Making Presentation Math Computable: Proposing a Context Sensitive Approach for Translating LaTeX to Computer Algebra Systems

André Greiner-Petter[1], Moritz Schubotz[1,2], Akiko Aizawa[3], and Bela Gipp[1]

[1]University of Wuppertal, Wuppertal, Germany
(andre.greiner-petter@zbmath.org, {last}@uni-wuppertal.de)
[2]FIZ-Karlsruhe, Berlin, Germany ({first.last}@fiz-karlsruhe.de)
[3]National Institute of Informatics, Tokyo, Japan ({last}@nii.ac.jp)

July 15, 2020

Abstract

Scientists increasingly rely on computer algebra systems and digital mathematical libraries to compute, validate, or experiment with mathematical formulae. However, the focus in digital mathematical libraries and scientific documents often lies more on an accurate presentation of the formulae rather than providing uniform access to the semantic information. But, presentational math formats do not provide exclusive access to the underlying semantic meanings. One has to derive the semantic information from the context. As a consequence, the workflow of experimenting and publishing in the Sciences often includes time-consuming, error-prone manual conversions between presentational and computational math formats. As a contribution to improve this workflow, we propose a context-sensitive approach that extracts semantic information from a given context, embeds the information into the given input, and converts the semantically enhanced expressions to computer algebra systems.

1 Introduction

The document preparation system LaTeX has become a de facto standard[1] for writing scientific papers in STEM disciplines over the last 30 years [1]. Numerous other editors, such as the editor for Wikipedia articles[2] or Microsoft Word [11], entirely or partially support LaTeX expressions. LaTeX provides a

[1] https://www.latex-project.org/ [Accessed 03-24-2020]

[2] https://en.wikipedia.org/wiki/Help:Displaying_a_formula [Accessed 03-24-2020]

syntax for printing mathematical formulae that is similar to the way a person would write the math by hand. Thus, LaTeX focuses on the presentation of formulae but does not explicitly carry their semantic information.

For a human reader, LaTeX's focus on formulae presentation is typically not a problem since readers can deduce the semantics of the formulae from the surrounding context and the reader's prior knowledge. Consider the Euler-Mascheroni constant represented by the Greek letter γ. Without further information, γ is just a Greek letter, often used to describe this mathematical constant but can also be used to represent curve parametrization, among other things. Based on the context, a human reader can interpret γ correctly and connect the letter with the semantic background. Computational systems, however, have issues identifying the correct semantics of formulae if the formulae do not provide enough context. For example, in LaTeX, γ is represented as `\gamma`.

Explicitly given semantic information in mathematical expressions becomes increasingly relevant in computational mathematics. Nowadays, many scientists also compute formulae from their papers [2, 3]. They evaluate specific values, create diagrams, and search or calculate practical solutions. Computer Algebra Systems (CAS) are software tools that allow for such computations and visualizations of mathematical expressions. CAS create their representations (hereafter referred to as CAS input) with the intent of creating an input syntax that is intuitive and easy to type. CAS input must be unambiguous to CAS. Otherwise, a CAS is unable to perform computations and visualizations. CAS input is not standardized; instead, each CAS provider has created its own syntax that differs from other systems [10]. The workflow of writing a paper, therefore, leads to the problem of continually transforming mathematical expressions from LaTeX to CAS input and back. Since LaTeX does not carry the semantic information explicitly, the CAS is unable to parse complex input directly. Thus, the author must perform the transformation manually, which is time-consuming and error-prone.

Transformations between CAS input and LaTeX are not straightforward and require substantial knowledge of the internal processes for the CAS [10]. Table 1 illustrates the differences in representations exemplified for a Jacobi polynomial [5]. The expression in generic LaTeX, i.e., general LaTeX without custom macros, sharply differs from the semantically unique terms in CAS inputs. To overcome the issue of missing explicit semantic information in LaTeX expressions, the National Institute of Standards and Technology (NIST) has developed a unique set of semantic LaTeX macros. NIST uses these macros for the Digital Library of Mathematical Functions (DLMF) [14] and the Digital Repository of Mathematical Formulae (DRMF) [4]. Both DLMF and DRMF macros enhance the search capabilities on the DLMF and DRMF websites and establish info boxes that provide short descriptions of the symbols, link to their definitions, and further literature. Table 1 shows that the semantically enhanced LaTeX is closer to the syntax supported by a CAS. In the following, we will refer to semantically enhanced LaTeX as semantic LaTeX, and general LaTeX expressions as generic LaTeX, respectively. In the following, we will propose a context-sensitive approach to convert the generic LaTeX expressions to CAS. The approach will

Systems	Representations
Rendered Version	$P_n^{(\alpha,\beta)}(\cos(a\Theta))$
Generic LaTeX	`P_n^{(\alpha,\beta)}(\cos(a\Theta))`
Semantic LaTeX	`\JacobiP{\alpha}{\beta}{n}@{\cos@{a\Theta}}`
CAS Maple	`JacobiP(n,alpha,beta,cos(a*Theta))`
CAS Mathematica	`JacobiP[n,\[Alpha],\[Beta],Cos[a \[CapitalTheta]]]`

Table 1: Representations of a Jacobi polynomial in different systems.

take advantage of existing tools and datasets.

