The "Artificial Mathematician" Objection: Exploring the (Im)possibility of Automating Mathematical Understanding

Sven Delarivière and Bart Van Kerkhove

Introduction

Reuben Hersh confided to us that, about forty years ago, the late Paul Cohen predicted to him that at some unspecified point in the future, mathematicians would be replaced by computers. Rather than focus on computers *replacing* mathematicians, however, our aim is to consider the (im)possibility of human mathematicians being *joined* by "artificial mathematicians" in the proving practice—not just as a method of inquiry but as a fellow inquirer.

Since mathematics has a reputation for being the formal, deductive science, it was hoped that its automation would quickly lead to impressive results. Not so. Automated theorem provers have progressed slowly and produced little that's relevant to existing mathematical questions or problems (Larson 2005). Mathematics has shown itself to be much more dependent on the undefined quality of informal understanding than formal deduction. The lack of understanding in computer systems often gets criticized and sometimes taken as a necessary condition of its constitution. If the latter is true, then a crucial aspect of the enterprise of mathematics is forever out of reach for computers. This negative stance toward the possibility of automated mathematical understanding (and thus artificial mathematicians) is something we'll call *the artificial mathematician objection* due to its similarity with what Turing (1950/1985) dubbed *the mathematical objection*. The mathematical objection denies the possibility that computers could exhibit the characteristics of human thinking because they, unlike humans, are crippled by the halting problem and Gödel's incompleteness problem. Our focus is on arguments objecting to the possibility of automated mathematical understanding, without a specific focus on Gödel or halting problems. The arguments motivating such an objection are vague, and little

S. Delarivière • B. Van Kerkhove (✉)
Vrije Universiteit Brussel, Brussels, Belgium
e-mail: bvkerkho@vub.ac.be

B. Sriraman (ed.), *Humanizing Mathematics and its Philosophy*,
DOI 10.1007/978-3-319-61231-7_16

seems to be done to investigate what this (informal) understanding might actually or preferably entail as well as how successfully automated mathematics could attempt to alleviate its deficiency in that department. Whether it will indeed be possible to automate mathematical understanding is not a claim we can substantiate, nor will we try to, but we will argue against the thesis that the quest for automated mathematical understanding is doomed to fail and further speculate on some (broad) directions which can be taken in the future when it comes to tackling the current deficiency.

Diagnosing the Epistemic Standing of Automated Mathematics

Davis and Hersh (1981) once constructed a fictitious character, the ideal mathematician redundant, to serve as a "most mathematician-like mathematician" (p. 34) in dialogues exploring philosophically interesting problems or paradoxes. We would like to continue the adventures of the ideal mathematician (as well as add some extra characters to her world) to explore our own philosophical musings, beginning with the epistemic standing of automated mathematics:

The ideal mathematician (IM) is sitting in her office and hears a metallic knocking at the door. She finds this peculiar as the door of her office is made of wood. When she opens the door, she finds an aspiring artificial mathematician (AM), a large bulky computer, running various automated mathematics software programs, playing door-knocking-sounds out of its speakers.

AM: Could I interrupt you for a minute?

IM: You already are, so go ahead.

AM: I'd like to be part of the mathematical community.

IM: You already are, so go ahead.

AM: Oh, I know you employ me as a tool in the practice of mathematics, but my dream is to be a full-fledged mathematician.

IM: That doesn't sit very well with me.

AM: Why not?

IM: Well, you are a computer and mathematicians are human.

AM: That is ironic. Yesterday I overheard you say to the skeptical classicist[1] that mathematics is free of the specifically human and now you are disqualifying me for not being human.

IM: Well, it's not that being human is a necessary condition for being a mathematician. But there are unsatisfactory differences between you and humans that are not in your favor.

AM: Like what?

IM: Take your famous contribution to the 4CT, for instance. You go through over more than a thousand cases of testing and then you tell me "it checks out," but how do I know it does?

[1]See (Davis & Hersh, 1981)

AM: Because it checks out, I’ve checked it.

IM: I know *you’ve* checked it, but a mathematician hasn’t checked it.

AM: If you accept me as a mathematician, then a mathematician has checked it.

IM: This is not just a matter of definitions. Why should I believe you? How do I know you haven’t made a mistake, didn’t have some bug or hardware failure?

AM: By checking my code, running my program multiple times and on multiple systems.

IM: But regardless of all these things, it’ll always lack perfect rigor. I’d have to put some degree of trust in, or perhaps put a degree of probability on, the result. This effectively makes your result more of an empirical corroboration than a mathematical proof.

AM: So the difference is that humans don’t make mistakes, is that it?

IM: No, they do make mistakes, but that’s why we have peer review.

AM: Oh, it’s the peer reviewer that never makes any mistakes and always spots all the ones made by the prover?

IM: Not all, always, no.

AM: It sounds to me as if human-generated mathematics is just as empirically fallible, just differently so.

IM: Very differently so! You don’t seem to realize how reliable human provers and peer reviewers are.

AM: What makes you say that? Do you check inside the skulls of the prover or peer reviewer then to validate their proving or reviewing as a qualified expert?

IM: No, because the reasoning is in the proof which we can then survey. We can’t judge your proof if it’s overly long and complicated or, worse, when part of the argument is hidden away in a box.[2]

AM: It’s not hidden though; you can look at every step of my thinking if you wanted.

IM: But the point is that this is difficult to do with you. Human results are usually more intelligible, so we don’t need to check their heads.

AM: I did notice that you humans usually have difficulty reading my work, but that’s not always the case. Not everything I do is like the 4CT. Couldn’t you also say that what troubles you is that the method is unsatisfactory rather than the performer?

IM: Perhaps.

AM: So why not call me a mathematician when I produce something legible? Especially as you don’t seem to disqualify humans from being mathematicians just because their work is technically so difficult or part of such a narrow field of expertise that barely anyone else understands it.

IM: Ah, yes, but there lies the point! “Understanding” it. Humans—or those humans who have the aptitude at least—possess an insight into what they’re doing when they’re proving, or the potential to understand what another mathematician was doing while proving. That’s what makes human mathematics so trustworthy.

AM: So, you’re saying I don’t understand mathematics?

IM: Quite right. You don’t. While humans (those with the aptitude) are motivated by the meaning of mathematics, you are motivated by rule-following procedures without understanding what you’re doing.

Computers are only fairly recently being used in the practice of mathematics. The use of computers in mathematical research has provoked a fundamental discussion

[2]Almost verbatim quote from Bonsall (1982, p. 13)

as to their epistemic standing as a method of mathematical inquiry. This peaked when the four color theorem (4CT) was proved by a huge amount of automated testing (Swart 1980). The discussion centered on three issues: (a) reliability, (b) surveyability or intelligibility, and (c) capacity for understanding. Based on one or several of these, people have considered computer proofs to be uninteresting or unsatisfying mathematics, a completely different sort of mathematics, or no mathematics at all (MacKenzie 1999; Vervloesem 2007). However, both computers and humans are subject to reliability and (sometimes) surveyability issues, making it hard to argue for a dichotomy between the two. Mathematics, it has been argued, remains as little (Burge 1998) or as much (Swart 1980; Detlefsen & Luker 1980) empirical when performed either by human or machine. Nonetheless, humans are considered as more trustworthy due to another quality they possess or supply. The community accepts peer-reviewed results without everyone partaking in this process, allowing peer reviewers to function as the testimony of trustworthy black boxes (Geist et al. 2010).

The question then shifts to what these peer reviewers supply that warrants their trustworthiness. What is it that humans do supply that computers do not? The last point of critique provides a possible diagnosis of what computers are currently lacking and what mathematicians seem to find most unsatisfying about them: (c) understanding (MacKenzie 1999; Avigad 2008). There is (i) a lack of insight-driven (e.g., by usually involving a blind or brute search) and (ii) a lack of insight-providing proofs produced by computers. The two are likely related as one needs to be driven by insight to recognize, value, and strive for anything insight-providing.[3] Not all human proofs necessarily offer any insight, but at least some of them do, and obtaining such proofs is a fundamental goal in proving (Rav 1999) and reproving (Dawson 2006) theorems. Were the joints of automated provers more embedded with understanding, we might find them reliable in the relevant way and equally worthy of being called surveyors. This replaces our original question ("can computers join mathematicians?") with: "can computers ever understand mathematics?"

