Proof Technology and Learning in Mathematics: Common Issues and Perspectives

Nicolas Balacheff, Thierry Boy de La Tour

PROOF TECHNOLOGY AND LEARNING IN MATHEMATICS: COMMON ISSUES AND PERSPECTIVES

Nicolas Balacheff
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
Nicolas.balacheff@imag.fr

Thierry Boy de la Tour
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
thierry.boy-de-la-tour@imag.fr

To cite this version:
Balacheff N., Boy de la Tour T. (2019) Proof Technology and Learning in Mathematics: Common Issues and Perspectives. In: Hana G., Reid D., de Villiers M. (eds) Proof Technology in Mathematics Research and Teaching (pp. 349-365). Berlin: Springer.

INTRODUCTION

Mathematical proof is undoubtedly the cornerstone of mathematics. Indeed, no mathematical work is definitively complete without the final QED. Mathematics educators know this centrality of proof, the challenges it presents in terms of teaching, and the complexity of its learning. The rapid spread of mathematical digital technology over the last three decades has changed the field of mathematical experience for learners and opened new avenues in the teaching of proof. The example of dynamic geometry, as Maria-Alessandra Mariotti observes in this volume, plays a very special role in this context. The availability of more and more powerful digital technologies has also changed the work of mathematicians to the point where the very notion of mathematical proof is being questioned. Beyond mechanizing computation, these technologies are now mechanizing mathematical reasoning and proving with unprecedented consequences: logicians and computer scientists have made such progress that not only are automated theorem proving tools available, but even proof assistants. How will these new technologies impact mathematics teaching and learning?

Written by researchers from both sides—from the fields of automated theorem proving and of mathematics education—the chapters in this volume unpack the complexity and meaning of this question, open new issues, offer responses, and invite readers to think about the next step. Which comprehensive multidisciplinary projects in the near future will trigger a breakthrough in offering the smartest technology-enhanced environments for the learning and teaching of mathematical proof?

As authors of this afterword, we also come from both sides. In the following, we share our discussion of the ideas suggested by the preceding chapters. The next sections reflect our readings and visions of the future—first, in relation to automated theorem proving and second, in relation to the design of technology-enhanced environments for the learning of proof in mathematics. In the epilogue, we venture a bit further to consider a world where the mechanization of proving is fully achieved.

AUTOMATED THEOREM PROVING

As the performance of computers and Automated Theorem Provers (ATPs)[1] increases, we can expect a greater number of users of these systems either through direct learning of the indispensable but austere skills they require or through the specialized tools that academies and industries may provide. Thus, as with computer algebra systems and numerical simulations, these tools are likely to impact all activities related to mathematics. For instance, in 1964, Martinus Veltman developed the Schoonschip program for symbolic evaluation of algebraic expressions. This contributed to his work on particle physics and, in 1999, this work won him the Nobel prize.

COMPUTER PROOFS

Even if ATPs have not yet contributed to winning a Fields Medal, they have settled certain conjectures. A recent example published in Heule, Kullmann, and Marek (2016) determined the maximal integer *n* for which the set {1, ... , n} can be partitioned in two parts such that none contains three numbers a, b, c such that $a^2+b^2=c^2$. This number happens to be 7824. It is actually easy to prove that the set {1, ... ,7824} admits such a partition simply by providing a bi-partition and running a simple program to check that no triple of integers in each part has the property above. It is much more difficult to prove that *all* bi-partitions of {1, ... , 7825} contain such triples of integers because they total 2^{7825}. It would require a very elaborate program to explore such a huge search space in a reasonable period of time. Fortunately, SAT-solvers (ATPs devoted to propositional logic) are very elaborate programs. Indeed, it is quite remarkable that a program that was not designed specifically to solve this problem was actually able to do so simply from a standard translation into propositional logic (and some adequate tuning that need not concern us here).

This being said, the computation took four years of CPU time, dispatched among 800 cores in parallel. The mere fact that this computation happened cannot be considered a proof of the conjecture: computing errors can occur (e.g., a cosmic ray may hit the wrong core at the wrong time) and the SAT-solver could be bugged. The specificity of ATPs compared to other mathematical software is that they are able to support their computations (in a successful event) by a proof. Thus the provability of the conjecture is reduced to the validity of the proof, and we only need to read it to be convinced. Alas, the proof produced by this computation took 200 terabytes of memory and can obviously not be read by a mathematician.

[1] We do not distinguish here between fully and semi-automated theorem provers. Since techniques of automated proving are present in both, we will use the acronym ATP in a generic way to make the reading smoother for non-experts. The context should be sufficient to identify the differences when necessary.

So what have we gained with this huge amount of data? Simply the fact that this proof can be checked for correctness by a simple program and independently of the SAT-solver that produced it (this is known as the *de Bruijn criterion*). Running this proof checker several times minimizes the probability of a computing error, thereby reducing the provability of the conjecture to the correctness of a simple program. Further, it is not difficult to prove that such simple proof checkers only accept valid proofs.

Of course, this kind of proof is very different from those produced by mathematicians. This is not surprising as mathematicians themselves have been unable to prove the conjecture. Only a computer has so far been able to find a proof. This is likely one characteristic of the future of ATPs in mathematics—they will help mathematicians with tasks they cannot achieve alone while also creating frustration. This is linked not only to the length of computer proofs, but also to the fact that many ATPs translate statements into an internal format (a normal form) that yields very uniform proofs. This uniformity benefits efficiency but not clarity.

MATHEMATICAL PROOFS

The concept of proof we have sketched so far is close to the notion of *witness*, as a natural number n is the witness of membership[2] of an element $f(n)$ in a recursively enumerable set $\{f(n) \mid n \in \mathbb{N}\}$ (where f is a computable total function). From this purely computational point of view, sets of theorems are nothing more than recursively enumerable sets, and theorems are produced by the rules of a meaningless game. This is fine for computers, but mathematicians have a habit of endowing mathematical statements with *meaning*. The standard set-theoretic semantics of first and higher order logic constitute a good approximation of this meaning (even if categorical or other semantics may seem more appropriate as a foundation). In this semantic context, we can only admit inference rules that preserve the meaning of mathematical statements. This is what guarantees the correctness of mathematical deductions (and excludes inductive reasoning, a common but incorrect inference that should not be confused with mathematical induction).

