Mathematical method and proof

Jeremy Avigad (avigad@cmu.edu)
Carnegie Mellon University

Abstract. On a traditional view, the primary role of a mathematical proof is to warrant the truth of the resulting theorem. This view fails to explain why it is very often the case that a new proof of a theorem is deemed important. Three case studies from elementary arithmetic show, informally, that there are many criteria by which ordinary proofs are valued. I argue that at least some of these criteria depend on the methods of inference the proofs employ, and that standard models of formal deduction are not well-equipped to support such evaluations. I discuss a model of proof that is used in the automated deduction community, and show that this model does better in that respect.

Keywords: Epistemology of mathematics, mathematical proof, automated deduction

1. Introduction

It is generally acknowledged that at least one goal of mathematics is to provide correct proofs of true theorems. Traditional approaches to the philosophy of mathematics have therefore, quite reasonably, tried to clarify standards of correctness and ground the notion of truth.

But even an informal survey of mathematical practice shows that a much broader range of terms is employed in the evaluation of mathematical developments: concepts can be fruitful, questions natural, solutions elegant, methods powerful, theorems deep, proofs insightful, research programs promising. Insofar as judgments like these channel the efforts and resources we devote the practice, it is both a philosophical and pragmatic challenge to clarify the meaning of such terms.[1]

Value judgments applied to mathematical proofs provide particularly interesting examples. For, on a traditional view, the role of a proof is to demonstrate *that* a theorem is true; but it is very often the case that new proofs of an old theorem are valued, a fact that is rendered utterly mysterious by the standard characterization. Salient examples of the phenomenon are Dedekind and Weber's algebraic proofs of the Riemann-Roch theorem, the Selberg-Erdös proofs of the Hadamard-de la Vallée Poussin prime number theorem, or the 150 or so proofs of the law of quadratic reciprocity that have been published since Gauss' *Disquisitiones Arithmeticae*[2]; but the phenomenon is ubiquitous, from the most elementary mathematical proofs to the most complex.[3]

Put simply, the challenge is to explain what can be gained from a proof beyond knowledge that the resulting theorem is true. Of course, one sense in which a proof may be viewed as constituting an advance is that it may actually establish a stronger or more general statement, from which the original theorem easily follows. But even in cases like these we need to account for the intuition that the proof can also augment our understanding of the original theorem itself, providing a better sense of *why* the theorem is true.

Such proofs are sometimes called *explanatory* in the philosophical literature, and there is a small but growing body of work on the notion of explanation in mathematics (e.g. [50, 38, 39]). I will use the term here only gingerly, for two reasons: first, the term is not so very often used in ordinary mathematical discourse; and, second, it is certainly not the *only* term which is used to voice positive judgments about proofs. Here, I would prefer to remain agnostic as to whether there is a single overarching concept that accounts for all such positive judgments, or rather a constellation of related notions; and also as to whether the particular virtues considered here are best labeled "explanatory." A further difficulty with respect to obtaining a satisfactory theory is that judgments often vary as to the relative merits of different proofs; this is why it is common to find a dozen specialists in a subject writing thirteen introductory textbooks. The best we can therefore hope for is a theory that clarifies the factors that underly such judgments and helps explain the differences, e.g. ascribing them to differences of context, purpose, or emphasis.[4]

We do have some fairly good intuitions as to some of the reasons that one may appreciate a particular proof. For example, we often value a proof when it exhibits methods that are powerful and informative; that is, we value methods that are generally and uniformly applicable, make it easy to follow a complex chain of inference, or provide useful information beyond the truth of the theorem that is being proved. As a philosophical thesis, however, this claim is lacking. For one thing, it is vague: I have not said what it means for a proof to "exhibit" a "method," let alone what it means for a method to be general and uniformly applicable; nor have I said anything about how methods help render a proof intelligible, or the types of information they can convey. A second objection is that the claim is rather toothless: few would deny that the attributes indicated are generally desirable.

My goal here is to suggest that the first objection can be reasonably addressed. In other words, it is possible to develop an analytic theory of proof and method that can do philosophical work, and, in particular, can be used to clarify such evaluatory terms. To that end, I will discuss a model of proof that is currently used in the field of automated deduction, and attempt to enlist the corresponding terminology and framework for a more conceptual analysis. If this is successful, the second objection noted above will, instead, become an asset: insofar as the terms can be made sense of, the result will be a philosophical claim that stands a good chance of being correct.

The analysis begun here rests on the central assumption that at least some of the value judgments that are commonly applied to mathematical proofs are actually derivative of value judgments applied to associated methods. This association can happen in at least two ways. Sometimes new methods are *introduced* in the course of a proof; for example, Gauss' sixth proof of the law of quadratic reciprocity introduced the method of Gauss sums, which paves the way to higher-order generalizations; and the Dedekind-Weber proof of the Riemann-Roch theorem was ground-breaking in its introduction of algebraic methods to the study of function spaces (cf. the discussion in Corfield [13]). Sometimes, in contrast, old results are reproved in order to *illustrate* the benefits of methods that have been introduced in the development of a more general theory. For example, Dedekind often went out of his way to show how the new methods developed in his theory of ideals result in perspicuous proofs of established theorems of number theory, from Fermat to Kummer (see, for example, [16, §26–27]). In both situations, praise for the proofs can be read, at least in part, as praise for the associated methods.

The project begun here should be situated with respect to the much broader program of developing a theory of mathematical understanding. This can involve characterizing various mathematical activities (e.g. computing, conjecturing, problem solving, theory building), as well as characterizing the complex network of mathematical goals and subgoals (e.g. proving certain theorems, classifying structures, understanding mathematical phenomena, discovering important truths). This larger program is dauntingly vague, broad, and open-ended, and my hope is to make incremental progress by isolating a small, interesting, and more manageable subtopic. Such a separation, however, will make my account in some ways unnatural and in very many ways incomplete, and so some reference to the broader context will help clarify the scope of the restricted endeavor.

What I hope to begin to understand here are those features of mathematical practice that make a proof intelligible, for example, enabling us to *see* that a conclusion Y follows from hypotheses $X_1, X_2, \ldots, X_n$ in a "straightforward" way. In other words, the kinds of *methods* I will focus on are best characterized as *methods of inference*. This way of setting things up blurs the distinction between the context of discovery and the context of justification: we verify that Y follows from $X_1, X_2, \ldots, X_n$ by searching for an appropriate justification, using appropriate methods. A more significant difference is one of scale: in this essay I will focus on the process of understanding relatively small inferential steps. This leaves out the more complex, creative, and heuristic processes involved in finding complex proofs *ab initio*, attacking open problems, or developing theoretical machinery to support such efforts. Attention *has* been given to such higher processes in the automated deduction literature (see, for example, the literature on *proof planning* and *rippling*, in which [8] was seminal). Understanding the lower-level processes that I address is certainly relevant to understanding the higher-level ones, but I will not speculate here as to whether the difference is primarily one of scale, or whether a qualitatively different type of analysis is needed.

The notion of a method can, and has, been fruitfully used to characterize other types of mathematical activity as well. That is to say, there is also a literature on methods of solving mathematical problems (see, for example, [45]), methods of forming new concepts and conjectures (Lenat's Ph.D. thesis [37] was an early and important contribution, and [11] is a more recent one), and so on. Although I will not consider these here either, it will become clear in the discussion below that such issues lurk nearby. For example, we will see informally that some proofs are informative because they show us how an associated problem can be solved; thus methods of proof are related to methods of problem solving. At times we will even find that higher-order methods are called for: for example, we often wish to speak of methods of proof that can be *generalized*, talk which can naturally be understood to imply that there are second-order methods that transform specific proof methods into more general ones.

The structure of this essay is as follows. In Section 2, I will present three basic yet important theorems from elementary number theory, and discuss, informally, some of the benefits of various proofs of each. In Section 3, I will call attention to some of the *features* of proofs that seem to be relevant to this discussion. After showing that standard models of formal deduction fail to make these features apparent, I will discuss the model of proof alluded to above, and show that it fares better in this regard. The passage from Section 2 to Section 3 will be seen to be a move from language that is vague to language that is overly specific. The challenge, then, is to formulate a framework that abstracts away features that are "implementation specific," yet remains concrete enough to be informative. In Section 4, I will speculate as to how we can develop such a theory.

To be clear, then, this essay does not offer a general theory of mathematical understanding, or even a fragment of one. It does not go so far as to provide a framework that explains how mathematical proofs are evaluated. It does, however, take some initial steps towards developing such a framework, using informal case studies to identify some features of proofs that a satisfactory theory must take into account.

It will become apparent that the approach I am advocating is resolutely syntactic. An anonymous referee has reasonably questioned whether such an approach will be able to deliver philosophical explanations that we will find satisfying, with respect to a general theory of mathematical understanding or even the narrower issues addressed here; or whether alternative, semantic approaches are more appropriate. Although this is the kind of question that cannot be resolved at the outset, it should be kept in mind throughout the inquiry. I return to this issue briefly in Section 4.

2. Case studies

In this section, I will discuss three theorems of elementary number theory, none of which require mathematical background beyond elementary algebra and arithmetic. We will see that all three were known to Euler in eighteenth century, and were historically important to the development of the subject. In each case, I will present three distinct proofs. An informal discussion of the various advantages of each will provide us with a starting point from which to begin a more careful analysis.

Historical details beyond those mentioned here can be found in [21, 28, 44, 53] and Stillwell's introduction to [16]. Ultimately, Dickson's exhaustive [17] is the definitive reference for developments in number theory through the end of the nineteenth century.

2.1. Fermat primes

If x and y are integers, we say that x *divides* y, written $x|y$, if there is an integer z such that $xz = y$. The integers ± 1 are called "units," and, since they divide 1, they divide every integer. An integer x not equal to ± 1 is called *irreducible* if it has no nontrivial divisors, that is, no divisor that is neither a unit nor a unit multiple of x. An integer x not equal to ± 1 is called *prime* if whenever $x|yz$, then $x|y$ or $x|z$.

What I have called "irreducible" is what often goes by "prime" in an elementary mathematics education. Fortunately, when it comes to the integers, there is no difference: every irreducible number is prime, and vice-versa. The harder direction, i.e. the fact that every irreducible number is prime, is a consequence of the fact that the greatest common divisor of any two positive integers can be expressed as a linear combination of these two integers. The *Euclidean algorithm* yields an explicit means of doing so, and that algorithm, in turn, relies on the *division algorithm*: given any integer x and nonzero dividend y, we can write $x = qy + r$, where q is the "quotient" and r is the "remainder," the latter satisfying $0 \leq r < |y|$.

You should note that on the definition above, both 5 and -5 are considered prime. In fact, they are essentially the *same* prime, since they differ by a multiplicative factor of a unit. Below, however, it will be more convenient to use the word "prime" to denote the *positive* primes. Let us therefore adopt this convention.

It turns out that the numbers $2^{2^0} + 1$, $2^{2^1} + 1$, $2^{2^2} + 1$, $2^{2^3} + 1$, and $2^{2^4} + 1$ are all prime. As early as 1640, Fermat conjectured that $2^{2^n} + 1$ is prime for every natural number n, and in 1659 he hinted that he had a proof. The statement, however, was refuted by Euler [22].

THEOREM 2.1. $2^{2^5} + 1$ *is not prime.*

Proof #1. A calculation shows that

$$2^{2^5} + 1 = 2^{32} + 1 = 4294967297 = 641 \cdot 6700417,$$

as required. □

Sometimes a proof is nothing more than a calculation. In some contexts, this is optimal: it can provide a straightforward verification, requiring little thought or background knowledge.

Some ingenuity, however, makes it possible to shorten the calculation considerably. The next proof[5] is naturally expressed using a notation for *congruence* that was introduced by Gauss [27], and was therefore unavailable to Euler. Two integers x and y are said to be *congruent modulo* a third integer z, written $x \equiv y \pmod{z}$, if z divides $x - y$. In other words, all the following statements are equivalent:

- $x \equiv y \pmod{z}$
- $z|(x - y)$
- $x - y = kz$, for some integer k.

It will be convenient below to pass between these various representations freely. The relation of being congruent modulo an integer z is an *equivalence* relation, which is to say, it is reflexive, symmetric, and transitive. Furthermore, it respects addition and multiplication; that is, if $x_1 \equiv y_1 \pmod{z}$ and $x_2 \equiv y_2 \pmod{z}$, then $x_1 + y_1 \equiv x_2 + y_2 \pmod{z}$ and $x_1 y_1 \equiv x_2 y_2 \pmod{z}$. These facts make it possible to transfer valid forms of reasoning about arithmetic equations to congruences.

Proof #2. First, note that $641 = 5 \cdot 2^7 + 1$, so

$$5 \cdot 2^7 \equiv -1 \pmod{641}.$$

Raising both sides to the fourth power, we have

$$5^4 \cdot 2^{28} \equiv 1 \pmod{641}.$$

On the other hand, we also have $641 = 5^4 + 2^4$, that is,

$$5^4 \equiv -2^4 \pmod{641}.$$

Multiplying both sides by 2^{28} we have

$$5^4 \cdot 2^{28} \equiv -2^{32} \pmod{641}.$$

From the second and fourth congruences we have

$$1 \equiv -2^{32} \pmod{641}.$$

In other words, $641 | 2^{32} + 1 = 2^{2^5} + 1$, as required. □

The use of congruence notation is by no means essential to the proof; for example, the second congruence, which is equivalent to the assertion that $641 | 5^4 \cdot 2^{7 \cdot 4} - 1$, can be obtained using the identity

$$(5^4 \cdot 2^{7 \cdot 4} - 1) = (5 \cdot 2^7 + 1)(5 \cdot 2^7 - 1)(5^2 \cdot 2^{7 \cdot 2} + 1).$$

This identity lies hidden in the appeal to the properties of the congruence relation in the proof above; the notation is effective in removing such clutter.

One thing that can be said immediately about this proof is that it requires less tedious calculation than the first. One can certainly make sense of this in terms of the number of computation steps, given certain algebraic and arithmetic operations as "basic." But we can find additional virtues in the second proof. It can be said, perhaps, to partially explain what is special about 641, i.e. the fact that it can be written both as $5 \cdot 2^7 + 1$ and $5^4 + 2^4$. It also makes good use of properties of exponentiation, thereby explaining why that operation is relevant in the statement of the theorem. The proof also suggests a more general method by which other Fermat numbers can be shown to be composite; this method, and a precise sense in which it can be viewed as a generalization of the calculation above, is given by Baaz [3].

