How Natural is Formal Mathematics?

Michael Junk
University of Constance
78457 Konstanz, Germany
michael.junk@uni-konstanz.de

Sebastian Sahli
University of Constance
78457 Konstanz, Germany
sebastian.sahli@uni-konstanz.de

Abstract

We present results from a lecture on mathematical modeling where both lectures and homework assignments are based on an editor which checks a formal syntax and the correct use of names. Since the lecture addresses students without prior training or special inclination to formal mathematics our observations shed some light on the ambivalent relation between formality and naturalness.

1 Background

The lecture on mathematical modeling is a mandatory course in the undergraduate program for math majors at the University of Constance. It is designed for students in the second semester but may also be attended later. With regards to contents, the lecture focuses on the formulation and definition of models for real-world-problems with particular emphasis on optimization problems, questions involving uncertainty and randomness, descriptions of point and rigid body motions and of processes which are governed by conservation and balancing principles. Since mathematical analysis of the resulting models is *not* a central issue, the modeling lecture is effectively a mathematical language course which practices the *definition* of patterns and structures. In order to stress this central aspect of mathematics, we are using a specific formalism that has been summarized in the language $\mathbb{M}$ATh (as acronym for ***m**eta-language for **m**odels, **a**lgorithms and **th**eories* [1, 2, 3]). To give a simple example of the pen and paper syntax used in the course, we consider the definition of a structure called *rectangle*

$$\text{rectangle} := [a, b \in \mathbb{R}_{>0} \textbf{ with } \text{area} := a \cdot b;\ \text{perimeter} := 2 \cdot (a + b)].$$

The definition with a square-bracket construct formalizes the otherwise more customary verbose style:

Definition 1 *A pair of two positive numbers a, b is called a rectangle. To a rectangle we associate its area $a \cdot b$ and its perimeter $2 \cdot (a + b)$.*

The statement $(1, 1)$: rectangle denotes that $(1, 1)$ *is a* rectangle (up to technicalities, it parallels the $\in$-relation for sets or the application of unary predicates). In subsequent problems (like finding a rectangle with given perimeter U of maximal area), the named components of rectangles can be used:

$$\begin{aligned}&\text{isoperimetricProblem} := [U \in \mathbb{R}_{>0} \textbf{ with}\\ &\quad \text{optimal} := \text{argmax}([R : \text{rectangle} \textbf{ with } R.\text{perimeter} = U] \mapsto R.\text{area})].\end{aligned}$$

In the first weeks, the students are accustomed to the syntax by translating back and forth from well known verbose definitions to their formal counterparts. Basic and common mistakes at that level concern syntax, variable

In: A. Editor, B. Coeditor (eds.): Proceedings of the XYZ Workshop, Location, Country, DD-MMM-YYYY, published at http://ceur-ws.org

scopes and basic type consistency (e. g. confusing an evaluation $f(x)$ with the function f). Unfortunately, these problems persist with a considerable fraction of the students throughout the course which interferes with the actual goal, the discussion of semantic aspects.

2 ModelChecker

Propelled by the Corona crisis with its need for digital teaching methods, we have introduced an editor called ModelChecker which allows to write down definitions of models in plain ASCII text using a fixed syntax. At the beginning of the semester, this syntax has been presented alongside the very similar pen and paper form. Apart from syntax checking, the tool also controls the correct use of previously defined names. This automated handling has various advantages: Due to the immediate and untiring feedback it raises awareness for these common mistakes and helps to avoid them before submission so that the correction process can concentrate on more important semantic issues. Moreover, formerly common discussions about acceptability of sloppy answers are tacitly transformed into more fruitful discussions about syntactic rules which ensure unambiguousness at machine level.

To help students focus on their actual modeling tasks, the ModelChecker allows to introduce names and symbols of previously known objects and operators without the need for detailed definitions. However, extensive usage of this option revealed a variety of different perceptions of fundamental concepts in set theory, analysis and linear algebra. During the lecture, this uncontrolled growth has been counteracted by stipulating an unambiguous usage of `union`, `intersection`, `cartesianProduct` and of arithmetic symbols like `+`, `-`, `*`, `/` as well as the finite `sum` and `product` for a large family of real vector spaces. The names and symbols are now included in the program from the start.

