to provide secure foundations for all mathematics. In particular this should include:
>
> **A formalization** of all mathematics; in other words all mathematical statements should be written in a precise formal language, and manipulated according to well defined rules.
>
> **Completeness:** a proof that all true mathematical statements can be proved in the formalism.
>
> **Consistency:** a proof that no contradiction can be obtained in the formalism of mathematics. This consistency proof should preferably use only "finitistic" reasoning about finite mathematical objects.
>
> **Conservation:** a proof that any result about "real objects" obtained using reasoning about "ideal objects" (such as uncountable sets) can be proved without using ideal objects.

*The research reported in this chapter is based on [Bundy *et al*, 2005, Jamnik & Bundy, 2005, Bundy, 2012]. It was supported by EPSRC grants GR/S01771, GR/S31099 and EP/N014758/1. Many thanks to Predrag Janičić and Alison Pease for drawing some of the images and to Andy Fugard for permission to use a diagram drawn by one of the participants in his study. Thanks to Gila Hanna and Andy Fugard for comments on an earlier version.

Decidability: there should be an algorithm for deciding the truth or falsity of any mathematical statement."

The problems with the goals of Completeness, Consistency and Decidability, that were revealed by Gödel's incompleteness theorems [Gödel, 1931], have been well documented, but in this chapter, we are focused on the goal of Formalisation.

Formalisation has become important in Computer Science as the basis of *automated theorem proving*, which has important practical applications, for instance, in the verification of the correctness of computer software and hardware [Robinson & Voronkov, 2001]. Using one of many formal logics, the axioms and rules of inference of a formal mathematical theory are represented as data-structures in an automated theorem prover. Programs are then written to construct formal proofs by deriving new formulae from old by applying these rules to them. In some provers, humans can interact and guide the development of the proof; in others, the development is completely automatic.

[Zach, 2009, p10] summarises the post-Gödel, modified form of the Formalisation goal of Hilbert's Programme as:

> "Although it is not possible to formalize all mathematics, it is possible to formalize essentially all the mathematics that anyone uses. In particular Zermelo-Fraenkel set theory, combined with first-order logic, gives a satisfactory and generally accepted formalism for essentially all current mathematics."
>
> *Our claim in this chapter is that there is a form of proof, which we will call* schematic proof*, that resists even this, generally accepted, Formalisation goal of the Hilbert Programme.*

As evidence to support our claim we will analyse Cauchy's Proof of Euler's Theorem as discussed, for instance, in [Lakatos, 1976]. Paraphrasing [Hilbert, 1930], we will take Hilbert's view of proof to be:

> *A proof is a sequence of formulae each of which is either an axiom or follows from earlier formulae by a rule of inference.*

There is a widespread assumption in the Mathematics community that the rigorous[1] proofs one finds in Mathematics papers and textbooks are an abstraction of this ideal. Instead of stating every formula in the sequence, some of them are omitted, the assumption being that a typical reader will be able to skip several formal steps at a time. De Bruijn's factor even gives a numerical value of approximately 4 to the typical ratio of the logical to the rigorous proof steps [Bruijn, 1980]. This was calculated by surveying and analysing a number of proofs produced in different automated theorem provers. Carrying out this formalisation part of Hilbert's Programme then consists of filling in the gaps: agreeing on an axiomatic system, such as Zermelo-Fraenkel set theory, deriving all the elementary lemmas that might be assumed in a proof, and interleaving the rigorous proof with the missing steps.

We will show that schematic proofs contradict this assumption and that a much more radical programme is required to formalise them — provided they are not faulty.

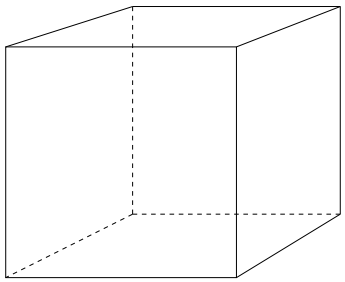
Schematic proofs are of interest to Mathematics educators for both, positive and negative reasons.

- On the positive side, they are more intuitive, natural and accessible, and can engender a deeper understanding of why a theorem holds than a Hilbertian proof can.
- On the negative side, they are error prone. A proof of their correctness for all cases is required, but is often omitted. If omitted, counter-examples can go undetected.