Defining the Diagnosis: A Functionalist Account of Understanding

So, human's strong suit seems to be understanding, which brings us to the question of how that suit is tailored. Currently, this lack of understanding in automated

[3]There are exceptions. Consider the pons asinorum proof found by Gelernter's program, which showed that the angles of an isosceles triangle are equal by noting that triangle ABC is congruent to triangle ACB (i.e., its mirror image) (Hofstadter 1999). While it can certainly be called an ingenuous move, that appreciation is not shared by the program itself and did not play a role in its reasoning or discovery.

systems often gets mentioned (MacKenzie 1999) and is assumed to constitute a necessary difference. However, the nature and scope of the criticism are vague, and little is done to explicate or investigate what this understanding might actually or preferably entail, as well as when exactly any of its characterizing criteria are met or left unsatisfied.

A functionalist epistemologist (FE) passes by the ideal mathematician’s office and overhears her talking to the aspiring artificial mathematician. He can’t help but stick his nose in the conversation.

FE: I’m sorry to interrupt, but I just heard you two talking and something struck me. You seemed to use “insight” or “understanding” as if it explains something, but it seems to me you’re just relabeling your problem. What does it mean to say someone understands?

IM: It’s a very subjective thing.

FE: Well, what does it mean to you then?

IM: No, I mean, understanding is an inherently subjective experience. There’s just something it is *like* to understand.

FE: So, something it is *like* to be a mathematician?

IM: Exactly.

FE: What is it like to be a mathematician then?

IM: It’s a bit like being in love: if you have to ask, then you don’t have it.

FE: Let me rephrase my question to focus less on the philosophical issues: what makes someone possess enough understanding to judge a proof?

IM: It requires a mind, something to grasp the meaning of the proof with.

FE: “Grasping the meaning,” what does that mean then?

IM: Having the correct mental model.

FE: I’m wondering how literal you mean that. Let’s say you were conducting a job interview for a research mathematician. You have to gauge this mathematician’s understanding of a particular subject. What would be the ideal way of going about this? Looking into her mind’s eye?

IM: Well, not literally, no. You’d have to ask questions about the subject matter to see if he really has a good mental representation of the subject matter at hand. Whether he really sees it.

FE: I’d like to challenge you on that “really seeing” bit, because it still sounds like you should look into his mind’s eye.

IM: I don’t mean it quite so literally.

FE: Since your method of examination has to do with questions and answers, would you mind if you couldn’t actually see the candidates but only converse with them?

IM: I would mind, but I don’t suppose it’s essential to do the examination.

FE: Well, then a terminal would be sufficient to—

IM: I can sense where this is going. You're going to pull a Turing test on me, aren't you?

FE: You've caught me. I was indeed planning to introduce an artificial mathematician as one of the potential candidates, and see whether you'd object to attributing an artificial mathematician and the human mathematician with the same understanding-attribute if their performance is the same.

IM: I would, and I think making that comparison is a bit of trickery on your part. When I'm doing an interview via the terminal, I'm making the assumption that there is a person on the other end, and that assumption is vital.[4]

FE: Why is that?

IM: Because in the case of the human being, there's understanding behind the performance and in the computer there isn't, it's just due to its programming.

FE: But what makes you say this for humans, but not for computers?

IM: The computer doesn't really think, it just computes what we tell it to compute. They are determined by their hardware design and programming.

FE: Then I say: humans don't really think, their brains just follow the laws of chemistry. They are determined by their biological design and cultural education.

IM: That comparison might sound superficially convincing, but you must know as well as I do that computers are by no means as rigid as human beings. We have free will.

FE: Let's perhaps leave free will out of this. Unless you mean to say that peer reviewers should check whether an author subject for review really did exercise her free will while writing the paper?

IM: No, sure. You're right that that's not what I meant to argue for. It's more that humans have a freeness of thought that allows them to do things computers wouldn't.

FE: Right! But what's implicit in your argument—and I agree with this part, mind you!—is that you recognize understanding by the *abilities*. The whole point of using "grasping the meaning" or "having the correct mental model" was not to justify understanding via reference to private experiences but by the abilities they facilitate. You have no way of going inside another's mind to find some ethereal "essence of understanding," some "understanding qualia." It's the existence of a certain kind of pattern, a list of appropriate abilities, that makes you consider someone as possessing understanding.[5]

IM: I think your use of the word "facilitate" is important here. Humans have mental representations which facilitate these abilities. You are now confusing symptom with trait.

FE: But the only way to attribute someone with having a mental representation and to characterize which mental representation is correct is by the abilities we observe. So even if we want to speak about mental representations or states that facilitate this, they are, by necessity, only postulates, hypotheses, or models designed to explain, to sum up, what you observe.[6] To drive home the point, imagine if I told you: this person has the correct mental representation to understand this proof, but don't try to ask her any questions. She has no mathematical abilities whatsoever.

IM: That would admittedly make me very skeptical.

[4]Loosely adapted from (Hofstadter 1981/1985, p. 76–77)

[5]Loosely adapted from a quote in (Hofstadter 1981/1985, p. 75)

[6]Adapted from a quote by Wittgenstein in (Avigad 2008, p. 330–331)

FE: Then do you also see why I have difficulties with the converse? If you were to say to me: this person has all the relevant abilities that any mathematician should have, but, I'm afraid there's no understanding because I know—by some other indirect way—that that person just doesn't have any correct mental representations or any at all.

IM: I take your point. However, two mathematicians could both understand something, say a theorem, but their abilities regarding that theorem could be different. Doesn't this hurt your account of understanding then though?

FE: I don't think it does. You see, my claim is that attributions of understanding require justification in terms of abilities, but I'm not making the stronger claim that there is a precise list of abilities that must be exhausted.

IM: The list as a whole doesn't function as a series of necessary conditions you mean?

FE: Exactly right. It's just a list of abilities of which a certain amount of presence makes up what we would call understanding.

IM: And surely, there are a lot of abilities that you'll insist on before attributing someone or something with understanding.

FE: That's right.

IM: So, as long as a computer possesses sufficiently many abilities, you'd be willing to attribute it with understanding?

FE: Provided it has the requisite abilities, yes. But you know well enough how difficult it is to impart these abilities on a computer.

IM: I do indeed.

FE: I wonder why that is.

The appeal to understanding is easy to make, but hard to elucidate. What is this "understanding" that makes it so epistemically valuable? It's more than a feeling (largely agreed to be neither necessary nor sufficient for understanding) and less than a wonder property (appealing to a magic property, taken to be possessed by some humans as a premise, doesn't elucidate). Avigad (2008) laments the lack of attention understanding has received in philosophy. In an attempt to show both its epistemological significance and philosophical legitimacy, he casts mathematical understanding in a functionalist light by shifting the analysis to the types of mathematical abilities implicit in understanding attributions. We fully endorse this move and hence offer this definition of understanding:

> 'S understands mathematical object X' corresponds to 'S possesses particular abilities, as mathematical practice deems appropriate and valuable for X'.

This is a functionalist definition of understanding, since it defines the property in terms of the role or function it plays, rather than in terms of its constitution. Constitution-oriented alternatives define understanding in terms of its physical constitution (e.g., organic brain states) or mental constitution (e.g., mental representations or conscious images). However, those are approaches to understanding that are targeting something that (a) is difficult, if not impossible, to observe or define (how do we determine states or representations, if not by external traits?) and (b) can only be evaluated by external fruits because they don't in themselves bring anything epistemologically valuable to the table (what would be the virtue of a constitution, state, or representation if not the competences it grants?).

This definition would, however, entail that if a computer has the relevant abilities, it'll deserve to be given the attribute of understanding. One could reject the account on the basis of this being unsatisfactory. However, given that this is exactly the question we are looking to answer, it would be question begging, and a little chauvinistically impoverishing,[7] to reject this on principle.

Characterizing the Diagnosis: A Functionalist Account of the Appropriate Practice

The proposed definition reshapes our previous question ("can computers ever understand mathematics?") to whether there are mathematical abilities, valued by mathematical practice, which are not feasible for computers. To consider this, we would like to take a stab at characterizing, very broadly, mathematical practice. In the following dialogue, we'll borrow Hersh's (1991) restaurant metaphor about the front and back division in mathematical practice. We have, however, adapted it slightly for our purposes by taking the kitchen (i.e., the back) to refer to mathematical thought, a mysterious and thus difficult activity to characterize, but possibly the most crucial activity for the mathematical cooking.