3 Experience

During the course, the students have convincingly demonstrated that the correct handling of formal mathematics is by far not natural. Studying the homeworks, it seems that the validity of expressions is frequently judged by *acoustic consistency* only. For example, *A is not empty* has been repeatedly formalized to `A : not(empty)` which passes the acoustic test under the translation $: \to$ *is* but is incorrect because `empty := [M : set with not(exists x in M)]` is a notion and not a `Bool` value. Another example is the expression `union([n in Nat] -> 2*n)` which sounds reasonable when vocalized as *the union of all numbers* $2n$ *with* n *ranging over the naturals* but it violates the rule that `union` expects set-valued functions as arguments. Surprised by the prevalent inability to detect such basic errors, we have set up a voluntary training module with 41 questions organized in five blocks which ask for a classification of elementary and short expressions into the categories *mapping, set, basic object* and *incorrect.* In their first attempt, only one student out of 23 passed with more than 75% correct answers while the average percentage was 54%, which perfectly reflects the difficulties revealed in the homeworks. Studying the most frequent errors, it can be observed that the categories are often chosen according to some vague *has to do with* relation. For example, `Def(x)` and `x --> y` both have to do with mappings and are often categorized as such but they actually represent sets. If more complex expressions need to be checked, the failure rate increases drastically which indicates that classification is rather based on personal experience with certain expressions than on strict recursive rule applications. Taking all this into account, it seems that the average math student uses the formal language of mathematics quite similar to a natural language where some words are replaced by symbols. While mental associations to language objects often point in the right directions, they may be too vague if it comes to details.

Since the selective advantage of a formal language arises exactly from those aspects where it differs from its natural counterpart we think that stressing these aspects (i. e. the strict and precise language rules) should be a central part of undergraduate math education. From our experience we can say that a tool which shifts a pure human-to-human communication to a human-via-machine-to-human mode alleviates this task considerably, because with the machine as a team-mate the importance of formal language rules is quite evident. Also, it seems more important *that* formal tools are used rather than *which* tools these are – and fortunately there is a rich selection to choose from (see for example the undergraduate project described in [4] using Isabelle or the one in [5] based on Mizar to name just a few).

References

[1] 𝕄ATh-Homepage, http://www.mmath.de/en.

[2] Junk, M., Hölle, S., Sahli, S.: *Formalized Mathematical Content in Lecture Notes on Modelling and Analysis*. In: Rabe, F., Farmer, W. M., Passmore, G. O. and Youssef, A. (eds.) Intelligent Computer Mathematics. Lecture Notes in Computer Science, vol. 11006, pp. 125-130. Springer (2018)

[3] Junk, M., Hölle, S., Sahli, S.: *A Meta Language for Mathematical Reasoning*. In: Osman Hasan, O., Kaliszyk, C., Naumowicz, A. (eds.) Proceedings of the Workshop Formal Mathematics for Mathematicians (FMM), Hagenberg, Austria (2018)

[4] Bayer J., David M., Pal A., Stock B.: *Beginners' Quest to Formalize Mathematics: A Feasibility Study in Isabelle*. In: Kaliszyk C., Brady E., Kohlhase A., Sacerdoti Coen C. (eds) Intelligent Computer Mathematics. CICM 2019. Lecture Notes in Computer Science, vol 11617. Springer, Cham (2019)

[5] Naumowicz, A.: *Testing Mizar User Interactivity in a University-level Introductory Course on Foundations of Mathematics*. Workshop Papers at 12th Conference on Intelligent Computer Mathematics CICM 2019, Prague, Czech Republic, July 8-12, 2019, online http://cl-informatik.uibk.ac.at/cek/cicm-wip-tentative/FMM1.pdf