Proceedings of the International Symposium on Mathematical Practice and Cognition , Alison Pease, Markus Guhe, and Alan Smaill (Eds.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

Interpreting Naproche – An algorithmic approach to the derivation-indicator view

Merlin Carl[1], **Peter Koepke**[1]

Abstract. In [1], Jody Azzouni proposes a derivation indicator view of mathematical practice and in particular of proofs.
The Naproche Project [5] takes proofs as linguistic entities whose semantics is given by corresponding formal derivations. This view is computationally implemented in the Naproche System where simple natural proof texts are automatically transformed into derivations, using techniques from computational linguistics, formal logic and automatic theorem proving. The development of the system has identified many ways in which parts of proof texts indicate elements of formal derivations. The Naproche Project can thus be seen as a support of the derivation indicator view, and we comment on some of the critique against derivation indication from this standpoint and our experience. Doing so, we extend earlier work by the second author on the relation between natural and formal proofs; since then, further development has shown the relation of DI to Naproche to be an interesting field that can (and should) be elaborated in detail.

1 THE NAPROCHE PROJECT

Naproche is a common project of mathematical logicians from Bonn, formal linguists from Duisburg/Essen and computer scientists from Cologne. Its name is derived from "NAtural language PROof CHEcking" and it aims at studying actual presentations of mathematical proofs from a logical and linguistic point of view. Hence, we are for example interested in questions concerning the underlying inference rules of natural proving, the role of background knowledge in understanding mathematical discourse and linguistic specialities of actual proof texts from mathematical practice.

The Naproche System is both an application and a way of empirical verification of results of our investigations. In short, it is a computer system where a mathematical text - consisting of axioms, definitions, lemmas, theorems and proofs, in a rich but restricted fragment of English including formulas - can be entered in a TeXMacs environment for automatic proof-check.
The first step in checking e.g. a proof found in a textbook thus consists in a reformulation of the given mathematical text in the Naproche language. A short example of what such a reformulation looks like is found below; the reader interested in more material and a description of the current Naproche language is invited to consider [5]. The Naproche language is (being) developed to resemble the mathematical vernacular as close as possible. Therefore, such reformulations are usually easy to do for the kind of texts we are considering at the moment. In particular, this can be done in a sentence-by-sentence fashion, without a preliminary complete understanding of the proof to be reformulated.
The processing is then continued by applying adapted techniques from formal linguistics, in particular the discourse representation theory of Kamp and Reyle [8], to generate a semantic representation in a nested box structure that we call "proof representation structures" (see [9] for details). To do so, the text is seen as a series of triggers for mental acts such as introducing or retracting an assumption, drawing a conclusion, introducing and naming an object, applying a proof technique (e.g. induction), recalling an earlier result, making a statement, announcing a proof, declaring it finished etc. Linguistic markers for these 'acts' are automatically recognized and trigger an appropriate processing. The obtained intermediate format is then used to deal with several linguistic and logical phenomena such as anaphorical constructions or the identification of the set of available assumptions at each part of the proof. From here, a rather routine algorithm computes a series of queries of the form 'deduce statement A from the set S of available assumptions and results already proved' in TPTP, a standard input format of automatic theorem provers. These are then used to check whether or not the claimed step can be carried out - resembling a human reader trying to fill in missing steps. As a by-product of this checking, one gets a formalized version of the text by simply concatenating the theorem prover's outputs.

Currently, there are various sample texts in the Naproche syntax available, dealing with group theory, logic, set theory and analysis. The most advanced is [6], an English reformulation of the first chapter of Edmund Landaus elementary textbook "Grundlagen der Analysis" [9].
For an impression of what a text in the current Naproche language looks like, consider the following short excerpt from the cited translation, Landaus 6th definition and 28th theorem, which is accepted by the latest version of the system:

Definition 6: Define $\ast$ recursively:
$x \ast 1 = x$.
$x \ast y' = (x \ast y) + x$.

Lemma 28a: For all y, $1 \ast y = y$.
Proof: By definition 6, $1 \ast 1 = 1$. Suppose $1 \ast y = y$. Then by definition 6, $1 \ast y' = (1 \ast y) + 1 = y + 1 = y'$. Thus by induction, for all y $1 \ast y = y$. Qed.

This fragment contains already several features such as the possibility of a recursive definition of a new operator, reference to a definition and the induction technique. Also, it is quite close

[1]Mathematical Institute of the University of Bonn

to the German text; it wouldn't look conspicuous if merely presented as an English version of the original.

Our system has several potential applications. People interested in this subject may appreciate it as a natural language interface to formal mathematics. Authors of introductory textbook notes could use it as a tool for authoring and checking. For first-year students, it could serve (and has served) as an automatic tutoring system for learning how to prove. And finally, the generation of proof texts that are readable for computers and humans alike is of course a contribution to the field of men-machine-communication.