In mathematical proofs found in textbooks, inference rules are generally implicit. Statements follow each other separated by "thus", "hence", and "therefore," sometimes requiring a great deal of thinking (and writing) just to understand why one statement is implied by others. By leaving standard or secondary details to the reader, a proof can be made shorter and thus emphasize its core arguments—those that the writer perceives as pivotal. Sometimes arguments are supported by figures, as is common in category theory and geometry, or in the schematic proofs in the chapter by Alan Bundy and Mateja Jamnik in this volume. It is clear that usual mathematical proofs do not meet the *de Bruijn criterion*; no simple program can check that they are correct.

[2] Let $S=\{f(n) \mid n \in \mathbb{N}\}$, then $x \in S$ iff $\exists n \in \mathbb{N}$ such that $x=f(n)$. Hence if $x \in S$, there is a witness n which proves this fact. On the opposite side, if $x \notin S$ then there is no witness to prove it.

Yet mathematicians do make mistakes—sometimes superficial, sometimes deep—and every mathematical proof is as likely to contain bugs as programs are. Every devoted referee knows how difficult and time consuming it can be to hunt for bugs in proofs written by colleagues. These are strong incentives to develop software that can automatically verify mathematical proofs. As computers can only handle syntax, this means that mathematical proofs should be translated into formal proofs in a convenient logic. This is a job for ATPs.

One pioneering project in this direction is the Mizar system developed by the group founded by Andrzej Trybulec in 1973 (see Bancerek et al., 2018). The aim of this system is to automatically check given texts in a language as close as possible to standard mathematics and hence readable by a human being. The author of an article is asked to fill in intermediary details (using feedback from the system) until Mizar achieves its verification. Of course, it is tempting to use such systems not simply to check a completed mathematical proof, but also to develop an incomplete proof in a partially automated way—in other words, to use the system as a *proof assistant* by exploiting its proving capabilities.

Many proof assistant systems are designed to provide this kind of help. Among the most popular are *PVS*, *Isabelle*, and *Coq* (see the chapter by Chantal Keller in this volume). All these systems have an input language that offers the possibility of writing mathematical definitions and statements, and triggering a number of automated proving tools. Many satisfy the *de Bruijn criterion* in the sense that statements are recognized as theorems only if they have a formal proof in the logic of the system.

This leads to a concept of proof as the information provided to a proof assistant that allows it to find a formal proof of a statement (or at least to verify it). This means that what is considered a proof depends on the state of the system (including its libraries of mathematical knowledge and automated tools). More fundamentally, the problem of verifying such proofs (including mathematical proofs) is undecidable, and contrary to formal proofs where inference rules should always be decidable relations.

One important aspect of proof assistants is the language in which they require users to write mathematics. Obviously these languages, called *logical frameworks*, should be both general enough to express any kind of mathematics and close enough to the standards of mathematical language. The burden of translating from the latter to the former is left to the user and would be very difficult to automatize. It requires a good knowledge of the target language, but also of the level of detail required for a verification to be possible. In terms of length, this translation can be expected to multiply the original text by a factor of approximately four. This is currently more time consuming than refereeing a paper but, of course, it is safer.

PROOF THEORY

We can also see proof assistants as systems that translate (some form of) mathematical proofs into formal proofs. We may thus examine the possibility of a reverse translation—from formal proofs to the kind of proofs that mathematicians write, and that we can read. This kind of translation would ideally

leave aside all the gory details that readers do not need to know and provide only the essence of the proof. Two important methods for structuring proofs are illustrated in the chapter by Rob Arthan and Paulo Oliva: factorizing recurring terms into *definitions* and splitting the proof between *lemmas*.

Alternatively, we could design ATPs that produce readable proofs directly. For instance, Pedro Quaresma and Vanda Santos address in their chapter the challenging problem of producing proofs in geometry that can be illustrated step by step on figures and illustrate that this is not simply a problem of proof search, but mostly of the logic in which proofs are searched for. Another example is the chapter by Mohan Ganesalingam and William Timothy Gowers where most of the effort is devoted to designing a logic that would "imitate human thought."

These problems are related to *proof theory*—the study of the structure of mathematical proofs—and particularly to the work of Gerhard Gentzen (see Takeuti, 1975). In his famous *sequent calculus*, theorems have the form $\Gamma \vdash \Delta$, where Γ and Δ are sequences of formulas (statements). The intended meaning is that the logical disjunction of the formulas on the right can be deduced from the formulas on the left. This grants a nice symmetry (or duality) between the inferences of formulas on the left and right sides. It is also undeniable that sequents of the form $\varphi \vdash \varphi$, for any formula φ, should be accepted as axioms.
Another obvious inference (at least when Δ is empty) is the *cut rule*

$$\frac{\Gamma \vdash \Delta, \varphi \quad \varphi, \Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

where φ can be considered as a lemma that is proved in the first premise and used in the second to complete the proof. This rule is inconvenient for ATPs because using it to attempt to prove the conclusion requires guessing the lemma. The fundamental result obtained by Gentzen, known as Gentzen's Hauptsatz or cut-elimination theorem, shows that in first order logic every proof can be transformed into a cut-free proof (possibly with a huge increase in size). Thus ATPs can conveniently search for cut-free proofs, which is close to the so-called tableaux method (see Hähnle, 2001). But such proofs could be much reduced if we could introduce cut rules, and therefore lemmas, by inverting cut-elimination. Recent results in this direction have been presented in Ebner, Hetzl, Leitsch, Reis, and Weller (2018).

