

System Description: XSL-based Translator of Mizar to LaTeX

Grzegorz Bancerek², Adam Naumowicz^{1*}, and Josef Urban^{2**}

¹ University of Bialystok

² Czech Technical University in Prague

Abstract. We describe a new version of the Mizar-to-LaTeX translator. The system has been re-implemented as XSL stylesheets instead of Pascal programs, allowing greater flexibility. It can now be used to generate both LaTeX/PDF and HTML with MathJax code. We also experimentally support generation of full proofs. Finally, the system is now available online and through the Mizar Emacs interface.

1 Introduction

The Mizar project [3] has since its inception in 1973 focused on formal representation of mathematics that would be as close possible to its representation in natural language. After the Mizar language emerged and a larger number of articles have been collected in the Mizar Mathematical Library (MML) [5,2], the Mizar team started to experiment with translating the Mizar articles into \LaTeX . This translation has been evolving for the last three decades [1,4,8].

Here we describe the most recent large re-implementation of the system which switches from custom Pascal programs to more flexible XSL processing, done primarily by the first author over the last eight years. This in turn builds on the XML-ization of the Mizar processing, done over the past decade [9,7]. The system can now produce both \LaTeX /PDF and HTML using MathJax for displaying formulas and terms. We experimentally support also translation of full proofs. The system is used for in-house production of the journal Formalized Mathematics¹ [6] and it is newly also available as a remote PDF/HTML service for the Mizar authors via the Emacs authoring environment for Mizar[10], similarly to the MizAR [11] ATP service for Mizar.

2 Summary of the Old Mizar-to- \LaTeX Translation

The previous incarnation of the translator as well as its history are described in detail in [1]. The complete process was carried out by a set of eight custom programs (`fmparse`, `newfmfrm`, `addfmfrm`, `fmfrm`, `resvar`, `fmnotats`,

*Partially supported by the project N^o 2017/01/X/ST6/00012 financed by the National Science Center, Poland and the COST Action EUTypes CA15123. The Mizar processing has been performed using the infrastructure of the University of Bialystok High Performance Computing Center.

**Supported by ERC Consolidator grant nr. 649043 *AI4REASON*.

¹ <https://www.degruyter.com/view/j/forma>

fmanalyz and jformat) run in a succession after preparing the article with the standard Mizar accommodator utility and before producing the final output with L^AT_EX using B^BT_EX metadata provided by the user/editor.

Let us only briefly recall that the translation of the non-formula text parts was done as a static mapping of the keywords. For example the commutativity property for functors was expressed by the sentence *Let us observe that the functor is commutative*. The translation of atomic formulas, terms, and types was done according to a database of L^AT_EX *translation patterns* for the Mizar symbols with arities (a *format* in the Mizar terminology). Each pattern consists of a control header (deciding e.g. about using the math mode in L^AT_EX, bracketing, etc.) and one or more proper patterns that map the format with arguments to L^AT_EX in various contexts. Following is an example of such a pattern for a typical PartUnion functor:²

```
O1 0 2
OPartUnion
mol@s#1#2; \bigcup_{\beta<_{#2}}\beta
```

This means that the Mizar term PartUnion(B, R) would be displayed as $\bigcup_{\beta <_R B} \beta$. Note that in this case the translation is quite nontrivial and it reveals information about how the symbol PartUnion is defined in Mizar.

The process of translation worked through several levels of generalization and it always used the pattern from the most specific level fitting the formula translated. All top-level elements of an article, i.e. theorems, definitions, schemes, reservations and global shortcuts were presented in the rendering, but proofs were not translated as a rule. Single letter variables occurring in an article were preserved while others were abbreviated into single letters with indices.

3 Description of the New Technology

The new technology of automated translation currently used for Mizar texts published in the *Formalized Mathematics* journal is based on XSL translation templates applied to the XML representation of the weakly-strict Mizar [7] encoding of the original Mizar input file (*.wsx file). However, the semantic representation generated by the Mizar verifier (*.xml file) is also used to decode links to external articles. All bibliographic metadata are first translated to special XML format and merged with information extracted from the Mizar article. Global (for the journal) L^AT_EX translation patterns are also kept in the XML pub.xml file. In the following sections we describe the basic functions of the main XSL stylesheets. They have been designed to perform well-defined simple iterative tasks within the process of generating the final L^AT_EX rendering for PDF and MathJax-enabled HTML presentation.