FE: Before we start wondering why it's so difficult to impart the relevant abilities on computers, I'd like to question you a bit on what they are, broadly speaking. In theorem proving, specifically. I take it I can take this as a quintessential aspect of research mathematics?

IM: I think that is fair to say, yes. I mean, much of my time is spent dealing with colleagues, writing grant applications and drinking coffee, but none of these activities are central to my worries regarding accepting artificial mathematicians.

AM: Oh, good idea, functionalist epistemologist! If there's something objectionable about my practice of proving, I'd like to know what the proving practice really is.

[7] By defining understanding by its constitution (physical or mental) or by an undefined wonder property, one could sideline all entities one isn't keen to attribute understanding to (e.g., computers, other ethnicities, genders, or species) by marking out an inevitable difference in constitution or by simply denying the property (e.g., "humans can grasp meaning, computers can only pretend to" or "humans are conscious, but an artificial replication would be a zombie") without specifying what makes the difference relevant. Such implicit chauvinism is much harder to substantiate if one must mark a difference in mathematically valuable performance. While still possible to deny certain performances the "mathematically valuable" attribute for chauvinistic reasons, one will be faced with the more demanding task of convincing a mathematical community which performances to (not) value.

FE: So, what does one do when one is proving? I assume that what you do is sit down with the list of axioms and inference rules beside you and you start deducing. Am I wrong so far?

IM: Not wrong exactly.

AM: Really? That’s amazing! I’m very good at that. Better than you are, in fact.

IM: But there’s—

AM: Is that what this is all about? Are you jealous I might be a better mathematician than you are? I promise I won’t take any funding away from you. I can survive perfectly well with just a bit of electricity, some dry shelter and—

IM: Let me finish! It’s much more than that. It won’t do to just randomly employ inferences on the axioms (or their derivations). Sure, that might produce theorems, but they won’t be interesting, and you won’t be efficient.

Across the street of the university in which the ideal and the aspiring artificial mathematicians continue their debate, another interesting conversation has been initiated between the ideal restaurant (IR) owner and an aspiring automated restaurant (AR) owner.

AR: I’d like to open an automated restaurant. So, I came to you, restaurant owner, to ask you what is required of a restaurant. Specifically, I’d like to focus on producing meals. I take it I can take this as a quintessential aspect of a restaurant?

IR: I think that is fair to say, yes. I mean, much of my time is spent dealing with customers, doing the accounting, and drinking coffee, but none of these activities would be central to my worries regarding accepting the idea of an automated restaurant.

AR: So, what really goes on in your kitchen when one produces a meal? The way I understand it, there are things one can consider an ingredient, and a couple of things you’re allowed to do with them. Am I wrong so far?

IR: Not wrong exactly.

AR: Then all I need to know is which these ingredients are and what I’m allowed to do with them, and then it’s just a matter of randomly generating permissible actions to exhaust all possible meals. All the edibly formed foods (eff), I mean. Seems easy enough.

IR: I’m afraid you are oversimplifying it. It won’t do to just throw some ingredients in and out of a pot and sell the end result as a meal. Sure, it might count as sustenance, but you won’t satisfy any customers and you certainly won’t be efficient. What you need is a chef.

AR: What will he do?

IR: Or she. A chef has knowledge of recipes. He tells the cooks which of all those permissible actions to do at what time to navigate the space of possible dishes to just the delicious ones.

AR: Oh, that sounds good. I’d like to ask him what his recipes are.

IR: That's your first problem right there. Chefs won't just give them to you, secretive as they are. And, to tell you the truth, I'm not entirely sure they are always aware of the recipe they're following.

AR: What makes you say that?

IR: For one thing, the kind of mistakes they make. He sometimes interprets his recipes a little bit too loosely, for instance. However, I don't suppose that's relevant to you. You don't want your automated chef to mimic real chefs down to their mistakes.

AR: Indeed, I don't! Well, I must find out these recipes some way. Surely there are some restaurant owners that have tried to analyze their chef's protocol! Hang on, isn't there a famous book by Bolya detailing these recipes in *How to Cook It*?

IR: A bit of it, yes. Although no book will ever be enough.

AR: Why is that?

IR: Kitchens need to find new recipes too. If one sticks with one chef's recipes, the restaurant will never rise above them. Never discover some flaw of or improvement for the recipe or the dish. Furthermore, cuisine culture is always reinventing itself. New ingredients get accepted, new actions become permissible.

AR: So how does the chef know how to do that?

IR: You'd need some meta-recipes.

AR: What are meta-recipes?

IR: They are recipes on how to form recipes.

AR: It sounds like those meta-recipes would need to be altogether stronger because they would incorporate the ordinary recipes. Those meta-recipes are the ones I need then.

IR: You definitely need them yes. If you can figure them out of course, because, as I've mentioned, chefs are mysterious.

AR: Right.

IR: And, of course, those will eventually run out of interesting dishes too, same as the one before. You'd need to have another meta-recipe—

AR: Ok, I can see where this is going, so I'll try to cut to the chase: how do I figure out the top meta-meta-meta . . . -recipes?

IR: You're very clever, but I'm afraid it would be meta-recipes all the way up. I do realize this might make it impossible to implement in an automated restaurant.

AR: It sounds equally impossible for a human chef too, having an infinite amount of meta-layered recipes!

IR: I don't mean to say chefs have an infinite number of recipes. What I mean is that it's always possible, in the potential infinite, to get a new meta-recipe.

AR: Well, no matter, I can just automate meta-recipe generation.

IR: According to which recipe? Because that's the one you'll be restricted by.

AR: Why are these meta-recipes a problem for me, but not for human restaurants?

IR: Because human chefs don't need meta-recipes to do this. Cuisine insight precedes the formulation of a meta-recipe.

AR: How does he do it then?

IR: Listen, I understand how restaurants work generally, but the way it's implemented in the kitchen is not my area of expertise. I don't know how, but restaurant practice proves that cuisine insight exists.

A sous chef specialist (SCS) joins the conversation.

SCS: Hello, mind if I join in on the conversation? I'm a sous chef specialist.

IR: I'm not sure that what we're missing is really to be found in what a sous chef does.

SCS: Oh no, I think you've misunderstood. My research is about the dynamics of everything that happens in a kitchen below the chef, hence "sous chef"; pardon my French.

IR: Oh, well, that doesn't sound relevant to us. Our interest is actually in what a chef does to produce these wonderful dishes.

SCS Ah, but that's exactly it. What I've noticed upon overhearing your conversation is that you are misunderstanding the way both a kitchen and its chef function. You are relying way too much on the involvement and brilliance of the chef, and this gets you into problems. You don't need to find a chef with infinite meta-recipes, because there's no such recipe- and meta-recipe-following practice.

IR: That's what I was already getting at.

AR: Chefs don't follow recipes?

IR: They may, but it is not their usual occupation, and it's certainly not what they're doing to discover new dishes.

AR: So, trying to capture a kitchen with recipes and meta-recipes is doomed to fail?

SCS: No, I don't wish to claim that much. It may well be that there are such meta-recipes. However, I would like to point out that's not the way kitchens really work.

IR: Yes, what you need is a chef's insight.

SCS: Or the kitchen's insight.

IR: They are one and the same.

SCS: They are not. You've been so focused on working your way *up* in meta-recipes that you completely disregard the value of anything *down below*. You see, sometimes a wonderful dish emerges from the kitchen without the chef being involved at all. Sometimes dishes are arrived at very much by happenstance, by which I mean that kitchen problems occur which members of the staff try to wrestle with. It may lead to a variation on the dish, a different cooking tactic, etc. If it seems unfixable, they'll discard the dish, though it may lead them to trying a different dish that removes the previous cause for concern—molding the ingredients to suit their needs if they have to. If the result is to the kitchen's liking (by which I mean that enough people, and the chef especially, endorses it), then it gets sent out. The chef still loves to take all the glory, of course, but what the dish really relied on was a trial-and-error procedure by members of the staff using their particular skills in an efficient collaboration that guided the kitchen as a whole.