Most of the logical frameworks mentioned above are formulated as sequent calculi with various features convenient for proving and expressing mathematical statements. An interesting example is the $\lambda\Pi$-Calculus Modulo Theory (Assaf et al., 2016) that offers the option of separating computations from deductions, thus hiding computations in a possible translation of such proofs in natural language. Sequent calculi offer many possibilities to express statements and also proofs in different forms, and to study their possible transformations. It is clear that the future of proof technology does not rely simply on the design of efficient ATPs but also, and fundamentally, on the development of proof theory.

DESIGNING TECHNOLOGY FOR ENHANCING PROOF LEARNING

PROOF TUTORS

Interest in ATP research for educational applications emerged in the early 70s with the belief that "building a theorem prover is an exciting alternative to the usual classroom presentation" (Goldstein, 1973, p.3). Technically, Goldstein's geometry prover was based on representations of procedural knowledge with the objective of "developing a high-level, 'natural' formalism for representing mathematical knowledge." (Ibid.). The identified challenge was to find efficient methods for "finding a proof amidst a surfeit of geometric knowledge" (Ibid.). However, the choice of backward chaining to model reasoning was not as natural as the formalism could have been. Nevins (1974), thanks to an "efficient representation" of the knowledge base and to "confining" the exploration to objects (points and lines) implicit in the statement but appearing in the diagram, proposed a solution with forward chaining. These seminal works prepared the ground for *Geometry Tutor* (Anderson, Boyle, & Yost, 1988)—the emblematic tutor for the learning of mathematical proof designed on the principles of the ACT* model of cognition (Anderson, 1983). *Geometry Tutor* was "based on considerable theoretical analysis of the characteristics of the problem domain and on a great empirical observation of students' behavior" (Anderson et al., 1988, p.4). It included an "ideal model" in order to "generate proofs in what we felt was a natural, human way—in contrast to some methods of proof" (Ibid. p.5) and "buggy models" to allow diagnosing student errors as production rules. The *Tutor* traces students' behaviour through its ideal and buggy models and provides immediate or "near immediate" feedback to keep them on the right track.

However, *Geometry Tutor* was criticized for the restricted possibilities it gave learners, the limited relevance of its feedback, and its static diagrams. These criticisms led to the *Angle Project* (Koedinger & Anderson, 1990) which gave a more active role to diagrams, and then to a new generation of Cognitive tutors including a cognitive model of the content to be learned and more subtle scaffolding strategies for feedback and advice (Anderson, Corbett, Koedinger, & Pelletier, 1995).

Although these tutors went to school and success was reported in various domains, they showed limits inherited from production rule systems, especially in relation to mathematical proof. These systems satisfied the requirements for modeling competences, but imposed severe limits on modeling mathematical problem solving due to their incapacity to properly express mathematical knowledge.

ATP research developed long before the birth of research on mathematical proof learning environments. It was focused on modeling human reasoning, mainly with a view to implementing propositional logic. The first significant project, the *Logic Theorist* (Newell & Simon, 1956), made a breakthrough. It had some educational use for the teaching of information processing languages (Stefferud, 1963), but this remained limited. Based on the *Principia Mathematica*, the Logic Theorist relied on mathematicians' introspection and a few empirical observations of students. This weakness was later overcome by a systematic observation of students solving proof problems in symbolic logic.

It led to an improved model, the *General Problem Solver* (Newell, Shaw, & Simon, 1959), but this model still had to face a much more demanding challenge: “problem solving is the battle of selection techniques against a space of possibilities that keeps expanding exponentially” (Ibid., p. 26). The lessons learned from the Logic Theorist shaped the design of the *Geometry Machine* (Gelernter, 1959) by *limiting the domain* and searching for a *specifics heuristic* (e.g., drawing benefit from diagrams) that could model “the discovery of proofs for theorems in elementary Euclidean plane geometry in the manner, let us say, of a high school sophomore” (Ibid., p.135). To some extent, the approach was pragmatic:

> [T]he machine is granted the same privileges enjoyed by the high-school student who is always assuming (i.e., introducing as additional axioms) the truth of a plethora of ‘obviously self-evident’ statements concerning, for example, the ordering properties of points on a line and the intersection properties of lines in a plane. (Ibid., p.139)

Such a statement and the evidence that the *Geometry Machine* could solve high school geometry problems opened some perspective for the design of Technology Enhanced Learning (TEL) environments for mathematical proving. Unfortunately that was not the case even though the Gelernter paper, first published in *Computer and Thoughts* in 1959, was reprinted in *Intelligent Tutoring Systems* in 1963. The convergence of ATP and educational technology fell short.

Several issues had to be addressed to make a convergence possible: the depth of the gap between machine and human ways of reasoning, the multiplicity of representations in geometry (including diagrams and hence reasoning based on visualization), the constraints on human–machine communication—just to mention a few.

Interestingly, while ATP research focused on computational models, the *Geometry Tutor* project came to recognize that “placing interface design ahead of production–systems design represents a major restructuring of our approach to tutor construction” (Anderson et al., 1995, p.35). This was because of the impact of the interface on the skills students acquire; the interface is the space in which they experience and learn. Beneath the interface, the implemented models must ensure that communication makes representations accessible to students and enables interactions that enhance learning—hence the following objective from the early 2000s which could be shared by many projects:

> Our long-term goal is to build an intelligent tutoring system for elementary geometry. Any proofs and constructions found by our automated geometry theorem prover [GRAMY] must be stated with the common ontology of Euclidean geometry—the axiomatized geometry system taught in schools. (Matsuda & VanLehn, 2004, p.3)

However, provers have to comply not only with the constraints of the interface, but also with the professional responsibilities of teachers. This imposes an additional filter: “the desired geometry theorem prover must not only be able to find a single comprehensible proof, it should also be able to find all proofs that are considered acceptable to instructors” (Ibid., p.4). Along with providing ATP features for mathematicians, computer-based tutors must take three additional categories of users into account: the curriculum decision makers (who specify the standard of mathematical validation at a given grade), the teachers (who orchestrate learning and decide what counts as a proof in relation to a

standard), and the learners (who are simultaneously constructing an understanding of proof and of the related content).