² http://mizar.uwb.edu.pl/version/current/html/pcomps_2#K1

3.1 addformat.xml

This is the main stylesheet responsible for selecting information to be translated from the weakly-strict Mizar representation. Identified symbols are matched with their format specification to be later replaced by concrete translations.

3.2 addtranslation.xml

This script augments the processed file with available translation patterns. An example of a concrete pattern (for the previously mentioned PartUnion functor) as extracted from the `pub.xml` file looks as follows:

```
<Translation
  voc="PCOMPS_2" kind="0" symbolnr="1" symbol="OPartUnion" argsnr="2"
  leftargsnr="0" rightargsnr="2" format="O1 0 2" header="mol@s#1#2;"
  priority="8" forcing="rqw" context1="1" context2="" TeX-mode="m">
  <pattern>
    \bigcup_{\beta{\&lt;_}{<X pos="lower" locus="2"/>}}{<X locus="1"/>}\beta
  </pattern>
</Translation>
```

The system also proposes formats for new definitions introduced in the current article. The `unknown_patterns.xml` stylesheet handles unknown patterns.

3.3 varrepr.xml

The task of this stylesheet is to replace any identifiers that contain names of Greek letters into corresponding L^AT_EX symbols. Longer variable identifiers are given standardized representations with subscripts (e.g. AA becomes A_1).

3.4 multipred.xml

This procedure is technically split into several passes of `multipred.xml`, `multipred2.xml` and `multipred3.xml` run in succession together with `prune.xml`. The goal is to locate in the input text a list of constructs with a qualifying format that can be printed together in a shortened form. E.g. in the PCOMPS_2 article, instead of the literal translation: “ G_9 is cover of P_6 , and G_9 is finer than F_9 ” the script produces “ G_9 is cover of P_6 and finer than F_9 ”.

3.5 transitive.xml

This stylesheet improves the quality of the translation by generating output of the form: “ $x < y < z$ ” instead of “ $x < y$ and $y < z$ ”. I.e., it joins consecutive predicates with shared arguments as is usually done in informal mathematics.

3.6 compress.xml

There are several independent stylesheets that compress and thus make more natural for the reader the occurrences of particular constructs in a common context. These are: `compress_let.xml`, `compress_for.xml`, `compress_-assume.xml`, `brackets.xml`, `compress_func.xml`, and `compressres.xml` for generalizations, quantifiers, assumptions, brackets, functor definitions with common arguments, and variable reservations, respectively.

3.7 recognize-programs.xml

The set of templates `recognize-programs.xml`, `recognize-programs2.xml` and `recognize-programs3.xml` is used to generate custom encoding of specific complex terms - representation of programs. E.g. a special format for a concrete program defined for a Mizar model of an abstract computer: $(((((\langle \text{if } d_1 > 0 \text{ goto } 2 \rangle \wedge \langle \text{halt}_{\text{SCM}} \rangle) \wedge \langle d_3 := (d_0) \rangle) \wedge \langle \text{SubFrom}(d_1, d_0) \rangle) \wedge \langle \text{if } d_1 = 0 \text{ goto } 1 \rangle) \wedge \langle d_4 := (d_2) \rangle) \wedge \langle d_2 := (d_3) \rangle) \wedge \langle \text{AddTo}(d_3, d_4) \rangle) \wedge \langle \text{SCM} - \text{goto } 3 \rangle$.

3.8 article2latex.xml and article2html.xml

Depending on the output format, based on the preparatory tasks performed by the previously mentioned stylesheets, one may choose to generate either \LaTeX code for a self-contained PDF article, or code embedded in an HTML document to be rendered by MathJax. If \LaTeX code is chosen, we finally run `pdflatex` using several Mizar-specific headers.

3.9 Remote Service and Processing Times

Since the translation toolchain has many components and relies on a number of custom tools and their versions installed, we make it available as an online service. This is similar to the solution taken for the MizAR [11] ATP service for Mizar. The Mizar users can now send their articles from Emacs to the service for PDF and HTML translation of their current work independently of the publication process of *Formalized Mathematics*. This is done by selecting the *Mizar->Remote solving->Produce PDF online* and *Mizar->Remote solving->Produce MathJax online* menu options, respectively.³ The remote processing of a basic Mizar article such as `XBOOLE_1` takes about 15 s and the whole MML can be processed locally on the server overnight. This is quite comparable to the speed of the earlier Pascal-written version of the toolchain.