[1]The word 'informal' is sometimes used instead of 'rigorous'. We avoid 'informal', as we will claim, in §4, that schematic proofs can also be formalised.

2 Cauchy's 'Proof' of Euler's Theorem

Polyhedra are the 3D version of polygons: objects whose faces are polygons. Examples include the five regular Platonic polyhedra: tetrahedron, cube, octahedron, dodecahedron and icosahedron, as well as many other semi-regular and regular objects. Euler's 'Theorem'[2] states that in any polyhedron, the following holds, $V - E + F = 2$, where V is the number of vertices, E the number of edges and F the number of faces. Figure 1 illustrates this 'theorem' in the case of the cube.



In a cube there are 8 vertices, 12 edges and 6 faces. So, $V - E + F = 8 - 12 + 6 = 2$.

Figure 1: Euler's Theorem in the case of the cube.

In [Lakatos, 1976], Imre Lakatos uses a rational reconstruction of the history of Euler's 'Theorem' as a vehicle to argue that the methodology of mathematics had evolved and become more sophisticated. He describes a fictional classroom in which a teacher leads a (very bright!) class through this history. The teacher starts by presenting Cauchy's 'proof'[3] of the theorem. The students then confront it with various counter-examples and suggest ways to cope with them.

Cauchy's 'proof' is couched as the following 'thought experiment', quoted from [Lakatos, 1976, p7-8]:

> **"Step 1:** Let us imagine the polyhedron to be hollow, with a surface made of thin rubber. If we cut out one of the faces, we can stretch the remaining surface flat on the blackboard, without tearing it. The faces and edges will be deformed, the edges may become curved, but V and E will not alter, so that if and only if $V - E + F = 2$ for the original polyhedron, $V - E + F = 1$ for this flat network — remember we have removed one face (Figure 2 top left shows the flat network for the case of a cube)."
>
> **"Step 2:** Now we triangulate our map — it does indeed look like a geographical map. We draw (possibly curvilinear) diagonals in those (possibly curvilinear) polygons which are not already (possibly curvilinear) triangles. By drawing each diagonal we increase both, E and F by one, so that the total $V - E + F$ will not be altered (Figure 2 top right)."
>
> **"Step 3:** From the triangulated network we now remove the triangles one by one. To remove a triangle we either remove an edge — upon which one face and one edge disappear (Figure 2 bottom left), or we remove two edges and a vertex — upon which one face, two edges and one vertex disappear (Figure 2 bottom right). Thus if $V - E + F = 1$ before a triangle is removed, it remains so after the triangle is removed. At the end of this procedure we get a single triangle. For this $V - E + F = 1$ holds true. Thus we have proved our conjecture."

[2]The scare quotes indicate that there are issues with proving Euler's 'Theorem'. These issues are the subject of this section. It would have been more accurate to call it 'Euler's Conjecture'.

[3]Again, the scare quotes indicate that there are issues with this alleged proof.

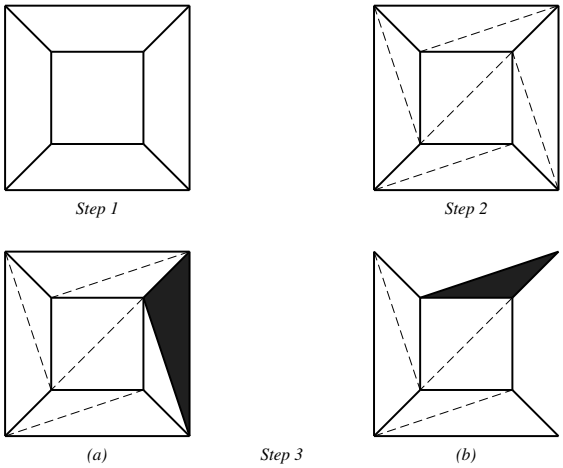
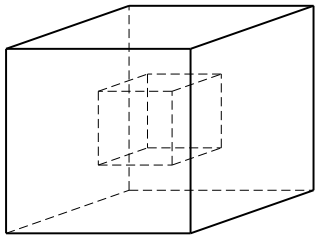


Figure 2: Cauchy's 'proof' applied to the cube.

The bulk of [Lakatos, 1976] consists of the presentation of various counter-examples to Euler's 'Theorem', followed by discussions of how these can be dealt with either by ruling them out as polyhedra or adapting the proof and/or theorem. Figures 3 and 4 give four such counter-examples.