1.1 Related Work

To the best of our knowledge, there is no system nor a theoretical concept yet that allows for translating LaTeX expressions to CAS and taking the context of the expression into account. Existing tools, such as the inbuild import/export functions of CAS, ignore context information and are therefore limited to simple, unambiguous cases (e.g., `\frac{1}{2}` or `\cos x`) [10].

We previously developed a system called LaCASt, that converts semantic LaTeX expressions to the CAS Maple and Mathematica [10]. LaCASt is essentially a rule-based engine that performs translations based on manually crafted patterns. The engine follows a modular concept, which allows for extending the system without additional coding, e.g., by extending or creating new lists of translation patterns. Cohl et al. [8] have shown that LaCASt is able to identify errors in digital mathematical libraries and CAS. However, LaCASt also does not consider the context of math formulae, since the necessary semantic information is encoded in the semantic macros. Moreover, the use of the semantic LaTeX dialect is currently limited to the DLMF and DRMF. Hence, the next step is to extend the system to work with generic LaTeX inputs.

2 Towards a Context-Sensitive Approach

LaCASt performs the translation based on parse trees, which are generated by the Part-of-Math (POM) tagger [7]. Similar to the Part-of-Speech (POS) taggers in natural language processing (NLP), the POM tagger also tags tokens with additional information. In its current state, the POM tagger does not consider context information. Thus, the parse tree generated by the POM tagger should not be misunderstood as a syntax tree of equations. Since semantic LaTeX is an extension of generic LaTeX, the POM tagger is also able to parse semantic LaTeX expressions. The POM tagger stores the information about tokens in a manually crafted database, called lexicons. The lexicons contain possible semantic information for symbols. For example, the lexicon entry for ζ contains twelve different meanings [7, 10]. Three of the twelve entries are special functions: the Weierstrass zeta function, the Riemann zeta function, and the Hurwitz zeta function. Each meaning also provides information about the structure of the function. For example, the Hurwitz zeta function $\zeta(s, a)$ has two arguments. The first argument is a complex variable, while the second is a complex parameter.

The semantic information of a mathematical formula is either given in the context or can be derived from the structure of the formula (e.g., when the notation of an expression is unambiguous). The lexicons of the POM tagger and the definitions of the semantic LaTeX macros provide a database of standardized notations of mathematical functions. Hence, this knowledgebase can be used to derive semantic information from the structure of an expression. To analyze the textual context, we can use the Mathematical Language Processor (MLP) [6]. The MLP aims to extract the textual descriptions, called definiens, from the context of a mathematical expression. The MLP focuses on single mathematical symbols, named identifiers. An identifier might also include the subscript since a symbol with a subscript is often interpreted as one mathematical object. The basic approach of the MLP is that candidates of definiens and identifiers are connected when the distance between them is small, i.e., fewer words appear between the identifier and its definiens. The score also considers the distance of identifier-definiens pairs to complex mathematical expressions that contain the identifier. Schubotz et al. [6] also presented ten patterns of phrases, defined by domain experts, that introduce a new pair of definiens and identifier, such as `<identifier> (is|are) <definiens>`. The authors reported the precision of $p = 0.4860$ and the recall of $r = 0.2806$ for their new machine learning approach. The concept of the MLP is implemented in a publicly available Java framework called mathosphere[3].

For the Jacobi polynomial from Table 1, $P_n^{(\alpha,\beta)}(x)$, mathosphere extracts four identifier P_n, α, β, and x rather than groups of tokens, such as $P_n^{(\alpha,\beta)}(x)$. Without considering $P_n^{(\alpha,\beta)}(x)$ as one mathematical object, it is challenging to identify α, β, and n as parameters and x as the variable. We addressed this issue in [12] by identifying so-called Mathematical Objects of Interest (MOI). MOI represent meaningful groups of tokens rather than single identifiers. In [12], we developed a search engine to find MOI by a given textual query. For example, the top-3 results for the search query '*Jacobi Polynomial*' were $P_n^{(\alpha,\beta)}(x)$, $P_n^{(\alpha,\beta)}$, and $\beta > -1$ (which is one of the constraints of Jacobi polynomials). The search engine allows for linking mathematical expressions with textual queries. The retrieved MOIs are based on the distributions of mathematical formulae in the corpus of arXiv[4] and zbMATH[5]. Hence, they represent common relevant expressions for a given textual query.