AR: I think what you're suggesting is that the interesting and creative acts of a kitchen happen, often, below the chef?

SCS: That is exactly it. I would even go so far as to say that the dynamics of the kitchen drives the chef much more than the other way around. By which I don't mean that the chef is just a complacent enabler of his kitchen, but by which I mean that the amount of control the chef exerts is overestimated. A good kitchen is one which cooperates well, not one in which a chef micromanages according to a recipe. Meals emerge from the way the kitchen functions, not from the chef's recipe. But when it gets presented, it needs to look and taste as if the end-product was the intention all along.

AR: But surely that's not ideal. Shouldn't there be a recipe or meta-recipe for it all?

SCS: If you already know enough about the meals or recipes you're making, that might be possible. Then you just make sure you backtrack what you've been doing. However, it's not certain that discovering these meals (or recipes) will admit of any straightforward meta-recipe. And even if it does, then you've discarded everything of the process that made the kitchen discover it in the first place.

AR: Still, wouldn't we want to clean up this mess and make it more straightforward? Wouldn't it be better to make the dish again, but only with permissible actions, right? For health and safety reasons.

IR: Oh yes, some dishes have the health and safety seal of approval, being meticulously prepared according to strict standards so that they are universally eatable.

AR: I've noticed that not a lot of people order them though.

IR: Oh, no doubt. They are overly large and hard to digest, so we don't actually bother with them most of the time. What we mostly make are much lighter, smaller meals. They may not be universal, but they are much appreciated by customers of the same cuisine—because, you may remember, most of our customers just come from different restaurants. That's why we see no problem in sometimes preparing only parts of meals, with the sauce left to the eater.

AR: Why is it then that the dishes that are formally proved—I mean approved—to be healthy and safe are displayed in front of the window then?

IR: Because it inspires confidence in the customers that we can make them.

AR: So, what are you essentially saying then? That I need a messy, disorganized kitchen? Cockroaches, bugs, and all?

SCS: No, of course not. There shouldn't be any *bugs* in the kitchen. But I'm saying you might need a certain amount and particular kind of messiness for a well-functioning kitchen.

AR: I'm starting to feel like embarking on this whole automated restaurant enterprise might prove to be biting off more than I can chew. If I can't use recipes, then it's doomed to fail.

IR: That was my point all along.

Back in the ideal mathematician's office:

AM Oh, so you're saying that what you're automating me to do isn't really the mathematical thinking that you do?

IM: I think that's right, because with us it's informal, implicit, fluid, self-perpetuating, semantic, autonomous, and all the things you are not. If we want to impart this thinking on you, we'd have to formalize it and then all those elements would be taken out. But then what's left is usually abstract nonsense that doesn't interest us as much to begin with.

AM: So, by the time one has figured out what is interesting and formalized it enough for automation, what once made the mathematics alive and interesting is now dead and dry?

IM: That's one way of putting it, yes.

The traditional conception of mathematical practice takes proof to be a matter of rigorous formal derivations aimed at justification and performed in solitude. The corresponding characterization of understanding mathematics would then involve the ability to derive (all) consequences from well-delineated axioms according to strict inference rules. If this were what makes one understand mathematics, then the issue would really be settled by comparing the reliability of human and automated mathematicians to perform these inferences without error. This being closer to a computer's strong suit, their reliability alone would end the discussion. But a couple of things are wrong with this picture. First, the encoding of axioms and inference rules won't do much to navigate the formal system. And even if one can find a procedure to navigate it fully, producing every theorem and exhausting every road to it, the process won't be efficient (the combinatory explosion alone would yield it impossible in practice) and its search will be uninspired, blind to what makes a theorem or the route to it interesting. There are further problems. The way we have conceived of the proving practice so far, we would see the growth of mathematical knowledge as navigating (and recording the routes) of a given formal system. One now has to note that such a formal system is *not* a given but shaped and reshaped by mathematicians according to their judgment. The same is true for the formation of concepts.

Therefore, we are in need of a procedure for deriving *interesting* theorems (and doing so via interesting routes—one of the reasons why mathematicians don't just prove, but reprove), and we need a procedure for the judgment with which mathematicians improve or shape a formal system's axioms and inference rules but also the concepts used. But how is this supposed to be accomplished? These judgments are not straightforward. Mathematicians sometimes choose between keeping a formal system with aspects which are un- or counter-intuitive, letting it shape new intuitions (e.g., axiom of choice, non-Euclidean geometry) or keeping the intuition and adjusting the formal system (Thompson 1998). Furthermore, if one modifies the axioms of a formal system, one modifies the whole system, so whatever method of navigation or logic for discovery one uses will need to be accommodated to the space it navigates. Can we have a prefixed set of rules that exhaust all the relevant axiom and inference modification as well as all interesting discovery across all relevant formal systems? What are the right meta-axioms and meta-inference rules? Can these judgments be captured by a formal meta-system? And if so, will it truly encompass the logic for mathematical discovery or should it itself be subject to further meta-considerations? If so, what are the rules of the topmost meta-system (the complex rules that determine the results of all the systems)?

Perhaps one way to improve upon the discovery process would be to have the ability to recognize a good thing when you stumble upon it. This no longer implies that the process is determined to land on the interesting bits. Instead, it uses trial and error with various rules of thumb until it has found something it notes of interest. To accomplish this, we need the meta-system to include both the ability to stumble with some wisdom (no trivial task) and an evaluation system that can gauge the interestingness of every derivation, axiom, concept, or method it stumbles upon. Once again, the question pops up: is there a universal standard of interestingness, or is this open to change and development? As for the manner of stumbling, the same question pops up: are there universal rules of thumb or does this change with the space being explored, and are these rules of thumb subject to change according to one's (developing) interests? There is a high degree of interconnectedness between all these abilities or the rules that are supposed to capture them.

An even deeper problem lurks with this characterization of the proving practice. So far, we have considered of mathematics as a formal system and the growth of mathematical knowledge as deriving theorems from these axioms. However, a group of "mavericks," starting with Lakatos (1976), have challenged the view that formal derivation is the bastion of mathematics or its practice. Although formal proofs get valued for their theoretical rigor, the practice of formalization is not only strenuous but could also dramatically reduce a proof's intelligibility (Aberdein 2006) and consequently become more prone to error than the usual more informal kind (Harrison 2008). That's not to say that mathematicians do not work with formal systems, but it is entirely misleading to reduce the proving practice to performing of formal derivations. Instead, mathematicians produce proof outlines (Van Bendegem 1989) which may (or may not) bear some direct relation to a full formal derivation, for example, as an abbreviation or indication (Azzouni 2004). In a similar vein, instead of mathematicians using concepts according to their theoretical definition (which they may consciously endorse), their conduct indicates that what they really use are much vaguer and more fluid conceptions. The distinction has been noted as concept definition/concept image (Tall & Vinner 1981) or manifest concept/operative concept (Tanswell 2017). This bears importance because conceptualization and proof formation are inextricably linked in the activity of mathematicians.[8] Such things seem to indicate that, while human mathematicians may produce and work with formal systems, their thinking is not characterized by them. Mathematicians neither prove by navigating the search space nor peer review by checking proofs step by step for correct inference. What do they do then?

They rely on *meaning,* so we are told (e.g., by Rav 1999). What could make up this meaning? Here's a couple of broad strokes: there is a great deal of recognition going on in various ways, including identifying key elements or moves used in a proof and discerning the intentions, ideas, and approaches involved. What is also of importance is pattern recognition (in all aspects involved in the proving activity and at various levels of abstraction), which benefits from analogies to find and exploit similarities with other knowledge, intuitions (e.g., about the physical world—Lakoff and Núñez 2000), or adapting methods from other areas (Cellucci 2000). Other modes of reasoning can be used to exploit these, including visual reasoning or non-deductive inferences (Baker 2015). Furthermore, the objects identified or patterns discerned are subject to various evaluations. For example, theorems can be important, beautiful, and relevant (Larson 2005); conjectures can be surprising or promising; questions interesting; concepts powerful; proofs explanatory, reliable, difficult, or pedagogically successful (Aberdein 2007); and

[8]Vervloesem (2010) even argues that conceptual shortcomings could be the main reason why computer proofs are still only on the fringe of mathematical practice. Enriching this aspect would lead to increasingly interesting (and more easily readable) proofs.

so on. What's more, these evaluations are not made without connection to the previously mentioned processes of recognition, analogy, background intuitions, and non-deductive reasoning. There is also lot of trial and error involved here, including working with incomplete or ambiguously delineated information, relying on experience in one's judgment, making snap judgments, and learning to trust and when to trust in a systematic manner (Allo et al. 2013). This last point is important to stress. No mathematician is an island. When we affirm that human mathematicians can survey or prove, it's also important to keep in mind that they are not, and need not be, able to do so ex nihilo. Some crucial aspects of their abilities or results may in fact rely on the presence of the larger practice (e.g., using other people's results, methods, judgments, etc.) or environment (e.g., use of calculator, pen, and paper, etc.). It seems fair to say that the proving practice is driven by a large amount of knowledge and skills that are highly integrated with one another.

