Noname manuscript No.
(will be inserted by the editor)

The Mizar Mathematical Library in OMDoc: Translation and Applications

Mihnea Iancu · Michael Kohlhase · Florian Rabe · Josef Urban

the date of receipt and acceptance should be inserted later

Abstract The Mizar Mathematical Library is one of the largest libraries of formalized mathematics. Its language is highly optimized for authoring by humans. As in natural languages, the meaning of an expression is influenced by its (mathematical) context in a way that is natural to humans, but harder to specify for machine manipulation. Thus its custom file format can make the access to the library difficult. Indeed, the Mizar system itself is currently the only system that can fully operate on the Mizar library.

This paper presents a translation of the Mizar library into the OMDoc format (**O**pen **M**athematical **Doc**uments), an XML-based representation format for mathematical knowledge. OMDoc is geared towards machine support and interoperability by making formula structure and context dependencies explicit. Thus, the Mizar library becomes accessible for a wide range of OMDoc-based tools for formal mathematics and knowledge management. We exemplify interoperability by indexing the translated library in the MathWebSearch engine, which provides an "applicable theorem search" service (almost) out of the box.

1 Introduction

Mizar [TB85] is a representation format for mathematics that is close to mathematical vernacular used in publications. Mizar is also a formal system for completing and verifying proofs written in the Mizar language. The continual development of the Mizar system has resulted in a centrally maintained library of mathematics: the Mizar mathematical library (MML). The MML is a collection of Mizar articles:

Mihnea Iancu
Computer Science, Jacobs University Bremen, E-mail: m.iancu@jacobs-university.de

Michael Kohlhase
Computer Science, Jacobs University Bremen, E-mail: m.kohlhase@jacobs-university.de

Florian Rabe
Computer Science, Jacobs University Bremen, E-mail: f.rabe@jacobs-university.de

Josef Urban
Computing and Information Sciences, Radboud University, E-mail: Josef.Urban@gmail.com

text-files that contain definitions, theorems, and proofs. Currently the MML (version 4.166.1132) contains over 1000 articles with over 50000 theorems and over 10000 definitions.[1] Introductory information on Mizar and the MML can be found in [TR99] and [Wie99]. For the rest of this paper, we assume that the reader is at least superficially familiar with these basic texts.

The Mizar language is based on Tarski-Grothendieck set theory [Try90] formalized in (unsorted) first-order logic.[2] In addition, Mizar provides a very expressive and flexible type system that features dependent types as well as predicate restrictions [Ban03]. The Mizar language — in particular the type system and the input syntax — are highly optimized for authoring by humans. Consider for instance the following theorem:

```
for A being set holds
  A is finite iff ex f being Function st rng f = A & dom f in omega
```

For a skilled mathematician this can be almost read and understood without Mizar-specific training. The downside of this is that the Mizar system is currently the only system that can fully operate on the Mizar library, and as a consequence, many feel that the Mizar library is locked up in a custom file format that excludes it from the methods and tools developed in the mathematical knowledge management (MKM) communities.

In this paper, we show how he have remedied this situation by describing and implementing a translation of the MML into the OMDoc/LF language. OMDoc (**O**pen **M**athematical **Doc**uments [Koh06]) is an XML-based representation format for mathematical knowledge geared towards making formula structure and context dependencies explicit for machine support. OMDoc is parametric in the underlying logical formalism, and we use its instantiation with the Edinburgh Logical Framework (LF, [HHP93]) to formally define the Mizar language.

This Mizar to OMDoc transformation completely rethinks the information architecture and indeed enhances the OMDoc language design in the process.

Our translation satisfies three requirements that are as indispensable for interoperability as they are hard to combine: (i) it preserves the human-oriented structure of Mizar expressions and declarations, e.g., the type system is not coded out; (ii) it uses only the generic representational infrastructure of a simple framework language (OMDoc/LF in our case) so that further processing is possible without hard-coding any idiosyncrasies of Mizar; (iii) the result can be verified based on a formal representation of the Mizar syntax and semantics in LF.