The hollow cube is a cube with a cubical hole in the middle. The values of V, E and F are all doubled. So $V - E + F = 16 - 24 + 12 = 4$.

Figure 3: The hollow cube: a counter-example to Euler's 'Theorem'.

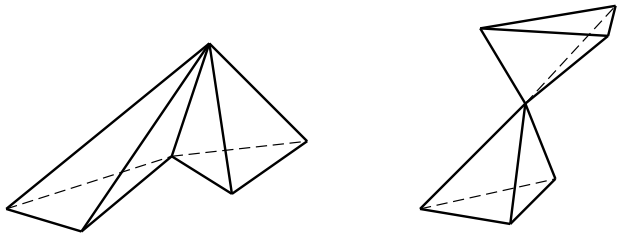
3 Schematic Proof

Cauchy's 'proof' in §2, quoted from [Lakatos, 1976, p7-8], has some unusual properties. It is an example of what we will call *schematic proof*. The hypothesis of this chapter is that:

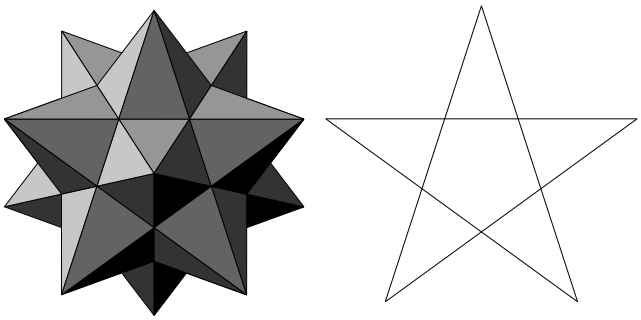
Schematic proofs resist Hilbert's Programme to unpack rigorous proofs into logical ones.

These unusual properties are:

1. The 'proof' is not Hilbertian, that is, it is not a sequence of formulae each of which is either an axiom or follows from earlier formulae by a rule of inference. Rather, it is a *procedure*,



$V - E + F = 5 - 10 + 8 = 3$ $\qquad$ $V - E + F = 7 - 12 + 8 = 3$



$V - E + F = 12 - 30 + 12 = -6$

Figure 4: Three more counter-examples to Euler's Theorem.

which is to be applied to any polyhedron, during which a property ($V - E + F = 1$) remains invariant.

2. This procedure will produce a different sequence of steps for each polyhedron, for example, when applied to a tetrahedron, the 'map' produced by step 1 will consist of three triangles, in contrast to the five quadrilaterals produced for a cube. Step 2 will then not be necessary and there will be fewer applications of step 3. Indeed, we can see the procedure as generating a *different* proof for each polyhedron.

3. The 'proof' is error prone, that is, some steps either cannot be applied to some polyhedra, or they produce unanticipated effects. There is no attempt to prove that the procedure will apply to all polyhedra, or that it will output a proof of Euler's 'Theorem' for each polyhedron.

4. The 'proof' is carried out in the absence of any definition of polyhedra. In a Hilbertian proof, definitions must precede proofs. The absence of a definition quickly becomes apparent. In the argument as to whether the hollow cube (Figure 3) is a polyhedron, two rival definitions are proffered [Lakatos, 1976, p14]. In one, a polyhedron is a solid, and the hollow cube is a polyhedron. In the other, a polyhedron is a collection of surfaces, and the hollow cube falls apart into two nested polyhedra. Even these definitions are not sufficient, as they still contain undefined terms, such as 'surface'.

5. The 'proof' was accepted for some time before counter-examples were discovered. Even then, according to Lakatos' account, there was considerable confusion about exactly what was wrong with the 'proof' and how it should be fixed. How is this possible? Hilbertian

proofs can be mechanically checked and any errors will quickly show up, for instance, a formula that does not follow by a rule of inference from earlier formulae. Even before the advent of computers, humans could be employed to check even a long proof.

The reason we call this 'proof' *schematic* is because of point 2 above. To present it as a single proof we need to abstract from the varying number of steps it produces. For this we need abstraction devices, such as ellipsis (cf Figure 6 in §5.1 and Figure 10 in §5.2 below).