The previous proof may leave one wondering, however, how Euler initially hit upon 641. A later paper gives a clue: Euler [23] showed that if x and y are relatively prime (that is, have no common factor other than ± 1), then every factor of $x^{2^n} + y^{2^n}$ is either 2 or of the form $2^{n+1}k + 1$; he also noted that (taking $x = 2$ and $y = 1$) this implies that any factor of $2^{2^5} + 1$ must have a factor of the form $64k + 1$. The proof relies on *Fermat's little theorem*, which asserts that if p is prime and x is any integer not divisible by p, $x^{p-1} \equiv 1 \pmod{p}$. Taking this theorem for granted, the following proof encapsulates Euler's observation.

Proof #3. Suppose we are looking for a prime divisor p of $2^{32} + 1$, that is, a solution to

$$2^{32} \equiv -1 \pmod{p}.$$

Squaring both sides, we wish to find a p satisfying

$$2^{64} \equiv 1 \pmod{p}.$$

By Fermat's little theorem we know

$$2^{p-1} \equiv 1 \pmod{p}.$$

Let d be the least positive integer satisfying $2^d \equiv 1 \pmod{p}$. Then d must divide $p - 1$; otherwise, we could write $p - 1 = qd + r$ with $0 \leq r < d$, in which case

$$2^{p-1} \equiv 2^{qd} 2^r \equiv (2^d)^q 2^r \equiv 2^r \equiv 1 \pmod{p},$$

contrary to the choice of d. By the same reasoning, d must divide 64, and so must be a power of 2. But d cannot be less than or equal to 32, because otherwise we would have $2^{32} \equiv 1 \pmod{p}$; by the first congruence, this would imply $-1 \equiv 1 \pmod{p}$, that is, $p|2$, contradicting the hypothesis that p is a prime dividing $2^{32} + 1$. So d has to be 64, and p has to be of the form $64k + 1$. The first few primes of this form are $197, 257, 449, 577$ and 641. Trial and error shows that 641 is the first one that works. □

As far as verification is concerned, this proof is certainly no savings over the first; in fact, the net result is that one has to do the same calculation (and more). But the proof is explicitly designed to show how 641 could have been discovered in practice. Here, too, the proof displays ideas that are useful in related contexts; for example, the same method can be used to show that $2^{2^4} + 1$ *is* prime.

In principle, the fact that $2^{32}+1$ is composite could have been discovered by a brute force enumeration. Proofs that provide more palatable alternatives in situations like these can provide interesting case studies. Consider, for example, the following special case of Ramsey's theorem, which is often given to students as an exercise:[6]

> Suppose any two people at a party are assumed to either mutually know each other or not. Then at any party with six people, there will either be a group of three people all of whom know each other, or a group of three people all of whom do not know each other.

Once again, this can be shown, in principle, by enumerating all 2^{15} possibilities, but exploiting symmetries inherent of the formulation cuts down on the number of cases dramatically. Label the six people a, b, c, d, e, and f. Then of the other five, either there will be three people that a knows, or three people that a does not know. Assume, without loss of generality, the former, and, relabeling if necessary, call them b, c, and d. If none of these three know each other, we are done; otherwise, two of them, say b and c, know each other, and a, b, c is the desired triple.

There are reasons to prefer such a proof over a computer-assisted verification, beyond the savings in time. For example, the proof above gives hints as to how one may easily find a counterexample among five people (start by picking two people for a to know, and two for a not to know); and it can, perhaps, be said to explain "why 6" (roughly, because $6 = 1 + (2 \cdot 2 + 1)$). Most importantly, it conveys ideas that will help prove generalizations; for example, for every k there is an n big enough such that the statement above holds with "6" and "3" replaced by "n" and "k."

The examples we have just considered also show that sometimes the additional information valued in a proof can involve methods of solving an associated problem. Consider the following three:

- Show that $2^{2^5} + 1$ is composite.
- Determine whether or not $2^{2^5} + 1$ is composite.
- Find a nontrivial factor of $2^{2^5} + 1$.

I will take it that, in each case, a satisfactory solution has to include an explicit or implicit proof that the answer is correct. (We tell our students ad infinitum that in mathematics one must always justify one's answer.) But the three instructions request different sorts of information: the first asks for a proof; the second for a decision; the third for a factor. Thus viewing the theorem in terms of an associated problem often makes it clearer what additional information one might want, and what types of generalizations may be sought.

Finally, let us take note of the role played by the definitions of divisibility and congruence in the proofs above. We have already observed that such definitions can allow one to transfer methods of reasoning that are effective in other contexts, or are subsumed under a more general framework. For example, we have seen that congruence modulo an integer is an *equivalence relation*, inheriting some of the properties of ordinary equality; and divisibility is a *partial order*, which is to say, the relation $x|y$ has some of the same properties of the $\leq$ relation on the integers, or the $\subseteq$ relation on sets.

Notice also that the definition of divisibility involves an existential quantifier, and thus, derivatively, the notion of congruence does also. The fact, for example, that $x|y$ and $y|z$ implies $x|z$, or that $x \equiv y \pmod{z}$ implies $xw \equiv yw \pmod{z}$, expand to first-order implications with existential quantifiers in the antecedents and the conclusion; and their proofs show how witnesses in the conclusion are instantiated, given witnesses to the hypotheses. Later appeal to these general lemmas then eliminates the need to exhibit witnesses explicitly in the proof. We have already seen this at play in the discussion following the second proof above. The use of definitions to facilitate quantificational reasoning is an important one in mathematics; in fact, Tappenden [51] suggests that Frege's notion of a *fruitful* definition rests precisely on the use of quantifiers.

2.2. Products of sums of squares

In the *Arithmetic*, Diophantus notes that the product of $5 = 2^2 + 1^2$ and $13 = 3^2 + 2^2$ is 65, which is again a sum of two squares. (In fact, 65 is equal to both $8^2 + 1^2$ and $7^2 + 4^2$.) This is an instance of the following:

THEOREM 2.2. *If x and y can each be written as a sum of two integer squares, then so can xy.*

Proof #1. Suppose $x = a^2 + b^2$, and $y = c^2 + d^2$. Then

$$xy = (ac - bd)^2 + (ad + bc)^2,$$

a sum of two squares. □

Writing xy as $(ac+bd)^2+(ad-bc)^2$ works just as well, accounting for the two representations of 65 indicated above. These equations are implicit in Diophantus, and according to Dickson [17, vol. II, p. 226], can be found explicitly in Leonardo Pisano's *Liber Quadratorum* of 1225. The simplicity of the calculation has an added payoff: the proof uses only the commutativity and associativity of addition and multiplication, the distributivity of multiplication over addition and subtraction, and the fact that subtraction is an inverse to addition; hence it shows that the theorem is true much more generally in any *commutative ring.*

Our second proof of Theorem 2.2 involves a detour through the theory of *Gaussian integers* $\mathbb{Z}[i]$, that is, complex numbers of the form $a + bi$, where a and b are integers, and i is a square root of -1. If $\alpha = u + vi$ is any complex number, its *conjugate*, $\overline{\alpha}$, is defined to be $u - vi$. It is easy to check that conjugation is an automorphism of the field of complex numbers, which is to say, it preserves addition and multiplication. (Roughly speaking, this reflects that from the point of view of the real numbers and the field operations, the elements i and $-i$ are indistinguishable.) In particular, $\overline{\alpha \cdot \beta} = \overline{\alpha} \cdot \overline{\beta}$ for any α and β.

The *norm* $N(\alpha)$ of a complex number α is defined to be $\alpha\overline{\alpha}$. From the definition it is easy to see that the norm is multiplicative as well, i.e.

$$N(\alpha\beta) = \alpha\beta \cdot \overline{\alpha\beta} = \alpha \cdot \beta \cdot \overline{\alpha} \cdot \overline{\beta} = \alpha\overline{\alpha} \cdot \beta\overline{\beta} = N(\alpha)N(\beta).$$

Notice that if $\alpha = a+bi$ is a Gaussian integer, then $N(\alpha) = a^2+b^2$ is an ordinary integer. Conversely, we can always write $a^2 + b^2 = N(a + bi)$. In other words, the integers that can be written as the sum of two squares are exactly those that are norms of Gaussian integers. This gives a remarkably short proof of Theorem 2.2:

Proof #2. Suppose $x = N(\alpha)$ and $y = N(\beta)$ are sums of two squares. Then $xy = N(\alpha\beta)$, a sum of two squares. □

This brevity is in a sense misleading, since, in the final accounting, the relevant properties of the norm function have to be proved as well. But this is tempered by the fact that the notion of the norm of a complex number is much more generally useful. The (positive) square root of the norm is usually called the *modulus* or *absolute value*, and corresponds to the distance from the origin to the associated point in the Euclidean plane. As a result, the norm and modulus have useful geometric significance, the latter playing a role similar to the usual absolute value on the real numbers. For example, the Gaussian integers also satisfy a form of the division algorithm: any two Gaussian integers α and β can be written $\alpha = \beta\eta + \rho$, where $N(\rho) < N(\beta)$. Thus one can show, *just as for the integers*, that the notions “prime” and “irreducible” coincide for the Gaussian integers. We will make use of this important fact below.

In short, one can argue that the expense incurred in deriving properties of the norm should be entered as a capital improvement, and not charged against our particular application. Only with this understanding does it make sense to say that the second proof is shorter than the first.[7]

Our second proof also leads to interesting generalizations. The complex numbers, $\mathbb{C}$, are an example of a two-dimensional associative division algebra over the real numbers, $\mathbb{R}$. A theorem of Frobenius from 1877 asserts that aside from $\mathbb{R}$ itself, there is only one other finite-dimensional structure of this sort, namely, Hamilton's four dimensional algebra $\mathbb{H}$, the *quarternions*. Indeed, the corresponding notion of quarternion norm yields a product rule for four squares, originally due to Euler. The structures $\mathbb{R}, \mathbb{C}, \mathbb{H}$ all have the technical properties of being *alternative* and *quadratic* real algebras, with no zero divisors. If one is willing to give up associativity, a theorem by Zorn from 1933 shows that there is only one more structure of this sort: Cayley's eight-dimensional algebra $\mathbb{O}$, the *octonians*. And, sure enough, the octonian norm yields a product rule for sums of eight squares. Zorn's structure theorem can be used to prove a theorem due to Hurwitz in 1898, to the effect that these are the *only* product laws for sums of squares of this sort.[8] Thus, our second proof yields generalizations that not only explain other product laws and bring them under a uniform framework, but, in fact, lead to an algebraic classification that explains why there are no others. A lovely presentation of the mathematical and historical details can be found in chapters by Koecher and Remmert in [18].

The proof has generalizations in other directions, as well. Below we will consider Euler's use of Gaussian integers to prove Theorem 2.2. This use was a harbinger of what is probably the most significant trend in nineteenth century number theory: the use of finite algebraic extensions of the rational numbers, like the Gaussian integers, to address questions about the ordinary integers. The notions of conjugate and norm generalize to such number fields, and are useful there for exactly the same reason they are useful in our proof; namely, they exploit symmetries and relate properties of the extension to properties of the ground field. Even today we share in the nineteenth century fascination at the fruitfulness of this transfer. In 1860, in his *Report on the theory of numbers*, H. J. S. Smith wrote that

> ...the complex numbers of Gauss, Jacobi, and M. Kummer force themselves upon our consideration, not because their properties are generalizations of the properties of ordinary integers, but because certain of the properties of integral numbers can only be explained by a reference to them. (Smith [46, Art. 64], quoted in Corry [14, pp. 91–92].)

This language is compelling and mysterious: what can it mean for mathematical objects to "force themselves upon us," and wherein lies their explanatory power? Our second proof of Theorem 2.2, as simple as it is, provides an illustrative example.

There is another sense in which this proof is historically significant. Much has been written about the late nineteenth century emphasis on "conceptual methods" over calculation, forcefully advocated by Riemann in his development of the theory of complex functions, and by Dedekind in his development of algebraic number theory. (See [26, 29, 35, 49] for characterizations of this emphasis, as well as [19, 20] for less sanguine views as to the effects on algebraic number theory.) For example, Dedekind writes:

> Even if there were such a theory, based on calculation, it still would not be of the highest degree of perfection, in my opinion. It is preferable, as in the modern theory of functions, to seek proofs based immediately on fundamental characteristics, rather than on calculation, and indeed to construct the theory in such a way that it is able to predict the results of calculation... (Dedekind [16, §12], quoted by Stein [49, page 245])

This language is equally mysterious: what can it mean to base proofs on "fundamental characteristics rather than calculation," yet somehow "predict the results of calculation"? Once again, an analysis of our second proof of Theorem 2.2 can serve as a starting point for attempts to understand the phenomenon.

There is a proof that is intermediate between the two we have seen so far:

Proof #3. Suppose $x = a^2 + b^2$ and $y = c^2 + d^2$. Then

$$\begin{aligned} xy &= (a^2+b^2)(c^2+d^2) \\ &= (a+bi)(a-bi)(c+di)(c-di) \\ &= (a+bi)(c+di)(a-bi)(c-di) \\ &= ((ac-bd)+(ad+bc)i)((ac-bd)-(ad+bc)i) \\ &= (ac-bd)^2+(ad+bc)^2, \end{aligned}$$

a sum of two squares. □

This is the proof given by Euler in his *Algebra* [24] of 1770. Cauchy gave essentially the same proof in his *Cours d'analyse* [10, VII §I] of 1821, after introducing the term "conjugate," and before launching into a detailed presentation of the complex numbers and their properties. Our third proof is more or less the result of "unwinding" our second proof, expanding the definition of norm and including the steps needed to establish the supporting lemmas. To the extent to which we recognize this proof as different, we see that these aspects of the presentation are important. In other words, the ways in which information and inferential steps are encapsulated in definitions and lemmas has at least some bearing on what we can say about a proof.

Even such a minor rewriting can make a difference. Presenting the proof this way, one is apt to note that the terms can be grouped differently into conjugate pairs,

$$xy = (a + bi)(c - di)(a - bi)(c + di),$$

yielding a second representation of $(a^2+b^2)(c^2+d^2)$ as a sum of squares, $(ac+bd)^2+(ad-bc)^2$. In the next section, we will consider the question as to exactly which integers can be represented as a sum of two squares. Having both representations of a product is relevant to determining the *number of ways* such integers can be represented, a problem of equally longstanding concern in number theory.

2.3. Representability by sums of squares

In this section we will consider three proofs of the following theorem:

THEOREM 2.3. *Every prime number congruent to 1 modulo 4 can be written as a sum of integer squares.*

Remember that saying that p is congruent to 1 modulo 4 is equivalent to saying that p is of the form $4k + 1$, or that $p - 1$ is a multiple of 4.