In the next section, we discuss related translations, the new information architecture of ours and the concrete implementation. Details about the translation can be found in [IKR11]. In section 3 we discuss interoperability and exemplify the tool support by integrating a novel search engine into a web-based frontend for Mizar formalization. Section 4 concludes the paper.

[1] See http://mmlquery.mizar.org/ for up-to-date statistics.

[2] Technically, theorem schemes and the Fraenkel operator of Mizar slightly transcend first-order expressivity, but the language is first-order in style.

2 Translating Mizar to OMDoc

The motivations for a translation of the Mizar library are not particular to the OMDoc format, and it is therefore not very surprising that the work reported on in this paper is not the first attempt to translate the Mizar library.

When processing Mizar text we have the choice between two levels of language: The **pattern-level** (presentation-level) language is the rich human-oriented external syntax in which Mizar articles are written; and the **constructor-level** (semantic level) is the machine-internal representation used in the Mizar system.

The Mizar project produces a hyper-linked, pretty-printed version of the assertions of a Mizar article for publication in the journal *Formalized Mathematics* [Ban06a] with its electronic counterpart the *Journal of Formalized Mathematics* [JFM]. This pattern-level translation generates a human-oriented presentation of Mizar articles, where formulae are presented in mathematical notation using LaTeX and parts of the Mizar logical language are verbalized.

There have been various early hand translations of selected Mizar articles for benchmarking automated theorem provers (see e.g. [DW97]). In 1997/8 Czeslaw Bylinski and Ingo Dahn translated the MML into a PROLOG syntax by extending the Mizar system with a custom (constructor-level) PROLOG generator. This was done independently of the main Mizar code base, and as a result soon desynchronized with it. A structure-preserving transformation of constructor-level Mizar to OMDoc has been also attempted in [BK07], but has remained partial. Constructor-level Mizar has been used for information retrieval purposes in the MMLQ system which provides a web interface to and a query language; see [Ban06b] for details.

Finally, the last author has defined a custom XML format in [Urb06c] covering initially mainly the constructor-level, and modified Mizar to use this format internally. In this way it is ensured that this XML format cannot desynchronize from the (often rapid) Mizar development, and can be used as a faithful translation layer for external tools. At the same time, this means that the format has to be quite Mizar-oriented, and its use in generic MKM systems typically requires a translation layer. One such layer — the MPTP system [Urb06b] — has been developed for translation targeted at automated theorem proving systems based on the TPTP syntax. The MPTP translation is mostly concerned with semantics and formulas, the presentation layer is practically omitted there, and a custom proof translation was only added experimentally later in [US08].

The work presented here is analogous to the MPTP translation in that it uses the Mizar XML as an API suitable for implementing a translation to a general mathematical format, allowing a number of independently developed general tools to work with the MML.

2.1 Representing the Mizar Language in OMDoc

OMDoc is a content markup format and data model for mathematical documents. It models mathematical content using three levels of abstraction:

Object Level: OMDoc uses OpenMath and MathML as established standards for the markup of *formulae*. Mizar types, terms and formulas correspond to this level.

Statement Level: OMDoc supplies original markup for explicitly representing the *declarations* and *assertions* in mathematical theories. Mizar definitions, theorems, schemes, notations, and registrations correspond to this level.

Theory Level: Finally OMDoc offers original markup that allows for clustering sets of statements into *theories* as well as specifying relations between them (inclusions, morphisms). Mizar articles and imports between them correspond to this level.

Core OMDoc concentrates on the structural relations between these mathematical concepts. It deliberately avoids fixing language primitives for them and abstracts from specific mathematical foundations. This is a crucial design choice that makes OMDoc a universal representation format while remaining manageably simple.