4 Conclusion and Future Work

The described re-implementation of the Mizar to \LaTeX translation system is based on the flexible and easily extensible XML/XSL technology rather than custom Pascal programs tied to the specific implementation of the core Mizar system. Thanks to this approach, a set of shared scripts and stylesheets can be used for the production of the printed editions of the journal *Formalized Mathematics* but also as an on-demand remote PDF/HTML service for the Mizar authors. Users of the Emacs authoring environment for Mizar can now experiment with the PDF and HTML translation of their current work.

The future work on the translation will focus on adding more natural language based linguistic features to the generated mathematical text. In particular,

³ Both menu items trigger the `mizar-tex-remote` function in the current `mizar.el` Emacs Lisp mode for Mizar available at <https://github.com/JUrban/mizarmode/blob/master/mizar.el>

the human-friendly presentation of the structure of (nested with varying level of importance) proofs will be investigated. It can now be experimented with by modifying the `addformat.xml` stylesheet which controls filtering relevant information from the original Mizar article and either the `article2latex.xml` or `article2html.xml` templates that generate the final specific rendering in L^AT_EX or MathJax/HTML format.

References

1. G. Bancerek. Automatic translation in formalized mathematics. *Mechanized Mathematics and Its Applications*, 5(2):19–31, 2006.
2. G. Bancerek, C. Byliński, A. Grabowski, A. Kornilowicz, R. Matuszewski, A. Naumowicz, and K. Pąk. The role of the Mizar Mathematical Library for interactive proof development in Mizar. *Journal of Automated Reasoning*, Nov 2017.
3. G. Bancerek, C. Byliński, A. Grabowski, A. Kornilowicz, R. Matuszewski, A. Naumowicz, K. Pąk, and J. Urban. Mizar: State-of-the-art and beyond. In M. Kerber, J. Carette, C. Kaliszyk, F. Rabe, and V. Sorge, editors, *Intelligent Computer Mathematics - International Conference, CICM 2015, Washington, DC, USA, July 13-17, 2015, Proceedings*, volume 9150 of *Lecture Notes in Computer Science*, pages 261–279. Springer, 2015.
4. G. Bancerek and P. Carlson. Mizar and the machine translation of mathematics documents. Available online at http://www.mizar.org/project/banc_carl93.ps.
5. A. Grabowski, A. Kornilowicz, and A. Naumowicz. Four decades of Mizar - foreword. *J. Autom. Reasoning*, 55(3):191–198, 2015.
6. A. Grabowski and C. Schwarzweller. Revisions as an essential tool to maintain mathematical repositories. In M. Kauers, M. Kerber, R. Miner, and W. Windsteiger, editors, *Towards Mechanized Mathematical Assistants, 14th Symposium, Calculemus 2007, 6th International Conference, MKM 2007, Hagenberg, Austria, June 27-30, 2007, Proceedings*, volume 4573 of *Lecture Notes in Computer Science*, pages 235–249. Springer, 2007.
7. A. Naumowicz and R. Piliszek. Accessing the Mizar library with a weakly strict Mizar parser. In M. Kohlhase, M. Johansson, B. R. Miller, L. de Moura, and F. W. Tompa, editors, *Intelligent Computer Mathematics - 9th International Conference, CICM 2016, Bialystok, Poland, July 25-29, 2016, Proceedings*, volume 9791 of *Lecture Notes in Computer Science*, pages 77–82. Springer, 2016.
8. A. Trybulec and P. Rudnicki. A collection of texed mizar abstracts. Available online at <http://www.mizar.org/project/TR-89-18.pdf>.
9. J. Urban. XML-izing Mizar: Making semantic processing and presentation of MML easy. In M. Kohlhase, editor, *MKM*, volume 3863 of *LNCS*, pages 346–360. Springer, 2005.
10. J. Urban. MizarMode - an integrated proof assistance tool for the Mizar way of formalizing mathematics. *J. Applied Logic*, 4(4):414 – 427, 2006.
11. J. Urban, P. Rudnicki, and G. Sutcliffe. ATP and presentation service for Mizar formalizations. *J. Autom. Reasoning*, 50:229–241, 2013.