OConcise – Towards an automatic mathematical assistant – Extended Abstract

Romana Ježek*

University of Vienna

Abstract

We introduce the software OConcise [8] for the semantic analysis of mathematical text. OConcise combines the already existing concept of using grammar sheets for parsing natural mathematical language with the automatic formalization of the parsed text and the verification of the mathematical proofs. In order to show that the software works we test the system on a chapter of the mathematical book "Universallogik" of W. Neumaier [13].

1 MOTIVATION. In the last decades, the computer has become more and more important for the work of mathematicians. Tools like Mathematica [6] and Matlab [5] are used as a matter of course, but there is rare use of proof assistants so far.
One reason could be that formal input requires text formalization, that is, if done manually, time-consuming and also significantly increases the text volume [18]. Some proof assistants accept an input in a controlled natural language, but such proof assistants require a different language, one always has to learn this language to make use of it. Large language models can be used with natural language, but they need a huge amount of data to be able to process the input, and their answers are not necessarily true.
OConcise is designed to support mathematical writing without the risk of producing incorrect information. In contrast to proof assistants, the input can be a normal LaTeX file without special annotations. The grammar of the text is passed in a separate type sheet that is explained in Section 2.1. These type sheets can be collected and used again such that the system will grow and understand more and more natural mathematical language.

2 THE FMATHL PROJECT. OConcise is part of the FMathL (Formal Mathematical Language) project [11], whose goal is to create a mathematical assistant that helps mathematicians with extracting information from literature, finding the important literature for their work, checking proofs and giving advices for writing new papers. To achieve this goal it should be able to:

1. read books and papers on mathematics and extract the knowledge contained in it
2. fill in gaps in a proofs and verify given results
3. communicate mathematical ideas and results between two systems
4. solve mathematical problems that are straightforward for a graduate student

For the first point the FMathL Type System [15], the dynamic generalized parser DynGenPar [10] and Concise [4], a program that stores the data in a structured way, have been created. OConcise enriches the system with the new capability of verifiying the proofs in a mathematical text. It is therefore the next step towards a mathematical assistant.

*supervised by Arnold Neumaier

2.1 The FMathL Type System. The semantic analysis in FMathL is based on a type system, a structure defining a semantical text representation and its grammar. A so-called type sheet tells the system how to interpret the text. Type sheets are created and extended interactively. Parts of the text are categorised and ordered into a generalized context-free grammar. A category contains the meaning of and the relationships between expressions. The syntactical rules for combining these expressions are defined in input and output usages. The input usage tells the system how the category should appear in the text being read, and the output usage tells the system how the text should be written. The semantical information is used for the automatic formalization. It is needed to avoid incorrect assignments or incorrect manipulation of mathematical objects. For example, adding an integer to a vector should not be accepted.

2.2 DynGenPar. The parser DynGenPar [10], developed as part of the FmathL project, produces a parse tree from a text and an imported grammar. DynGenPar can read PMCFG grammars [16] which is important for text written in natural language, which is usually not context-free [14]. The problem of ambiguity is solved by showing the user the possible alternatives.

2.3 Concise. Concise [4] was created by Domes [3] and Kofler [9] based on the theoretical foundation of A. Neumaier [12]. It integrates the parser DynGenPar and the type sheets to store the semantic information of a text into a graph. In the IDE the graph can be viewed and manipulated by the user. It also provides a text view of the graph and limited automatic writing of text files. Since Concise is not open source we decided to create the open source version OConcise for the further development.

3 PROOF VERIFICATION. After collecting all the information of the text OConcise tries to verify the proof steps. For the verification we use Vampire [1], a powerful automated theorem prover that supports first order logic, arithmetic, induction and also higher order logic [2]. Janota [7] investigated how much Vampire can automatically prove and disprove and got a very good result.
To make the text readable for Vampire, the output of DynGenPar is transformed into the TPTP format [17], an input format of Vampire. Vampire tries to prove a conjecture by negating it and finding a contradiction. The result can be:

1. UNSAT: the conjecture is valid
2. SAT: the conjecture is not valid
3. TIME OVER: the answer is unknown

Vampire prints status messages until either a strategy succeeds or the time limit is reached. Vampire uses the SZS standards for output by default and reports a human-readable proof. In OConcise the information will be passed to the user.