In contrast to the theorems of Sections 2.2 and 2.1, proving Theorem 2.3 requires some sophistication. I have included a discussion of some of the proofs here because I felt that the subsequent analysis would be bolstered by an example of a “nontrivial” theorem of mathematics. On the other hand, most of the themes that arise have already made an appearance in the previous examples, and the conclusions I wish to draw will be summarized at the beginning of Section 3.1. Therefore, the reader who is eager to get to the point may well wish to skip this section on a first reading, and leave the more extended case study for a rainy day.

Note that every odd number is congruent to either 1 or 3 modulo 4, and so the square of an odd number is congruent to 1 modulo 4. Similarly, the square of any even number is congruent to 0 modulo 4, and so the sum of any two squares is always congruent to either 0, 1, or 2 modulo 4. This shows that no prime congruent to 3 modulo 4 can be written as a sum of squares. Since 2 is the only even prime, and $2 = 1^2 + 1^2$, Theorem 2.3 yields a precise characterization of the primes that can be written as sums of two squares.

In fact, it yields more. Suppose a positive integer $n > 2$ is written as a product of powers of distinct primes,

$$n = p_1^{a_1} p_2^{a_2} \dots p_k^{a_k}.$$

The preceding theorem and discussion, combined with Theorem 2.2, tells us that if those primes p_i that are congruent to 3 modulo 4 (if any) occur to an even power (and so, are perfect squares), then n can be written as a sum of squares. In fact, the converse also holds; which is to say that if a prime congruent to 3 modulo 4 occurs with an odd exponent in the prime factorization of n, n *cannot* be written as a sum of squares. Proving this fact is somewhat easier than proving Theorem 2.3.[9] Thus, Theorem 2.3 is the most difficult component in the following characterization of the integers that can be written as the sum of two squares:

THEOREM 2.4. *A positive integer n can be written as a sum of two squares if every prime congruent to 3 modulo 4 occurring in the factorization of n occurs to an even power.*

Theorem 2.4 was stated, without proof, by Girard in 1632. We have seen that an interest in the types of integers than can be written as sums of two squares traces back to Diophantus, and, indeed, Theorem 2.3 appears as one of Fermat's marginal notes to his copy of Bachet's edition of the *Arithmetic*. In letters to Pascal, Digby, and Carcavi, in 1654, 1658, and 1659, respectively, Fermat claimed to have a proof of Theorem 2.3; in the last, he said he used the "method of infinite descent," of which more will be said below. (Further historical details can be found in [21, 28, 44, 53], and there is an exhaustive historical account in Dickson [17, Vol. II Chap. VI].) All the proofs we will consider rely on the following lemma:

LEMMA 2.5. *If $p \equiv 1 \pmod 4$, there is a natural number m such that $m^2 \equiv -1 \pmod p$.*

Note that by the observations above, the hypothesis is e.g. equivalent to saying that p is of the form $4n + 1$, and the conclusion is equivalent to saying that p divides $m^2 + 1$. For completeness, I will sketch various proofs of Lemma 2.5 in a footnote,[10] but these will not be needed in the discussion that follows.

The first proof we will consider is adapted from Euler's original proof from 1747:[11]

LEMMA 2.6. *Let $x = a^2 + b^2$ and $p = c^2 + d^2$ each be a sum of two squares, with p prime. If $p|x$, then x/p is also a sum of two squares.*

Proof. By hypothesis, p divides x, so it also divides

$$a^2p - c^2x = a^2(c^2 + d^2) - c^2(a^2 + b^2) = a^2d^2 - b^2c^2 = (ad - bc)(ad + bc).$$

Since p is prime, it must divide one of these two factors. Suppose it divides $(ad - bc)$. Using one of the formulas for the product of sums of squares, write

$$px = (a^2 + b^2)(c^2 + d^2) = (ad - bc)^2 + (ac + bd)^2.$$

Since p divides the left-hand side and $ad-bc$, it must also divide $ac+bd$. Dividing both sides of the equation by p^2 yields

$$x/p = ((ad - bc)/p)^2 + ((ac + bd)/p)^2,$$

as required. If, instead, p divides $(ad + bc)$, use the product formula

$$px = (ad + bc)^2 + (ac - bd)^2$$

and proceed similarly. □

Proof #1. By Lemma 2.5, it suffices to show that every prime number p dividing a number of the form $m^2 + 1$ can be written as a sum of two squares. Suppose otherwise; then there is a *smallest* prime p that divides a number of the form $m^2 + 1$ and cannot be written as a sum of two squares. Pick such an m corresponding to this p, and by the division algorithm, write $m = qp + r$, with $0 \leq r < p$. Then p divides

$$m^2 + 1 = (qp + r)^2 + 1 = q^2p^2 + 2pqr + r^2 + 1.$$

Since p divides the first two terms on the right, it must also divide r^2+1. Write $r^2 + 1 = py$; since $r < p$ we have $r^2 < p^2$, and so $r^2 + 1 < p^2$. (If r^2+1 were exactly equal to p^2, we would have $p^2-r^2 = (p+r)(p-r) = 1$, contradicting the fact that $p \geq 2$.) Hence $y < p$. Factor y into primes $q_1, \ldots, q_l$; then each q_i is less than p, and so, by our assumption on p, can be written as a sum of squares. Applying Lemma 2.6 l times, we conclude that p can be written as a sum of squares, contrary to our hypothesis. □

As I have presented it, the proof is nonconstructive; instead of showing how p can be written as a sum of squares, it shows that the contrary assumption is contradictory. Of course, if one believes the conclusion, one can find a sum of squares by a methodical search. But the argument above can easily be turned into a direct proof. Given a prime p of the form $4n + 1$, the second and fourth proofs sketched in footnote 4 show, explicitly, how to obtain an m such that p divides $m^2 + 1$. The next lemma and the proof of the theorem then show, explicitly, how to reduce the problem of writing the prime p as a sum of two squares to the problem of writing the smaller primes $q_1 \ldots q_l$, which divide $r^2 + 1$, as sums of two squares. (In fact, the algorithm can be improved; see the discussion in [21, Section 2.6].)

This proof, then, has a lot going for it; it is elementary, straightforward, and computationally informative. It also illustrates Fermat's oft used "method of descent," that is, showing how a putative counterexample in the positive integers can be repeatedly replaced by a smaller one.

The formula $x^2 + y^2$ is an instance of a *binary quadratic form with integer coefficients*, which are expressions of the form $ax^2 + bxy + cy^2$, with a, b, c integers. I will call these "forms" for short, and use (a, b, c) to denote the form with with given coefficients. The values one obtains by substituting integer values for x and y are called the integers *represented by* the form. Thus Theorem 2.4 solves one instance of the problem of determining which integers can be represented by a given form. The second proof we will consider uses the notion of *equivalence* of forms, which was introduced by Lagrange and further developed by Gauss, and used by both to address the more general problem.

Consider what happens when we make the substitutions

$$\begin{aligned} x &= rx' + sy' \\ y &= tx' + uy' \end{aligned}$$

The reader can check by straightforward calculation that the form $ax^2 + bxy + cy^2$ becomes a new form $a'x'^2 + b'x'y' + c'y'^2$ in the variables x', y', where

$$\begin{aligned} a' &= ar^2 + brt + ct^2 \\ b' &= 2ars + b(ru + st) + 2ctu \\ c' &= as^2 + bsu + cu^2 \end{aligned}$$

I will say that the form (a, b, c) has been *transformed* into (a', b', c') by the transformation

$$S = \begin{pmatrix} r & s \\ t & u \end{pmatrix}.$$

Clearly, any integer represented by (a', b', c') can be represented by (a, b, c); if $a'x'^2 + b'x'y' + c'y'^2 = n$, then $x = rx' + sy'$ and $y = tx' + uy'$ is a solution to $ax^2 + bxy + cy^2$.

Under what conditions can (a', b', c') be transformed *back* into (a, b, c)? A bit of algebraic manipulation shows that if $\delta = ru - ts$ is nonzero, the transformation

$$\begin{pmatrix} u/\delta & -t/\delta \\ -s/\delta & r/\delta \end{pmatrix}$$

brings x', y' back to x, y. If $\delta = \pm 1$, the entries above will be integers, in which case the argument above shows that the two forms will represent exactly the same values. One can check that the process works the other way round: applying the second transformation to a quadratic form and then the first brings one back to the initial starting point; and the value $\delta' = (ru - ts)/\delta^2$ associated with the second transformation is also ± 1. Two forms that are related this way are said to be *equivalent*, and the associated transformations are said to be *unimodular.* Clearly every form is equivalent to itself, and we have just seen that if (a, b, c) is equivalent to (a', b', c'), then (a', b', c') is equivalent to (a, b, c). Another straightforward calculation shows that the result of composing two unimodular transformations is again a unimodular transformation, so that equivalence is transitive as well. In other words, equivalence really *is* an equivalence relation.

We need one last ingredient. The *discriminant* of the form (a, b, c) is defined to be the integer $b^2 - 4ac$. A straightforward calculation shows that if (a, b, c) and (a', b', c') are equivalent forms, they have the same discriminant; in other words, the discriminant is an *invariant* of the equivalence relation.

LEMMA 2.7. *Every form is equivalent to a form* (a, b, c) *in which* $|b| \leq |a| \leq |c|$.

Proof. Notice that the unimodular transformation

$$\begin{pmatrix} 1 & s \\ 0 & 1 \end{pmatrix}$$

transforms (a, b, c) into a form (a', b', c') in which $a' = a$ and $b' = 2as+b$. By a suitable choice of s, we can always guarantee that $|b'| \leq |a'| = |a|$. (To do so, first note that without loss of generality, we may assume that a is positive; otherwise, solve the problem with $-a$ in place of a and then replace s by $-s$. Assuming a is positive, use the division algorithm to write $-b = (2a)s + r$, where $0 \leq r < 2a$. If $r > a$, replace s by $s + 1$ and r by $r - 2a$, so $-b = (2a)s + r$, with $|r| \leq a$. Then $b' = -r = (2a)s + b$ satisfies $|b'| = |r| < |a|$, as required.)

If $|a'| \leq |c'|$, we are done. Otherwise, the unimodular transformation

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

transforms (a', b', c') into a form (a'', b'', c'') in which $a'' = c'$, so that now $|a''| = |c'| < |a|$. We now return to the first step with (a'', b'', c'') in place of (a, b, c); the fact that $|a|$ decreases at each step guarantees that the algorithm must ultimately terminate successfully. □

A form (a, b, c) satisfying the conclusion of the lemma is said to be *reduced.* Note that in any reduced form we have

$$c^2 = |c|^2 \geq |a||c| \geq |a|^2 = a^2 \geq |b|^2 = b^2.$$

If ac is positive, then $4ac - b^2$ is positive, and we have

$$4ac - b^2 \geq 4a^2 - a^2 = 3a^2.$$

If ac is negative, then $4ac - b^2$ is negative, and we have

$$b^2 - 4ac = b^2 + 4|ac| \geq 4|ac| \geq 4a^2 > 3a^2.$$

Either way, we have shown that in any reduced form, $3a^2$ is less than or equal to the absolute value of the discriminant, $|b^2 - 4ac|$. This tells us that there are only finitely many reduced forms with a given discriminant, since there are only finitely many values of a and b that are small enough in absolute value, and these determine c.

Now consider the reduced form $x^2 + y^2$, which has discriminant -4. Note that if $ax^2+bxy+cy^2$ is also in reduced form and has discriminant -4, then $3a^2 \leq 4$, so a can only be -1, 0, or 1. Trying these same possibilities for b shows that the only reduced forms with discriminant -4 are x^2+y^2 and $-x^2-y^2$. In other words, *any form with discriminant* -4 *that represents a positive integer is equivalent to* x^2+y^2. This gives us an easy proof of our main theorem:

Proof # 2. Suppose p is of the form $4n + 1$. By Lemma 2.5, choose m so that $p|m^2 + 1$. Then p is clearly represented by the form $px^2 + 2mxy + \frac{m^2+1}{p}y^2$, taking $x = 1$ and $y = 0$. This form has discriminant -4, and so, by the preceding discussion, is equivalent to $x^2 + y^2$. □

There is a lot to like about this proof. The argument shows, straightforwardly, how one can transform the form $px^2 + 2mxy + \frac{m^2+1}{p}y^2$ to $x^2 + y^2$, and hence how to transform the integers $1, 0$ representing p in the first form into integers representing p in the second. As in the first proof, it is easy to see what is getting smaller at each stage. This provides us with not just an explicit algorithm, but also a strong sense as to *why* the theorem is true.

It also provides a general strategy for studying other forms. Indeed, the argument generalizes immediately to forms like $x^2 + 2y^2$ and $x^2 + 3y^2$, where one can again show that all positive-valued forms with the corresponding discriminants are equivalent. The fact that there are *inequivalent* forms with the same discriminant as $x^2 + 4y^2$ helps explain comparatively anomalous behavior of numbers represented by this latter form. (See, for example, the helpful discussion in Stillwell's introduction to [16].) It also raises the question of determining the number of inequivalent forms of a given discriminant. For suitable discriminants, this is known as the *class number* of an associated finite field extension of $\mathbb{Q}$, the determination of which plays a central role in modern number theory.

But there is more we can say. The notion used above to represent transformations may call to mind the matrices one encounters in an introductory course in linear algebra. This is no coincidence. If one associates to the form (a, b, c) the symmetric matrix

$$A = \begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix},$$

then for every x and y the value $ax^2 + bxy + y^2$ can be obtained by the matrix product

$$\begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

involving A. The discriminant of the form is just -4 times the determinant of A. The form corresponding to the transformation S described above is just the one associated to the matrix product $S^t AS$, where S^t denotes the *transpose* of S, that is, the result of exchanging entries of S across the main diagonal. The composition of two transformations corresponds to the product of the associated matrices; unimodular transformations correspond to matrices with determinant ±1; and the fact that equivalent forms represent the same integers simply reflects the fact that the corresponding matrices have inverses with integer entries. In short, the proof can be recast as a perspicuous and fruitful application of the methods of linear algebra, which, by the end of the nineteenth century, had become a central tool in arithmetic, algebra, geometry, and analysis. In fact, Gauss' implicit use of ideas from linear algebra in his analysis of forms was instrumental in the development of the theory of matrices and determinants (see [33]). Thus, we appreciate our second proof because it makes effective use of linear algebra, and, indeed, played a part in the historical development of this very useful collection of tools.