For the case of Mizar, this means that core OMDoc does not feature exact analogues to Mizar's sophisticated definition principles. Neither can it adequately represent the theoremhood property that defines the semantics of Mizar formulae. This is not surprising because these features are highly specific to the syntax and semantics of individual languages. The extension of core OMDoc with such language-specific features is the role of *pragmatic* OMDoc, which we will discuss next.

Core OMDoc uses a minimal set of conceptually orthogonal representational primitives, resulting in expressions with canonical structure, which simplifies the theoretical analysis and implementation of core OMDoc expressions (e.g., see [KRZ10]).

Pragmatic OMDoc, on the other hand, strikes a balance between verbosity and formality. It permits introducing a complex representational infrastructure that provides both a formal semantics and is intuitive for humans. In particular, the semantics of these language-specific extensions is defined entirely within core OMDoc. Thus, OMDoc can provide multiple "pragmatic vocabularies" catering to different communities and their tastes.

OMDoc achieves a pragmatic object level by being parametric in the foundational framework in which the syntax and semantics of a language are formalized. More concretely, a logical framework — LF in our case — is described as an OMDoc theory. Then a logic — Mizar's first-order logic in our case — is defined as another OMDoc theory with *meta-theory* LF; this in turn serves as the meta-theory for the actual object language of interest — in our case Mizar's Tarski-Grothendieck set theory. Finally, individual Mizar articles are represented as (conservative) extensions of this theory. The respective meta-theory induces the pragmatic semantics of the object theories: In particular, the LF type system induces the definition of well-formedness and provability of Mizar formulae. For the details, we refer to the MMT fragment [RK11] of OMDoc for the general framework and to [IR11] for the formalization of Mizar in LF.

To achieve a pragmatic statement level, we make use of statement patterns, which were introduced recently in MMT by Fulya Horozal and the third author. A statement pattern introduces a new kind of statement together with concrete syntax for it. Moreover, it defines the semantics of these statements in terms of core OMDoc. For example, standard first-order logic is defined using three patterns for function symbols, predicate symbols, and axioms/theorems, respectively. A pattern can have free variables, e.g., for the arity of a function symbol; and specific instances of a pattern must provide substitutions for these free variables. To define Mizar, we need to add several sophisticated patterns, e.g., we need a single pattern

for case-based implicit function symbol definitions. This way, we are able to give a formalization of the syntax and semantics of both the object level and statement level of Mizar. [3] Effectively, we are able to recover a fragment of OMDoc that is isomorphic to Mizar.

2.2 Translating the Mizar Library to OMDoc

The representation of Mizar in OMDoc was the crucial prerequisite for translating the MML to OMDoc: It provides an ideal translation target because the non-trivial aspects of the translation can be delegated to the generic elaboration mechanisms implemented in the MMT toolkit.

Every Mizar article is translated to an OMDoc theory that includes the OMDoc theory for Tarski-Grothendieck set theory and thus those for Mizar and LF. The acyclic imports-relation between Mizar articles can be directly translated to the inclusion relation between OMDoc theories. Then for each statement within the article's body, we identify the respective statement pattern and generate the corresponding OMDoc instance. For a simple example, let us look at a theorem statement from the article `BOOLE`

```
theorem for X being set holds X \/ {} = X
```

Mizar exports it as the following constructor-level XML representation:

```
<Proposition line="34" col="5" plevel="" propnr="1">
  <For pid="0" vid="1">
    <Typ kind="M" nr="1" pid="1" aid="HIDDEN" absnr="1"><Cluster/><Cluster/></Typ>
    <Pred kind="R" nr="1" pid="9" aid="HIDDEN" absnr="1">
      <Func kind="K" nr="6" pid="6" aid="XBOOLE_0" absnr="2">
        <Var nr="1"/>
        <Func kind="K" nr="5" pid="5" aid="XBOOLE_0" absnr="1"/>
      </Func>
      <Var nr="1"/>
    </Pred>
  </For>
</Proposition>
```

We will first translate the asserted formula — an *object level* expression — to an OpenMath object.

