Symbolic Informalization: Fluent, Productive, Multilingual

Aarne Ranta

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
aarne.ranta@cse.gu.se

Abstract

Symbolic informalization enables a reliable conversion of formal mathematics to natural language. It can also be used for creating training data for neural autoformalization. This paper outlines the project Informath, which aims to show how symbolic informalization can produce text with natural variation, be developed with a reasonable effort, and address multiple formal and natural languages simultaneously.

1 Introduction

Informalization, translating from formal to natural language is a useful technique when communicating the results of formal mathematics to the rest of the world. Mizar's generation of LaTeX [2] is an early example of this, and the extraction of texts from Coq proofs [3] is another one from the 1990s.

A more recent application of informalization is to help the opposite direction, **autoformalization**. This trend started with [15], using Mizar's informalization [2] for producing training data for neural machine translation. Mizar's informalization is **symbolic**, based on rules, whereas [7] uses a neural language model also for informalization.

Symbolic informalization can, at least in theory, be made 100% reliable, and therefore used for automatic processing without human control. Neural methods can usually not give such guarantees. Thus the neural informalization used in [7] produces results that are reported to be 74% correct. The authors accept this and state three arguments against using symbolic informalization tools: these tools

- "result in natural language content that lacks the inherent diversity and flexibility in expression: they are rigid and not natural-language-like",
- "are hard to design and implement",
- "differ a lot for different formal languages, hence the approach is not scalable for multiple formal languages".

Since we do not want to throw away the advantages of symbolic informalization, we take these claims as challenges to be met. They have been a stimulus for the project Informath, which builds on the tradition of symbolic approaches, in particular [3, 6, 8, 4, 11, 9]. The following brief overview of Informath aims to show how it deals with the three challenges.

2 The Informath project

Informath builds on two technologies:

- Dedukti [1], a logical framework designed to be an interlingua for many other frameworks and equipped with conversion tools between them.
- GF (Grammatical Framework, [10]), a grammar formalism that uses a logical framework to define interlingual representations (abstract syntaxes) for natural languages.

The use of Dedukti readily solves the third problem mentioned in [7]: multiple formal languages. Informath addresses at the moment four formal languages simultaneously: Dedukti itself, Agda, Lean, and Rocq (formerly Coq). With the use of GF on the natural language side, currently three languages are addressed: English, French, and Swedish.

Here is an example of a Dedukti formula and its renderings in each language:

```
Dedukti: (a : Elem Int) -> (c : Elem Int) -> Proof (and (odd a) (odd c)) ->
  Proof (forall Int (b => even (plus (times a b) (times b c))))
Agda: (a : Int) -> (c : Int) -> and (odd a) (odd c) ->
  all Int (\b -> even (plus (times a b) (times b c)))
Rocq: forall a : Int, forall c : Int, (odd a /\ odd c ->
  All Int (fun b => even (a * b + b * c)))
Lean: (a c : Int) (_ : odd a /\ odd c) : All Int (\b => even (a * b + b * c))
```

English: Let $a, c \in Z$. Assume that a and c are odd. Then $ab + bc$ is even for every integer b.
French: Soient $a, c \in Z$. Supposons que a et c sont impairs. Alors $ab + bc$ est pair pour tous les entiers b.
Swedish: Låt $a, c \in Z$. Anta att a och c är udda. Då är $ab + bc$ jämnt för alla heltal b.

The second problem, difficult design and implementation, is addressed by using GF and its Resource Grammar Library (RGL), which takes care or morphological and syntactic details and provided a high-level API to programmers. This is an advantage over context-free grammars familiar from compilers and used in, e.g., ForTheL [8]. In order to recognize variations such "is" vs. "are" in a context-free grammar, the simplest way is to treat them as free variants without requiring agreement to the subject. This is tolerated, sometimes even desired, in systems aimed to parse natural language input. But systems for informalization should guarantee grammaticality. Even a simple sentence such as "x is even" needs several context-free rules to be accurate in English, if also negated and inverted sentences are to be covered. In French, over 50 variations are needed because of gender and mood. In GF, a single RGL expression, `mkCl x even_A` (building a clause with an adjectival predicate), is enough for both languages.

In addition to helping grammar writers, the abstractions provided by GF and RGL support the use machine learning to extract rules from text [13]. In the Informath project, we have used neural parsing to help extract mathematical terminology from Wikidata labels [12]. Since machine learning is by nature uncertain, such rules must be manually checked. However, they need only be checked once to become a part of a 100% reliable symbolic system.