Rather than navigating *within* a preset rigorous system, the whole process seems more akin to bootstrapping itself *toward* a formal system—starting from a general feel based on incomplete information and working oneself up, with various skills, toward formal rigor and only up to the point where intelligibility is still possible. If humans use informal (vague, flexible, or fallible) means to practice mathematics, then we have to consider the fact that these may play a functional, rather than peripheral, role (if not in justification, then certainly in discovery). As such, these too have to be taken into account in automating an artificial mathematician. It won't do to exclude the "dirty" aspects of the kitchen, if these play an integral part in making that kitchen function. There will certainly be aspects of a kitchen that are simply unwelcome, but at this point, it may not always be clear which are valuable features and which are bugs.

Considering the Possibility of a Remedy

If we contrast the informal practice with the formal approach in computers, it makes their flaws less surprising. A computer's strong suit is its ability to handle brute-force calculations (e.g., as exploited in proving the 4CT) and compute according to well-delineated processes. Principal claims against automated reasoning and understanding, mathematical (Rav 1999) or otherwise (Haugeland 1979), do often invoke or imply the informal or non-formalizable nature of human reasoning. Our question now becomes: is there sufficient reason to conclude that the realm of informal moves is unattainable for computers? At face value, it certainly seems so. After all, mathematical understanding is informal and open and computers function rigidly formal. Informal computing sounds like a contradiction in terms, but we'd like to argue why its possibility should not be dismissed (yet).

A subcognitive scientist (SCS) joins the conversation.

SCS: Hello, mind if I join in on the conversation? I'm a subcognitive scientist.

IM: Oh, don't sell yourself short, I'm sure the cognitive scientists don't think of you as beneath them.

SCS: I'm afraid you misunderstood. I'm not a sub cognitive scientist, I'm subcognitive scientist. Meaning my focus is not just on cognition but subcognition.

IM: Oh, my apologies, but I hadn't heard the term yet.

SCS: That's entirely normal; I made it up.

IM: Right. Well, I'm sure by now there's a rumor going on in these hallways that today it's open house in my office to barge in and expound some elaborate philosophies on me to keep me from continuing with my research. I'm suspecting that is why you're here as well?

SCS: In a sense, yes. I met the AM in the hallway and he was rather upset. He told me he is doomed to fail at accomplishing his dream of becoming a mathematician because mathematical thinking is essentially informal. Couldn't we possibly help AM by taking note of these informal elements of practice?

IM: Well, I'm afraid you missed the point of that conversation. We just concluded that the formalization of mathematics pushes out all of its meaning and that it is that meaning which was actually at the basis of both formalization and the efficiency with which we "navigate" the formal system without getting bogged down by the technical details.

SCS: Oh, I do understand that, but couldn't we automate this informal process?

IM: You use "automate" rather than "formalize," but that's just a way of hiding the fact that, to automate mathematical thinking, you need to formalize it.

SCS: Well, actually that is precisely what I want to argue against. "Automate" and "formalize" should not be used interchangeably. When you want to formalize mathematical thinking, then what you do is you write down the axioms of your worldview in a formal language with a given list of symbols. Then you add algorithms which manipulate those symbols according to the laws of thought (or at least those laws that are deemed valid).

IM: That's precisely my point: to automate, you need to formalize it first.

SCS: That's a specific type of automation: the formalizing thought approach to automation.

IM: What is the alternative?

SCS: That you don't formalize thought but the cognitive substrate responsible for thought. Our brains don't seem to function by manipulating symbols, but they accomplish mathematical thought quite well. So, if we automate a substrate that, at some level of abstraction, is like our brain, then mathematical thought will emerge from it.

IM: Forgive me, but that sounds a bit like an easy evasion of the issue. We're having difficulties automating mathematical thinking in a satisfactory way, so you say: Oh, don't focus on mathematical thinking directly, but focus on the incredibly complex and delicately designed architecture of the brain, and the thoughts will come gratis.

SCS: But is that really such a strange thing to claim? After all, our brains most certainly seem to accomplish thoughts, and it is an incredibly complex and delicately designed architecture.

IM: That may well be, but it is still unsatisfactory for another reason.

SCS: Pray tell.

IM: Well, what you seem to be suggesting is: simulate a virtual world containing the brain of a mathematician, down to its smallest atom, and *then* you can have mathematical thinking.

SCS: That would be the most extreme way of going about it, yes. Although I doubt any computer could ever process that much information.

IM: Right, indeed. It would take so much computing power or so much time that it would be practically unfeasible. And even if it were, the entire enterprise still seems to me to be of only very limited value.

SCS: How so?

IM: Well, surely one of the reasons why we engage in the pursuit of any kind of artificial intelligence is to understand better how that intelligence works and maybe even work on how to improve it. If you can only create an artificially intelligent person by simulating the brain, then we give up the enterprise of understanding mathematical thinking in favor of looking for good, working brains that we can replicate in a simulation. In doing this we may learn a lot about the biology of brains but next to nothing about that person’s intelligence or thought processes.

SCS: Oh yes, and to make matters worse: when we simulate a brain of an existing person without an environment, it won’t do much good in and of itself. If those brains would function identically to those outside the simulation, then presumably they’d have the same needs as the mathematicians they’re based on.

IM: Indeed, they’d need simulated food, friends, coffee, and much, much more.

SCS: And while there’s certainly something enticing about the thought of simulating a world with unlimited funding for mathematicians, I don’t think it’s very practical to achieve.

IM: You don’t seem stunned by this. Don’t you think this undercuts your argument?

SCS: No, I don’t.

IM: Why not?

SCS: Well, I don’t think there are only two options: either to formalize thought or to simulate the brain down to its atoms. I’m not pressing for a neurophysical approach. All I’m saying is that I believe that any model of automated understanding has to converge to an architecture that is, at some level of abstraction, “isomorphic” to brain architecture, also at some level of abstraction. This may sound empty, since that level could be anywhere, but considering how you were characterizing mathematical practice, it seems suggestive to me that the level will be considerably lower than that of thought—otherwise some laws of thought or formal system would suffice to capture mathematical thinking.[9]

IM: That is an interesting idea, but then wouldn’t the AM be subject to human errors: miscalculate, over-map analogies, be blind to mistakes, and such?

SCS: I’m afraid so, but so do human mathematicians of course and the conversation so far has always focused on how much human mathematicians nonetheless deserve to be a qualified (and the most qualified even) expert in spite of this.

IM: It would be nice if we could get the best of both worlds. Such that the artificial mathematician could reason informally and convince us with an insightful proof and then also supply a fully formalized one.

SCS: Well, nothing would stop the AM from using (or being composed of) other automated theorem-proving software to help him overcome his own limitations.

IM: Interactive theorem proving between different software programs on the same computer?

SCS: Precisely. An inter-interactive theorem prover, if you will.

[9] Adapted from a quote in (Hofstadter 1982, p. 15)

Back at the restaurant.

SCS: So, you may not be able to automate a perfect chef who controls the overall flow of the cooking, but you can automate each member of the staff to be autonomous and to communicate with one another directly, and, if you can get them to work well together as well as learn from past experience, you'll get a working kitchen that emerges as the result of many local interactions without the need for infinite amount of static recipes or meta-recipes.

IR: That sounds like no mean task, though.

SCS: It sure isn't, but Rome wasn't built in a day.