All identifiers are translated to OpenMath symbols, which are identified by their theory and name. Note that Mizar exports the identifiers in the MML text above as pointers to arrays of constructors in Mizar articles. For instance, the identifier `set` is identified as the first mode definition in article `HIDDEN` by the `aid`, `kind` and `absnr` attributes in line 4. This representation is necessary to disambiguate the overloaded user-visible names. We reuse this naming scheme and translate `set` to the `<OMS module="HIDDEN" name="M1"/>`.

In general, we have to distinguish several cases. Logical symbols and constructs are implicit in Mizar and do not have an internal Mizar identifier. In OMDoc, those are declared in the respective meta-theory and translated accordingly. For example, the operator `for _ being _ st _` is translated to `<OMS module="mizar" name="`*`for`*`"/>`. Non-logical symbols that are fixed in Mizar's set theory such as `set` are translated to symbols declared in the theories `HIDDEN` and `TARSKI`. Finally, every article can refer to all locally declared or included identifiers. For example, the

[3] The corresponding pragmatic theory level is ongoing work, but not needed for Mizar.

above theorem refers in line 5 to the 5th functor declared in the article XBOOLE_0, i.e., {}. It is translated to the symbol <OMS module="XBOOLE_0" name="K5"/>. This process works because we translate all articles to OMDoc theories in dependency order.

Mizar expression forming constructs like the Pred element are translated into OpenMath applications using the OMA element: Its first child is the applied constructor (a function, predicate, mode, attribute, structure, aggregator, or selector identifier) and the remaining children are the arguments. Similarly, Mizar quantifiers like the For element are translated into OpenMath bindings using the OMBIND element and LF's higher-order abstract syntax. As we see, the translation of objects only poses little problems given the XML representation. To understand the translation of statement level constructs, let us continue with the theorem above.

In core OMDoc, it is translated to a symbol using the Curry-Howard representation of formulae-as-types and proofs-as-terms. That means, we generate a symbol with name T1 in the theory XBOOLE_0 whose type is given by the asserted formula. Our translation currently does not cover proofs because Mizar can only export partial proof objects. If proof objects were exported, we would similarly translate a proof to an OpenMath object, using the proof constructors which are already part of the OMDoc theory mizar. This OpenMath object would then occur as the definiens of the symbol T1. In particular, because we are using the typing relation of LF, OMDoc can guarantee the correctness of the proof.

While formally adequate, the Curry-Howard representation is not appealing to many users. Therefore, we use pragmatic OMDoc instead: We declare a pattern for theorems in the OMDoc theory mizar of the form

$$\textbf{pattern } \texttt{theorem} \textbf{ for } f : \mathit{prop}, p : \mathit{proof}\ f \qquad \rightsquigarrow \qquad \mathit{thm} : f = p$$

Here we use an abbreviated OMDoc syntax where boldface keywords correspond to OMDoc's XML elements. theorem is the name of the pattern, and f and p are free variables of the pattern that act as placeholders. Then, if F is the asserted formula and P its proof, we generate in the theory XBOOLE_0 an instance of the form

$$\textbf{instance } \texttt{T1} \textbf{ of } \texttt{theorem} \textbf{ with } f = F, p = P,$$

which elaborates to the core OMDoc declaration

$$\mathit{T1}/\mathit{thm} : \mathit{proof}\ F = P.$$

Among the statement level constructs of Mizar, the translation of theorems is the simplest case. We need more involved patterns for the remaining statements of the Mizar language. In particular, these are the definitions (of functors, predicates, modes, attributes, or structures), schemes, and registrations, all of which come in multiple variants, e.g., case-based functor definition without default case. The complete list of over 30 patterns that are part of the Mizar representation in OMDoc can be found in [IKR11].

2.3 Implementation

The OMDoc theories defining the Mizar *language* have been written manually once and for all. The translation of the Mizar *library* must be automated to be scalable.

