A Web Platform for Hosting the Mizar Mathematical Library

Daichi Yamamichi[1], Seigo Shigenaka[1], Kazuhisa Nakasho[1] and Katsumi Wasaki[2]

[1]Yamaguchi University, 2-16-1 Tokiwa-dai, Ube, Yamaguchi, Japan
[2]Shinshu University, 4-6-17 Wakasato, Nagano, Japan

Abstract

In this paper, we report on developing a web platform for hosting the Mizar Mathematical Library (MML). As the size of formalized mathematical libraries has been drastically increasing in recent years, there has been a growing demand for tools that support efficient and comprehensive browsing and searching of those libraries. A Wiki function for adding comments to the HTMLized MML and three types of search functions (article name, symbol name, and theorem) are implemented in this platform. It is also designed to maintain consistency during library updates by using Git as its backend. This platform aims at long-term use, with the highest priority on usability, extendability, and interoperability. We are planning to add features such as a dependency graph component in the future.

Keywords

Mathematical knowledge management, Mizar Mathematical Library, QED manifesto, Web service

1. Introduction

Research on proof assistant systems has made remarkable progress in recent years. However, the goals of the QED manifesto [1] have not been achieved yet. F. Wiedijk (2007) [2] lists the challenges facing the QED project as "*only very few people are working on formalization of mathematics*" and "*formalized mathematics does not resemble real mathematics at all.*" To increase the participation of mathematicians and accelerate library development, we need to improve the browsability and searchability of libraries and formal languages themselves.

The MathWiki Project[1] aims to improve the readability of formalized mathematical libraries and make them accessible to wider communities. A Wiki for Mizar [3], Large Formal Wikis for Coq/CoRN [4], and Agora System for Flyspeck Project [5] were proposed in this project. However, since the MathWiki Project was terminated in 2014, its contents cannot follow the libraries' updates. Many other projects build Wikis for mathematical libraries like MathWiki. ProofWiki [6], the Stacks Project[2], and The Lean Mathematical Library [7] accumulate mathematical libraries and convert them into highly readable HTML documents. However, these systems do not collaborate with advanced search engines or graphical tools for visualizing library dependencies.

FMM 2021: Fifth Workshop on Formal Mathematics for Mathematicians, July 30–31, 2021, Timisoara, Romania
nakasho@yamaguchi-u.ac.jp (K. Nakasho)
0000-0003-1110-4342 (K. Nakasho)

[1]https://www.nwo.nl/en/projects/612066825
[2]https://stacks.math.columbia.edu/

We have developed the emwiki system, a Wiki service hosting the Mizar Mathematics Library (MML) [8] while coping with the existing systems' challenges. The emwiki system is a web platform based on the Django framework and is featured by its sophisticated usability, extendability, and interoperability. Currently, this service is deployed on a server at Shinshu University and is available at https://em1.cs.shinshu-u.ac.jp/emwiki/release/.

2. Wiki Function

The Wiki feature is implemented to embed additional comments for reading and understanding articles in the MML. Although users cannot edit mathematical statements written in the formal language itself, they can add comments to theorems and definitions. We reused the HTMLized MML [9] to improve the convenience of the MML. The HTMLized MML is a document in which the MML is converted to HTML format, and reference relationships are expressed as hyperlinks. Besides, the HTMLized MML has highlight and proof folding functions. M. Iancu et al.(2013) [10] developed a conversion program from the MML to OMDoc format [11]. OMDoc is a markup language for mathematical documents that can be used to write down semantics and presentation. However, OMDoc is less presentation-oriented and requires a more rigorous description than TeX. Therefore, we adopted TeX syntax for mathematical expressions in comments, which is familiar to mathematicians. MathJax [12] is used as a mathematical expression rendering engine. We also implemented a real-time preview feature to check rendering results while editing.

The version tracking feature manages the history of comments. This function is achieved by using Git, a distributed version control system, as its backend. Moreover, it is necessary to maintain the linkage between theorems/definitions and comments during library updates. Our system exploits the Git merge function for this purpose. As theorems/definitions do not have persistent identifiers in the Mizar language, the most effective way to identify theorems/definitions before and after a library update is to track text differences. Our system embeds comments into the MML directly. Since the comments are written just before the theorems/definitions, the administrators can maintain the consistency of the linkage between the theorems/definitions and comments using the 3-way merge function during library updates.