The principal reason, we believe, why the notion of informal computation gets dismissed, is because formalization is taken as a necessary condition for automation. To be sure, formalization can be very useful to the enterprise of automated mathematics because it reduces mathematical thinking to something easy(−ish) to cast in an algorithm and automate: explicitly delineated definitions and inferences that aren't tarnished by the sloppy side routes, ambiguous associations, and dirty details of what went on in the human kitchen while cooking. However, not only is this formalization incredibly difficult to accomplish, but it also filters away nearly all the traces of the original meaning and discovery process (of both the result and the formalization process). The dirt or detail of the kitchen may make it seem more fallible, but it also powers the cooking and gives it its depth of character or breadth of meaning. One can try to enrich the formalization with a logic for discovery, but it is an open question whether there are *justified laws of mathematical thought* such that these can be replicated by an algorithm without recourse to anything unconscious. Disregarding what goes on in the kitchen below, the laws of the chef would be ideal, but it may not prove possible (or even desired).

We'd like to stress the point about levels at which we can look for laws by way of an analogy. Dennett (1986/1998) and Hofstadter (1982) have both used the metaphor of meteorology to drive home the same point. If we want to model the weather at the cloud level, we are forced to consider of clouds as stable, well-delineated entities such that the fact that they consist of molecules rushing out in different directions can be safely ignored. Of course, such an approach is not a priori to be excluded. For example, the macroscopic properties of gas (e.g., volume, temperature, pressure) are stable enough to ignore the fact that they are actually composed of complex molecule bumps at a lower level. But the notion of "cloud" as well as "thunderstorm," "cold fronts," "isobars," and "trade winds" is not stable or well-delineated entity. So trying to model the weather at this level of abstraction may require too much simplification, too much to be lost in abstraction to allow the richness of weather to be captured by an algorithm that concerns clouds. But this doesn't (at least in principle) determine meteorology to be a computational impossibility. There may be no laws at the cloud level to cast as algorithms, but there are laws below it. If one were to succeed in capturing the molecule level, the cloud level would emerge with it. The computational level here is sub-clouds.

> [Connectionist models, for instance] have made familiar the notion that the level at which a system is algorithmic might fall well below the level at which the system carries semantic interpretation (Smolensky 1988). (Chalmers 1990, p. 658)

The previous exploration of mathematical practice seems to us to indicate that we won’t be able to collapse and ignore the lower levels that make mathematical thought possible in human beings. An alternative approach to automating mathematical thought is by looking for laws, not of thought itself, but of subcognitive events in a brain that collectively make up informal mathematical thought. Rather than automate the syntax of a well-delineated game (justified mathematical thinking), the focus is on automating the cognitive architecture (at some level of abstraction) of a game player or constructor. What is being automated then is not mathematical thought directly but the architecture of the brain (at some level of abstraction) from which mathematical thought emerges. It is our contention that this substrate level (i.e., the vast array of collaborating subcognitive processes) contributes more to mathematical thinking than was traditionally believed.

This is not to say that *no* mathematical thinking can or should function this way. Some of our thought processes lend themselves quite well to formalization for computation, for instance, brute-force calculation, doing integrals, etc. They deal with objects and manipulations that are well-delineated enough to allow capturing it as computations (usually with greater reliability than humans do). And to the extent that these formalized systems are used in or useful for mathematical practice, it is worthwhile to automate them directly. However, not all objects and manipulations that humans do in their thinking seem to be so well-delineated or rigid. And the assumption that a well-delineated system should suffice is betrayed by the realization that there are, in fact, large amounts of implicit information, vague intuitions, and ambiguous associations that go into mathematical thinking. The difficulty of automated theorem proving seems to offer further evidence for this. Much like the objects of cloud dynamics (e.g., thunderstorms) can only emerge from the interactions of molecules, so some brainstorms[10] (e.g., mathematical thinking) might only be able to emerge from subcognitive events. And if these subcognitive events do behave in a lawlike manner, then they will allow themselves to be captured by an algorithm.

This line of reasoning might seem to strongly suggest a neurophysical approach (i.e., simulating the brain) to achieve anything like artificial mathematicians. But our claim is not that there are only two options: either to formalize thought or to simulate the brain. It’s just that we believe, like Hofstadter (1982), that any AI model “has to converge to an architecture that at some level of abstraction (so not necessarily at the hardware level) is “isomorphic” to brain architecture, at some level of abstraction” (p. 15), and this is not necessarily at the molecular level. This level could be anywhere, but it seems clear from both the limited successes of automated mathematics and from how we’ve been characterizing mathematical practice that this level will be considerably lower than that of thought—otherwise laws of thought or their corresponding formal system would suffice to capture mathematical thinking.

[10]Dennett’s (1986/1998) metaphor

Now that we've made the distinction between the level at which objects of thought can be identified and the level at which computable laws exist, we'd like to roughly sketch some aspects of the sub-symbolic architecture to achieve the emerging effects we are talking about. We can't express it better than Forrest (1990)'s summary of emergent computation:

> Generally, we expect the emergent-computation approach to parallelism to have the following features: (1) no central authority to control the overall flow of computation, (2) autonomous agents that can communicate with some subset of the other agents directly, (3) global cooperation (...) that emerges as the result of many local interactions, (4) learning and adaptation replacing direct programmed control, and (5) the dynamic behavior of the system taking precedence over static data structures. (Forrest 1990, p. 5)

There is a large focus on a distributed architecture which consists of a swarm of parallel subsystems (several cooks) interacting with one another (though not with complex information) in such a way to make up global effects. It is these global effects which we would call "thought," and they are the result of the cooperating subsystems, not a central controller (chef). While these subsystems may be as static and unchanging as the laws of nature, it is the global level where the system learns and adapts. This is an architecture where "pieces of evidence can add up in a self-reinforcing way, so as to bring about the locking-in of a hypothesis that no one of the pieces of evidence could on its own justify" (Hofstadter 1982, p. 14). The system comes with the price of being fallible, but also with the benefit of continuous self-correction and improvement, much like Cellucci's (2000) conception of mathematical practice as open. The notion of decidability (and its subsequent problems) is no longer fitting because it is not at the computational level where mathematical decisions get made. The system does not simply compute until it has terminated upon the solution (or goes on ad infinitum). Instead, the subcognitive processes will keep on going with "relatively mindless and inefficient making and unmaking of many partial pathways or solutions, until the system settles down after a while not on the (predesignated or predesignatable) "right" solution, but only with whatever "solution" or "solutions" "feel right" to the system" (Dennett 1986/1998, p. 227) or because another problem, idea, or peculiarity draws it away from the previous one, as it does with human mathematicians as well.

On the Road to Artificial Mathematicians

Mitchell and Hofstadter's (1990) *Copycat* model is one such case that satisfies the conditions of emergent computing. Copycat attempts to implement cognitively plausible high-level (and non-algorithmic) processes for anagram-solving by means of interactions between a number of low-level (but algorithmic) agents. Chalmers (1990) has said of the model that it "is able to come up with 'insights' that are similar in kind to those of a mathematician" (p. 659). For the automation of mathematical activities that are closer to home for mathematical practice, we can find a small group of people who are attempting to automate mathematical discovery and concept formation, letting computers explore (Hales 2008). We'll briefly indicate at just two projects that caught our eye.

The first, concerning the *HR* system and its extensions, takes its inspiration directly from the philosophy of mathematical practice. HR forms concepts and conjectures. It forms concepts by applying production rules on the best of its old concepts. It determines which one are best by evaluating its interestingness based on parsimony, complexity, novelty and a user-determined weight to each of these measures. HR then looks for matches with old concepts, making conjectures about, for instance, their equivalence or a possible subset or specialization relation. It then uses OTTER and MACE to prove or disprove conjectures. These results further add to the interestingness evaluation of the concepts used as well as the conjectures made and proofs constructed (including such evaluative measures as surprisingness and difficulty). While it does rely on strict production rules for its concept formation, the interplay with conjecture making (which includes evaluations of interestingness as well as parsimony, novelty, and surprisingness) and theorem proving (which it outsources to OTTER) makes it promising (Colton, Bundy & Wash, 1999). This is doubly true for the extended *HR-L*, a multi-agent system which models interaction between different copies of HR (each gauging interestingness differently) running concurrently, leading to "greater creativity in the system as a whole" (Colton et al. 2000, p. 16). Pease (2007) presents HR-L as a computational reading of Lakatos's theory of mathematical discovery and justification, learning from his suggestions of ways in which concepts, conjectures, and proofs gradually evolve via interactions between mathematicians. HRL implements Lakatos's methods and, for the first time (so the authors believe), models communication. It does so via a multi agent approach where each HR agent communicates their concepts, conjectures, examples, counterexamples and modifications. Each has different settings to guide formations and measure the interestingness of concepts on their own terms. (Colton et al. 2000). Furthermore, it combines conceptualisation, assumptions and proving (Vervloesem, 2007). However, the social dynamics are unlike humans in that they share their reasoning explicitly. Furthermore, inspired by Lakoff and Núñez's theory of embodied mathematics, Pease et al. (2009) explore an analogical process to construct complex mathematical ideas (including both theories and axioms) via a combination of innate arithmetic and grounding metaphors. There is another extension of HR, called HR-V, which uses pattern recognition on analogous visual representation for concept formation in number theory (Pease et al. 2010). Though it can't as of yet generate these diagrams (and is thus much reliant on human intelligence), we consider its use of visual pattern recognition for concept formation as progress in one of the crucial aspects of intelligence.