But the importance of our second proof runs even deeper than that. The argument exploited a number of general strategies: introducing an equivalence relation that filters out representational features that are subordinate to the solution of the problem, assigning a suitable invariant to the associated equivalence classes, and choosing canonical representatives whenever possible. These strategies are pervasive in modern mathematics, and nineteenth century mathematicians were eminently conscious of this fact. In this respect as well, our second proof is commonly viewed as an early and important archetype. Thus we can admire the proof for exhibiting one of the most generally valuable strategies in modern mathematics, and, indeed, for being instrumental in the development thereof.

The last proof we will consider makes use of the Gaussian integers, and, in particular, the following key fact:

LEMMA 2.8. *Every irreducible element of* $\mathbb{Z}[i]$ *is prime.*

We have already noted in Section 2.2 that given the notion of the norm of a Gaussian integer, the lemma can be proved much the same way one proves the corresponding statement for the integers. With this in hand, we have a quick proof of Theorem 2.3:[12]

Proof #3. By the lemma, let m be such that $p|m^2+1$. Passing to the Gaussian integers, we have

$$p|m^2+1=(m+i)(m-i).$$

On the other hand, p does not divide either of $m+i$ or $m-i$, since the quotients $m/p+i/p$ and $m/p-i/p$ are not Gaussian integers. So p, when considered as a Gaussian integer, is not prime. Hence, by Lemma 2.8, it is not irreducible. Hence, it can be written $p=xy$, where x and y are Gaussian integers that are not units. Taking norms, we have $p^2=N(p)=N(xy)=N(x)N(y)$. But now this is an equation in the positive integers; since neither of $N(x),N(y)$ is equal to 1, we have $N(x)=N(y)=p$, so p is a sum of two squares. □

Proof #3 is remarkably short. To be sure, it requires Lemma 2.8, which is the key component in showing that the Gaussian integers satisfy the unique factorization property; and the proof of this lemma requires work. But as Dedekind was fond of pointing out, once one is careful to identify the properties of the integers that are used to prove unique factorization there, the generalization to the Gaussian integers comes at little extra cost. The axiomatic characterization of a *Euclidean domain* makes it possible to account for both these instances by subsuming them under a more general theorem, and makes our third proof seem like a bargain.

This proof is, in fact, constructive; the greatest common divisor of $m+i$ and p can be computed by the Euclidean algorithm, and yields a nontrivial factor x of p satisfying $p=N(x)$. But the details of the algorithm are relegated to more fundamental aspects of the theory, leaving the focus of the proof on the algebraic properties of the Gaussian integers.

Finally, our proof fares well with respect to generality and fruitfulness. Similar methods can be used in any finite extension of the rationals satisfying unique factorization. This makes it possible to transfer intuitions about the natural numbers to intuitions about the rings of "integers" in these more general fields, and use these intuitions to understand complex phenomena in the ordinary integers. The fact that there are such extensions for which unique factorization fails was the primary impetus to the theory of ideal divisors, which began with

Kummer and received fuller, though distinct, treatments in the hands of Kronecker and Dedekind. This theory, which managed to restore the phenomenon of unique factorization through the creation of an enlarged domain of "ideal" prime factors, was the most important nineteenth century development in algebraic number theory, and many natural questions about quadratic forms and the like can usefully be posed in this general framework. For example, the problem of determining the class number of a form, described above, translates to the problem of determining the cardinality of an associated group of ideal divisors. In [16, §27], Dedekind shows how our Theorem 2.3 follows from a much more general theorem, typical of the theory, due to Kummer.

There are many other proofs of Theorem 2.3, including proofs using continued fractions, by Hermite [32] and Smith [47].[13] A proof using Minkowski's important geometric methods can be found e.g. in [31]. In recent years, Conway [12] has provided an intuitive and visual representation of the Gauss-Lagrange reduction procedure. Aigner and Ziegler provide a proof by Don Zagier [57] in their *Proofs from the Book*, the title of which is a reference to Paul Erdös' oft-repeated claim that God has a book with the most elegant proof of every mathematical theorem. Whether or not one agrees with their assessment of Zagier's argument, their choice shows that mathematicians can still wax enthusiastic at the appearance of new proof, more than 350 years after the theorem was apparently first proved by Fermat, and almost 250 years after a proof was published by Euler.

3. Towards a better understanding of proof

3.1. Reflection on the case studies

Our case studies have provided us with a corpus of examples, in which we have discerned a grab bag of virtues that mathematical proofs can enjoy. Some of these virtues may be classified as explanatory: a proof can explain how it might have been discovered, how an associated problem was solved, or why certain features of the statement of the theorem are relevant. Proofs may also establish stronger statements than the theorem they purport to prove; they may introduce definitions and methods that are useful in other contexts; they may introduce definitions and methods that can fruitfully be generalized; or they may suggest solutions to more a general problem. They can also suggest related theorems and questions. We can add a few more fairly obvious virtues to the list: a good proof should be easy to read, easy to remember, and easy to reconstruct. Sometimes our criteria are at odds with one another: for example, we may value a proof for providing explicit algorithmic information, whereas we may value another proof for downplaying or suppressing calculational detail.[14]

This informal analysis should be viewed as a *starting point* for philosophical inquiry, rather than as a satisfactory conclusion. What, exactly, does it mean to say that a proof shows us how a problem is solved? How, exactly, do proofs reveal or suppress algorithmic information? Precisely what features of a mathematical presentation make it easy to follow? The challenge now is to clarify what it is that we think proofs are doing, and understand the mechanisms by which they do it.

If we are to take the informal discussion in Section 2 seriously, the general character of the remarks will put serious constraints on the way we try to account for the data. For example, all of the following were implicit in the informal analysis:

1. A proof is some kind of communicable text (which may involve diagrams) that, in particular, provides sufficient information to establish that the purported theorem is true.

2. Beyond correctness, proofs can be evaluated with respect to differing (and sometimes competing) desiderata.

3. Higher-level features of the presentation of a proof, such as the organizational role of lemmas and definitions, are relevant to the evaluations.

4. The evaluations, with respect to both correctness and other standards of merit, are carried out with respect to appropriate background contexts.

This list clarifies what a general philosophical theory of proof should do. Among other things, it should spell out the various standards by which proofs are evaluated, as well as the types of contextual information that are relevant to the evaluations. First and foremost, however, it should provide an understanding of "proof" that is robust enough to support such a study. The remainder of this essay takes some initial steps towards developing such an understanding.

The model of proof standardly used in mathematical logic today is that of formal axiomatic deduction.[15] This formal notion is supposed to provide an explication of the informal notion of proof, one that explains the virtue by which an informal proof is judged to be *correct*, as well as what it means for a theorem to be a *deductive consequence* of some assumptions. I take this theory to be one of the true mathematical and philosophical success stories of the late nineteenth and early twentieth centuries; it provides a solid basis for mathematical and philosophical theorizing, one that is more robust than anyone before Frege could expect. However, the theory of deduction was not designed to address the broader epistemological issues we are concerned with here, and, *in that respect*, we can identify ways in which the model falls short.

Consider, for example, the role of definitions in a proof. Our informal discussion called attention to the ways in which notions like divisibility, congruence, and norm aid our understanding of a proof. More extensive historical narratives support this point of view. In his book, *The emergence of the abstract group concept* [56], Hans Wussing traces the rise of the notion of a group in algebra, number theory, and geometry, in the nineteenth century. The text distinguishes between early, *implicit* uses of group-theoretic reasoning, to conscious, *explicit* uses of the group concept by the century's close. This strongly presupposes that there is an important difference between the former and the latter, that is, between considering particular instances of groups and using certain types of reasoning, and explicitly labeling the instances as such and identifying the patterns of reasoning in use. To support this type of analysis we need a model of proof that clearly distinguishes between the two.

In standard logic textbooks, however, definitions are usually treated outside the deductive framework; in other words, one views definienda as meta-theoretic names for the formulas they stand for, with the understanding that in the "real" formal proof it is actually the definientia that appear.[16] If one is working in the language of set theory, for example, occurrences of the group notion become buried in a haze of quantifiers, connectives, and epsilons; and it is hard to differentiate "explicit" uses of the notion from undistinguished appearances of the defining formula, or any of its logical equivalences.

Similarly, it is not clear how to analyze the role of contextual background knowledge in the standard logical model. Our discussion shows that proofs are evaluated not just with respect to a particular set of goals and values, but also with respect to a set of resources that are assumed to be generally available. From the point of view of axiomatic deduction, however, a proof is a self-contained warrant, whose correctness is judged solely in the context of the relevant axiomatic system.

In short, standard models of deduction currently used in mathematical logic cannot easily support the type of analysis we are after, for the simple reason that *they were not designed to.* Thus we need a model of proof that is better suited to the work we are now asking of it. In the next section, I will consider a model that rises better to the task.

3.2. The view from automated deduction

On the formal notion of deduction, a proof is (more or less) a sequence of assertions, each one of which is a principle of logic or a basic mathematical axiom, or which follows from previous assertions by a logical rule of inference. But proofs in an ordinary mathematical text don't look much like these formal derivations. For example, in a standard undergraduate textbook one often finds phrases like the following:

> "... the first law may be proved by induction on n."
>
> "... by successive applications of the definition, the associative law, the induction assumption, and the definition again."
>
> "By choice of m, $P(k)$ will be true for all $k < m$."
>
> "Hence, by the well-ordering postulate..."
>
> "From this formula it is clear that..."
>
> "This reduction can be repeated on b and r_1..."
>
> "This can be done by expressing the successive remainders r_i in terms of a and b..."
>
> "By the definition of a prime..."
>
> "On multiplying through by b..."
>
> "... by the second induction principle, we can assume $P(b)$ and $P(c)$ to be true..."
>
> "Continue this process until no primes are left on one side of the resulting equation..."
>
> "Collecting these occurrences, ..."
>
> "By definition, the hypothesis states that..."
>
> "... Theorem 10 allows us to conclude ..."

These examples are taken from actual proofs in Birkhoff and MacLane's *A Survey of Modern Algebra* [6]. In fact, they are all found in Chapter 1, which develops the basic properties of the integers needed in Section 2 above. What these snippets indicate is that "real" proofs often contain more elaborate instructions as to how one can "see" that an assertion follows from its predecessors. The usual story is that such proofs are simply higher-level, informal texts that indicate the existence of the lower-level formal ones; i.e. they are recipes, or descriptions, that provide enough information, in principle, for a fastidious formalizer to fill in every last detail.

The observation I would like to make here is that these two features of ordinary proofs – the informality, and the level of detail – are independent of one another. On one hand, one can imagine a tedious informal proof in which every inferential step is spelled out in complete detail. On the other hand, and more interesting for our purposes, it is also possible to imagine higher-level proofs that are nonetheless presented in a language that has been fully specified, so that the resulting proofs can be checked by purely mechanical procedures.

The evidence that it is possible to imagine such languages, proofs, and verification procedures is that, in fact, they exist. The last few decades have seen the advent of mechanized proof assistants, which are designed to facilitate the development of formally verified axiomatic proofs. These include systems like Mizar, HOL, Isabelle, PVS, Coq, NuPrl, and ACL2, and many others. (Some of these were designed with the goal of formalizing specifically mathematical theories, others with the goal of proving the correctness of various hardware and software specifications; and, whatever the origins, most of the systems have been adapted and developed to support both purposes.) Proof development is an interactive process between the user, who has some informal proof in mind, and the machine, which keeps the user painfully honest. Though systems incorporate different interface enhancements to help the user along, the final products are always "proof scripts." From the user's point of view, a proof script provides a (semi-)intelligible representation of the proof he or she had in mind; from the machine's point of view, the proof script provides explicit instructions for constructing a low-level axiomatic proof of the traditional sort.

Most proof assistants support a type of interaction based on *goal refinement*. First, one specifies the theorem that one intends to prove; this is tantamount to declaring a certain goal. Then one iteratively applies *methods* that reduce a current goal to others that are hopefully simpler. At any point in the process, the set of goals to be met constitute the *state*; when this set is empty, the theorem in question has been proved. The most basic types of methods are those that invoke a logical inference or a previously proved theorem, or expand a definition. More complex methods are built up from these. Correctness is guaranteed by the fact that ultimately the only way a complex method is allowed to modify the system's state is by applying the basic ones.

Using system known as Isabelle [58, 41], for example, one can construct proofs in a version of Church's higher-order logic. Goals are natural deduction sequents of the form $X_1, X_2, \ldots, X_n \Rightarrow Y$, representing the task of deriving Y from hypotheses $X_1, X_2, \ldots, X_n$.[17] The command **apply** *(rule andI)* applies the logical "and introduction" rule, which reduces a goal of the form

$$X_1, X_2, \ldots, X_n \Rightarrow Y \wedge Z$$

to the two subgoals

$$X_1, X_2, \ldots, X_n \Rightarrow Y$$
$$X_1, X_2, \ldots, X_n \Rightarrow Z.$$

Applying the command **apply** *(erule andE)* applies the logical "and elimination" rule, which reduces a goal of the form

$$X_1, X_2, \ldots, X_n, Y \wedge Z \Rightarrow W$$

to the subgoal

$$X_1, X_2, \ldots, X_n, Y, Z \Rightarrow W.$$

A branch of reductions is completed when one is reduced to a trivial subgoal of the form

$$X_1, X_2, \ldots, X_n, Y \Rightarrow Y,$$

which is finished off by the command **apply** *(assumption)*. Thus, even though one is always working backwards from goals, the sequent form allows one to reason both forwards from hypotheses and backwards from a conclusion.

In Isabelle, one may also apply more powerful, automated methods. For example, each of the commands on the following list is paired with an informal translation:

apply (*induct-tac x*)	"Use induction on x."
apply (*unfold Definition-2a*)	"Expand Definition 2a."
apply (*simp add*: *Equation-a*)	"Simplify, using Equation a."
apply (*auto add*: *Lemma-3*)	"Straightforward, using Lemma 3."
apply (*arith*)	"Use arithmetic reasoning."