4 PROOF OF CONCEPT. OConcise is tested on a chapter of the book "Universallogik" by W. Neumaier [13]. The main purpose is to verify the proofs in the book by formalizing the necessary axioms automatically and testing the proof steps with Vampire.

References

[1] Ahmed Bhayat, Márton Hajdú, Petra Hozzová, Laura Kovács, Jakob Rath, Michael Rawson, Giles Reger, Martin Riener, Martin Suda, Johannes Schoisswohl, Andrei Voronkov, Ioan Dragan, Bernhard Gleiss, Kryštof Hoder, Evgeny Kotelnikov, Bernhard Kragl, Simon Robillard, and Chalmers. Vampire, Version 5.0.1. https://github.com/vprover/vampire?tab=readme-ov-file, 2026. Accessed: 2026-03-26.

[2] Filip Bártek, Ahmed Bhayat, Robin Coutelier, Márton Hajdu, Matthias Hetzenberger, Petra Hozzová, Laura Kovács, Jakob Rath, Michael Rawson, Giles Reger, Martin Suda, Johannes Schoisswohl, and Andrei Voronkov. The vampire diary. In *Computer Aided Verification*, pages 57–71, Cham, 2025. Springer Nature Switzerland.

[3] Ferenc Domes. *Rigorous techniques for continuous constraint satisfaction problems*. PhD thesis, University of Vienna, 2010.

[4] Ferenc Domes. Concise, Version 0.9. https://www.mat.univie.ac.at/~dferi/concise.html, 2012. Accessed: 2026-03-26.

[5] The MathWorks Inc. Matlab. https://www.mathworks.com, 1984–2025. Accessed: 2026-04-26.

[6] Wolfram Research, Inc. Mathematica. https://www.wolfram.com/mathematica, 1988-2025. Champaign, IL, Accessed: 2026-04-26.

[7] Mikolás Janota. Experimental results for vampire on the equational theories project. *CoRR*, abs/2508.15856, 2025.

[8] Romana Ježek. OConcise. https://github.com/jezekr/OConcise. Accessed: 2026-05-03.

[9] Kevin Kofler. *Dynamic generalized parsing and natural mathematical language*. PhD thesis, University of Vienna, 2017.

[10] Kevin Kofler and Arnold Neumaier. Dyngenpar – a dynamic generalized parser for common mathematical language. In *Proceedings of CICM 2012*, number 7362 in Lecture Notes in Artificial Intelligence. Springer, 2012.

[11] Arnold Neumaier. FMathL. https://www.arnold-neumaier.at/FMathL.html, 2010. Accessed: 2026-03-26.

[12] Arnold Neumaier and Peter Schodl. A framework for representing and processing arbitrary mathematics. *Proc. Int. Conf. Knowledge Engineering and Ontology Development (J. Filipe and J.L.G. Dietz, eds.)*, pages 476–479, 2010.

[13] Wilfried Neumaier. *Universallogik: Eine Synthese Klassischer Logiken von Aristoteles, Leibniz, Boole, Frege, Peano, Cantor, Zermelo ; Verbale Logik: Ein Grammatik-Kalkül Nach Ideen von Leibniz Und Peano*. Georg Olms Verlag, Hildesheim, 2020.

[14] Aarne Ranta. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford, 2011. ISBN-10: 1-57586-626-9 (Paper), 1-57586-627-7 (Cloth).

[15] Peter Schodl. *Foundations for a self-reflective, context-aware semantic representation of mathematical specifications*. PhD thesis, University of Vienna, 2011.

[16] Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.

[17] Geoff Sutcliffe. Stepping Stones in the TPTP World. In C. Benzmüller, M. Heule, and R. Schmidt, editors, *Proceedings of the 12th International Joint Conference on Automated Reasoning*, number 14739 in Lecture Notes in Artificial Intelligence, pages 30–50, 2024.

[18] Freek Wiedijk. The de bruijn factor. 2000. https://api.semanticscholar.org/CorpusID:16411752.