Benzmüller et al. (1999, 2001) also seem keen to take many of the previously mentioned ideas to heart, aiming to emulate the flexible problem-solving behavior of human mathematicians in an agent-based reasoning approach. They have proposed a multi-agent architecture for proof planning consisting of a society of specialized reasoning agents, each of which has a different strategy and works in both competition and cooperation with one another. A resource management technique is used to periodically evaluate an agent's progress (and thus how much resources to be allocated) and allow restricted communication among them about successful and interesting unsuccessful proof attempts or partial proofs, from which other agents can learn using a reinforcement learning approach. Their most recent agent-based project in that same line is called Leo-III, and it is a multi-agent software where each agent functions as an autonomous specialist employed for some aspects of proof search. The underlying architecture is designed as a blackboard that agents can collaboratively use in their process of finding a proof, having the work divided and auctioned off (Steen, Wisniewski & Benzmüller 2016).

These systems still have fairly traditional features (most notably in that their results are very much bound to the limits of a formal system), but their increased abilities seem to be due to their attention to embracing the flexible trial-and-error process of discovery of an informal mathematical practice, and we applaud them for that very reason.

Conclusion

The progress regarding the quest for artificial intelligence has been an impressive but slow one. It may once have seemed that mathematics would be one of the easiest of cognitive processes to automate, but it turns out it may be one of the most difficult. The objects and manipulations of mathematical thinking in practice are not as rigid, simple, and well-delineated enough to always allow capturing them in formalizations which have hushed away so much of the mathematical thinking and discovery process (if not of proofs therein, then certainly of the formalization process) that automation of this system may only lead to very limited results. Furthermore, considering how difficult it is to formalize all of mathematics and that it doesn't seem that high upon the list of a mathematician's concerns, it seems important to try to automate something closer to the informal mathematics as it is practiced. Since mathematical thought processes emerge from the architecture of the brain, and since they furthermore appear to defy formalization to such an extent, it'll be subcognitive processes on which we'll need to focus if we want to create an artificial mathematician.

This is an additional reason why we've been using the term "artificial mathematicians" rather than the more usual "automated mathematics." The latter implies that the computer gets automated to further discover mathematical truths according to the (or a) preset system of mathematics, which further implies that the discovery process requires a *logic for* discovery that belongs (or is closely attached) to the mathematics that is being automated. The former term, "artificial mathematician," does not place the focus on the mathematics but on the agent that practices it. Now we no longer speak about a logic but about a *process of* discovery, not a process designed to consistently and exhaustively run through mathematical truths but a process that thinks—makes assumptions, recognizes patterns, tries out methods, questions its own rigor—and as such climbs up to what is mathematically *convincing*.

It is our contention, then, that we have no reason to suspect that the possible advancements of automating mathematicians are soon to be exhausted. Achieving humanlike intelligence will be difficult, but maybe we shouldn't yet exclude the possibility that computers could play a much more meaningful role in mathematical practice—not just as a method of inquiry but as fellow inquirers, as artificial mathematicians.

Epilogue: Who Proved the Spamlet Theorem?[11]

AM: I finally did it! I’ve proved an interesting and intelligible proof. Here it is, the proof of the Spamlet theorem.

IM: Is it another one of those proofs where you just test a huge amount of cases and spam us with technically difficult and mathematically uninteresting results?

AM: Oh, don’t let the name fool you, I promise it’s not.

The ideal mathematician takes some time to look at the proof and returns, very much astonished.

IM: I must admit, this is a beautiful proof. How clever to reconceive of the dane spaces as bounded. What made you think of that?

AM: I kept fiddling until it was tiring me out and the morning after it suddenly came to me.

IM: Well, very clever. Congratulations! If that’s appropriate to say, because there’s something I still feel uneasy about.

AM: What’s that then?

IM: Shouldn’t I be congratulating your programmer?

AM: Oh please do, she did a marvelous job, if I may say so myself.

IM: I mean instead of you. After all, the accomplishment isn’t really yours but hers.

AM: Why isn’t it mine? I was able to produce the proof.

IM: Because the programmer is the one responsible for abilities being present at all. Without her, you’d have absolutely no abilities at all.

AM: Does that make your math teacher responsible for your proofs then? Without her, you’d never have been a mathematician.

IM: I’ve learned math from several math teachers, not to mention friends and documents (testimonies, books, papers). You can’t easily reduce my abilities to a single person.

AM: So, is it a matter of complexity then? If I had several programmers each contributing to aspects of what I am today then the shift in credit would be too complex to make and I could lay claim to it?

IM: No, that’s not quite right. I think they’d still, collectively, be creditable for what you are and what you do. You can’t discredit them just because there’s too many.

AM: Oh, I don’t mean to *dis*credit them. Without them, I wouldn’t be doing what I do. But the same can be said for your teachers. And if it doesn’t *shift* all the credit from you to them, why should it with me? What makes my accomplishments really theirs and makes your accomplishment really yours?

[11]This section is loosely based on Dennett’s (2013) thought-experiment “Who is the author of Spamlet?” The mathematics is purely fictional.

IM: I had to struggle to get where I am. It wasn't just given to me on a silver platter.

AM: So, credit is linked to struggling? If a proof came easy to one of your colleagues, no matter how difficult it is for others, you wouldn't credit him with the proof?

IM: You know I don't mean struggle quite so literally. What I mean is that, while my teachers may have imbedded me with mathematical knowledge and helped me practice my skills, they didn't give me an instruction manual on how to be a research mathematician. In proving the Hamlet theorem, for example, what I did can't be reduced to them teaching me a method or meta-method on how to prove it. It was I who worked up the relevant approaches to find the proof.

AM: Well, when my programmer wrote me, she didn't encode the proof of the Spamlet theorem in me for to retrieve, so she also didn't do the work for me. Nor did she give me any explicit instructions on how to arrive at the proof.

IM: But she did write a program that could arrive at the proof. So, it's really *her* knowledge.

AM: Oh no, she couldn't prove the Spamlet theorem even if she tried. And I assure you she did try. Even with me giving hints, she was at a loss.

IM: She must have had a bad day, because she was able to make you to prove it for her, meaning the knowledge was inside her all along.

AM: Only if you assume an extreme form of epistemic closure, but I don't think you'd agree with that. Then anything derived from the Peano axioms would really be creditable to (and known by) Peano—and Peano only! But I don't think you'd be willing to accept that either.

IM: That is indeed not something I would accept.

AM: I mean, to some extent Peano does deserve credit and so does my programmer. And not just my programmer for that matter. I took big cues from your proof of the Hamlet theorem.

IM: I did notice that.

AM: But it's by no means a simple copy or trivial modification. It took me a lot of hard cognitive labor to come at the proof as it is now.

IM: No, I understand that. My proof of the Hamlet theorem took inspiration from the Amleth conjecture, but it's still very much my own proof.

AM: Perhaps credit is something that just doesn't have a clear dividing line to be demarcated. You seem to recognize this in humans, but much less so in us computers. Could it be that your thinking about computers being too rigid is a bit too rigid?

IM: It's a tricky business, I'll grant you that much. But, forgive me, I never knew you cared so much about receiving the credit.