The user management function allows the administrators to track and block specific users. The Wiki function stores a history of editors and their revisions, and when a comment is accidentally rewritten, it is possible to contact the user who wrote it and roll back it. Also, if a malicious comment is found, it will be deleted, and the user who wrote it will be blocked. The user management function is available in any component on our platform.

The screenshot of the Wiki feature is shown in Fig. 1.

3. Search Function

The emwiki system has three types of search components. The two simple search components accept article and symbol names as input. As of 2018, the MML contains 1,290 articles and 8,852 symbols [8], and being able to search these articles and symbols efficiently is critical for users. The excessive use of symbol overloading in the MML makes it difficult to understand the library. MML Reference [13] generates HTML documents from the MML to help users understand its

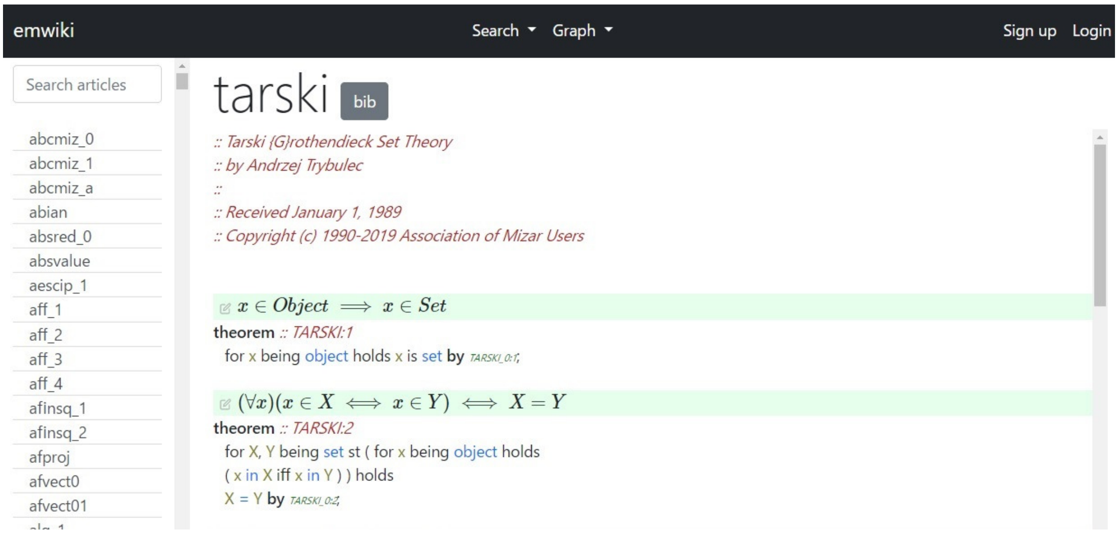

Figure 1: emwiki

symbols. Each HTML page contains symbol definitions, referrers, and references. It also has an incremental search function for symbol names, enabling users to search with a fast and intuitive operation. We integrated MML Reference into emwiki and linked this search function with the Wiki function.

We also developed a flexible theorem search component. For a long time, grep has been used for full-text search in the MML [14]. MML Query [15], developed by G. Bancerek et al. in 2001, has dramatically improved the efficiency of MML search. However, MML Query is not easy for beginners because it has its own query language. MML Query is a pattern-matching-based search system. Therefore it is not good at searching equivalently modified theorems. The Alcor system [16], developed by P. Carins et al., provides an LSI-based search function for the Mizar Mathematical Library. We have implemented a similar LSI-based search component into the emwiki system. The search component does not require its own query language, and the user can perform searches by inputting the desired theorems in the Mizar language.

4. Visualization of Library Dependencies

The dependency graph component is designed to encourage library maintainers to refactor their libraries by visualizing the file dependencies in the MML. The MML has been maintained for the past 30 years by the University of Bialystok [17] and is continuously refactored using specialized tools to minimize article dependencies and remove cyclic dependencies among groups [18]. However, some of the Mizar language specifications hinder the refactoring of the MML [19]. The Mizar language does not have namespace or package features. Also, the length of the Mizar file name is limited to 8 characters due to the legacy of MS-DOS constraints. Therefore, all of the 1,290 articles with ambiguous file names are currently arranged in a flat structure in the MML. Furthermore, the Mizar language has a constructor overloading feature and a context-sensitive grammar in which the last imported constructor takes precedence. These language specifications should be improved in the future, but we must continue to maintain our vast library of assets in the meantime.