FROM EMPIRICAL VALIDATION TO MATHEMATICAL PROOF

How to assess the validity of learners' statements is a question throughout primary and secondary schooling. Indeed, the means and criteria of assessment from the perspective of the learner and the perspective of the teacher at each grade level differ in a significant way. Even at the university level, students continue to shape their understanding of what counts as a mathematical proof; for example, despite being familiar with proving in advanced algebra or calculus, they are often destabilized by proving in discrete mathematics (e.g., graph theory). However, the gap between advanced learning and early learning constitutes another order of magnitude. At the earliest stages, validation is dominated by naive induction and empirical verification. It is constrained by language, discourse structures, and the available representations of mathematical objects and relations. This approach gradually develops into more elaborate types of argumentation based on generic examples or thought experiments. These early forms of argumentation reflect the everyday structure of argumentative discourse, which is not devoid of logic but can display the ill-defined content and implicitness of ad hoc situations. Hence, at some point in the middle school curriculum, mathematics teachers face the challenge of introducing students to the sociomathematical norms of argumentation (Yackel & Cobb, 1996, p.466 sqq) and ultimately to the basics of mathematical proof. This is also a challenge for the design of ATP-based TEL environments, whether based on proof assistants for pedagogical use or proof tutors.

Learning mathematical proof is less an evolution than an epistemological revolution: students must evolve from the position of practitioner, driven by the practical realities of a situation, to the position of theoretician, driven by the requirements of knowledge. The validity of a mathematical statement does not depend on the beliefs and conceptions of the proposer (*epistemic value*, Duval, 2007, p.138) or of a community, but only on the relations it has with other previously validated statements following given norms of inference which ensure that it has a place within a mathematical theory (*ontic value*, Hanna, 2017). Being aware of the existence of such a reference theory bounding the discourse supporting the validity of a mathematical statement is as critical as understanding and mastering logical rules—hence the claim that a *mathematical theorem* is not a statement backed by a proof but, as Mariotti states in her chapter, by "the system of statement, proof and theory." This conception of theorem is consistent with the conception driving the design of TEL environments. It offers a starting point for the convergence of both fields of research by emphasizing that decisions must be based not only on the nature and complexity of proofs, but also, and inseparably, on the underlying theory.

This definition of theorem builds a first bridge between the two sides of a proof—validation and explanation. Explanation, in the sense of Duval (1992), refers here to a system of relations in which the statement to be explained finds its place. While solving a problem comforts the positive epistemic value of the solution statement, its meaning comes from the coherence of this statement's membership with the problem-solver system of knowledge. Hence, a consequence for the relevance

and viability of the pedagogical use of ATP-based TEL environments is that these environments should either facilitate learners' appropriation of the epistemic value of the solution (providing an argumentation) or situate the proof in a theoretical framework compatible and coherent with the learners' system of knowledge (providing an explanation). The difficulty in doing so comes from the inherent formal nature of ATPs, although logicians and computer scientists are aware that "at the research level, mathematics is often quite vaguely formulated because mathematicians can usually rely on the deep understanding and intuition of themselves and their fellows to keep them out of trouble" (Harrison, Urban, & Wiedijk, 2014, p.57). As a matter of fact, in a less "radical" way, this also holds true for the learner.

When students propose a proof, and teachers present a proof to the class, they are supposed to find the right balance between what must be made explicit and what is tacitly accepted. This does not mean a lack of rigour or a tolerance of vague approximations. It is just a property of human communication that applies to mathematics as it does to other domains: a mathematical proof is a discourse that should be both convincing and meaningful. Indeed, throughout its history, mathematics has framed its discourse and established a standard centered on formalization as a means of ensuring the rigour and validity of its outcomes. But as the Bourbaki group—the paragon of formalism—claimed, this mathematical discourse remains "naïve" insofar as it includes natural language and necessarily admits some shortcuts in proofs.

Looking at proof as a discourse reveals an unresolved tension. In the most classic teaching tradition, solving a problem is presented step by step in such a way that the end produces at once both the solution and the proof. This is reinforced by the use of representations such as two-column proofs, flow-chart proofs, and standardized paragraph proofs. These reify the mathematical norm, monitor or scaffold students' activity, and serve as a professional pedagogical resource (Weiss, Herbst, & Chen, 2009). The private activity of the student, like the private activity of the mathematician, is foreign to this ideal picture. When not reduced to the application of a few theorems or the implementation of a standard reasoning, problem solving is driven and supported by *heuristic argumentation* (Duval, 1992, p.51). Heuristic argumentation is not meant to convince, but to allow choices in a context where not all statements and possibilities have been strictly verified or even made explicit, as is the case, for example, with plausible reasoning (Polya, 1945). Writing a proof consists of logically structuring, making explicit, and verifying all the statements that relate the hypothesis to the claimed solution. In terms of the content, a continuity exists between the heuristic argumentation and the proof, precisely observed and conceptualized by Paulo Boero as the "cognitive unity of theorems":

> During the production of the conjecture, the student progressively works out his/her statement through an intensive argumentative activity functionally intermingled with the justification of the plausibility of his/her choices. During the subsequent statement-proving stage, the student links up with this process in a coherent way, organising some of the previously produced arguments according to a logical chain. (Garuti, Boero, & Lemut, 1998)

When associated with a purely deductive problem-solving process, this continuity facilitates a natural transition to proof. But when the problem has some complexity, abduction and induction having played

a key role, a structural distance emerges between argumentation and proof (Pedemonte, 2007) even though the discursive surface structure of argumentation may be close to the structure of a proof (Duval, 1992, p.38). The transition from problem solving to proving is not straightforward, and students have to achieve it in a demanding context. First, the representations and controls they use may lead to technical difficulties. Second, they have to adapt to sociomathematical norms they are in the process of learning and understanding. Third, they have to comply with the teacher's evolving expectations. Despite the analogies between the work of mathematicians and students, there are differences of an epistemological nature that result from the process of teaching mathematical proof (Herbst & Balacheff, 2009) and from the distance from professional values and norms (Dawkins & Weber, 2016).

