Exploiting Semantic Annotations in Math Information Retrieval

Petra Sojka, Martin Liška, Michal Růžička, David Fornánek
NLP Centre, Faculty of Informatics, Masaryk University, Botanicalka 86a, 602 00 Brno, Czech Republic
sojka@fi.muni.cz

Abstract

The design and architecture of MiS (Math Indexer and Searcher), a system for mathematics retrieval is presented, and design decisions are discussed. We argue for an approach based on combining Presentation Content and MathML, using a similarity of math substrates, semantic annotations by Mathematical Subject Classification code extensions, statistical semantics keywords generated by topic modeling (LDA), and math corpus preprocessing to disambiguate the content and find the system collocations. The whole is summarized implemented as a math-aware search engine based on the MDX system and the state of the art system Apache Lucene. Scalability issues were checked against more than 400,000 arXiv documents with 158 million mathematical formulae. Almost three billion MathML substrates were indexed using a Software-competent Lucene.

1. Introduction and Motivation

The solution to the problem of relevant, easy, and precise mathematical formulae retrieval lies at the heart of building digital mathematical libraries (DML). There have been numerous attempts to solve this problem, but none have found widespread adoption and satisfaction within the wider mathematics community. And as yet, there is no widely accepted agreement on the math search format.

Tokenization

Tokenization is a straightforward process of obtaining substrates from an input formula. MiS makes use of Presentation MathML markup where all logical units are parsed in XML tags which makes obtaining all substrates a question of tree traversal. The inner representation of each formula is an XML node encapsulating all the member child nodes. This means the highest level formula—as it appears in the input document—is represented by a node named “math”. All logical subparts of an input formula are obtained and passed on to modification algorithms.

Formulae Modifications

MiS performs three forms of modification algorithms, the goal of which is to create several more or less generalized representations of all formulae obtained through the tokenization process. These steps allow the system to return similar matches to the user query while preserving the formula structure and equality.

Ordering

Let us take a simple example: $a + 1$ and the query $a + 1$. This would not match even though it is perfectly equal. This is why a simple ordering of the operands the commutative operations, addition and multiplication, is used. It tries to order arguments of these operations in the alphabetical order of the XML nodes denoting the operands whenever possible—It considers the priority of other relevant operators in the formula. The system applies this function to the formula being indexed as well as to the query expression. Applied to the example above, the XML node denoting variable $a$ is named “nr”, the node denoting number 1 is named “int”, “nr” and “int” therefore $1$ would be exchanged for $a$ and $1$ would match.

Unification of Variables and Constants

Let us take another example: $a \cdot b$ and $x \cdot y$. Again, these would not match even though the difference is only in the variables used. MiS employs a process that unifies variables in expressions while taking bound variables into account. All variables are substituted with a unified symbol and the search process is carried out. Applied to the example, both expressions would unify to $u_1 \cdot u_2$, and would match.

Ordering

Let us take a simple example: $x + y$ and $y + x$. This would lead to the indexing of millions of docs and searching for any symbol would end up matching all of the documents containing it.

4. Evaluation and Implementation

For large scale evaluation, we needed an experimental implementation and a corpus of mathematical tests. The Math Indexer and Searcher is written in Java. The role of the tokenizing and searching core is performed by Apache Lucene 3.1.0. The mathematical part of document processing can be seen as a standalone plugin extension to any full text library, however it would need custom integration for each one. In the case of Apache, a custom Tokenizer (MathTokenizer) has been implemented. For the textual content of documents, Lucene’s StandardAnalyzer is employed. In MathTokenizer, TermAttributes are used for carrying string of math expressions and PresentationAttribute for storing weights of formulae. Lucerne’s practical scoring function for every hit document if the query with each query term is as follows:

$$\text{score}(d, q) = \sum \text{weights}(d) \cdot \text{weights}(q)$$

Corpus of Mathematical Documents MiREC

A document corpus MiREC (version 2013.1.324) was used to evaluate the behaviour of the system we created. The documents come from the arXiv.org project they have been extracted from arXiv into HTML+MathML (both Content and Presentation) [11].

We were able to gather a great amount of documents in MiREC corpus versus 44,794 from the MiS test corpus. The math indexing engine was performed by Apache Lucene documents containing 158,106,118 mathematical formulae. 3,210,314.146 expressions were indexed and the resulting size of the index is 47 GB. Sizes of uncompressed and compressed files are 124 GB and 5 GB respectively. MiREC corpora are available to the community for download from MiREC website [http://http://mi.mii.st/web/mirec/miirec].

We have to come to the conclusion that the unification of variables interfere less with original formula meaning than the unification of numbers constants. For this reason,Their variable should be higher...e., less discriminative.