To avoid misunderstandings, let us remark on what Naproche is not: It is not and is not meant to be a tool for understanding research papers in any advanced area of mathematics. The idea of e.g. replacing the process of peer review by automatic checkers is pure science fiction. Also, there are and probably will always be figures in the vernacular that exceed the abilities of the system. Therefore, some preparatory work from the user is required. Checking a completely unadjusted text will usually not work.
The next goal is a check of an English reformulation of the whole of Landaus "Grundlagen der Analysis". To achieve this, the inventory of proof figures, logical features, grammar and vocabulary is constantly being extended. Furthermore, we plan broader text corpus studies as a starting point for further empirically solid research on mathematical reasoning and speech practice.

2 DERIVATION INDICATION

Another interesting aspect is in how far Naproche can help to understand the meaning of "understanding" in the context of proofs. In [11], Jeremy Avigad proposes a notion of understanding that refers to abilities that someone with understanding should have. The points he lists include the ability to formalize, fill in gaps, verify correctness and spot mistakes. In this context, one might state that Naproche is in a weak sense capable of understanding elementary mathematical texts.

In [1], Jody Azzouni looks for an explanation for what he calls the "benign fixation of mathematical practice", i.e. the fact that mathematics, when seen as a social practice, is remarkably stable. This particularly applies to questions concerning what counts as a proof. His claim is that mathematical proofs are "indicators" of formal derivations, symbolic sequences of some logical calculus that can be checked mechanically for correctness and thus cannot be meaningfully disputed any more. Hence, derivations serve as background certificates for the discursive stability of a natural proof.

This view bears an obvious analogy with the way computer scientists deal with algorithms that one might label "Turing program indicator view": Usually, "algorithms" are identified with programs for a Turing machine (or one of many equivalent formal notions, depending on the taste of the author). However, actual Turing programs virtually never appear in any publication containing algorithms. Instead, areas as combinatorial optimization or algorithmic graph theory use natural language descriptions of recipies to indicate Turing programs. The way from one to the other is far from trivial: Quite often an actual implementation remains a challenging problem. Much more is merely "described" rather than "implemented".
Thus, if Turing programs correspond to formal proofs, and informal process descriptions to natural proofs, the situations resemble each other closely and the well-accepted "Turing program indicator view" becomes the algorithmic counterpart to the DI-view.
In recursion theory, where algorithms become objects of consideration of other algorithms more regularly than mathematical proofs become objects of mathematical proofs, there is a common practice called "proof by Church-Turing-Thesis": Given an informal high-level description of a procedure, the existence of a Turing program for the same purpose is taken for granted. This step is all-important, even though one is rarely interested in carrying it out in detail: Probably nobody wants to see e.g. a Turing program enumerating two incompatible Turing degrees. Nevertheless, a Turing program is a good representation for what underlies our certainty that the algorithm can be carried out in all details, i.e. made unambiguous.

While the DI-view is an inspiring perspective in our opinion, Azzouni does not explain the precise meaning of what he means by "indication", nor does he give a hint how it works. In particular, it remains to explain how mathematical practice could be fixed by derivation indication hundreds of years before formal derivations were introduced in the beginning of the 20th century by Frege, Russell and others.

In order for the derivation indicator explanation to work, there must be a clear sense for the process of "indication" - proof practice would not be fixed by referring to derivations if there was no generally accepted way how to go from a natural proof to a derivation. But if there is such a procedure, however complicated, then by the Church-Turing thesis, it is algorithmically implementable. It is these algorithms that we are looking for.
Consider again the excerpt from our reformulation of Landau cited above. The indicators, or, as we call them, triggers, are easy to identify: "Definition 6", for example, announces the introduction of a new notion which is given the number 6 for further reference. In the proof, "suppose" indicates the introduction of an assumption, "then" prepares that the following is deducible from the available assumptions, "by induction" gives a hint on the proof procedure to be used, while "qed" marks the end of the proof and hence the claim that at this point, the proof goal should follow from what has been said so far.

Our expectation is that the language of more advanced textbooks than [10], though more elaborate and complicated, can be handled in a similar manner. This is supported by the view many mathematicians take on proofs and their relation to derivations: Derivations are seen as representations of discursive limit objects that would arise if every step of a proof was carried out in full detail.
Let's have a look at the following quote by Saunders MacLane [12]:

"As to precision, we have now stated an absolute standard of rigor: A mathematical proof is rigorous when it is (or could be)

written out in the first-order predicate language L(ε) as a sequence of inferences from the axioms ZFC, each inference made according to one of the stated rules. [...] When a proof is in doubt, its repair is usually a partial approximation to the fully formal version."