Results from research on the learning of mathematical proof show that the contribution of ATPs requires understanding and solving new problems. Some design issues are common—mainly bridging the gap between human reasoning and ATP models, and delivering readable write ups. But addressing learning raises other issues such as: (i) bridging the gap between argumentation and proof, (ii) facilitating the transition from problem solving to proving, and (iii) scaffolding the transition from empirical validation to the mathematical norms at play in the classroom.

TECHNOLOGY ENHANCED LEARNING OF MATHEMATICAL PROOF

The largest body of research on technology and the learning of proof has focused on Dynamic Geometry Environments (DGEs) (see, e.g., Sinclair et al., 2017). This is an effect of the influence of Logo and the concept of microworld born in the 80s. The strength of DGEs is that they provide students with a *field of experience* (Boero et al., 1995) for exploring mathematical facts, making conjectures, shaping their argumentation, and proposing proofs (Baccaglini-Frank & Mariotti, 2010). Because research on proof tutors was essentially related to AI and cognitive science, the contribution of ATP research to research on mathematics education has until recently been rather limited (Hauer, Kovács, Recio, & Vélez, 2018, p.2). One of the many DGEs, *Cinderella*, includes a randomized theorem prover that offers the possibility of checking and proving properties. But, as Kortenkamp & Richter-Gebert (2004) point out: "it has been developed as a tool for mathematicians in research, publishing and teaching" (p.1). Actually, to mathematics educators, ATP technology seems far from satisfying the constraints of the educational context. Some ATP researchers are well aware of this critique: "[ATP] generated proofs are unnatural and incomprehensible as these systems do not approach geometry problems like how secondary school students are taught" (Wang & Su, 2017, p.10). But are models that approach mathematics in the same way as learners do necessary to TEL environments? The example of *Cinderella* suggests this is not strictly the case:

> Sometimes the semantics of a user's action can be unclear. For example, when he tries to insert the intersection of three lines by placing a point on it, it is unclear whether he assumes that these always meet in one point or not. A software that "knows" whether the three lines are always concurrent can react properly in that situation: When they are, it is not necessary to ask which pair of points should define the intersection, as all three define the same one. When they are not, the software can signalize that it needs the attention of the user to resolve an ambiguity. (Kortenkamp & Richter-Gebert, 2004, p.8)

Cinderella did not primarily target an educational audience. But the seminal project *Cabri-géomètre*, which does, included an algebraic oracle that could verify properties and support student inquiries with messages like: “The property is true in general*”* or *“*The property seems true on the figure but is false in general. A counterexample can be proffered*”* or “The property is true for this figure but Cabri-géomètre cannot tell for the general case*”* (Laborde, 1990, p.137). This feature was exploited to develop the proof microworld *Cabri-Euclide* (Luengo, 1997).

The question then is: which services can ATP provide in a TEL environment? Or, more precisely, what kind of feedback could ATP provide while the student is engaged in problem solving and proving?

Depending on whether the student is solving a problem or validating a solution, the feedback could target the strategy rather than a particular rule, the related knowledge and not the logical thread, or the standard for a successful achievement at a given grade and not the underlying logical structure. It could immediately spot the misuse of a theorem or a flaw in reasoning, or wait to provide a counterexample to the proposed proof. It could be textual or visual and, in the case of geometry, preferably both. In this sense, geometry is the best case for addressing the difficulties in designing such environments, especially for early learning.

A DGE is a natural working space that enables free heuristic argumentation for solving a geometrical problem. These environments have a built-in validating feedback that is prompted by messing up a construction (Healy, Hoelzl, Hoyles, & Noss, 1994). ATP could provide further services in terms of confirming or invalidating certain properties on demand. For example, the DGE *GeoGebra* “[can answer] a query posed by a user about the truth or falsity of any geometric statement” or “present further hypotheses that should be considered for the proposition to become true" (Hauer, Kovács, Recio, & Vélez, 2018, p.2). Teachers may expect a service such as scaffolding the writing of a proof, hence providing feedback on a textual representation of the solution to a problem. Still, feedback could have a textual or a visual form.

Feedback responds to actions of the student who, in turn, can respond with new actions and hence with new decisions on how to solve a problem or shape a proof. Unlike ATP, the reasoning of the student is semi-empirical rather than apodictic. It is not exact and stable, and may inherit from experience as well as formal learning. Let’s call this knowledge under construction a *conception* (which is not necessarily a misconception). One could characterize it as the operators, controls, and semiotic representations the student uses in order to solve a given set of problems (Balacheff, 2010). The feedback may lead to an evolution of the reasoning, the proof, or related conceptions. From the perspective of early learning, an ATP-based environment is less a tool than a rational agent able to communicate and argue, and build its reasoning on information gathered from student activities and conceptions as they evolve (Luengo, 1999).

Nowadays, digital technologies can be used by mathematicians to solve problems and supplement their proofs. Significantly, the editors of the *Annals of Mathematics* outline the journal's conditions for considering “computer-assisted proof of important mathematical theorems”:

> The human part of the proof, which reduces the original mathematical problem to one tractable by the computer, will be refereed for correctness in the traditional manner. The computer part may not be checked line-by-line, but will be examined for the methods by which the authors have eliminated or minimized possible sources of error [...] We will print the human part of the paper in an issue of the Annals. The authors will provide the computer code, documentation necessary to understand it, and the computer output, all of which will be maintained on the Annals of Mathematics website online. (Annals of Mathematics, retrieved 01-22-2019)

Thanks to their progress, ATP-based tools can be expected to contribute to solving the most important problems in mathematics or checking very long and complex proofs such as Fermat's last theorem, and hence become part of the “computer-assisted proofs” the editors of the *Annals of Mathematics* mention.