To illustrate, the following proof script shows that for every integer x and y, and every natural number n, if $x|y$, then $x^n|y^n$.

```
theorem (a::int) dvd b ==> a^n dvd b^n
  apply (induct-tac n)
  apply (subst power-0)+
  apply (rule zdvd-1-left)
  apply (subst power-Suc)+
  apply (rule zdvd-zmult-mono)
  apply (assumption)+
  done
```

When writing proofs in Isabelle, one uses expressions like *a^n dvd b^n* in place of $a^n|b^n$. At the risk of causing some confusion, I will use Isabelle notation when displaying proof commands, but ordinary mathematical notation when describing the resulting states. The **theorem** command declares the initial goal,

$a|b \Rightarrow a^n|b^n$,

representing the theorem to be proved. In the command, the notation (*a::int*) specifies that the variable a is supposed to range over integers, whereas the fact that b ranges over integers as well is then inferred from the context. The first command declares that the proof is to proceed by induction on n, resulting in two subgoals:

$a|b \Rightarrow a^0|b^0$
$a|b, a^n|b^n \Rightarrow a^{n+1}|b^{n+1}$

The theorem *power-0* states that $x^0 = 1$ for any integer x, and so the next command repeatedly substitutes 1 for terms of the form x^0. This reduces the first goal to

$a|b \Rightarrow 1|1$.

This is polished off by the theorem *zdvd-1-left*, which asserts that $1|x$ for any integer x. The next command replaces a^{n+1} by $a \cdot a^n$ and b^{n+1} by $b \cdot b^n$ in the second goal. The theorem *zdvd-zmult-mono* asserts

$x|y, w|z \Rightarrow x \cdot w|y \cdot z$,

and applying it to the current goal yields two subgoals,

$a|b, a^n|b^n \Rightarrow a|b$
$a|b, a^n|b^n \Rightarrow a^n|b^n$.

Each of these is finished off simply by noting that the desired conclusion is one of the hypotheses.

You may reasonably object that the script above looks nothing like an ordinary proof. Even a seasoned Isabelle veteran will have a hard time determining the outcome of each instruction, without the computer's interactive responses or the kind of play-by-play account provided above. This is even more true when one uses more advanced methods, whose behaviors are complex and open-ended. Perusal of any math text quickly reveals the problem: the substance of an ordinary proof invariably lies in the *statements*, rather than the *instructions*. In other words, an ordinary proof is essentially a sequence of assertions; the instructions provide the minimum guidance needed for a competent reader to verify that each assertion follows from previous ones, but these instructions play a supporting role, and may be left out entirely when the appropriate justification can be inferred.

The good news is that that there are proof languages that are designed to capture this style of proof. Andrzej Trybulec's *Mizar* language [43] is an early and important example. More recently, Markus Wenzel ([55]; cf. also [54]) developed a similar proof language, called *Isar*, and implemented it in the Isabelle framework. Here is an *Isar* proof of the theorem above:

```
theorem (a::int) dvd b ==> a^n dvd b^n
proof -
  assume a dvd b
  show a^n dvd b^n
  proof (induct n)
    show a^0 dvd b^0
    proof -
      have a^0 = 1
        by (rule power-0)
      moreover have (1 dvd b^0)
        by (rule zdvd-1-left)
      ultimately show ?thesis
        by simp
    qed
  next
    fix n
    assume a^n dvd b^n
    show a ^ Suc n dvd b ^ Suc n
    proof -
      from prems have a * a^n dvd b * b^n
        by (intro zdvd-zmult-mono)
      moreover have a ^ Suc n = a * a^n
        by (rule power-Suc)
      moreover have b ^ Suc n = b * b^n
        by (rule power-Suc)
      ultimately show ?thesis
        by simp
    qed
  qed
qed
```

The proof proceeds by induction on n. The base case, and the inductive hypothesis, are established by appeal to the same theorems as before; the difference is that the outcomes are made explicit. (In the proof, the word *?thesis* refers to the claim being justified; thus, in the first instance, it refers to the base case $a^0|b^0$, and, in the second, it refers to the claim that $a^{n+1}|b^{n+1}$. The word *prems* refers to the local premises, that is, the assumptions that are in place in the current proof context.) With respect to readability, this presentation, although not perfect, is a step in the right direction. Concerning proof style, the Isar tutorial [40] advises:

> *Do not manipulate the proof state into a particular form by applying tactics but state the desired form explicitly and let the tactic verify from this form that the original goal follows.*

Here, "tactic" is just Isabelle terminology for "method." The tutorial continues to note that following the advice "yields more readable and also more robust proofs." This readability has a lot to do with the fact that the resulting formal proofs are closer to ordinary, informal ones.

In fact, Wenzel's implementation works by translating each proof written in the Isar language into a sequence of applications of Isabelle's methods. The philosophical advance embodied in this achievement is that ordinary mathematical texts can be understood in terms of the goal-refinement model. Wenzel's thesis and the Isar documentation show how this can be done; roughly, a simple finite state model does the bookkeeping, and various mathematical buzzwords ("hence," "thus," "have," "from," "with," "show," "moreover," and so on) provide flexible ways of specifying which hypotheses and assertions are relevant at each point in the text. It is surprising how far one can get with this model.

Incidentally, it turns out that the theorem that I have chosen as an example has a one line proof:

```
theorem (a::int) dvd b ==> a^n dvd b^n
  by (induct n, auto intro: zdvd-zmult-mono)
```

This translates, roughly, to the following informal proof: "Use induction on n. The verification is straightforward, using theorem *zdvd-zmult-mono*." Thus, here we are in the fortuitous situation where a proof that is easy for us is also easy for the system. Alas, all too often, this is not the case.

In any event, since its syntax is fully specified, the Isabelle/Isar proof language is an example of a formal language, and we have seen that it provides a higher-level characterization of a mathematical proof. Whether one chooses to make the assertions or the intermediate states more prominent, the two types of formal text rest on the same model of proof at the core: a proof is a specification of a sequence of methods of inference, each of which transforms (reduces) the epistemic requirements needed to verify that the purported theorem follows from the relevant axioms and definitions. It is important to emphasize that this characterization extends the standard notion of correctness in a conservative way: what makes a proof script valid is simply that it is an effective warrant for the kind of low-level axiomatic deduction that logicians know and love. From the point of view of correctness, then, the new notion is just an embellishment of the standard logical model. The hope, however, is that the higher-level formulation will better support an analysis of the broader evaluatory terms that we are concerned with here.

Consider the list of observations presented in Section 3.1. The first was that a proof should be understood as some kind of communicable text; proof scripts certainly have that character. The second was that proof can be evaluated with respect to different desiderata. Further work is necessary to determine whether or not the model can support the kinds of analysis we are interested in, but here, at least, we can begin to assess its prospects.

The third observation was that our model of proof should take lemmas and definitions seriously. Recall that one of our objections to the use of formal axiomatic deduction as a suitable model was that definitions are usually assumed to be expanded in the metatheory. In contrast, Isabelle *never* expands a definition, unless one explicitly instructs the system to do so. Doing otherwise would defeat the purpose of using definitions in the first place. Typically, we introduce a definition to avoid having to repeat the definiens; after establishing the relevant properties of the defined notion, we rely on these as far as possible.

Because definitions have a recognizable status in Isabelle, one can easily do an exhaustive search on theorems and rules in the current environment in which a certain definiendum occurs. Indeed, much of the Isabelle's development has focused on providing adequate support for the various types of definitions one wants to use, in practice. The issues are subtle, because definitions affect the behavior of automated methods by providing patterns that methods can match against: the mere occurrence of the token for "norm" or "group" can be used to trigger the invocation of lemmas and rules in a search procedure, or to instruct the simplifier to express appropriate terms in canonical forms that are justified by the group axioms or properties of norms. Thus, tokens like "norm" and "group" may have a number of associations in the system, including their definitions in particular instances; theorems, lemmas, and rules in which they occur, which can be made available to the general automated reasoners and term simplifiers in various ways; more specialized automated methods that are designed to act on states in which these tokens are found; implicit or explicit definitions with respect to general algebraic structures, of which particular definitions are instances; and so on. Thus the mechanisms for handling definitions have a tremendous effect on the system's ability to construct proofs simply and automatically. It would be surprising if the study of these "pragmatic" mechanisms were to have to have no positive effects on the development of a more robust epistemology of mathematics.[18]

The final observation was that we would like to be able to evaluate individual proofs against a suitable background context, whereas, in contrast, traditional axiomatic proofs stand alone. Mechanized proof assistants, however, distinguish between the underlying axiomatic system, standard libraries of theorems, specialized libraries of theorems, lower- and higher-level methods, as well as general and more specialized methods. A particular proof script can therefore only be understood and evaluated with respect to the more general resources available to the system. (The fact that libraries and automated methods can change while the proof assistant is under development is the constant bane of formalizers.) Thus, careful attention to the practice of automated deduction should help us in our conceptual analysis by providing us, at least, with concrete, working models of mathematical context.

3.3. A case study, revisited

The attention we have given to mechanized proof assistants suggests a certain methodology for getting at the methods that are implicit in an informally presented proof. Start by translating the proof into a formal proof language, as straightforwardly as possible, with the goal of verifying it mechanically. You will find that the steps in an ordinary proof are too large, and the instructions too vague; the computer needs more information in order to be able to fill in the gaps. Making this information explicit will require you to reflect carefully on the theorems, simplification rules, and other mechanisms by which you are able to recognize the ordinary proof as valid. Once you spell out all the details, then, you will have before you a formal representation of the background context and methods that are needed to make the original proof intelligible.

This, reflexively, provides you with a partial explanation of why these methods are valuable: they make it possible to read the proof at hand. The case is bolstered considerably when the same methods are shown to be more generally useful. Thus, for example, a single proof may help one initially uncover certain mechanisms for reasoning about groups or norms; a comparative analysis will help show how these mechanisms function more widely.

Of course, our discussion shows that there are other virtues we would like to ascribe to methods. We would like to understand what it means to say that a method shows how a related problem can be solved, or how to obtain an algorithm; or, for example, what it means for a method to be "generalizable" to other contexts. The type of analysis just described provides a starting point, by providing concrete representations of the "methods" in question. This lays a foundation, on which we can begin to build a more elaborate theory.

Let us consider, as an example, the product law for sums of squares that we studied in Section 2.2. The first proof has a short Isabelle formalization:

```
theorem EX (x::int) y. (a^2 + b^2) * (c^2 + d^2) = x^2 + y^2
proof (rule exI)+
  show (a^2 + b^2) * (c^2 + d^2) = (a * c − b * d)^2 +
    (a * d + b * c)^2
    by (simp add: zadd-zmult-distrib zadd-zmult-distrib2
       zdiff-zmult-distrib  zdiff-zmult-distrib2 power2-eq-square)
  qed
```

In other words, we simply provide explicit terms that express the product of $(a^2 + b^2)$ and $(c^2 + d^2)$ as sums of squares. Verifying that these terms do the job is a straightforward calculation using basic properties of integers, the distributivity of multiplication over addition and subtraction in particular.

Analyzing our second proof, the one that uses the concept of the norm of a Gaussian integer, requires a good deal more work. The appendix to this essay contains an Isabelle development of the theory of Gaussian integers that is just barely sufficient to prove the theorem at hand. (It is modeled after Jacques Fleuriot's development of the complex numbers, which is included in the Isabelle 2004 distribution.) The reader need not be concerned with the specific mechanisms invoked to define the Gaussian integers as new objects; the net effect is that we then have variables ranging over Gaussian integers, a function *gauss*(*a*,*b*) that turns a pair of integers a and b into the Gaussian integer $a + bi$, as well as functions *gauss-re*(*x*) and *gauss-im*(*x*) that return the real and imaginary parts of a Gaussian integer, respectively. These satisfy the expected identities:

```
  lemma [simp]: gauss-re(gauss(a,b)) = a
  lemma [simp]: gauss-im(gauss(a,b)) = b
  lemma gauss-gauss-re-im-conv [simp]:
     gauss(gauss-re(z),gauss-im(z)) = z
  lemma gauss-gauss-eq [simp]:
     (gauss(a,b) = gauss(c,d)) = (a = c & b = d)
```

The annotation [*simp*] in the statements of these theorems tells the system that these equalities should always be applied in the left-to-right direction when simplifying terms. The last lemma is an important boolean identity, telling the system that demonstrating the equality of two Gaussian integers amounts to proving the identity of the real and imaginary components; when we are presented with Gaussian integers in these terms, we should always simplify the former task to the latter. Our ability to do so depends on the fact that *gauss* is injective, which is to say, our representations are unique.

The number of preparatory lemmas may seem daunting, but, at least, most of the proofs are one-liners. Recall that the methods "auto" and "simp" are essentially Isabelle's way of saying "obvious," at least given the background resources and those explicitly provided. So, for example, the lemmas

```
  lemma gauss-mult-commute [simp]: (w::gauss-int) * z = z * w
  lemma gauss-mult-assoc [simp]: ((u::gauss-int) * v) * w = u * (v * w)
```

establish that the multiplication we have defined for Gaussian integers is commutative and associative. These identities are made available to the automated simplifier. The proofs are entirely straightforward, given the definition of multiplication for Gaussian integers, and the relevant properties of multiplication and addition for ordinary integers. The lemma

```
lemma gauss-conj-mult: gauss-conj(w) * gauss-conj(z) =
  gauss-conj(w * z)
```

shows that conjugation is multiplicative; again, this is easy, given the definitions of conjugation and multiplication. The Lemma *gauss-norm-conj* establishes the relationship between the norm and the conjugate. The lemma

```
lemma gauss-norm-mult [simp]: gauss-norm(x) * gauss-norm(y) =
    gauss-norm(x * y)
```

asserts that the norm is multiplicative, a fact that follows easily from the two lemmas *gauss-norm-conj* and *gauss-conj-mult*.

The final lemma and theorem provide the dénouement. The lemma shows that every sum of squares is the norm of a Gaussian integer:

```
lemma sum-squares-eq-norm-gauss: a^2 + b^2 = gauss-norm(gauss(a,b))
  by (simp add: gauss-norm-def)
```

This is immediate, given the definition of norm. The final theorem is our Theorem 2.2:

```
theorem EX (x::int) y. (a^2 + b^2) * (c^2 + d^2) = x^2 + y^2
  by (auto simp add: sum-squares-eq-norm-gauss)
```

In other words, the theorem is obvious, given the preceeding lemma.

Despite the length, this example is misleadingly simple; the fact that almost every lemma has a one-line proof shows that in this case most of the verification can be reduced to unwinding definitions, simplifying terms, and using basic logical inferences. Generally speaking, automated deduction begins to get hard (and therefore interesting) when this is not the case. At one end of the spectrum of mathematical activity, there is routine calculation and verification, where the appropriate means of proceeding is clear and straightforward; at the other, there is blind search and divine inspiration. Mathematical methods are designed to shift as much as possible to the first side, so that serious thought and hard work can be reserved for tasks that are truly difficult. The project proposed here is to better understand how they do this.