We have implemented the translation as a complex processing pipeline within the MMT tool.

1. Mizar is run over the whole library transforming `miz` input files to `xml`. This represents the state before the work presented here.
2. Our translator reads all `xml` files (in dependency order) and parses them into custom Scala classes that model the constructor level Mizar language.
3. The translator then translates each article into Scala classes for OMDoc. The latter are part of the MMT tool, which serializes them as `omdoc` files. At this point the Mizar-specific part of the translation is over.
4. The generic algorithms in the MMT tool elaborate the pragmatic OMDoc to core OMDoc.

All involved files are available at https://tntbase.mathweb.org/repos/oaff/mml/. The translator consists of 88 Scala classes and 47 Scala objects, which are available at https://svn.kwarc.info/repos/MMT/src/mmt-mizar. Running the steps 2-4 of the pipeline on all 1129 files of the MML takes 41:40 minutes using a Intel Core i5-2410M Processor and 4GB of RAM. A fine grained breakdown is given in the table below.

Format	Total file size (raw/zipped)	Generation time
Mizar source	77.5 MB / 13.5 MB	—
Mizar XML	8.6 GB / 425.0 MB	—
pragmatic OMDoc	894.8 MB / 22.7 MB	29:46 min
core OMDoc	1015.3 MB / 25.3 MB	11:54 min

3 Interoperability with OMDoc-based Applications

The Mizar to OMDoc translation interfaces the Mizar system and library to a variety of systems and tools developed directly for OMDoc or interfaced to it. In this section, we will exemplify this by indexing the translated library in the MathWebSearch engine, a web service capable of crawling, indexing and querying content-rich representation formats. Up to now, the MathWebSearch engine has mainly been used on semi-formal mathematical content to answer instance queries, which can help users find partially recalled formulae [KŞ06].

3.1 Applicable Theorem Search (ATS) via Unification Queries

For Mizar we can also index formula schemata: variables quantified by the universal quantifier $\forall$ and the dependent types constructor Π give rise to meta-variables in the index, given a suitable extension of the MathWebSearch crawler. These metavariables allow to pose unification queries to MathWebSearch and thus query for theorems applicable to a given situation. We fortify our intuition on this with an example: say we want to prove the reflexivity of ordinal division (every ordinal divides itself)

```
for k being Ordinal ex a being Ordinal st k = k *^ a
```

The first (straightforward) step in the proof[4] is to first fix `k` so that it becomes a (local) constant of type `Ordinal`, and the goal changes to:

```
ex a being Ordinal st k = k *^ a
```

which reduces the proof to finding the theorem `ORDINAL2:39`:[5]

```
for A being Ordinal holds ( 1 *^ A = A & A *^ 1 = A )
```

which directly concludes the proof after splitting the conjunction. In this simple example (chosen for the ease of exposition), it is reasonable to assume that a knowledgeable Mizar user knows where to find `ORDINAL2:39`, but this might not be the same for new users of the large Mizar library, or even experienced users who have not worked with ordinals in Mizar before. We all know situations, where some remote theorem would have saved us days or months of work. In our situation, the existential structure of the subgoal above can directly be used as a unification query `k =k *^ $a`, where `$a` is a query variable. Note that this query unifies with subformula `A *^ 1 = A` of `ORDINAL2:39`. To enhance coverage we have extended the crawler of MathWebSearch to include the symmetric version of all equalities, so that we find `$A= $A *^ 1` in the index. Indeed MathWebSearch reports a hit for our query with unifier $\sigma := \{\$A \mapsto k, \$a \mapsto 1\}$. Note that σ already gives most of the information needed to apply `ORDINAL2:39` in the proof.

3.2 Realizing an ATS Service for Mizar

To index MML in MathWebSearch, the MMT tool generates a MathWebSearch harvest — essentially the list of all OpenMath objects occurring in the OMDoc files together with the URIs of their occurrences. The harvests for the MML take up 154.2 MB (4.4 MB zipped), the substitution tree index uses 891 MB of RAM and can be generated in 5:40 min. After indexing the MathWebSearch service is ready to respond to search queries.