Nevertheless, this progress is not sufficient for educational needs; research needs to take further steps to solve the specific problems raised by learning. Such improvement is possible by integrating a *didactic heuristic* into the best suited tools, as well as a distributed architecture for allocating services. The latter reflects Gelernter's vision of splitting complexity among specialized entities: a “syntax computer,” a “diagram computer,” and a “heuristic computer” (Gelernter, 1959, p.139). For example, a “write-up computer” that uses the services of an ATP engine could be in charge of high level communication. This suggests looking precisely at what a multi-agent architecture could offer (Webber, Pesty, & Balacheff, 2002). As to the former, the specificity of learning situations in mathematics creates favorable conditions for ATP searches for a solution to a problem or the verification of a proof. First, it is possible to use the hypothesis of “proving in a close world” (Trilling, 1996): any statement is true because claimed so by the problem statement, or can be proved with limited effort. Second, the ATP logical model can be augmented with a complete description of a priori accepted statements (theorems, definitions), instantiating the theory and completing the triplet (*statement*, *proof*, *theory*). Third, such a *theory* need not be complete with respect to mathematics itself, nor to one of its subdomains. It could be a *micro-theory* in the sense of Minsky (i.e., sufficient for solving a cluster of problems at a specific grade level). Fourth, the degree of implicitness and the desirable structure of the proof can be specified by the teacher providing multiple examples. Comparing proofs replaces the obligation to conform to an “ideal” proof.

Finally, what could a TEL environment look like that takes advancements in research on mathematics education into account? First, it would facilitate identifying different types of activity and enabling navigation among them. A problem-solving space (e.g., a mathematical microworld) could be identified and associated with a formulation space (e.g., a digital notebook for expressing and freely linking mathematical statements) and with a validation space (for expressing a proof in line with the curriculum; e.g., as a two column proof, flowchart proof, graph, or paragraphs). Second, the ATP would be initialized with axioms and theorems chosen by the teacher or stipulated by the curriculum. This initialization would create a theoretical reference framework in line with Mariotti's model of theorem (statement, proof, theory) and be easily accessible to students. It would give students a foundation for understanding that they are proving a statement within a theory.

The problem-solving space would be open to students' strategies and friendly to empirical validation. The formulation space would set the context and gather data for grounding ATP services such as verifying proofs, diagnosing flaws, building counterexamples, comparing proofs, or providing hints. The productive links between the different spaces would create a digital *mathematical working space* responding to students' early need for support in pragmatic and theoretical thinking (see the chapter by Philippe Richard, Fabienne Venant, & Michel Gagnon). These links would also enhance the ATP's ability to get an accurate picture of student activity from their DGE diagrams (Caferra, Peltier, & Puitg, 2001). Some of these features are already included in existing designs—*Cabri-Euclide* (Luengo, 1997); *Baghera* (Soury-Lavergne, 2003); *Advanced Geometry Tutor* (Matsuda & VanLehn, 2004); *AgentGeom* (Cobo, Fortuny, Puertas, & Richard, 2007); *PACT Geometry Tutor* (Aleven, 2010; and *QED-Tutrix* (Leduc, 2016)—hence making these suggestions plausible. Integrating them fully and using the full power of ATP would be a breakthrough.

EPILOGUE

The editors of this book remind us of the dream of Vladimir Voevodsky: "to have an electronic proof assistant check any theorem before its publication, as a way to accelerate the reviewing process." This dream could be extended further, questioning the need for both mathematicians and mathematics teachers.

The development of collaborative mathematics through online social media, as addressed in the chapter by Lorenzo Lane and his colleagues, opens the same question as other collaborative online activities: can participants be replaced by machines? This is an important problem for online poker or chess where machines now exist that are much better players than humans. This is of course not the case for mathematics, but it suggests a kind of Turing test for ATPs which could be granted the status of *artificial mathematicians* provided they were able to participate in a collaborative mathematical project and remain undetected by human participants. To pass this test, a machine would need to be able to read mathematical texts, develop mathematical ideas related to the project, and write them down as a human would (this last part of the test is indeed considered in the first chapter of this volume). This seems impossibly difficult, but may be more accessible to machines than the kind of knowledge needed to live in a human society with a human body, which machines can hardly experience. At least artificial mathematicians would be more worthy of artificial intelligence than the famous chatbot Tay, more adequately classified among the artificial deplorables.

But the challenge of designing such systems cannot be disconnected from the social significance of mathematics and the reasons that compel us to search, write, and teach mathematical proofs. One may argue that the reason for funding these activities is essentially economic, but as an economy is always required for funding anything, this is not a specific explanation. Jean Dieudonné asserted that mathematics exist "for the honor of the human mind." There is grandeur in this view of mathematics, but it seems incompatible with the use of computers to overcome our deficiencies. Whatever the reason may be, it certainly has to do with the discrepancy between our natural abilities and the

requirements of rational thinking. This suggests that learning mathematical proofs is not simply a challenge for students (or their teachers), but part of normal mathematical activity and deeply intertwined with searching and writing proofs, and learning mathematics. As noted by Jeremy Avigad in this volume, after having worked with the *Lean* ATP, "logic [was] relegated to the background, and the courses turned to the fundamental building blocks of mathematics." Both mathematicians and learners are engaged in constructing new mathematical content and shaping new mathematical concepts and tools. Mathematical proof as the cornerstone should not hide the beauty and essence of this construction.

Ultimately, one is always a student who tries to find a proof, and always a teacher who writes it down. The nature of the relation between these two aspects is hard to fathom, but should be considered a major issue in automated theorem proving and the teaching of proof.

REFERENCES

Aleven, V. (2010). Rule-Based Cognitive Modeling for Intelligent Tutoring Systems. In R. Nkambou, J. Bourdeau, & R. Mizoguchi (Eds.), *Advances in Intelligent Tutoring Systems* (Vol. 308, pp. 33–62). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-14363-2_3

Anderson, J. R. (1983). *The Architecture of Cognition*. Cambridge, MA, USA: Harvard University Press.