This gives a hint to view the apparent difference between proofs and derivations as a matter of granularity. The increasing degree of detail can be observed through the stages of mathematical research: From high-level discussions about proof plans to written sketches, research papers, textbook proofs, explanations thereof in lectures, elaborations thereof in tutorial classes over almost gap-free proofs as in Landau to derivations of the kind Naproche produces.

3 CRITICISM

Since its first publication, several points of criticism have been raised against the DI-view, see e.g. [3] and [4]. Based on our experiences with analyzing proofs and the Naproche systems, we feel that we can substantially react to some of them.

In [3], Yehuda Rav claims that proofs contain an "irreducible semantic component", by which he means "intentional objects" that "cannot be formalized but require the active interpretative participation of the reader". Taken for granted that understanding is an active process that needs to be carried out, we don't agree that this can only be done by humans. Even the current Naproche system generates an interpretation of a text through an active process involving language processing and automatic reasoning - one might say that the system questions each proof step, then tries to find reasons for it. Nevertheless, Naproche is an algorithmic system in any sense of the word - one which indeed does "the work of mechanically recognizing the validity of even an informal proof" ([3], p. 304).

One might object here that the Naproche language, in spite of resembling the mathematical vernacular, is still by definition itself a formal language and that the actual formalization is therefore carried out by the reader who reformulates the text, not by the system.
We think that even the current stage of Naproche gives a strong hint that the frontier between formal and natural languages is rather a question of degree. If the 'formal languages' reach a degree of resemblance to the vernacular that makes formalization a question of basic changes of words and formulations, much more of the work of generating an interpretation is done by the machine than by the human reader involved – in this sense, the human reader can be eliminated from the process of understanding.

Later on, Rav refers to the formalization of Landau in the Automath language. He claims, citing Feferman, that the fact the checking of the resulting derivation was successful is no argument in favour of the correctness of the original text, since a human reader was needed to understand the text prior to its formalization, and "understanding subsumes checking". But where the reading process can be replaced by a mere routine reformulation – which, based on our experience, appears to be the case for considerable parts of basic mathematical literature - this problem is at least heavily reduced.

While a complete formalization of a proof may indeed require a deep understanding of the original material, possibly even a deeper one than a usual reader will obtain, a reformulation of e.g. Landau's proofs in the Naproche syntax requires much less.
It is in fact closer to a translation of a proof into a different language than to a formalization in, say, predicate calculus.
This does not presuppose understanding to a degree that 'subsumes checking': It is well possible (and has probably already happened several times) that A, a native German speaker without knowledge of foreign languages, writes a text that he considers to be a proof. B, who knows German and English reads it, considers it correct and, translates it to English, and C, who knows English, but not German, spots a mistake in the translated text. The translation done by B cannot have subsumed checking, otherwise B would have spotted the mistake. So translation in another natural language does not necessarily subsume checking.
The question is then whether semi-formal languages can serve the same purpose at least for some tasks. Based on the first author's personal experience in writing Naproche texts, even the basic language of Naproche is already good enough for this. The translation of, say, the first chapter of Landau, was (on purpose) mostly done without an actual thought about the content of the original text, let alone a checking. If there was a mistake in this chapter, it could well have been discovered by Naproche though it evaded the person adapting the text.

It is also said that formalizations, even where possible, do not contain new information, nor do they enhance the rigor or security of an argument that can easily be seen to be correct by understanding it. Let us, however, consider again our main sample text [10]. It contains nothing but fairly elementary theorems in a fairly elementary area, and its author, Edmund Landau, was an accomplished mathematician of international rank and fame. Yet, in the introduction, he mentions that a colleague spotted a flaw in his supposedly clear and complete presentation that lead him to reorganize some of the material.
It therefore appears that there is space for an 'epistemic gain' even in the most basic examples of informal standard mathematics. Attempts to formalize an argument are a possible way to obtain a higher degree of certitude. Reformulating texts in a semi-formal language like that of Naproche is one way to make this gain cheaper.

Other points referring to the impreciseness of terms like "algorithmic system", "indication" etc. are, in our view, no counterarguments but open research fields. Furthermore, it should be noted that these arguments could also be raised against "algorithm indication", which nevertheless does not seem to cause any problems.

4 FUTURE PROSPECTS AND RELATED WORK

There are also related projects going on at the moment.

While, to the authors' knowledge, there is no other group that focuses on the connection between informal and formal proofs, there are already several proof checking systems that also emphasize the readability of their respective input language.

In the Mizar system, started in 1973 by Trybulec, theorems of advanced mathematics are formulated and proved by an active community, and results are published in the journal Formalized Mathematics.