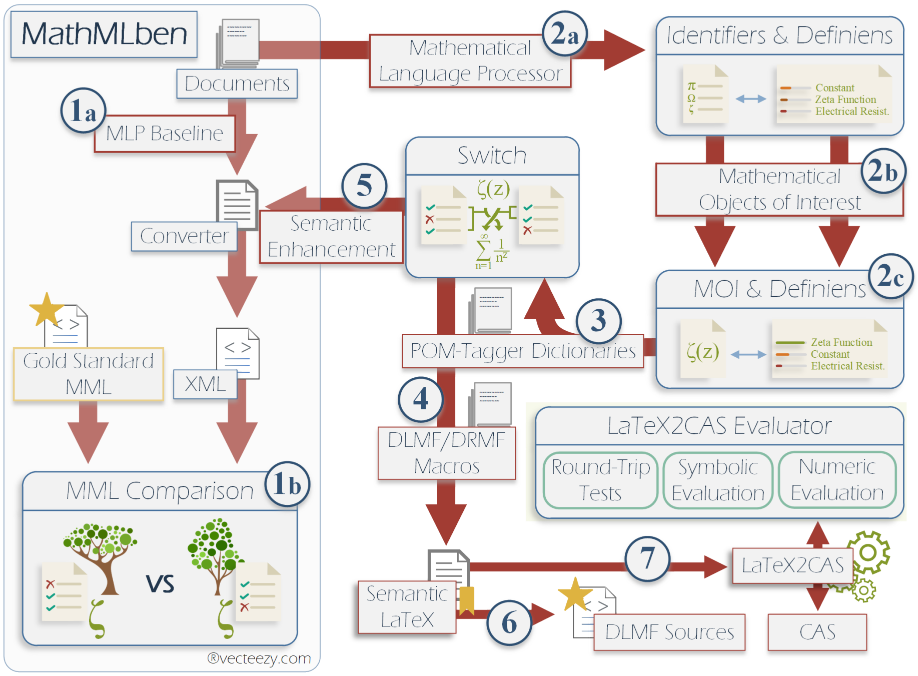

Figure 1: Pipeline of the proposed context-sensitive conversion process. The project extracts semantic information from real-world documents (2), enhances the mathematical input expressions with the extracted information (3-4), and transforms the math into CAS representations in the final step (6-7).

3 Conversion and Evaluation Pipeline

Figure 1 illustrates the pipeline of the proposed system to convert generic LaTeX expressions to CAS. The figure contains numbered badges that represent the different steps in the system. Steps 2-5 represent the conversion pipeline, while steps 1, 6, and 7 are different ways to evaluate the system. Mathosphere [6] will serve as the baseline. With MathMLben [9], a benchmark for MathML, we tested the performance of several LaTeX to MathML conversion tools. MathMLben provides a manually crafted semantically annotated dataset for 300 mathematical formulae. We evaluate mathosphere on this annotated dataset in step 1a.

The conversion pipeline starts with mathosphere (step 2a) to extract identifier-definiens pairs from the given context. Since mathosphere only considers single identifiers, we will use the developed search engine in [12] to derive MOIs for the extracted definiens (step 2b). The identified MOIs can be matched against

[3] `https://github.com/ag-gipp/mathosphere` [Accessed 03-24-2020]
[4] `https://arxiv.org` [Accessed 03-24-2020]
[5] `https://zbmath.org` [Accessed 03-24-2020]

complex expressions in the context. Therefore, we end up with MOI-definiens pairs in step 2c, where the scores are calculated based on the relevance of MOIs and the original scores generated by mathosphere.

Once we extracted the MOI-definiens pairs, we replace the generic LaTeX expressions by their semantic counterparts (steps 3-4). This can be done based on the lexicons of the POM tagger and the DLMF Macro definition files, which both provide information about the argument layout of functions. This information is important to identify fixed notations, i.e., P in $P_n^{(\alpha,\beta)}(x)$, and the variables/parameters, i.e., α, β, n, and x in $P_n^{(\alpha,\beta)}(x)$. After these steps, we have the option to evaluate the system in three different ways.

First, we improve the conversion process of LaTeX to MathML conversion tools by considering the extracted MOI-definiens pairs. Thus, we can measure the improvement of considering the context against the results in the MathMLben benchmark tests in [9], which did not use the information from the context. Second, we evaluate the generated semantic LaTeX expressions on the DLMF dataset. The DLMF is internally written in semantic LaTeX, but provides external access to the generic LaTeX version of each formula. Hence, the DLMF can be interpreted as a manually annotated dataset of LaTeX expressions. Third, we use the evaluation system of LaCASt [8], which uses CAS to check if a translated equation is still valid after the translation system. The latter is useful to compare the performance of the conversion from LaTeX to CAS with manually (semantic LaTeX from the DLMF) and automatically (proposed pipeline) annotated semantic information.