Even in our simple example, however, interesting phenomena emerge. For example, consider the following lemma:

```
lemma gauss-gauss-ex-intro [intro]: EX z. P(z) ==>
  EX x y. P(gauss(x,y))
```

This tells us that to prove that a property P holds of $x + iy$ for some x and y, it suffices to prove that P holds of some Gaussian integer. This inference is declared suitable for use by the automated reasoners. We carry out steps like this implicitly when reasoning about Gaussian integers, and it may be hard to believe that a proof assistant has to be told, explicitly, to do the same. But, in general, working backwards using rules like this does not always preserve validity; that is does in this case is guaranteed by the fact that *gauss* is surjective. Thus, our formalization, in getting us to uncover the principles of reasoning that should be automatic, at the same time forces us to identify the features of our domain that are basic to reasoning about it.

Note that in addition to associativity and commutativity, we also provide the simplifier with a lemma that embodies a funny combination of the two:

lemma *gauss-mult-left-commute* [*simp*]:
  $(u{::}\mathit{gauss\text{-}int}) * (v * w) = v * (u * w)$

There is a good reason for this. Clearly there is a problem with declaring a term like $a + b$, in general, to be a simplification of $b + a$: iterated application can leave the system "simplifying" ad infinitum. Isabelle's simplifier is smart enough to recognize such "permutative conversion rules," and will apply them only in cases where doing so results in a reduction with respect to a somewhat arbitrary ordering of terms. But this renders associativity and commutativity too weak. Adding left commutativity for addition to the mix has the net effect that a nested sum of terms is rewritten so that the terms appear in a fixed order, with parentheses grouped to the right; this convention makes it possible to match such terms. The epistemological moral is that a proper *understanding* of the arithmetic operations requires not just knowing *that* they satisfy associativity and commutativity, but also knowing *how* to make use of this fact. In particular, a certain faculty is required to ignore parenthetical groupings in iterative applications of an associative operation, and to recognize that sums like $a + b + c$ and $c + a + b$ are equal. With complex expressions, we are apt to do this by ticking off terms; in any event, it is a capability that is available to any practicing mathematician, and one that is usually carried out without comment or fanfare. It is only the discipline of formalization that brings this to the fore.

Another subtlety that emerges has to do with the handling of integers as a subdomain of the complex numbers. In Section 2.2, I noted that if z is any Gaussian integer, then $z\overline{z}$ is an integer. This is not exactly true; $z\overline{z}$ is really a *Gaussian* integer whose imaginary part happens to be 0. The statement only becomes true when one takes these Gaussian integers to be identified with their ordinary integer counterparts. If one views the integers as a subset of the Gaussian integers, one has to recognize that this subset is closed under the operations of addition and multiplication; if, alternatively, one views the integers as embedded in the Gaussian integers via the mapping $x \mapsto x + 0i$, one needs to recognize that this function respects the arithmetic operations. In the formalization in the appendix, the predicate *gauss-IsInt* holds of the Gaussian integers that have imaginary part 0. Then, the two lemmas

lemma *gauss-mult-int* [*simp*]: *gauss-IsInt* x ==> *gauss-IsInt* y ==>
gauss-IsInt $(x * y)$
lemma *gauss-mult-int-eq* [*simp*]: *gauss-IsInt* x ==> *gauss-IsInt* y ==>
gauss-re x * *gauss-re* y = *gauss-re*$(x * y)$

show, first, that the collection of Gaussian integers with this property is closed under multiplication; and, second, that the effect of such a multiplication is simply to multiply the real parts. Thus, once again, the act of formalization forces us to articulate a pattern of reasoning that typically passes unnoticed.

Other rules that are declared as "simplifications" in the appendix merit further consideration. For example, the final proof requires the fact that our statement of *gauss-norm-mult* declares the term *gauss-norm*$(x * y)$ to be a simplification of *gauss-norm*(x) * *gauss-norm*(y). But, in general, is this a good thing? Further formalization efforts may show that it is not always desirable to have automated methods apply this rewriting strategy. We may then choose to remove the declaration, in which case the identity has to be added explicitly to the list of resources in our formal proof of Theorem 2.2. To make matters worse, we may decide that the reverse direction constitutes a better default simplication,[19] in which case we would have to explicitly tell the simplifier to treat this case as an exception.

What this shows is that calculation becomes more complex when one is forced to use identities in which there is no clearly preferred direction for rewriting terms. The distributivity laws for multiplication over addition and subtraction are examples of such identities: sometimes one wants to multiply a term through a sum or difference, whereas at other times it is desirable to factor a term outside an expression. Having to specify the appropriate means of proceeding at each stage can be tedious; the alternative is to train the automated methods to pick up contextual clues, as we do, to determine what types of rewriting are likely to be fruitful in specific instances.

Finally, it is interesting to note that there is some redundancy in our formalization. One of our simplification rules,

lemma *gauss-mult-gauss* [*simp*]: *gauss*(a,b) * *gauss*(c,d) =
gauss$(a * c - b * d,\ a * d + b * c)$

is simply the multiplication rule for Gaussian integers. If we remove the declaration to the simplifier, every proof in the appendix still goes through, unchanged. The same is true if we remove the rule *gauss-gauss-ex-intro*, discussed above. But if we remove *both* of these, the final proof fails. What happens is that the system gets stuck trying to find appropriate terms s and t satisfying

$$gauss\text{-}norm(gauss(a,b)\ *\ gauss(c,d)) = gauss\text{-}norm(gauss(s,t)).$$

In other words, the last theorem is obvious only if we employ *either* the multiplication rule for Gaussian integers, *or* a faculty to recognize that the specific terms are irrelevant in this case. Determining which of these strategies is more natural or more useful, in this specific case or more generally, is no easy matter.

The fact that so many subtle issues emerge from such a simple example suggests that there is a wealth of insight to be harvested from even slightly more complex examples. At present, the faculties by which we navigate even the most familiar mathematical terrain are far from clear. This fact can be expressed in the form of a slogan: *what is obvious is not at all obvious.*

At any rate, the discussion up to this point has been intended to show that mechanically assisted formalization can help us detect the various methods of inference that are needed to make an ordinary mathematical proof intelligible. The reflections in Section 3.1 provide a sense of some of the criteria by which such methods may be evaluated; the next step is to formulate these criteria more precisely. While I will not begin to undertake this broader project here, let me briefly indicate two directions in which formal work with Isabelle may again provide some insight.

The notion of *generality* of method was a constant theme in the discussion in Section 2. Isabelle supports the notion of an *axiomatic type class*, that is, an axiomatic characterization of a class of structures, of which particular domains may be shown to be instances. In Section 2, we noted that the first proof of Theorem 2.2 works, more generally, for any commutative ring. In fact, there are Isabelle formalizations of the notion of a commutative ring, and our formalization of the first proof works equally well for such an axiomatic class, provided we cite the more general distributivity laws. Similarly, we noted in Section 2 that the notion of "norm" makes sense for more general classes of structures. With more work, we can axiomatize, for example, the relevant properties of finite field extensions of the rational numbers, and show that the Gaussian integers are a particular instance; most of the theorems in the appendix can then be proved in the more general framework. Thus, one can show how a more general body of methods can be used to support a formal proof of Theorem 2.2. This falls short of characterizing the sense in which the specific methods associated to Gaussian integers are *generalizable*; that is, it does not characterize the higher-level methods (or heuristics) one can use to obtain appropriate generalizations. But it does provide a clear sense in which methods developed to reason about the Gaussian integers are *instances* of more general ones.

Also discussed in Section 2 was the notion that some methods show us how an associated problem was solved. Isabelle was not designed to solve mathematical problems, other than the problem of finding a proof. But there are ways we can begin to creep up on such issues. For example, in Isabelle's interactive mode, one may specify a theorem with metamathematical parameters that are to be instantiated. Writing

theorem $(a\hat{}2 + b\hat{}2) * (c\hat{}2 + d\hat{}2) = ?x\hat{}2 + ?y\hat{}2$

declares the goal of finding *terms* to substitute for $?x$ and $?y$ and a proof of the resulting theorem. In our example, if one issues the *same proof script before*, the system happily reports that a theorem has successfully been proved; $?x$ and $?y$ are instantiated to $ac - bd$ and $ad + bc$, respectively. Thus, we have a precise sense in which these methods provide additional information, beyond the fact that the statement of the theorem is true; that is, they show us how to find specific witnesses for x and y.

4. Towards a general theory

Adapting a system like Isabelle for use in our project of understanding value judgments that are applied to proofs involves an awkward type mismatch, in that we are using a *specific* implementation of a proof language to address *general* questions about the nature of proofs. In our analysis, we are not so much concerned with the fact that certain definitions, theorems, and methods *in Isabelle* make it possible for *that particular system* to verify a proof script; but, rather, that a certain body of definitions, theorems, and methods *in mathematics* make it possible for a *mathematical cognizer* to understand a certain proof. Thus, we need to develop a way of speaking about methods and proofs at a level of abstraction that strips away whatever it is we take to be ad-hoc and specific to a certain implementation. At the same time, such a framework has to be concrete enough to support a rigorous analysis.

Some essential features of our framework can easily be discerned. First of all, we need an appropriate notion of a *proof state*, which characterizes, among other things, the locally-available knowledge and the immediate subgoals at each stage of a proof in progress. I have implicitly assumed that such a state can be represented syntactically, which is to say, it can be stored, communicated, and acted upon by a computational agent. The second essential component of the account is that of a *method*, that is, an algorithmic procedure which acts on a proof state and transforms it into another. It is these methods that are to be the basic objects of evaluation.

At this level of generality, however, the framework is unlikely to be useful. For example, if we take the initial proof state to be simply a statement of the theorem to be proved, there is a single method that always succeeds in finding a proof if there is one: blind, systematic search. Of course, in practice, this is a lousy way to proceed. Instead of methods that are generally foolproof but impractical, we seek methods that are effective in particular contexts. Characterizing such methods will require more nuanced ways of describing both proof states and the algorithms that act upon them. But then we are pushed back to the problem of overspecificity: what more can we do beyond choosing a particular representation of proof states, and a particular "programming language" for methods?

Here I am encouraged by historical precedent. Before the nineteenth century it may have seemed unlikely that any neat theory could account for the correctness of the bewildering range of styles and methods of mathematical argumentation. Now, a couple of hundred years later, the modern theory of mathematical proof provides just such a theory. Achieving our modern understanding required both philosophical and mathematical reflection, as well as a good deal of mucking around, and it was a long time before the outlines of a robust and stable theory began to emerge. Eventually, conceptual pieces began to fall into place, terminology and notation began to stabilize, important deductive systems like first-order logic and higher-order logic were isolated, semantic notions were clarified, and interesting axiomatic systems like set theory and arithmetic were identified. It seems to me unlikely that we can obtain a similarly robust theory of proof, of the kind described above, without reconciling ourselves to a period of untidy exploration.

Pushing the analogy may be fruitful. One of the factors that contributed to the identification of first-order logic as an important fragment of reasoning was its characterization in non-syntactic terms. Deductive systems for first-order logic vary widely in choice of primitives, axioms, and rules; what they all have in common is that they give rise to a notion of consequence that is sound and complete for first-order semantics. Analogously, we can ask: can methods of inference be fruitfully characterized in more "semantic" or algebraic terms?

The development of an appropriate framework has to go hand in hand with initial attempts to answer the types of questions that the framework is supposed to address. Here are some:

1. What methods of inference are required to understand proof X?

2. What are the methods of inference that are used in the branch of mathematics X?

3. Are there useful and informative ways of characterizing and classifying methods?

4. What are the *types* of methods that are used in the branch of mathematics X?

5. What are the types of methods that are used in mathematics *simpliciter*?

6. To what extent do methods vary across the branches of mathematics?

7. How do methods from the different branches interact?

8. In what contexts is the collection of methods X useful?

9. What are the methodological / epistemic benefits of methods X?

10. What are the methodological / epistemic benefits of methods of type X?

Here I am using “method” to refer specifically to the kind of low-level methods of inference we have been discussing, so this list does not even begin to address broader issues related to problem solving, generalization, and the like. It is common in mathematics to classify various methods as algebraic, analytic, combinatorial, geometric, and so on, and one might hope to shed light on such a taxonomy. Aside from logical and philosophical interest, this could also raise interesting mathematical questions; for example, it could provide a clear sense to the question as to whether a particular result can be obtained by certain methods.

As noted in the introduction, the framework I have proposed is based on a distinctly syntactic view of mathematical practice. A benefit is that the philosophical analysis does not presuppose or depend on any substantial portion of this practice; all that is needed at the core is a theory of syntactic entities and computational procedures. On the other hand, from a naturalist perspective, it would be perfectly legitimate to bring the full weight of our contemporary mathematical understanding to bear. One may therefore wonder whether a more semantic framework would be more appropriate. For example, a referee suggests that the "higher-order" methods, like generalization, alluded to in the introduction are better understood in terms of *analogies* between semantic objects.

I admit to a bias against such approaches. Put crudely, I doubt that accounting for the utility of the notion of a group in terms of references to actual groups will have much explanatory value. Furthermore, I expect that insights from a semantic account can easily be translated into syntactic terms: simply speak of "uses of the term 'group'" rather than "references to groups." But, to be fair, this misses the point of the referee's objection: what is at issue is the most natural level of description rather than inter-translatability, and the approach I have suggested may simply miss the conceptual forest for the syntactic trees.

I see, however, no reason that different perspectives should not be developed in parallel. I think it likely that they will converge in the limit, and that there is much to be gained by understanding the relationships between them. Ultimately, only time will tell which perspectives yield the most insight. In the meanwhile, you pays your money, and takes your chances.

My claim in the introduction that a good theory of proof will help explain the ways in which certain methods of inference render a proof intelligible may suggest that the program I am proposing has a psychologistic component, aiming to clarify human cognition. Indeed, it may well be the case that the kind of theory I am after can inform such an empirical study, and can, in turn, benefit from the results. Similarly, I expect it can be informed by historical and contemporary mathematical case studies, and can, in turn help us understand these cases. I hope my discussion also suggests that a good theory can be informed, and can serve to inform, research in automated deduction; and that it can benefit from an appropriate *mathematical* understanding, and provide specifically mathematical insights.

That said, let me make it clear that the type of theory I am after is neither psychological nor historical in nature. By that, I only mean to say that I believe it possible to develop a general epistemological framework for characterizing mathematical methods and goals in terms that are independent of these disciplines. The approach I have described has a Kantian transcendental flavor: taking, as a starting point, the fact that ordinary mathematical proofs *are* intelligible, the challenge is to characterize the cognitive mechanisms that make them so. It also has a phenomenological feel: what must be accounted for is not the nature of mathematical objects in and of themselves, but, rather, our representations of these objects, and the way we interact with these representations in our mathematical experience.