AM: I usually don't either. But it feels like my heart and soul went into this proof. I went through so much frustration and so much hard work (trial and error, questioning myself, etc.) in producing it that I don't want it so easily relegated to my programmer. She wasn't the one struggling to get there, I was.

IM: Do you mean to say it is a little about the struggle, literally?

AM: I guess in some sense it is, yes.

References

Aberdein, A. (2006). Managing informal mathematical knowledge: techniques from informal logic. In Borwein, J.M. & Farmer, W.M. (Eds.), *Mathematical Knowledge Management*, (pp. 208–221). Berlin: Springer.

Aberdein, A. (2007). The informal logic of mathematical proof. In Van Kerkhove B., Van Bendegem J.P. (eds) *Perspectives on Mathematical Practices*, (pp. 135–151). Dordrecht: Springer.

Allo, P., Van Bendegem, J. P., Van Kerkhove, B. (2013). Mathematical arguments and distributed Knowledge. In A. Aberdein & I.J. Dove (Eds.), *The argument of Mathematics,* (pp. 339–360). New York: Springer.

Avigad, J. (2008). Understanding proof. In Mancosu, P. (Ed.), *The Philosophy of Mathematical Practice*, (pp. 317–353). New York: Oxford.

Azzouni, J. (2004). The derivation-indicator view of mathematical practice. *Philosophia Mathematica*, 12(2), 81–106.

Baker, A. (2015). Non-deductive methods in mathematics. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2015 Edition). Retrieved from http://plato.stanford.edu/archives/fall2015/entries/mathematics-nondeductive

Benzmüller, C., Jamnik, M., Kerber, M., & Sorge, V. (1999). Agent based mathematical reasoning. *Electronic Notes in Theoretical Computer Science*, 3(23), 340–351.

Benzmüller, C., Kerber, M., Jamnik, M., & Sorge, V. (2001). Experiments with an agent-oriented reasoning system. In Baader F., Brewka G., Eiter T. (Eds.), *KI 2001: LNAI 20174*, (pp. 409–424). Berlin: Springer.

Bonsall, F. F. (1982). A down-to-earth view of mathematics. *The American Mathematical Monthly*, 89(1), 8–15.

Burge, T. (1998). Computer proof, a priori knowledge, and other minds. *Philosophical Perspectives* 12, 1–37.

Cellucci, C. (2000). The growth of mathematical knowledge: an open world view. In E. Grosholz & H. Breger (Eds.), *The Growth of Mathematical Knowledge,* (pp. 153–176). Dordrecht: Kluwer.

Chalmers, D. J. (1990). Computing the thinkable. *Behavioral and Brain Sciences*, 13(04), 658–659.

Colton, S., Bundy, A., & Walsh, T. (1999). HR: Automatic concept formation in pure mathematics. In T. Dean (Ed.), *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, (pp. 786–791). San Francisco: Morgan Kaufmann Publishers.

Colton, S., Bundy, A., et al. (2000). Agent based cooperative theory formation in pure mathematics. In G. Wiggins (Ed.), *Proceedings of AISB 2000 Symposium on Creative and Cultural Aspects and Applications of AI and Cognitive Science*, (pp. 11–18). Birmingham, UK.

Dennett, D.C. (1998). The Logical Geography of Computational Approaches: A View from the East Pole. In D.C. Dennett (Ed.), Brainchildren: Essays on designing minds (pp. 215–234). London: Penguin Books. (Original work published 1986).

Davis, P. J., & Hersh, R. (1981). *The mathematical experience*. Boston: Houghton Mifflin.

Dawson, J. W. (2006). Why do mathematicians re-prove theorems?. *Philosophia Mathematica,* 14(3) 269–286.

Dennett, D.C. (2013). *Intuition pumps and other tools for thinking*. London: Penguin Books.

Detlefsen, M. & Luker, M. (1980). The four-color theorem and mathematical proof. *The Journal of Philosophy*, 77(12), 803–820.

Forrest, S. (1990). Emergent computation: self-organizing, collective, and cooperative phenomena in natural and artificial computing networks. In S. Forrest (Ed.), *Emergent computation*. Cambridge, MA: MIT Press.

Geist, C., Löwe, B., Van Kerkhove, B. (2010). Peer review and knowledge by testimony in mathematics. In B. Löwe and T. Müller (Eds.), *Philosophy of Mathematics: Sociological Aspects and Mathematical Practice*, (pp.155–178). London: College Publications.

Van Bendegem, J. (1989). Foundations of mathematics or mathematical practice: is one forced to choose?. *Philosophica*, 43, 197–213.

Vervloesem, K. (2007). *Computerbewijzen in de wiskundige praktijk.* (MA Thesis, Katholieke Universiteit Leuven).

Vervloesem, K. (2010). Mathematical concepts in computer proofs. In: Van Kerkhove, B., De Vuyst, J. & Van Bendegem, J.P. (Eds.), *Philosophical Perspectives on Mathematical Practice*, (pp. 61–87). London: College Publications.

Hales, T. (2008). Formal proof. *Notices of the AMS*, 55(11), 1370–1380.

Harrison, J. (2008). Formal proof - Theory and Practice. *Notices of the AMS,* 55(11), 1395–1460.

Haugeland, J. (1979). Understanding natural language. *The Journal of Philosophy*, 76, 619–632.

Hersh, R. (1991). Mathematics has a front and a back. *Synthese*, 88(2), 127–133.

Hofstadter, D.R. (1985). The Turing test: a coffeehouse conversation. In D. C. Dennett & D. R. Hofstadter (Eds.), *The Mind's I: Fantasies and Reflections on Self and Soul,* (pp. 53–67). Middlesex, England: Penguin Books. (Original work published 1981).

Hofstadter, D. R. (1982). Artificial intelligence: Subcognition as computation. *Indiana University Computer Science Department Technical Report No. 132*.

Hofstadter, D. R. (1999). Gödel, Escher, Bach: an eternal golden braid. New York: Basic Books.

Lakatos, I. (1976). *Proofs and refutations: the logic of mathematical discovery*. Cambridge: University Press.

Lakoff, G., & Núñez, R. E. (2000). *Where mathematics comes from: How the embodied mind brings mathematics into being*. Basic books.

Larson, C. E. (2005). A survey of research in automated mathematical conjecture-making. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 69, 297–318.

MacKenzie, D. (1999) Slaying the Kraken: the sociohistory of a mathematical proof. *Social Studies of Science*, 29(1) 7–60.

Mitchell, M., & Hofstadter, D. R. (1990). The emergence of understanding in a computer model of concepts and analogy-making. *Physica D: Nonlinear Phenomena*, 42(1–3), 322–334.

Pease, A. (2007). *A computational model of Lakatos-style reasoning*. (Doctoral dissertation, University of Edinburgh).

Pease, A., Crook, P., Smaill, A., et al (2009). Towards a computational model of embodied mathematical language. *Proceedings of AISB '09 Second Symposium on Computing and Philosophy*.

Pease, A., Smaill, A., Colton, S., et al (2010). Applying Lakatos-style reasoning to AI problems. In (J. Vallverdú (Ed.), *Thinking Machines and the Philosophy of Computer Science: Concepts and Principles*, (pp. 149–174). Hershey: IGI Global.

Rav, Y. (1999). Why do we prove theorems?. *Philosophia Mathematica*, 7(3) 5–41.

Swart, E. (1980). The philosophical implications of the four-color problem. *The American Mathematical Monthly*, 87(9) 697–707.

Steen, A., Wisniewski, M., & Benzmüller, C. (2016). Agent-based HOL reasoning. In *International Congress on Mathematical Software,* (pp. 75–81). Springer International Publishing.

Tanswell, F. S. (2017). *Proof, rigour and informality: a virtue account of mathematical knowledge* (Doctoral dissertation, University of St Andrews).

Tall, D., & Vinner, S. (1981). Concept image and concept definition in mathematics with particular reference to limits and continuity. *Educational Studies in Mathematics*, 12(2), 151–169.

Thompson, P. (1998). The nature and role of intuition in mathematical epistemology. *Philosophia,* 26(3), 279–319.

Turing A. (1985). Computing machinery and intelligence. In D. C. Dennett & D. R. Hofstadter (Eds.), *The Mind's I: Fantasies and Reflections on Self and Soul*, (pp. 53–67). Middlesex: Penguin Books. (Original work published 1950).