Even though the two extensions to the index creation were triggered by the Mizar ATS application, we have realized them more generally and integrated them into the MMT system: The properties of being universal quantifiers and symmetric relations can be formulated for logics in general, by specifying logic views (MMT morphisms between meta-theories) from a special theory of "universalness" and "symmetry" respectively. Given a suitable development of the logics, an MMT-aware MathWebSearch crawler can determine the universal quantifiers and symmetric logical constants and treat them specially in the way described above.

To make the ATS service usable for an average Mizar author, it needs to be embedded into a development environment. For that we are experimenting with the MizAR [US10] system, a frontend for verification, presentation, and automated reasoning services for Mizar developed by the last author. We have extended MizAR by a facility to generate MathWebSearch queries (in MML form; after all we do not want the Mizar user to have to learn to pose MathWebSearch queries in the native content MathML form). For that we employ the transformation described in

[4] Indeed the actual proof in the Mizar library proceeds like this: The definition of ordinal division is at http://mizar.uwb.edu.pl/version/7.12.01_4.166.1132/arytm_3.html#D3, click on the "reflexivity" keyword there and then on the "proof" keyword.

[5] http://mizar.uwb.edu.pl/version/7.12.01_4.166.1132/html/ordinal2.html#T39

Section 2.3 again: The query (we directly use existentially quantified variables as query variables for familiarity) is loaded into Mizar and exported as XML constructor form. Our translator transforms into OMDoc and content MathML, which is then passed on to the MathWebSearch web service via a HTTP POST request. MathWebSearch answers this request with a list of hits, where a hit consists of a URI reference which specifies the matching subterm together with a unifying substitution. Note that since the MathWebSearch index is fed by MizAR, the URIs are MizAR URIs[6], which can be used to cross-link results to the Mizar source. In contrast to index creation, efficiency of query translation is of essence, but our translation pipeline is fast enough in practice as queries are relatively small.

Figure 1 shows the result of the integration described above. Here, we are in the proof of reflexivity of ordinal division as described above. The built-in proof support by the Mizar engine has failed, and the MizAR system has called the MathWebSearch engine on the existential subgoal (which can be directly translated into a MathWebSearch query as described above). MathWebSearch returns the two hits as clickable MizAR links, the second[7] one allows to complete the proof directly.

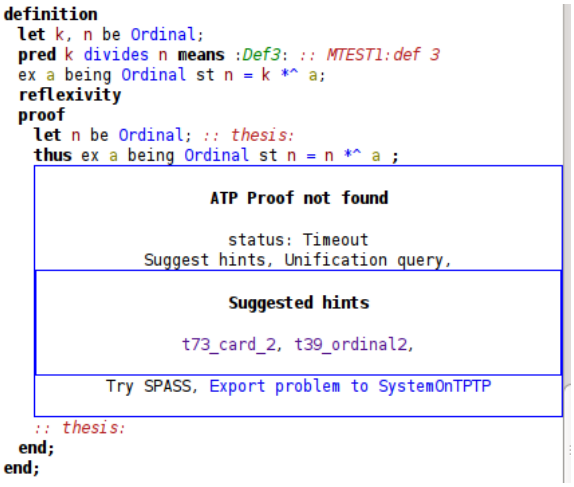

Fig. 1 A Mizar Proof Attempt with ATS Hints

4 Conclusion

In this article we have described a translation of the Mizar mathematical language to the OMDoc format and exhibited the interoperability afforded by interfacing Mizar to an OMDoc-based service: the applicable theorem search service. The only integration work necessary was on the MizAR side: the transformation had to be integrated into index creation and query translation, and a user interface for displaying results had to be developed. Note that even this simple integration yields a new service for the Mizar community: earlier information retrieval services like the MoMM [Urb06a] and MMLQ [Ban06b] systems provided subsumption-like and database-like queries different from what unification queries provide. We plan to investigate those in more details in the future in the MizAR setting, which we also plan to integrate within the new MizarWiki platform [UARG10].