4 Formalisation of Schematic Proofs

We have argued that schematic proofs are not Hilbertian and that this presents an obstacle to their formalisation within the usual assumptions of Hilbert's Programme, that is, the 'filling in the gaps' process outlined in §1. However, we will argue that it *is* possible to give a logical account of schematic proofs, albeit a more complex one. This logical account will use the *constructive ω-rule* [Shoenfield, 1959].

The ω-rule for the natural numbers 0, 1, 2, ... is:

$$\frac{\phi(0), \quad \phi(1), \quad \phi(2), \quad \ldots}{\forall x.\phi(x)}$$

that is, we can infer that $\phi(x)$ for all natural numbers x provided we can prove $\phi(n)$ for $n = 0, 1, 2, \ldots$. The ω-rule is clearly not a very practical rule of inference, since it requires the proof of an infinite number of premises to prove its conclusion. A Hilbertian proof using it would consist of an infinite sequence of formulae. Its use is usually confined to theoretical discussions, for instance, in Gödel's incompleteness theorems [Gödel, 1931].

The constructive ω-rule is a refinement of the ω-rule that *can* be used in practical proofs. It has the additional requirement that the $\phi(n)$ premises be proved in a *uniform* way, that is, that there exists an effective procedure, $proof_\phi$, which takes a natural number n as input and returns a proof of $\phi(n)$ as output. We will write this as $proof_\phi(n) \vdash \phi(n)$. The procedure $proof_\phi$ formalises our notion of schematic proof. Applied to the domain of polyhedra, rather than natural numbers, it could be used to formalise Cauchy's 'proof' of Euler's 'theorem' given in §2. In particular, the procedure, which given a polyhedron, proves Euler's 'theorem' for it, can be interpreted as the procedure $proof_\phi$, but for polyhedra rather than natural numbers.

The constructive ω-rule has been automated to generate schematic proofs in the domain of natural numbers. Siani Baker's PhD thesis [Baker, 1993] automated the schematic proofs of theorems of Peano Arithmetic. Mateja Jamnik's PhD thesis [Jamnik, 2001] automated the proofs in [Nelsen, 1993] (see §5.1). These automated provers start with a theorem to prove and one or two instances of the proof of this theorem for particular numbers: essentially just calculations of concrete instances of the theorems. These proof instances are then generalised to a procedure $proof_\phi$.

Both of these automated provers then do something omitted in Cauchy's 'proof': the meta-proof that the procedure generated a proof for each $\phi(n)$. This is done by mathematical induction[4] on the natural numbers, that is, the proof that:

$$proof_\phi(0) \vdash \phi(0)$$
$$proof_\phi(n) \vdash \phi(n) \implies proof_\phi(n+1) \vdash \phi(n+1)$$

Not only was a comparable meta-proof omitted by Cauchy, it's hard to see how he could have provided it. The natural numbers have a recursive structure, which lends itself to inductive proof. The set of polyhedra has no known recursive structure, so a similar proof plan is not available. As we have seen in §2, the absence of this meta-proof is a fatal flaw. It allows the possibility that there is a polyhedron *poly* say, for which:

$$proof_\phi(poly) \nvdash \phi(poly)$$

[4]Not to be confused with, the regrettably similarly named, *philosophical induction*, which is a rule of conjecture, not deduction, or with the *omega*-rule. See Figure 5.

Mathematical Induction	Philosophical Induction	ω-rule
$\dfrac{\phi(0), \quad \forall n\ \phi(n) \implies \phi(n+1)}{\forall n.\ \phi(n)}$	$\dfrac{\phi(n_1), \quad \phi(n_2) \quad \ldots \quad \phi(n_m)}{\forall n.\ \phi(n)}$	$\dfrac{\phi(0), \quad \phi(1) \quad \phi(2) \quad \ldots}{\forall n.\ \phi(n)}$

Figure 5: Difference between mathematical induction, philosophical induction and the ω-rule.

where ϕ is Euler's 'Theorem'. As we saw in §2, there are many such polyhedra. This illustrates the negative side of schematic proof, which we mentioned at the end of §1, that educators need to be aware of.