The developing team of Isabelle is working on a 'human-readable structured proof language' named Isar.

We are collaborating with the VeriMathDoc project, which includes the PLATO program. Their goal is to develop a mathematical assistant system that naturally supports mathematicians in the development, authoring and publication of mathematical work.

In his dissertation, Claus Zinn developed a system for processing mathematical texts [17]. In his work, he also introduces extended DRSs, called Proof Representation Structures. Despite the equality of their names, the PRS notions of Naproche and the work of Zinn are quite different.

The Naproche project itself is still at an early stage of development. Even though it already reaches its goal to mediate between easy natural proofs such as the excerpt from Landau cited above and fully formalized, mechanically checkable derivations, a lot is still to be done before one can approach more sophisticated textbooks. On the linguistic side, grammar and vocabulary are permanently expanded while more and more linguistic phenomena like resolution of anaphora or presuppositions are taken into account. Corpus studies are planned and carried out to get information about common proof strategies, argumentative figures and how they are indicated.
Also, we attempt to make the current CNL closer to the actual natural language; as an example, the current way to trigger the retraction of an assumption by 'thus' is quite unsatisfying. In reading mathematical texts, one encounters virtually never a doubt on what the open assumptions are without such artificial rules.
How this works is one big question for the time being.
On the logical side, one easily notices that implicit background knowledge such as spatial imagination or intuitive type systems play a large role in understanding mathematical discourses. For example, human readers easily deal with objects such as sets, sequences, lists etc. without using explicit knowledge on concatenations, permutations or set theory. Our goal is to model this by providing the system with a weak set theory. Here, it is quite a challenge to find the right balance between premises that are on the one hand strong enough to allow usual references to sets to be processed, and harmless enough to keep the search space in control. (E.g., one might try to use ZFC as the background theory, which is certainly strong enough. But then the prover would have to deal with infinitely many axioms in each step, which does not look very promising.)
Talking about complexity issues, these already are a permanent threat to incautious expansions of the system. When gaps in natural proofs become too big to be filled in by an ATP in an acceptable amount of time, the proof check is unsuccessful. One way to approach this difficulty is to exploit implicit information and hints on how to proceed given in the proof more efficiently. Still, if the search space consists of all accessible theorems, checking of longer texts might soon become unfeasible. This can be avoided by carefully selecting the premises that are likely to be useful for the current step. The selecting algorithms are still at an experimental stage, yet even these result in a enormous reduction of running times.
We are optimistic that these developments will allow us to check much more and more advanced texts. For the near future, we plan a mechanical check of further chapters of Landaus 'Grundlagen der Analysis', including those where set theory becomes unavoidable, and of Euklids 'Elements'.

5 CONCLUSIONS

We hope to have made plausible that the DI-view on mathematical practice can lead to interesting research and concrete, useful technical applications. In our opinion, this view is rather an inspiring credo, or a "regulative imperative", than a complete explanation of the stunning stability of mathematical practice. In the long run, the Naproche project could be a way of empirically sustaining this perspective. It is our conviction that enriched formalisms in the spirit of the Naproche language can provide a bridge between what are called the "two streams in the philosophy of mathematics" and lead to a notion of proof that incorporates the advantages of both.

REFERENCES

[1] Azzouni, Jody (2006): Tracking Reason (Oxford University Press)
[2] Azzouni, Jody (2004): The derivation-indicator view of mathematical practice (Philosophia Mathematica (3) 12, 81-105)
[3] Rav, Yehuda (2007): A Critique of a Formalist-Mechanist Version of the Justification of Arguments in Mathematician's Proof Practices (Philosophia Mathematica (III) 15, 291-320)
[4] Carlo Celluci (2008): Why proof? What is a proof? (Deduction, Computation, Experiment. Exploring the Effectiveness of Proof, Springer Berlin)
[5] The Naproche homepage: www.naproche.net
[6] Carl M., Cramer M., Kühlwein D.: Landau in Naproche-Syntax (available at [5])
[7] Koepke, Peter: Mathematical Proofs as Derivation-Indicators: Theory and Implementation (Talk given in Utrecht 2009)
[8] Kamp/Reyle (1993): From discourse to logic (Dordrecht: Kluwer Academic)
[9] Nickolay Kolev (2008): Generating Proof Representation Structures in the Project NAPROCHE (available at [5])
[10] Landau, Edmund (1965): Die Grundlagen der Analysis (Wissenschaftliche Buchgesellschaft Darmstadt)
[11] Avigad, Jeremy (2008): Understanding Proofs (in: The Philosophy of Mathematical Practice, Oxford University Press)
[12] MacLane, Saunders (1986): Mathematics, Form and Function (Springer)