In mathematical knowledge management (MKM), several types of research were conducted on analyzing and visualizing the dependencies of libraries. J. Alama (2011) [20] created a database of reference relations among theorems, definitions, and notations in the MML and provided a way to access the dependency graph through a web interface. J. Heras et al. (2014) [21] provides a tool to visualize the dependency graph of the HoTT library written in Coq. J. Alama et al. (2012) [22] extracted the dependencies in the Coq and Mizar libraries and performed a quantitative comparative analysis of the library features. R. Marcus et al. (2020) [23] proposed the TGView3D system, which renders the dependencies of formalized mathematical libraries as 3D graphs in a hybrid of force-directed and hierarchical layouts.

In this study, we adopt hierarchical and clustering graphs. The hierarchical graph is intended to clarify the flow of dependencies and utilize refactoring, such as minimizing dependencies and eliminating circular references among groups. The clustering graph aims to place semantically close articles nearby and provides helpful information for grouping articles. Although our dependency graph generator for the Mizar articles and online dependency graph created by the generator is not implemented into emwiki yet, they are available at GitHub[3] and GitHub Pages[4], respectively.

In the Mizar language, reference relationships between articles are described in the environment part at the beginning of an article. The environment part is categorized into ten items according to the type of components being imported, and the merged set of articles belonging to nine items, excluding the vocabulary part, was adopted as the directed edges of our dependency graphs. The reason for excluding the vocabulary part is that it plays a role in determining token units in lexical analysis but does not necessarily represent the relationship between articles [24]. Since Mizar does not recursively load external files written in environment sections, a transitive reduction is performed to cut redundant edges before constructing the dependency graph. Many extra dependencies will remain and deteriorate the graph's visibility if this procedure is omitted. The data structure is saved in dot and sfdp formats of Graphviz and is drawn in a web browser using Cytoscape.js library. The component also provides functions such as highlighting nearby nodes, searching nodes, moving nodes, hyperlinking to articles, and zooming in/out. The detailed usage of the dependency graph component can be found in the help menu in the upper right corner.

5. Conclusions and Future Work

In this paper, we have developed the emwiki system as a web platform for hosting the MML. This system is differentiated from existing services hosting mathematical libraries in usability, extendability, and interoperability. One of the most severe difficulties in the long-term use of a service hosting formalized mathematical libraries is to keep up with library updates. In

[3]https://github.com/mimosa-project/emgraph
[4]https://azsgws.github.io/demo-emgraph/

this study, we attempted to solve this problem using Git and its merge function to maintain reference relationships during version updates. Since this platform is built using Django, a new component development is stylized as adding Django applications. Besides, all components are allowed to access its version-controlled library, user management function, and database. The components implemented on the platform share web pages and data with each other. For example, MML Reference, now integrated into the system, shares a Wiki function and web pages.

For the dependency graph component, several issues remain. While the hierarchical graph could properly place article dependencies, the clustering graph has not been able to classify articles into meaningful groups. Mizar articles have metadata such as the Mathematics Subject Classification (MSC)[5]. We are considering taking this information into account for graph layout and including it as additional information for article search. We also think it will be effective to analyze the vocabulary contained in the articles and reflect the semantic classification in the layout algorithm. Another beneficial enhancement is drawing theorem graphs. However, to handle as many nodes as the theorems in the MML requires a more powerful graph database and drawing library than the conservative library we currently use.

This project aims to improve the browsability, searchability, and comprehensive understandability of formalized mathematical libraries. Although the system currently supports only Mizar, it could be extended to other languages. It is also significant to research and develop other graphical components to improve the browsing and searching formalized mathematical libraries.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP20K19863.