Scalability Testing and Efficiency

We have devised a scalability test to see how the system behaves with an increasing number of documents and formulae indexed. Subsets containing 10,000, 50, 500, 2000 and the complete 324,300 documents were gradually indexed and several values were measured: the running time of the index preprocessing and the average query time. Indexing time of this corpus was 1378.82 min, giving the average query time. Indexing time of this corpus was 1378.82 min, giving the average query time.

Results

MiS demonstrated the ability to index and search a relatively vast corpus of real scientific documents. Its usability is highly elevated thanks to its preprocessing functions together with formula weighting model. The ability to search for exact and similar formulae and substrates, more so with customizable relevancy computation, demonstrates an unquestionable contribution to the whole search experience.

We have created a demo web interface WebMiS which is publicly available on the MiS web page [http://http://http://mi.mii.st/web/mirec/miirec/]. Our WebMiS interface supports queries in two different notations—in Presentation and MathML. Mathematical queries are alternatively canonicalized using canonicalizer program developed primarily for semantic information retrieval process through topic to assistant search of math queries. Properly to query...university’s results retrieval. Portability of the interface is increased by using MathJax for rendering of mathematical formulae in snippets.

Scalability

We have devised a scalability test to see how the system behaves with an increasing number of documents and formulae indexed. Subsets containing 10,000, 50, 500, 2000 and the complete 324,300 documents were gradually indexed and several values were measured: the running time of the index preprocessing and the average query time. Indexing time of this corpus was 1378.82 min, giving the average query time.

Table 2: Scalability test results (run on 32 GB RAM, quad core AMD Opteron™ Processor 8500 driven machine).

<table>
<thead>
<tr>
<th>Documents</th>
<th>Indexing Time (min)</th>
<th>Query Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10,000</td>
<td>1378.82</td>
<td>0.00</td>
</tr>
<tr>
<td>50,000</td>
<td>1378.82</td>
<td>0.00</td>
</tr>
<tr>
<td>500,000</td>
<td>1378.82</td>
<td>0.00</td>
</tr>
<tr>
<td>2,000,000</td>
<td>1378.82</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Everything you can imagine is real. (Pablo Picasso)

References

Natural Language Processing techniques as used in corpus management systems are also useful for document analysis (classifications, named entity recognition, word sketches). They are able to reach web scalability and avoid inference problems. The main ideas are to 1) augment surface texts (including math formulae) with additional linked representations bearing semantic information (expanded forms of the text, canonical text and subformulae) for indexing, including support for indexing structural information (expressed as Content MathML, or other tree structures) and 2) use semantic user preferences to order found documents.

Semantic Math-aware search is a gateway to the vast treasure of knowledge in Digital (Mathematical) Libraries (DML) as EUDML. There are two main types of searches.

The goal of a navigational (exploratory) search is to locate documents or web pages related to the user's intention usually expressed as a sparse set of key-words. For example, in EUDML one might type a theorem (e.g. Pythagoras) and want to refer to all theorems related to that topic. Both types of searches benefit from proper handling of semantics—that is, the meanings of words and their relations to each other depend on possibly multiple domains of interest defined by user e.g. as a Mathematical Subject Classification (MSC) 2010.


We have also described several semantic annotations and enhancements that use full MathML, in addition to standard ‘bag-of-word’ indexing, as a Lucene plug-in component. They represent the structure of more general (sub)formulae employed in MathML.

Knowledge Graph is also indexed. Word terms of formulae expanded as vocalized for reading (e.g. using the Google Text-to-Speech API) and increased time-consuming tasks.


3. Conclusions, Credits

We have presented an approach to mathematics searching and indexing—the architecture and design of the MathML system. The feasibility of our approach has been verified on large corpora of real mathematical papers from arXiv.org. Scalability tests have confirmed that the computing power needed for real math similarity computations is readily available; this would allow the use of this technology for projects on a European or world-wide scale.

References


We have realized that the key to quality retrieval is to normalize mathematical expressions by converting them into the canonical representation. We have not found any tools to fit this goal, so we are implementing a new, modular one, in Java (using Sun’s Math Toolkit). The main design implementation is simple, extremely extensible and flexible.

Normalization consists of a dozen of canonicalization modules, both for Presentation and Content MathML. We preferably use Content MathML over Presentation MathML, and eventually will convert even Presentation MathML to it in the future. The semantic enhancements thus have to be computed, disambiguated and indexed in advance, on the fly. We are introducing our indexing terms with several types of semantic annotations—topical, canonical formulae terms and interlingual terms for multilingual retrieval.

Figures 4: Web interface of MIaS for The European Digital Mathematics Library

Acknowledgements. Our work has been partially supported by the European Community’s Joint Programming Initiatives on e-Science (Polly Policy Supporter, “Open access to scientific information”, Grant Agreement No. 250553).