How should we gauge the success of such a theory? Of course, by the usual philosophical standards: its internal coherence and consistency, the extent to which it accords with intuition, and the extent to which it provides a useful conceptual apparatus for those disciplines that touch upon such epistemological issues. It is not clear to me whether there is anything else one has a right to expect from the philosophy of mathematics; in any event, these goals are certainly enough to justify the effort.

Acknowledgements

This paper is partially the result of a seminar on mathematical structuralism that I taught jointly with Ken Manders in Fall 2002, and I am grateful to Manders and the seminar participants for their input. I read parts of an early draft at an informal workshop organized by Manders in July 2003, and parts of a version very close to this one both at a workshop organized by the University of Irvine's Department of Logic and Philosophy of Science in March of 2004 and at the Chicago meeting of the American Philosophical Association in April of 2004. I am grateful to Andrew Arana, Clemens Ballerin, Matthew Frank, Jukka Keranen, Erica Lucast, Paolo Mancosu, Douglas Marshall, Tobias Nipkow, Penelope Maddy, John Stillwell, William Tait, an anonymous referee, and others for comments, suggestions, and encouragement.

Notes

[1] Many have raised issues like these, and I am not claiming originality or priority in that respect. In particular, Manders has long been emphasizing the need for more general theories of mathematical *understanding*, which would presumably address questions like the ones I raise here. But if I were to try to attribute to him a particular way of framing the issues, I would run the risk of mischaracterizing his views; so, instead, this note will have to suffice to acknowledge his general influence.

[2] A list, with references, can be found in Lemmermeyer [36, Appendix B].

[3] John Stillwell has suggested to me that it would be fruitful to consider various proofs of the fundamental theorem of arithmetic and the Pythagorean theorem in the same vein.

[4] In my view, Steiner [50], in particular, does not sufficiently acknowledge this. His analysis proceeds by comparing multiple proofs of sample theorems, and noting positive and negative features of the various proofs; I often find myself in disagreement only at the point where he judges a particular proof to be the most explanatory *simpliciter*. So, here I will strive to provide a framework in which one can clarify such evaluatory claims, without trying to provide a single uniform measure.

After circulating a draft of this paper, I received a copy of [30], which adopts a similar attitude, and urges a "bottom-up" methodology similar to the one I follow here. It also provides a forceful criticism of the conclusions in [50].

[5] This is essentially the one given by Coxeter [15, p. 27], who credits M. Kraitchick, and, later but independently, J. E. Hoffmann. It can also be found in Hardy and Wright [31], which cites (the first edition of) Coxeter [15], but credits Kraitchick and Bennett. A presentation in terms of congruences can be found in Baaz [3], which, however, mistakenly attributes the proof to Euler.

[6] Dana Scott used this example, in discussing the notion of mathematical proof, in a colloquium he gave at Carnegie Mellon, in the spring of 2002.

[7] There is a subtle interplay between such local and global considerations, that is, between valuing lemmas and definitions for their ability to help us understand a particular proof, and for their utility in more general contexts. In commenting on this paper, William Tait has emphasized that one should be careful not to devalue the former in favor of the latter. For example, even though Dedekind's notion of an ideal is now ubiquitous in commutative algebra, Dedekind was clearly pleased with its role in the develoment of the unique factorization theorem for algebraic integers, *before* its more global utility was established.

[8] that is, in which the terms used to express the product are real bilinear forms in the values that are squared and summed in the factors.

[9] See e.g. [21, Section 1.7], [28, Section 12.6], or [31, §366].

[10] One way to prove this is to appeal to Fermat's little theorem, which asserts that if p does not divide x, $x^{p-1} \equiv 1 \pmod{p}$. In particular, if p is of the form $4n+1$, each of the numbers $1, 2, \ldots, 4n$ satisfies the equation $x^{4n} \equiv 1 \pmod{p}$, and hence $x^{4n} - 1 \equiv (x^{2n}+1)(x^{2n}-1) \equiv 0 \pmod{p}$. By Lagrange's theorem, the polynomial $x^{2n} - 1$ has at most $2n$ roots modulo p; thus the remaining $2n$ numbers between 1 and $4n$ satisfy $x^{2n} + 1 \equiv 0 \pmod{p}$, that is, $(x^n)^2 \equiv -1 \pmod{p}$.

Another way to prove Lemma 2.5 is to appeal to Wilson's theorem, which asserts that $(p-1)! \equiv 0 \pmod{p}$ when p is prime. When $p - 1 = 4n$, note that the numbers $2n+1, 2n+2, \ldots, 4n-1, 4n$ are congruent, respectively, to $-2n, -2n-1, ..., -2, -1$ modulo p, which implies $(2n+1)(2n+2)\cdots(4n-1)(4n) \equiv (-1)^{2n}(2n)! \equiv (2n)! \pmod{p}$. Appealing to Wilson's theorem, we have $(4n)! \equiv (2n!)^2 \equiv -1 \pmod{p}$, so we can let $m = (2n!)$ in the statement of the lemma. This proof was given by Lagrange in 1771.

A third way to proceed is to first prove the weaker statement that there are relatively prime x and y such that $p|x^2+y^2$ by iteratively applying a differences operator to the sequence $1^{2n}, 2^{2n}, \ldots, 4^{4n}$, as did Euler in 1749 (cf. the next footnote). The desired conclusion follows from the fact that y has a multiplicative inverse modulo p.

A very direct and elegant fourth proof can be found in [1]: partition the $p-1$ nonzero residue classes modulo p into sets of the form $\{x, -x, x^{-1}, -x^{-1}\}$, where $-x$ and x^{-1} denote, respectively, the additive and multiplicative inverses of x modulo p. Most of these sets have four elements, but some collapsing can occur. When p is an odd prime, x and $-x$ are always distinct; but $x = x^{-1}$ exactly when $x^2 = 1$, i.e $x = \pm 1$, resulting in one set with two elements. When $p-1$ is a multiple of 4, there has to be exactly one other two-element set: this occurs when $x = -x^{-1}$, which is equivalent to the assertion that $x^2 \equiv 1 \pmod{p}$.

Some textbooks use the fact that the group of $4n$ nonzero residues modulo p is cyclic, and hence has an element m of order 4. But every proof of this fact that I know of (i.e. that there is a primitive element modulo any prime) uses Lagrange's theorem, and constitutes essentially a strengthened form of the second argument above.

Proofs of Fermat's theorem, Lagrange's theorem, and Wilson's theorem can be found in almost any elementary textbook on number theory; see for example [31, 42].

[11] Euler's proof assumed a slightly weaker form of Lemma 2.5, which he was unable to prove until 1749. See the discussions in [21, 44, 53], and also footnote 4.

[12] This proof is commonly found in textbooks today, but I do not know who first discovered it. It was certainly accessible to Gauss, who proved Lemma 2.8 in 1828. Dickson [17, II, p. 233] notes that L. Wantzel states in a paper of 1848 that the use of Lemma 2.8 provides the simplest proof of the fact that every prime divisor of a sum of two squares is again a sum of two squares.

[13] I can't resist including one last proof, adapted from [42], which is very elementary and direct. Using Lemma 2.5, start with an integer m such that $m^2 \equiv 1 \pmod{p}$. First, I claim that there is a solution to $x \equiv my \pmod{p}$, with $0 < |x|, |y| < \sqrt{p}$. To see this, consider the values $u - mv$ for all pairs u, v satisfying $0 \leq u, v < \sqrt{p}$. Since there $(1 + \lfloor\sqrt{p}\rfloor)^2 \geq p$ such pairs, there are two distinct pairs u_0, v_0 and u_1, v_1 such that $u_0 - mv_0 \equiv u_1 - mv_1 \pmod{p}$. Let $x = u_0 - u_1$, and let $y = v_0 - v_1$. At least one of these is nonzero since the pairs are distinct, and so they satisfy the requirements of the claim.

Now note that we have $x^2 \equiv m^2y^2 \equiv -y^2 \pmod{p}$, so that $p|x^2 + y^2$. Since $0 < x^2 + y^2 < 2p$, the only possibility is that $x^2 + y^2 = p$.

[14] This tension was a focal point of foundational debate in the late nineteenth century, and even though modern mathematics embraces a full range of viewpoints, from explicitly computational to resolutely abstract, such differences can still incite passion in serious practitioners of the subject. Although an adequate treatment of the topic is well beyond the scope of this essay, a few clarificatory words are in order.

From the bias of a modern, set-theoretic, point of view, we can distinguish between statements that are constructively valid, and statements that are true but not constructively valid. A statement is "constructively valid" if it remains true on a computational reading of its quantifiers; for example, a constructive reading of a theorem of the form "for every x there is a y such that ..." should provide an algorithm for producing a y from any given x. A simple instance of a statement that, on the modern view, is true but not constructively true is the following: "for every Turing machine x, there is a number y, such that x halts in y steps when started on empty input, if it halts at all."

It can happen that a nonconstructive proof can have a constructively valid conclusion. Saying that a proof is nonconstructive means that it relies on theorems that are not constructively valid, or, more generally, on methods of reasoning, like proof by contradiction and the law of the excluded middle, that do not in general guarantee constructive validity. Indeed, nonconstructive methods are often praised for making it possible to obtain even explicitly computational results more easily.

Mathematical logic and the theory of computability provide a clear sense in which a statement can be constructively valid, or not. There are also various characterizations of constructively valid methods of proof; see, for example, [4, 52, 7]. Proof theory, in an extended version of Hilbert's program, provides many ways in which nonconsructive theories can be interpreted in constructive terms; many of the articles in *The Handbook of Proof Theory* [9] can be interpreted in this light (see also, e.g., [2]). Recent work in logic has even focused on ways of extracting useful constructive information from nonconstructive proofs in practice; see, for example, [34, 5].

The discussion in Section 2 shows, however, that even when a proof is constructive, we may pass judgment as to whether it makes computational information explicit or not; or that we can declare that even though a proof is nonconstructive, an associated algorithm is "easily obtained." In other words, in ordinary practice, there are degrees of salience and more subtle ways in which constructive information can be transmitted. These more nuanced distinctions are not captured well by the standard logical models, but these types of judgments *a*re of interest here.

[15] For concreteness, we can fix on the notion of computably-axiomatized theories in many-sorted classical first-order logic, and, in fact, little will be lost if we focus on theories axiomatized by finitely many schemata. Note that such theories include deductive systems for higher-order logic, which can be expressed in such a many-sorted first-order framework, as well as axiomatic set theory. From our perspective it matters little whether one prefers an axiomatic, natural deduction, or sequent formulation, since these are easily and efficiently intertranslatable. Anyone who wishes to include intuitionistic or type-theoretic foundational frameworks as well is welcome to do so; all I am assuming is that the systems we consider are syntactically specified, in such a way that there are effective procedures for verifying the well-formedness of assertions and validity of inferences.

[16] To be sure, logicians also sometimes use the notion of a *definitional extension* of a theory, in which one extends the language of the original theory with new function and relation symbols, together with their defining axioms. But outside the field of proof complexity (where one is interested in the effects of such extensions on lengths of proofs), the notion does not play an important role in the subsequent development of the theory; which is to say, after the first chapter it is rare that any assertion in a logic textbook depends on whether one is thinking in terms of definitional extensions or definitions in the metatheory.

[17] I am simplifying somewhat. Isabelle's goals are actually higher-order sequents, which is to say, hypotheses can themselves be sequents of the form $W_1, W_2, \ldots, W_m \Rightarrow Z$; and one is allowed to use variables and universal quantifiers ranging over arbitrary terms in higher-order logic. For details, see [41], as well as the other documentation at [58].

[18] For example, a token with a suitable list of associations may be able to stand duty for the notion of a "mathematical concept." It could help explain, e.g., how it is that we can sometimes identify an implicit historical use of a concept, before a precise definition is in place; how a concept can be instantiated in different foundational frameworks; or how mathematical concepts can change over time, and yet preserve some of the same meaning. I am not yet convinced, however, that for our purposes talk of concepts has any benefits over more direct talk of methods and the tokens they detect; so, for the time being, I will stick with the latter.

[19] In fact, Tobias Nipkow tells me that this would be his initial impulse. Isabelle 2004's standard HOL library does not currently declare either simplication rule for the absolute value function on ordered rings, and more thought and experimentation is needed to determine whether it should.

References

1. Aigner, M. and G. M. Ziegler: 2001, *Proofs from The Book.* Berlin: Springer-Verlag, second edition.

2. Avigad, J.: 2000, 'Interpreting classical theories in constructive ones'. *J. Symbolic Logic* **65**(4), 1785–1812.
3. Baaz, M.: 1999, 'Note on the generalization of calculations'. *Theoretical Computer Science* **224**, 3–11.
4. Beeson, M. J.: 1985, *Foundations of Constructive Mathematics.* Berlin: Springer-Verlag.
5. Berger, U., H. Schwichtenberg, and M. Seisenberger: 2001, 'The Warshall algorithm and Dickson's Lemma: two examples of realistic program extraction'. *Journal of Automated Reasoning* **26**, 205–221.
6. Birkhoff, G. and S. Mac Lane: 1965, *A survey of modern algebra*, Third edition. New York: The Macmillan Co.
7. Bridges, D. and S. Reeves: 1999, 'Constructive mathematics in theory and programming practice'. *Philosophia Mathematica* **7**, 65–104.
8. Bundy, A.: 1988, 'The use of explicit plans to guide inductive proofs'. In: E. Lusk and R. Overbeek (eds.): *9th International Conference on Automated Deduction*, Vol. 310 of *Lecture Notes in Computer Science*. Berlin, pp. 111–120.
9. Buss, S. R. (ed.): 1998, *The Handbook of Proof Theory.* Amsterdam: North-Holland.
10. Cauchy, A.-L.: 1821, *Cours d'analyse de l'École Royale Polytechnique. Première partie: Analyse algébrique.* Paris:. Reprinted in Cauchy's *Ouvres completes*, Gauthier-Villars, Paris, 1882-1919, deuxième série, vol. 3.
11. Colton, S., A. Bundy, and T. Walsh: 1999, 'Automatic Concept Formation in Pure Mathematics'. In: T. L. Dean (ed.): *Automatic concept formation in pure mathematics. Proceedings of the 16th International Joint Conference on Artificial Intelligence.* San Francisco, pp. 786–793.
12. Conway, J. H.: 1997, *The sensual (quadratic) form*, Vol. 26 of *Carus Mathematical Monographs.* Washington, DC: Mathematical Association of America.
13. Corfield, D.: 2003, *Towards a philosophy of real mathematics.* Cambridge: Cambridge University Press.
14. Corry, L.: 1996, *Modern algebra and the rise of mathematical structures*, Vol. 17 of *Science Networks. Historical Studies.* Basel: Birkhäuser Verlag.
15. Coxeter, H. S. M.: 1969, *Introduction to geometry.* New York: John Wiley & Sons Inc., second edition.
16. Dedekind, R.: 1877, *Sur la théorie des nombres entiers algébrique.* Paris: Gauthier-Villars. Also *Bulletin des sciences mathématiques et astronomiques (1)*, 11 (1876) 278–288, (2), 1 (1877) 17–41, 69–92, 144–164, 207–248; parts also in Dedekind's *Werke*, vol. 3, 263–296. Translated as *Theory of algebraic integers* with an editorial introduction by John Stillwell, Cambridge University Press, Cambridge, 1996.
17. Dickson, L. E.: 1966, *History of the theory of numbers. Vol. I: Divisibility and primality. Vol. II: Diophantine analysis. Vol. III: Quadratic and higher forms.* New York: Chelsea Publishing Co.
18. Ebbinghaus, H.-D., H. Hermes, F. Hirzebruch, M. Koecher, K. Mainzer, J. Neukirch, A. Prestel, and R. Remmert: 1990, *Numbers*, Vol. 123 of *Graduate Texts in Mathematics.* New York: Springer-Verlag. With an introduction by K. Lamotke, Translated from the second German edition by H. L. S. Orde, Translation edited and with a preface by J. H. Ewing.
19. Edwards, H. M.: 1980, 'The genesis of ideal theory'. *Archive for History of Exact Sciences* **23**, 321–378.
20. Edwards, H. M.: 1992, 'Mathematical ideas, ideals, and ideology'. *Math. Intelligencer* **14**(2), 6–19.