The situation with the extraction of schematic proofs is somewhat reminiscent of the way that computer programs are typically developed. Programmers start with a few development examples of the input/output relationship that the program is intended to exhibit, and the series of steps it should go through in these concrete instances. They then generalise from the steps for concrete instances into a general procedure. The generalised procedure is then tested on further examples. This testing can only cover a finite subset of what is usually a potentially infinite set. So, there is no guarantee that the procedure will not subsequently fail, especially on a kind of example that the programmers did not think of. Ideally, the programmers would verify the correctness of the program via the kind of meta-proof discussed above. In Computing, this is called a *verification proof*. Unfortunately, verification proofs require a high skill level and are time consuming. As a result, they are rarely undertaken, except in highly safety or security critical applications. When they are undertaken, then they, like Cauchy's 'proof', often consist of showing that some invariant is preserved at each step of the procedure. This invariant might, for instance, be a safety or security property, or it may assert the absence of deadlock or of other kinds of error.

5 Schematic Proofs are Common

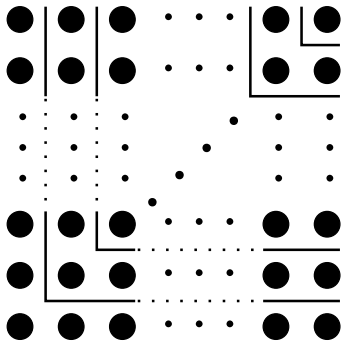
Our claim in §1, that schematic proofs resist Hilbert's Programme, would have little force if the set of schematic proofs was very small, for example, consisting only of Cauchy's faulty proof. In this section, we argue that they are commonplace. We will present two forms of evidence:

- Nelsen's "Proofs without words" [Nelsen, 1993]; and
- An analysis of human reasoning about recursive programs [Jamnik & Bundy, 2005, Fugard, 2005].

5.1 Nelsen's Proofs Without Words

In [Nelsen, 1993], Roger Nelsen shows how the truth of a mathematical theorem can often be convincingly demonstrated with a well chosen diagram. Some of these theorems are about the natural numbers, one of which is illustrated in Figure 6. It is these proofs that Mateja Jamnik automated in [Jamnik, 2001] using the constructive ω-rule. Her Diamond program used the procedure outline in §4. That is, a proof-generating procedure was abstracted from the proofs of a couple of special cases, say n=3 and n=4, then a meta-proof showed that this procedure would generate a correct proof for each natural number n. The success of her project shows that some of Nelsen's proofs without words can be characterised as schematic. Further examples of such schematic proofs are given in Figure 7.

Nelsen's diagrammatic proofs illustrate the positive side of schematic proofs that we argued at the end of §1. Many educators have long found them an intuitive and accessible alternative to Hilbertian proofs.



The diagram gives a proof of the theorem $n^2 = 1+3+\cdots+(2n-1)$. The diagram can be viewed as describing both, the left and right hand sides of the equation. The whole square represents n^2. Each of the L-shapes represents one of the odd numbers summed on the right-hand side.

Figure 6: A proof without words.

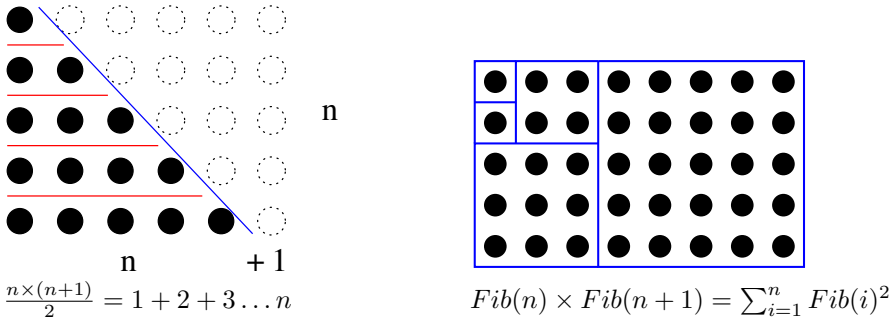


Figure 7: Further examples of proofs without words.