4 Conclusion

We presented a novel context-sensitive approach to convert mathematical LaTeX expressions to CAS. The proposed pipeline based on existing tools and datasets, such as MLP [6], POM tagger [7], LaCASt [10], and MathMLben [9]. Realizing the proposed pipeline is part of our current research.

Acknowledgments

This work was supported by the German Research Foundation (DFG grant GI-1259-1).

References

[1] A. Gaudeul. “Do Open Source Developers Respond to Competition?: The (La)TeX Case Study”. In: *Review of Network Economics* 6.2 (2006), p. 9. DOI: 10.2139/ssrn.908946.

[2] N. P. Karampetakis and A. I. G. Vardulakis. "Special issue on the use of computer algebra systems for computer aided control system design". In: *International Journal of Control* 79.11 (Nov. 2006), pp. 1313–1320. DOI: 10.1080/00207170600882346.

[3] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra (3. ed.)* Cambridge University Press, 2013.

[4] H. S. Cohl et al. "Growing the Digital Repository of Mathematical Formulae with Generic LaTeX Sources". In: *Proc. CICM*. Ed. by M. Kerber et al. Vol. 9150. Springer, 2015, pp. 280–287. DOI: 10.1007/978-3-319-20615-8_18.

[5] H. S. Cohl et al. "Semantic Preserving Bijective Mappings of Mathematical Formulae Between Document Preparation Systems and Computer Algebra Systems". In: *Proc. CICM*. Ed. by H. Geuvers et al. Vol. 10383. Springer, 2017, pp. 115–131. DOI: 10.1007/978-3-319-62075-6_9.

[6] M. Schubotz et al. "Evaluating and Improving the Extraction of Mathematical Identifier Definitions". In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 8th International Conference of the CLEF Association, CLEF 2017, Dublin, Ireland, September 11-14, 2017, Proceedings*. Ed. by G. J. F. Jones et al. Vol. 10456. Springer, 2017, pp. 82–94. DOI: 10.1007/978-3-319-65813-1_7.

[7] A. Youssef. "Part-of-Math Tagging and Applications". In: *Lecture Notes in Computer Science*. Ed. by H. Geuvers et al. Vol. 10383. Springer, 2017, pp. 356–374. DOI: 10.1007/978-3-319-62075-6_25.

[8] H. S. Cohl, A. Greiner-Petter, and M. Schubotz. "Automated Symbolic and Numerical Testing of DLMF Formulae Using Computer Algebra Systems". In: *Proc. CICM*. Ed. by F. Rabe et al. Vol. 11006. Springer, 2018, pp. 39–52. DOI: 10.1007/978-3-319-96812-4_4.

[9] M. Schubotz et al. "Improving the Representation and Conversion of Mathematical Formulae by Considering their Textual Context". In: *Proc. ACM IEEE JCDL*. Ed. by J. Chen et al. Fort Worth, USA: ACM, 2018, pp. 233–242. DOI: 10.1145/3197026.3197058.

[10] A. Greiner-Petter et al. "Semantic Preserving Bijective Mappings for Expressions involving Special Functions in Computer Algebra Systems and Document Preparation Systems". In: *Aslib Journal of Information Management* 71.3 (July 2019), pp. 415–439. DOI: 10.1108/AJIM-08-2018-0185.

[11] D. Matthews. "Craft beautiful equations in Word with LaTeX". In: *Nature* 570.7760 (June 2019), pp. 263–264. DOI: 10.1038/d41586-019-01796-1.

[12] A. Greiner-Petter et al. "Discovering Mathematical Objects of Interest - A Study of Mathematical Notations". In: *Proceedings of The Web Conference 2020 (WWW'20), April 20–24, 2020, Taipei, Taiwan*. Apr. 2020. DOI: 10.1145/3366423.3380218.

[14] *NIST Digital Library of Mathematical Functions*. `http://dlmf.nist.gov/`, Release 1.0.25 of 2019-12-15. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and M. A. McClain, eds.

Listing 1: Use the following BibTeX code to cite this article

```
@inproceedings{Greiner-Petter2020c,
  author = {Greiner-Petter, Andr\'{e} and Schubotz, Moritz
     and
        Aizawa, Akiko and Gipp, Bela},
  title  = {Making Presentation Math Computable:
        Proposing a Context Sensitive Approach for
        Translation LaTeX to Computer Algebra Systems},
  booktitle = {Mathematial software -- ICMS 2020},
  series   = {Lecture Notes in Computer Science},
  volume   = {12097},
  publisher = {Springer},
  year    = {2020}
}
```