References

[1] K. Gödel, The QED manifesto, Lecture Notes in Artificial Intelligence 814 (1994) 238–251.
[2] F. Wiedijk, The QED manifesto revisited, Studies in Logic, Grammar and Rhetoric 10 (2007) 121–133.
[3] J. Urban, J. Alama, P. Rudnicki, H. Geuvers, A wiki for Mizar: Motivation, considerations, and initial prototype, in: International Conference on Intelligent Computer Mathematics, Springer, 2010, pp. 455–469.
[4] J. Alama, K. Brink, L. Mamane, J. Urban, Large formal wikis: Issues and solutions, in: International Conference on Intelligent Computer Mathematics, Springer, 2011, pp. 133–148.
[5] C. Tankink, C. Kaliszyk, J. Urban, H. Geuvers, Formal mathematics on display: A wiki for Flyspeck, in: International Conference on Intelligent Computer Mathematics, Springer, 2013, pp. 152–167.
[6] J. Vyskocil, J. Urban, Disambiguating ProofWiki into Mizar: First steps, AITP 2018 (2018).

[5]https://msc2020.org/

[7] F. van Doorn, G. Ebner, R. Y. Lewis, Maintaining a library of formal mathematics, in: International Conference on Intelligent Computer Mathematics, Springer, 2020, pp. 251–267.
[8] G. Bancerek, C. Byliński, A. Grabowski, A. Korniłowicz, R. Matuszewski, A. Naumowicz, K. Pak, The role of the Mizar Mathematical Library for interactive proof development in Mizar, Journal of Automated Reasoning 61 (2018) 9–32.
[9] J. Urban, P. Rudnicki, G. Sutcliffe, ATP and presentation service for Mizar formalizations, Journal of Automated Reasoning 50 (2013) 229–241.
[10] M. Iancu, M. Kohlhase, F. Rabe, J. Urban, The Mizar Mathematical Library in OMDoc: translation and applications, Journal of Automated Reasoning 50 (2013) 191–202.
[11] M. Kohlhase, OMDoc–An Open Markup Format for Mathematical Documents [version 1.2]: Foreword by Alan Bundy, volume 4180, Springer, 2006.
[12] D. Cervone, MathJax: a platform for mathematics on the Web, Notices of the AMS 59 (2012) 312–316.
[13] K. Nakasho, Y. Shidama, Documentation generator focusing on symbols for the HTML-ized Mizar library, in: International Conference on Intelligent Computer Mathematics, Springer, 2015, pp. 343–347.
[14] R. Matuszewski, P. Rudnicki, Mizar: the first 30 years, Mechanized mathematics and its applications 4 (2005) 3–24.
[15] G. Bancerek, P. Rudnicki, Information retrieval in MML, in: International Conference on Mathematical Knowledge Management, Springer, 2003, pp. 119–132.
[16] P. Cairns, J. Gow, Integrating searching and authoring in mizar, Journal of Automated Reasoning 39 (2007) 141–160.
[17] A. Grabowski, A. Korniłowicz, A. Naumowicz, Four decades of Mizar, Journal of Automated Reasoning 55 (2015) 191–198.
[18] A. Naumowicz, Tools for MML environment analysis, in: International Conference on Intelligent Computer Mathematics, Springer, 2015, pp. 348–352.
[19] P. Rudnicki, A. Trybulec, On the integrity of a repository of formalized mathematics, in: International Conference on Mathematical Knowledge Management, Springer, 2003, pp. 162–174.
[20] J. Alama, Mizar-items: exploring fine-grained dependencies in the Mizar mathematical library, in: International Conference on Intelligent Computer Mathematics, Springer, 2011, pp. 276–277.
[21] J. Heras, E. Komendantskaya, HoTT formalisation in Coq: Dependency graphs & ML4PG, arXiv preprint arXiv:1403.2531 (2014).
[22] J. Alama, L. Mamane, J. Urban, Dependencies in formal mathematics: Applications and extraction for Coq and Mizar, in: International Conference on Intelligent Computer Mathematics, Springer, 2012, pp. 1–16.
[23] R. Marcus, M. Kohlhase, F. Rabe, Tgview3d: A system for 3-dimensional visualization of theory graphs, in: International Conference on Intelligent Computer Mathematics, Springer, 2020, pp. 290–296.
[24] P. Rudnicki, A. Trybulec, Mathematical knowledge management in Mizar, in: Proc. of MKM 2001, Citeseer, 2001.