The biggest challenge raised in [7] is the first one: diversity and flexibility in expression. The tool we use for this is symbolic NLG (Natural Language Generation, [14]), which can easily produce hundreds paraphrases of a given sentence. Developing a grammar for these paraphrases is guided by existing mathematical texts, and the goal is to cover the full variation of mathematical language described in [5].

3 First results

In Summer 2024, a grammar was built based on over 5000 mathematical terms in Wikidata, together with a syntax that covered most of ForTheL and Naproche [12]. In Winter 2025, a complete framework was built to translate between the formal and natural languages mentioned above, together with examples from arithmetic, set theory, and the "100 theorems" of [16]. An ongoing Master's thesis at Chalmers (by Pei Huang) uses symbolic data to reproduce the experiments of [7]. The focus has been on definitions and statements: while Informath can also informalize Dedukti proofs, the results are not good enough without extra NLG work. Informath code is available in https://github.com/aarneranta/informath.

References

[1] Ali Assaf, Guillaume Burel, Raphaël Cauderlier, David Delahaye, Gilles Dowek, Catherine Dubois, Frédéric Gilbert, Pierre Halmagrand, Olivier Hermant, and Ronan Saillard. Dedukti: a logical framework based on the $\lambda\pi$-calculus modulo theory, 2023.

[2] Grzegorz Bancerek, Czesław Byliński, Adam Grabowski, Artur Korniłowicz, Roman Matuszewski, Adam Naumowicz, Karol Pak, and Josef Urban. Mizar: State-of-the-art and beyond. In Manfred Kerber, Jacques Carette, Cezary Kaliszyk, Florian Rabe, and Volker Sorge, editors, *Intelligent Computer Mathematics*, pages 261–279, Cham, 2015. Springer International Publishing.

[3] Y. Coscoy, G. Kahn, and L. Thery. Extracting text from proofs. In M. Dezani-Ciancaglini and G. Plotkin, editors, *Proc. Second Int. Conf. on Typed Lambda Calculi and Applications*, volume 902 of *LNCS*, pages 109–123, 1995.

[4] Marcos Cramer, Bernhard Fisseni, Peter Koepke, Daniel Kühlwein, Bernhard Schröder, and Jip Veldman. The Naproche Project Controlled Natural Language Proof Checking of Mathematical Texts. In Norbert E. Fuchs, editor, *CNL*, volume 5972 of *LNCS*, pages 170–186. Springer, 2009.

[5] Mohan Ganesalingam. *The Language of Mathematics: A Linguistic and Philosophical Investigation.* Springer, 2013.

[6] T. Hallgren and A. Ranta. An Extensible Proof Text Editor. In M. Parigot and A. Voronkov, editors, *LPAR-2000*, volume 1955 of *LNCS/LNAI*, pages 70–84. Springer, 2000. http://www.cse.chalmers.se/~aarne/articles/lpar2000.pdf.

[7] Albert Q. Jiang, Wenda Li, and Mateja Jamnik. Multi-language diversity benefits autoformalization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 83600–83626. Curran Associates, Inc., 2024.

[8] Andrei Paskevich. The Syntax and Semantics of The ForTheL Language, 2007.

[9] Shashank Pathak. GFLean: An Autoformalisation Framework for Lean via GF, 2024.

[10] Aarne Ranta. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford, 2011.

[11] Aarne Ranta. Translating between language and logic: What is easy and what is difficult. In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *Automated Deduction – CADE-23*, pages 5–25, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[12] Aarne Ranta. Towards multilingual autoformalization and informalization of mathematics. In *SLTC-2024, Swedish Language Technology Conference*, 2024. https://sltc2024.github.io/abstracts/ranta.pdf.

[13] Aarne Ranta, Krasimir Angelov, Normunds Gruzitis, and Prasanth Kolachina. Abstract Syntax as Interlingua: Scaling Up the Grammatical Framework from Controlled Languages to Robust Pipelines. *Computational Linguistics*, 46(2):425–486, 06 2020.

[14] Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, 2000.

[15] Qingxiang Wang, Cezary Kaliszyk, and Josef Urban. First experiments with neural translation of informal to formal mathematics, 2018.

[16] Freek Wiedijk. Formalizing 100 theorems, 2025. https://www.cs.ru.nl/~freek/100/.