[6] In fact, the URIs are not processed by MathWebSearch in any way, just reported back when they correspond to a hit.

[7] Note that result list ordering is almost completely unexplored territory for mathematical search; see [PK11] for a discussion.

Currently, the Mizar to OMDoc translation does not cover proofs in the Mizar library (arguably one of the most important parts), and we are planning to extend the coverage soon.[8] To the extent that Mizar can export proofs, this will be straightforward using our existing formalization of Mizar's inference rules in LF. The OMizar format currently only exists at the data structure level in the MMT system. We also want to give it an XML syntax, so that it can be used as a more accessible representation format for Mizar– and thus interoperability and/or storage format. We hope that this will also inform us in the endeavor of creating a truly universal pragmatic level of the OMDoc format.

Finally, the rudimentary ATS service we have presented in this paper serves as a valuable test corpus and evaluation target for the MathWebSearch engine. The ATS has already triggered extensions in the index creation as we have discussed above. In the future, we will tackle types and restrictions, an issue we have all but swept under the carpet in this paper that is of utmost practical importance as the experience with the Mizar library shows. On the one hand we want to extend the leftmost substitution trees used as an index structure in MathWebSearch to cover typed unification. On the other hand, we want to extend the index so that it can handle restrictions, i.e., if we have a theorem of the form $\forall xA \Rightarrow \forall yB \Rightarrow C$, the restrictions A and B should be passed along with the variables x and y, respectively. However, in contrast to type constraints, which can be resolved eagerly during typed unification, restrictions are residuated (i.e., accumulated in the result to be discharged by the user or later reasoning processes). In settings like the Mizar logic which has a complex, dependently typed type system, we conjecture that typed unification and residuation have to be mixed to be effective. Note that simple forms of typed unification are almost trivial to integrate into MathWebSearch, but we want to make the treatment of types and restrictions logic-independent. We conjecture, that this can be done in the same manner we have discussed for the universal quantifiers and symmetry of equality via logic views.

Acknowledgements We gratefully acknowledge the contribution of two colleagues in this article: *Fulya Horozal* for sharing her work on patterns and their elaboration at a very early stage, and *Corneliu Prodescu* for his help with and extensions of the MathWebSearch system.

References

ACD+10. Serge Autexier, Jacques Calmet, David Delahaye, Patrick D. F. Ion, Laurence Rideau, Renaud Rioboo, and Alan P. Sexton, editors. *Intelligent Computer Mathematics, 10th International Conference, AISC 2010, 17th Symposium, Calculemus 2010, and 9th International Conference, MKM 2010, Paris, France, July 5-10, 2010. Proceedings*, volume 6167 of *Lecture Notes in Computer Science*. Springer, 2010.

Ban03. Grzegorz Bancerek. On the structure of Mizar types. *Electronic Notes in Theoretical Computer Science*, 85(7), 2003.

Ban06a. Grzegorz Bancerek. Automatic translation if Formalized Mathematics. *Mechanized Mathematics and Its Applications*, 5(2):19–31, 2006.

[8] Note that handling of proofs has been typically delayed to the second phase in similar Mizar exporting projects like MPTP and MML Query, because a lot of useful functionality can be developed already without proofs.

Ban06b. Grzegorz Bancerek. Information retrieval and rendering with MML Query. In Jon Borwein and William M. Farmer, editors, *Mathematical Knowledge Management (MKM)*, number 4108 in LNAI, pages 266–279. Springer Verlag, 2006.