21. Edwards, H. M.: 1996, *Fermat's last theorem: a genetic introduction to algebraic number theory*, Vol. 50 of *Graduate Texts in Mathematics*. New York: Springer-Verlag. Corrected reprint of the 1977 original.
22. Euler, L.: (1732/3) 1738, 'Observationes de theoremate quodam Fermatiano aliisque ad numeros primos spectantibus'. *Comm. Ac. Petrop.* **6**, 103–107. Reprinted in Volume 2 of [25], pp. 1–5.
23. Euler, L.: (1747/48) 1750, 'Theoremata circa divisores numerorum'. *N. Comm. Ac. Petrop.* **1**, 20–48. Reprinted in Volume 2 of [25], pp. 62–85.
24. Euler, L.: 1770, *Vollständige Anleitung zur Algebra.* St. Petersberg: Kays. Akademie der Wissenschaften. Reproduced in Volume 1 of [25].
25. Euler, L.: 1911–1956, *Opera Omnia. Series Prima: Opera Mathematica.* Geneva: Societas Scientiarum Naturalium Helveticae. 29 volumes.
26. Ferreirós, J.: 1999, *Labyrinth of thought: a history of set theory and its role in modern mathematics*, Vol. 23 of *Science Networks. Historical Studies.* Basel: Birkhäuser Verlag.
27. Gauss, C. F.: 1801, *Disquisitiones Arithmeticae.* Leipzig: G. Fleischer.
28. Goldman, J. R.: 1998, *The queen of mathematics: a historically motivated guide to number theory.* Wellesley, MA: A K Peters Ltd.
29. Gray, J.: 1992, 'The nineteenth-century revolution in mathematical ontology'. In: *Revolutions in mathematics*, Oxford Sci. Publ. New York: Oxford Univ. Press, pp. 226–248.
30. Hafner, J. and P. Mancosu, 'The varieties of mathematical explanation'. To appear in K. Joergensen, P. Mancosu, et al., editors, *Visualization, Explanation and Reasoning Styles in Mathematics*, Kluwer.
31. Hardy, G. H. and E. M. Wright: 1979, *An Introduction to the Theory of Numbers.* Oxford, fifth edition.
32. Hermite, C.: 1848, 'Théorème relatif aux nombres entiers'. *Journal de Mathématique pures et appliquées* **13**, 15. Reprinted in Hermite's *Ouevres*, Gauthier-Villars, Paris, 1905–17, page 264.
33. Knobloch, E.: 1994, 'From Gauss to Weierstrass: determinant theory and its historical evaluations'. In: *The intersection of history and mathematics*, Vol. 15 of *Sci. Networks Hist. Stud.* Basel: Birkhäuser, pp. 51–66.
34. Kohlenbach, U., 'Some logical metatheorems with applications in functional analysis'. To appear in the Transactions of the American Mathematical Society.
35. Laugwitz, D.: 1999, *Bernhard Riemann 1826–1866: turning points in the conception of mathematics.* Boston, MA: Birkhäuser Boston Inc. Translated from the 1996 German original by Abe Shenitzer with the editorial assistance of the author, Hardy Grant, and Sarah Shenitzer.
36. Lemmermeyer, F.: 2000, *Reciprocity laws: from Euler to Eisenstein*, Springer Monographs in Mathematics. Berlin: Springer-Verlag.
37. Lenat, D. B.: 1976, 'AM : an artificial intelligence approach to discovery in mathematics as heuristic search'. Ph.D. thesis, Stanford.
38. Mancosu, P.: 2000, 'On mathematical explanation'. In: E. Grosholz and H. Breger (eds.): *The growth of mathematical knowledge.* The Netherlands: Kluwer Academic Publishers, pp. 103–119.
39. Mancosu, P.: 2001, 'Mathematical explanation: problems and prospects'. *Topoi* **20**, 97–117.
40. Nipkow, T.: 2003, 'Structured Proofs in Isar/HOL'. In: H. Geuvers and F. Wiedijk (eds.): *Types for Proofs and Programs (TYPES 2002)*, Vol. 2646 of *Lecture Notes in Computer Science.* Berlin, pp. 259–278. Available under "documentation" at [58].

41. Nipkow, T., L. C. Paulson, and M. Wenzel: 2002, *Isabelle/HOL. A proof assistant for higher-order logic*, Vol. 2283 of *Lecture Notes in Computer Science*. Berlin: Springer-Verlag.
42. Niven, I., H. S. Zuckerman, and H. L. Montgomery: 1991, *An introduction to the theory of numbers*. New York: John Wiley & Sons Inc., fifth edition.
43. Rudnicki, P. and A. Trybulec: 1999, 'On equivalents of well-foundedness'. *Journal of Automated Reasoning* **23**(3–4), 197–234.
44. Scharlau, W. and H. Opolka: 1985, *From Fermat to Minkowski: lectures on the theory of numbers and its historical development*, Undergraduate Texts in Mathematics. New York: Springer-Verlag. Translated from the German by Walter K. Bühler and Gary Cornell.
45. Schoenfeld, A. H.: 1985, *Mathematical Problem Solving*. Orlando, Florida: Academic Press.
46. Smith, H. J. S., 'Report on the Theory of Numbers'. Originally published as a report to the British Association in six parts between 1859 and 1865. Reprinted by Chelsea both as a separate volume and in [48].
47. Smith, H. J. S.: 1855, 'De compositione numerorum primorum formae $4\lambda+1$ ex duobus quadratis'. *Journal für die reine und angewandte Mathematik (Crelle's Journal)* **L.**, 91–92. Reprinted in [48], pp. 33-34.
48. Smith, H. J. S.: 1965, *Collected Mathematical Papers*. Bronx: Chelsea Publishing Company. Edited by J.W.L. Glaisher. Originally published by Clarendon Press, Oxford, 1894.
49. Stein, H.: 1988, 'Logos, Logic, and Logistiké'. In: W. Aspray and P. Kitcher (eds.): *History and Philosophy of Modern Mathematics*. University of Minnesota, pp. 238–259.
50. Steiner, M.: 1978, 'Mathematical Explanation'. *Philosophical Studies* **34**, 133–151.
51. Tappenden, J.: 1995, 'Extending knowledge and 'fruitful concepts': Fregean themes in the philosophy of mathematics'. *Noûs*.
52. Troelstra, A. S. and D. van Dalen: 1988, *Constructivism in Mathematics: An Introduction*. Amsterdam: North-Holland. Volumes 1 and 2.
53. Weil, A.: 1984, *Number theory: an approach through history, from Hammurapi to Legendre*. Boston, MA: Birkhäuser Boston Inc.
54. Wenzel, M.: 1999, 'Isar – A Generic Interpretative Approach to Readable Formal Proof Documents.'. In: Y. Bertot, G. Dowek, A. Hirschowitz, C. Paulin, and L. Théry (eds.): *Theorem Proving in Higher Order Logics, 12th International Conference, TPHOLs'99, Nice, France, September, 1999, Proceedings*, Vol. 1690 of *Lecture Notes in Computer Science*. Berlin, pp. 167–184.
55. Wenzel, M.: 2002, 'Isabelle/Isar — a versatile environment for human-readable formal proof documents'. Ph.D. thesis, Institut für Informatik, Technische Universität München.
56. Wussing, H.: 1984, *The genesis of the abstract group concept: a contribution to the history of the origin of abstract group theory*. Cambridge, MA: MIT Press. Translated from the German by Abe Shenitzer and Hardy Grant.
57. Zagier, D.: 1990, 'A one-sentence proof that every prime $p \equiv 1 \pmod 4$ is a sum of two squares'. *Amer. Math. Monthly* **97**(2), 144.
58. 'The Isabelle theorem proving environment'. Developed by Larry Paulson at Cambridge University and Tobias Nipkow at TU Munich. http://www.cl.cam.ac.uk/Research/HVG/Isabelle/index.html.

Appendix

```
theory GaussInt = Main:

typedef gauss-int = {p::(int*int). True}
  by auto

instance
  gauss-int :: times..

constdefs
  gauss :: int * int => gauss-int
  gauss p == Abs-gauss-int(p)

  gauss-re :: gauss-int => int
  gauss-re(z) == fst(Rep-gauss-int z)

  gauss-im :: gauss-int => int
  gauss-im(z) == snd(Rep-gauss-int z)

  gauss-conj :: gauss-int => gauss-int
  gauss-conj z == gauss(gauss-re z, -gauss-im z)

  gauss-norm :: gauss-int => int
  gauss-norm z == gauss-re(z) ^ 2 + gauss-im(z) ^ 2

  gauss-IsInt :: gauss-int => bool
  gauss-IsInt z == (gauss-im z = 0)

defs
  gauss-mult-def:
  w * z == gauss(gauss-re(w) * gauss-re(z) - gauss-im(w) * gauss-im(z),
     gauss-re(w) * gauss-im(z) + gauss-im(w) * gauss-re(z))

lemma [simp]: Rep-gauss-int(Abs-gauss-int(a,b)) = (a,b)
  by (rule Abs-gauss-int-inverse, simp add: gauss-int-def)

lemma [simp]: (Abs-gauss-int(a,b) = Abs-gauss-int(c,d)) = ((a,b) = (c,d))
  by (simp add: Abs-gauss-int-inject gauss-int-def)

lemma [simp]: gauss-re(gauss(a,b)) = a
  by (simp add: gauss-re-def gauss-def)

lemma [simp]: gauss-im(gauss(a,b)) = b
  by (auto simp add: gauss-im-def gauss-def)
```

```
lemma gauss-gauss-re-im-conv [simp]: gauss(gauss-re(z),gauss-im(z)) = z
  by (auto simp add: gauss-def gauss-re-def gauss-im-def
      Rep-gauss-int-inverse)

lemma gauss-gauss-eq [simp]:
    (gauss(a,b) = gauss(c,d)) =(a = c & b = d)
  by (auto simp add: gauss-def)

lemma gauss-mult-gauss [simp]:
    gauss(a,b) * gauss(c,d) = gauss(a * c − b * d, a * d + b * c)
  by (auto simp add: gauss-mult-def)

lemma gauss-gauss-ex-intro [intro]: EX z. P(z) ==> EX x y. P(gauss(x,y))
  apply (erule exE)
  apply (subgoal-tac P(gauss(gauss-re(z),gauss-im(z))))
  by (auto simp del: gauss-gauss-re-im-conv, simp)

lemma gauss-mult-int [simp]: gauss-IsInt x ==> gauss-IsInt y ==>
    gauss-IsInt (x * y)
  by (simp add: gauss-IsInt-def gauss-mult-def)

lemma gauss-mult-int-eq [simp]: gauss-IsInt x ==> gauss-IsInt y ==>
    gauss-re x * gauss-re y = gauss-re(x * y)
  by (simp add: gauss-IsInt-def gauss-mult-def)

lemma gauss-mult-commute [simp]: (w::gauss-int) * z = z * w
  by (auto simp add: gauss-mult-def zmult-commute zadd-commute)

lemma gauss-mult-assoc [simp]: ((u::gauss-int) * v) * w = u * (v * w)
  by (auto simp add: gauss-mult-def zmult-ac zadd-zmult-distrib
    zadd-zmult-distrib2 zdiff-zmult-distrib zdiff-zmult-distrib2)

lemma gauss-mult-left-commute [simp]: (u::gauss-int) * (v * w) = v * (u *
w)
  by (auto simp add: gauss-mult-def zmult-ac zadd-zmult-distrib
    zadd-zmult-distrib2 zdiff-zmult-distrib zdiff-zmult-distrib2)

lemma gauss-conj-mult: gauss-conj(w) * gauss-conj(z) = gauss-conj(w * z)
  by (simp add: gauss-conj-def gauss-mult-def)

lemma gauss-mult-conj-self: z * gauss-conj(z) = gauss(gauss-norm(z),0)
  by (auto simp add: gauss-norm-def gauss-conj-def gauss-mult-def
    power2-eq-square)

lemma gauss-norm-conj: gauss-norm(z) = gauss-re(z * gauss-conj(z))
  by (simp add: gauss-mult-conj-self)
```

```
lemma gauss-mult-conj-self-int [simp]: gauss-IsInt (x * gauss-conj x)
  by (simp add: gauss-mult-conj-self gauss-IsInt-def)

lemma gauss-norm-mult [simp]: gauss-norm(x) * gauss-norm(y) =
    gauss-norm(x * y)
  by (simp add: gauss-norm-conj gauss-conj-mult)

lemma sum-squares-eq-norm-gauss: a^2 + b^2 = gauss-norm(gauss(a,b))
  by (simp add: gauss-norm-def)

theorem EX (x::int) y. (a^2 + b^2) * (c^2 + d^2) = x^2 + y^2
  by (auto simp add: sum-squares-eq-norm-gauss)

end
```