5.2 Human Use of Schematic Proof

In his MSc project [Fugard, 2005], Andy Fugard reported on some experiments to observe human reasoning about recursive programs. Participants were presented with conjectures, some of which were true and some false. For instance, he asked participants about the (true) *rotate-length conjecture*:

$$rot(len(l), l) = l \tag{1}$$

where l is a list, such as $[a, b, c]$, $len(l)$ is the length of the list, say 3, and $rot(n, l)$ rotates n times the list l, where one rotation is to move the first element to the end of the list, for example $[a, b, c]$ to $[b, c, a]$. The formal recursive definition of rot is:

$$\begin{aligned} rot(0, l) &= l \\ rot(n+1, [\,]) &= [\,] \\ rot(n+1, [h|t]) &= rot(n, app(t, [h])) \end{aligned}$$

where app appends one list to the end of another and $[h|t]$ is a list where h is the first element and t is the rest of the list. In Fugard's experiments the function names were replaced with arbitrary

letters so that participants were not influenced by the meanings implied by names such as *rot*, *len* and *app*.

The Hilbertian proofs of theorems about recursive programs use mathematical induction. Fugard did not observe any of his participants attempting an inductive proof. Had they done so, they would have encountered a difficulty. The rotate-length conjecture (1) is surprisingly tricky to prove. It requires an intermediate lemma. For instance, the following generalised rotate-length conjecture works as an intermediate lemma:

$$rot(len(l), app(l, k)) = app(k, l) \tag{2}$$

where k is also a list. Lemma (2) is (surprisingly) easier to prove than (1) and, in fact, (1) is an immediate corollary from (2) by setting k to be the empty list. Fugard's participants, however, found it harder to see that (2) was true and harder to prove than (1).

The participants' reasoning often used diagrams consisting of a succession of lists being rotated, in which the last list was the same as the first. These lists often started as concrete ones for particular lists, but were then converted into schematic ones via the use of ellipsis, or similar abstraction devices. Diagrams often showed signs of alteration, where a concrete list was converted into a more schematic one. Figure 8 shows an example. These diagrams approximated the more formal accounts of a concrete and schematic proof given in Figures 9 and 10, respectively.

A concrete diagram was first drawn on the left and then extended on the right using ellipsis and annotation to indicate generality.

Figure 8: Example diagram drawn by one of Fugard's participants for rotate-length conjecture.

$$\begin{aligned} rot(len([a_1, a_2, a_3]), [a_1, a_2, a_3]) &= rot(len([a_2, a_3]), [a_2, a_3, a_1]) \\ &= rot(len([a_3]), [a_3, a_1, a_2]) \\ &= rot(len([]), [a_1, a_2, a_3]) \\ &= rot(0, [a_1, a_2, a_3]) \\ &= [a_1, a_2, a_3] \end{aligned}$$

Figure 9: A concrete proof of the rotate-length theorem for $n = 3$.

One can speculate why people might prefer schematic proofs. Firstly, it's a common observation of Mathematics and Computing teachers that students find recursion and mathematical induction puzzling and difficult when they first encounter them. They are thought, for instance, to be circular: defining a function in terms of itself, assuming the conclusion you want to prove. On the other hand, most people are familiar at an early age with philosophical induction (see Figure 5) and the generalisation from concrete examples to a general one. They are used to developing general procedures to deal with new situations by generalising from successful concrete cases. It

$$
\begin{aligned}
rot(len([a_1, a_2, a_3, \ldots, a_n]), [a_1, a_2, a_3, \ldots, a_n]) &= rot(len([a_2, a_3, \ldots, a_n]), [a_2, a_3, \ldots, a_n, a_1]) \\
&= rot(len([a_3, \ldots, a_n]), [a_3, \ldots, a_n, a_1, a_2]) \\
&\vdots \\
&= rot(len([]), [a_1, a_2, a_3, \ldots, a_n]) \\
&= rot(0, [a_1, a_2, a_3, \ldots, a_n]) \\
&= [a_1, a_2, a_3, \ldots, a_n]
\end{aligned}
$$

Figure 10: A schematic proof of the rotate-length theorem.

is, thus, very natural to apply this technique to mathematical problems — even though it is error prone. As we argued at the end of §1, educators will, therefore, often find that students understand schematic proofs more readily than they do Hilbertian ones.

The contrast between the accessibility of schematic and Hilbertian proofs is acutely illustrated in [Lakatos, 1976, chap. 2]. It relates Poincaré's Hilbertian proof of a special case of Euler's 'Theorem'. Not only is the proof extremely technical, but the uncertainty about its correctness is transferred from the proof itself to whether the encoding of polyhedra into incidence matrices is faithful.