BK07. Grzegorz Bancerek and Michael Kohlhase. Towards a Mizar Mathematical Library in OMDoc format. In R. Matuszewski and A. Zalewska, editors, *From Insight to Proof: Festschrift in Honour of Andrzej Trybulec*, volume 10:23 of *Studies in Logic, Grammar and Rhetoric*, pages 265–275. University of Białystok, 2007.

DW97. Ingo Dahn and Christoph Wernhard. First order proof problems extracted from an article in the mizar mathematical library. In Ulrich Furbach and Maria Paola Bonacina, editors, *Proceedings of the International Workshop on First order Theorem Proving*, number 97-50 in RISC-Linz Report Series, pages 58–62. Johannes Kepler Universität Linz, 1997.

HHP93. R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the Association for Computing Machinery*, 40(1):143–184, 1993.

IKR11. Mihnea Iancu, Michael Kohlhase, and Florian Rabe. Translating the Mizar Mathematical Library into OMDoc format. KWARC report, Jacobs University Bremen, 2011.

IR11. M. Iancu and F. Rabe. Formalizing Foundations of Mathematics. *Mathematical Structures in Computer Science*, 21(4):883–911, 2011.

JFM. Journal of formalized mathematics.

Koh06. Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, August 2006.

KRZ10. Michael Kohlhase, Florian Rabe, and Vyacheslav Zholudev. Towards MKM in the large: Modular representation and scalable software architecture. In Serge Autexier, Jacques Calmet, David Delahaye, Patrick D. F. Ion, Laurence Rideau, Renaud Rioboo, and Alan P. Sexton, editors, *Intelligent Computer Mathematics*, number 6167 in LNAI. Springer Verlag, 2010.

KŞ06. Michael Kohlhase and Ioan Şucan. A search engine for mathematical formulae. In Tetsuo Ida, Jacques Calmet, and Dongming Wang, editors, *Proceedings of Artificial Intelligence and Symbolic Computation, AISC'2006*, number 4120 in LNAI, pages 241–253. Springer Verlag, 2006.

PK11. Corneliu C. Prodescu and Michael Kohlhase. Mathwebsearch 0.5 - open formula search engine. In *Wissens- und Erfahrungsmanagement LWA (Lernen, Wissensentdeckung und Adaptivität) Conference Proceedings*, sep 2011.

RK11. Florian Rabe and Michael Kohlhase. A scalable module system. Manuscript, submitted to Information & Computation, 2011.

TB85. A. Trybulec and H. Blair. Computer Assisted Reasoning with MIZAR. In A. Joshi, editor, *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 26–28, 1985.

TR99. Andrej Trybulec and Piotr Rudnicki. On equivalents of well-foundedness. *Journal of Automated Reasoning*, 23(3-4):197–234, 1999.

Try90. Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.

UARG10. Josef Urban, Jesse Alama, Piotr Rudnicki, and Herman Geuvers. A wiki for mizar: Motivation, considerations, and initial prototype. In Autexier et al. [ACD+10], pages 455–469.

Urb06a. Josef Urban. Momm - fast interreduction and retrieval in large libraries of formalized mathematics. *International Journal on Artificial Intelligence Tools*, 15(1):109–130, 2006.

Urb06b. Josef Urban. Mptp 0.2: Design, implementation, and initial experiments. *J. Autom. Reasoning*, 37(1-2):21–43, 2006.

Urb06c. Josef Urban. XML-izing Mizar: making semantic processing and presentation of MML easy. In Michael Kohlhase, editor, *Mathematical Knowledge Management, MKM'05*, number 3863 in LNAI, pages 346 – 360. Springer Verlag, 2006.

US08. Josef Urban and Geoff Sutcliffe. Atp-based cross-verification of mizar proofs: Method, systems, and first experiments. *Mathematics in Computer Science*, 2(2):231–251, 2008.

US10. Josef Urban and Geoff Sutcliffe. Automated reasoning and presentation support for formalizing mathematics in mizar. In Autexier et al. [ACD+10], pages 132–146.

Wie99. Freek Wiedijk. Mizar: An impression, 1999.