Anderson, J. R., Boyle, C. F., & Yost, G. (1988). *The Geometry Proof Tutor* (Advanced Computer Tutoring Project). Carnegie-Mellon University, Pittsburgh, PA 15213. Retrieved from http://act-r.psy.cmu.edu/wordpress/wp-content/uploads/2012/12/124GeoTutor.ABYost.pdf

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive Tutors: Lessons Learned. *Journal of the Learning Sciences*, *4*(2), 167–207. https://doi.org/10.1207/s15327809jls0402_2

Annals of mathematics. (n.d.). Statement by the Editors on Computer-Assisted Proofs. Retrieved 22 January 2019, from http://annals.math.princeton.edu/board

Assaf, A., Burel, G., Cauderlier, R., Delahaye, D., Dowek, G., Dubois, C., … Saillard, R. (2016). Expressing theories in the λΠ-calculus modulo theory and in the Dedukti system. Presented at the 22nd International Conference on Types for Proofs and Programs (TYPES 2016), Novi Sad, Serbia: Springer.

Baccaglini-Frank, A., & Mariotti, M. A. (2010). Generating Conjectures in Dynamic Geometry: The Maintaining Dragging Model. *International Journal of Computers for Mathematical Learning*, *15*(3), 225–253. https://doi.org/10.1007/s10758-010-9169-3

Balacheff, N. (2010). Bridging knowing and proving in mathematics An essay from a didactical perspective. In G. Hanna, H. N. Jahnke, & H. Pulte (Eds.), *Explanation and Proof in Mathematics* (pp. 115–135). Springer Berlin Heidelberg.

Bancerek, G., Byliński, C., Grabowski, A., Korniłowicz, A., Matuszewski, R., Naumowicz, A., & Pąk, K. (2018). The Role of the Mizar Mathematical Library for Interactive Proof Development in Mizar. *Journal of Automated Reasoning*, *61*(1–4), 9–32. https://doi.org/10.1007/s10817-017-9440-6.

Boero, P., Dapueto, C., Ferrari, P., Ferrero, E., Garuti, R., Lemut, E., … Scali, E. (1995). Aspects of the mathematics - culture relationship in mathematics teaching-learning in compulsory school. In L. Meira & D. Carraher (Eds.), *Proceedings of the Annual Conference of the International Group for the Psychology of Mathematics Education* (p. 17 pages). Recife. Retrieved from http://didmat.dima.unige.it/progetti/COFIN/biblio/art_boero/boero%26c_PME_XIX.pdf

Caferra, R., Peltier, N., & Puitg, F. (2001). Emphasizing Human Techniques in Automated Geometry Theorem Proving: A Practical Realization. In J. Richter-Gebert & D. Wang (Eds.) (Vol. LNAI 2061, pp. 268–305). Presented at the Workshop on Automated Deduction in Geometry, Zurich, Switzlerland: Springer Berlin Heidelberg. Retrieved from https://link-springer-com.gaelnomade-1.grenet.fr/content/pdf/10.1007%2F3-540-45410-1.pdf

Cobo, P., Fortuny, J. M., Puertas, E., & Richard, P. R. (2007). AgentGeom: a multiagent system for pedagogical support in geometric proof problems. *International Journal of Computers for Mathematical Learning*, *12*(1), 57–79. https://doi.org/10.1007/s10758-007-9111-5

Dawkins, P. C., & Weber, K. (2016). Values and norms of proof for mathematicians and students. *Educational Studies in Mathematics*, *95*(2), 123–142. https://doi.org/10.1007/s10649-016-9740-5

Duval, R. (1992). Argumenter, prouver, expliquer : continuité ou rupture cognitive ? *Petit x*, (31), 37–61.

Duval, R. (2007). Cognitive functioning and the understanding of mathematical processes of proof. In P. Boero (Ed.), *Theorems in school: from history, epistemology and cognition to classroom practice* (pp. 137–161). Sense Publishers.

Ebner, G., Hetzl, S., Leitsch, A., Reis, G., & Weller, D. (2018). On the Generation of Quantified Lemmas. *Journal of Automated Reasoning*, 1–32. https://doi.org/10.1007/s10817-018-9462-8

Garuti, R., Boero, P., & Lemut, E. (1998). Cognitive unity of theorems and difficulties of proof. In A. Olivier & K. Newstead (Eds.), *Proceedings of the 22th Conference of the International Group for the Psychology of Mathematics Education* (Vol. 2, pp. 345–352). Stellenbosch (SA). Retrieved from http://www.mat.ufrgs.br/~portosil/garuti.html

Gelernter, H. (1959). Realization of a Geometry-Theorem Proving Machine. In J. H. Siekmann & G. Wrightson (Eds.), *Automation of Reasoning* (pp. 99–122). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-81952-0_8

Goldstein, I. (1973). *Elementary geometry theorem proving* (AIM No. 280) (p. 46). MIT AI laboratory. Retrieved from https://dspace.mit.edu/bitstream/handle/1721.1/5798/AIM-280.pdf?sequence=2

Hähnle, R. (2001). Tableaux and Related Methods. In A. Robinson & A. Voronkov (Eds.), *Handbook of Automated Reasoning* (Vol. 1, pp. 101–178). Elsevier Science B.V.

Hanna, G. (2017). Connecting two different views of mathematical explanation. In *Enabling Mathematical Cultures*. Mathematical Institute, University of Oxford. Retrieved from https://enablingmaths.wordpress.com/abstracts/

Harrison, J., Urban, J., & Wiedijk, F. (2014). History of Interactive Theorem Proving. In *Handbook of the History of Logic* (Vol. 9, pp. 135–214). Elsevier. https://doi.org/10.1016/B978-0-444-51624-4.50004-6

Hauer, B., Kovács, Z., Recio, T., & Vélez, P. (2018). Automated reasoning in elementary geometry: towards inquiry learning. *Pädagogische Horizonte*, *2*(2), 14.