6 Discussion

We have defined a common class of proofs, which we call *schematic*, that we claim resist the part of Hilbert's Programme that asserts that all proofs can be readily formalised as a sequence of formulae that are either axioms or follow from earlier formulae by a rule of inference. Schematic proofs *can* be formalised — using the constructive ω-rule — but not as Hilbertian proofs. Rather, their formalisation consists of a procedure that generates a different Hilbertian proof for each concrete case, that is, different in the sense that each such concrete case consists of a different sequence of proof steps.

Moreover, schematic proofs are error prone in that they can yield unexpected counter-examples. Schematic proofs can be shown to be error free if a meta-proof is conducted to prove that the proof generation procedure produces correct proofs for all inputs. Unfortunately, this part of the process is often omitted. Also, schematic proofs can be conducted in the absence of definitions that would be crucial in a Hilbertian context. This is possible because, apart from the meta-proof, schematic proving is just based on an analysis and generalisation of the concrete proofs of a few examples. It is possible to agree that these are examples of a concept without formally defining that concept. If the meta-proof is omitted, as it often is, the issue of definitions only arises when potential counter-examples are discovered, and it is necessary to decide whether they are really examples of the concept.

The formalisation part of Hilbert's Programme has, here-to-fore, largely avoided the criticism that the Completeness, Consistency and Decidability goals have attracted because of Gödel's incompleteness theorems. We have argued that this Formalisation goal of the Programme is also not as straightforward as it is sometimes assumed to be.

References

[Baker, 1993] Baker, S. (1993). *Aspects of the Constructive Omega Rule within Automated Deduction*. Unpublished Ph.D. thesis, Edinburgh.

[Bruijn, 1980] Bruijn, N. G. de. (1980). A survey of the project Automath. In Seldin, J. P. and Hindley, J. R., (eds.), *To H. B. Curry; Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 579–606. Academic Press.

[Bundy, 2012] Bundy, A. (2012). Reasoning about representations in autonomous systems: what Pólya and Lakatos have to say. In McFarland, D., Stenning, K. and McGonigle-Chalmers, M., (eds.), *The Complex Mind: An Interdisciplinary Approach*, chapter 9, pages 167–183. Palgrave Macmillan.

[Bundy *et al*, 2005] Bundy, A., Jamnik, M. and Fugard, A. (2005). What is a proof? *Phil. Trans. R. Soc A*, 363(1835):2377–2392.

[Fugard, 2005] Fugard, A.J.B. (2005). An exploration of the psychology of mathematical intuition. Unpublished M.Sc. thesis, School of Informatics, Edinburgh University.

[Gödel, 1931] Gödel, K. (1931). Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatsh. Math. Phys.*, 38:173–198. English translation in [van Heijenoort, 1967].

[Hilbert, 1930] Hilbert, D. (1930). *Die Grundlebung der elementahren Zahlenlehre*, volume 104. Mathematische Annalen.

[Jamnik & Bundy, 2005] Jamnik, M. and Bundy, A. (2005). *Psychological validity of schematic proofs*, volume LNCS 2605 of *Lecture Notes in Computer Science*, pages 321–341. Springer-Verlag GmbH.

[Jamnik, 2001] Jamnik, M. (2001). *Mathematical Reasoning with Diagrams: From Intuition to Automation*. CSLI Press, Stanford, CA.

[Lakatos, 1976] Lakatos, I. (1976). *Proofs and Refutations: The Logic of Mathematical Discovery*. Cambridge University Press.

[Nelsen, 1993] Nelsen, R. B. (1993). *Proofs without Words: Exercises in Visual Thinking*. The Mathematical Association of America.

[Robinson & Voronkov, 2001] Robinson, Alan and Voronkov, Andrei, (eds.). (2001). *Handbook of Automated Reasoning*. Elsevier, 2 volumes.

[Shoenfield, 1959] Shoenfield, J. R. (1959). On a restricted ω-rule. *Bulletin de l'Académie Polonaise des Sciences : S'erie des sciences mathematiques, astronomiques et physiques*, 7:405–7.

[van Heijenoort, 1967] van Heijenoort, Jan. (1967). *From Frege to Gödel: a source book in Mathematical Logic, 1879-1931*. Harvard University Press, Cambridge, Mass.

[Zach, 2009] Zach, R. (2009). Hilbert's program. *Stanford Encyclopedia of Philosophy*.