Healy, L., Hoelzl, R., Hoyles, C., & Noss, R. (1994). Messing up. *Micromath*, *10*(1), 14–17.

Herbst, P., & Balacheff, N. (2009). Proving and Knowing in Public: The Nature of Proof in a Classroom. In D. A. Stylianou, M. L. Blanton, & E. J. Knuth (Eds.), *Teaching and learning proof across the grades: a K-16 perspective* (pp. 40–63). New York: Routledge.

Heule, M. J. H., Kullmann, O., & Marek., V. W. (2016). Solving and Verifying the boolean Pythagorean Triples problem via Cube-and-Conquer (Vol. LNCS 9710, pp. 228–245). Presented at the SAT 2016, Springer. https://doi.org/10.1007/978-3-319-40970-2_15

Koedinger, K., & Anderson, J. R. (1990). Theoretical and Empirical Motivations for the Design of ANGLE: A New Geometry Learning Environment. Presented at the Knowledge-Based Environments for Leaming and Teaching, Slandford University. Retrieved from http://pact.cs.cmu.edu/pubs/Koedinger,%20Anderson%20-90.pdf

Kortenkamp, U., & Richter-Gebert, J. R. (2004). Using automatic theorem proving to improve the usability of geometry software. In *Proceedings of MathUI 2004* (p. 12). Retrieved from https://pdfs.semanticscholar.org/8892/faa455ea7442438d3f126bd05ba4d8c51e81.pdf

Laborde, J.-M. (1990). *Cabri-géomètre - Manuel de l'utilisateur*.

Leduc, N. (2016). *QED-Tutrix : Système tutoriel intelligent pour l'accompagnement d'élèves en situation de résolution de problèmes de démonstration en géométrie plane*. Université de Montréal, Montréal.

Luengo, V. (1997). *Cabri-Euclide : un micromonde de preuve intégrant la réfutation*. Université Joseph Fourier (Grenoble 1), Grenoble. Retrieved from https://www.researchgate.net/publication/34765259_Cabri-euclide_un_micromonde_de_preuve_integrant_la_refutation_principes_didactiques_et_informatiques_Realisation

Luengo, V. (1999). Semi-Empirical Agent to Learn Mathematical Proof. In *Proceedings of Artificial Intelligence in education (AIED 99)* (p. 10). Le Mans, France: Amsterdam : IOS.

Matsuda, N., & VanLehn, K. (2004). GRAMY: A Geometry Theorem Prover Capable of Construction. *Journal of Automated Reasoning*, *32*(1), 3–33. https://doi.org/10.1023/B:JARS.0000021960.39761.b7

Nevins, A. J. (1974). *Plane geometry theorem proving using forward chaining* (AIM No. 303) (p. 35). MIT AI laboratory. Retrieved from https://dspace.mit.edu/bitstream/handle/1721.1/6218/AIM-303.pdf?sequence=2

Newell, A., Shaw, & Simon, H. (1959). *Report On A General Problem-Solving Program* (p. 27). RAND Coporation. Retrieved from http://bitsavers.trailing-edge.com/pdf/rand/ipl/P-1584_Report_On_A_General_Problem-Solving_Program_Feb59.pdf

Newell, A., & Simon, H. (1956). *The logic theory machine - a complex information processing system* (No. P-868) (p. 40). The Rand Corporation. Retrieved from http://shelf1.library.cmu.edu/IMLS/MindModels/logictheorymachine.pdf

Pedemonte, B. (2007). How can the relationship between argumentation and proof be analysed? *Educational Studies in Mathematics*, *66*(1), 23–41. https://doi.org/10.1007/s10649-006-9057-x

Polya, G. (1945). *How to Solve It*. Princeton University Press. Retrieved from https://press.princeton.edu/titles/669.html

Sinclair, N., Bartolini Bussi, M. G., de Villiers, M., Jones, K., Kortenkamp, U., Leung, A., & Owens, K. (2017). Geometry Education, Including the Use of New Technologies: A Survey of Recent Research. In G. Kaiser (Ed.), *Proceedings of the 13th International Congress on Mathematical Education* (pp. 277–287). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-62597-3_18

Soury-Lavergne, S. (ed. . (2003). *Baghera an hybrid and emergent educational society* (Cahier du laboratoire Leibniz No. 81). Laboratoire Leibniz - IMAG.

Stefferud, E. (1963). *The logic theory machine: a model of heuristic program* (Memorandum No. RM-3731-CC) (p. 198 pages). The Rand Corporation. Retrieved from https://history-computer.com/Library/Logic%20Theorist%20memorandum.pdf

Takeuti, G. (1975). *Proof Theory*. Amsterdam: North Holland.

Trilling, L. (1996). Rétrospective sur le projet Mentoniezh. *Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation*, *3*(2), 157–162. https://doi.org/10.3406/stice.1996.1294

Wang, K., & Su, Z. (2017). Interactive, Intelligent Tutoring for Auxiliary Constructions in Geometry Proofs. *ArXiv:1711.07154 [Cs, Math]*. Retrieved from http://arxiv.org/abs/1711.07154

Webber, C., Pesty, S., & Balacheff, N. (2002). A multi-agent and emergent approach to student modelling. In F. van Harmelen (Ed.), *15th European Conference on Artificial Intelligence (ECAI 2002)* (pp. 98–102). IOS Press. Retrieved from https://telearn.archives-ouvertes.fr/hal-00003043

Weiss, M., Herbst, P., & Chen, C. (2009). Teachers' perspectives on "authentic mathematics" and the two-column proof form. *Educational Studies in Mathematics*, *70*(3), 275–293. https://doi.org/10.1007/s10649-008-9144-2

Yackel, E., & Cobb, P. (1996). Sociomathematical Norms, Argumentation, and Autonomy in Mathematics. *Journal for Research in Mathematics Education*, *27*(4), 458–477. https://doi.org/10.2307/749877