MathWebSearch at NTCIR-11

Radu Hambasan
Jacobs University Bremen
r.hambasan@jacobs-university.de

Michael Kohlhase
Jacobs University Bremen
m.kohlhase@jacobs-university.de

Corneliu Prodescu
Jacobs University Bremen
c.prodescu@jacobs-university.de

ABSTRACT

We present and analyze the results of the MATHWEBSEARCH (MWS) system in the Math-2 task in the NTCIR-11 Information Retrieval challenge.

MWS is a content-based full-text search engine that focuses on low-latency query answering for interactive applications. It combines a powerful exact formula unification/matching with the full-text search capabilities of ElasticSearch to achieve simultaneous full-text search for mathematical/technical documents.

MWS 1.0 focuses on scalability (memory footprint, index persistence), integration of keyword- and formula search, and hit presentation issues. It forms a stable basis for future research into extended query languages and user-interaction issues. The system has been integrated into high-profile information systems like Zentralblatt Math.

In this paper we describe MWS 1.0, evaluate the submission and results for the NTCIR-11 Math-2 Task and conclude with future work suggested by the task results.

Team Name

KWARC

Subtasks

Math Retrieval (English)

Keywords

NTCIR, text and formula search, MathML

1. INTRODUCTION

Mathematics-aware information retrieval (MIR) has often been touted as one of the "killer applications" of computer support in STEM (Science, Technology, Engineering, and Mathematics). Indeed, the yearly production of published research in mathematics exceeds 120 thousand journal articles a year alone; the amount of internal technical documents in company archives certainly exceeds this quantity by far; being able to access the knowledge locked up in them will become a determining factor for commercial success.

To create a community and data set for MIR research, development, and evaluation, the well-established NTCIR (NII Test Collection for IR Systems) Project introduced a pilot task for MIR at NTCIR-10 in 2013 [AKO13]. The NTCIR-11 Math2 task [NTM] builds on that pilot and provides a data set of $100,000$ papers in XHTML+MathML translated by LaTeXML [LTX] from the Cornell ePrint arXiv [Sta+10] and a set of 50 queries. Results are submitted as lists of up to 1000 ranked hits (paragraphs) for each of the queries. These are judged against correct hits curated by mathematicians using the TReC evaluator [Tre].

MWS is a web application that provides low-latency answers to full-text queries which consist of keywords and formulae. MWS front-ends convert formula schemata (with query variables) into content MathML expressions [Aus+10], which the MWS formula indexer answers by unification and combines the results with keyword results from a text search engine. The modular architecture and standardized formats makes MWS applicable to a wide range of querying tasks – web-based formula search engines or editing support services. Unification queries over content MathML expressions form the basis of an expressive query language with well-defined semantics. As substitution instances of the original query, MWS results are highly significant, if the encoding of data set and search query are adequate – i.e. do not drop or spuriously introduce salient semantic features.

In the next section, we will describe the MWS 1.0 system, detailing the improvements over the NTCIR-10 version described in [KP13]. Section 3 presents and evaluates the results obtained by MWS in the the NTCIR-11 math task. Section 4 concludes the paper with tabulation of future work planned as an answer to the lessons learnt from participating in the challenge.

Acknowledgments

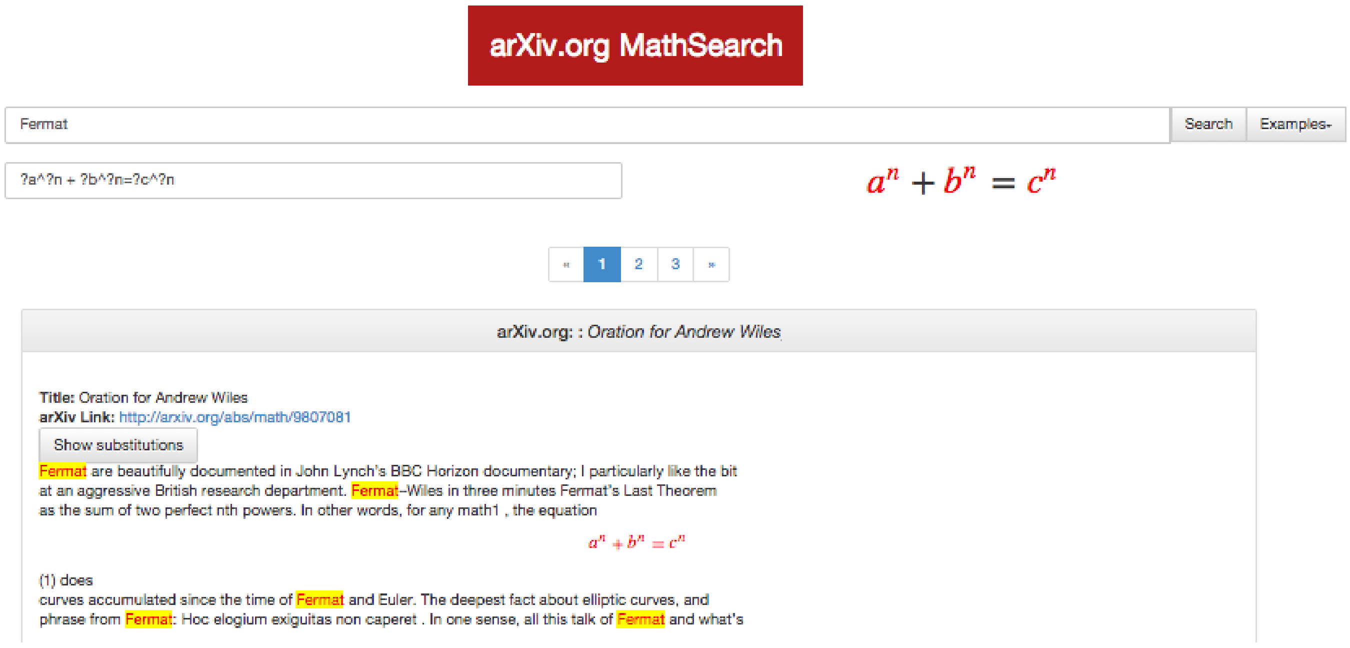
The work reported here here has been partially supported by the Leibniz association under grant SAW-2012-FIZ_KA-2. We gratefully acknowledge support on the web front-end by Tom Wiesing and discussions by Mihnea Iancu.

2. THE MATHWEBSEARCH SYSTEM

MWS is a complete system for crawling, indexing and searching documents that contain formulae and text. The special treatment of the formula modality constitutes a major step towards information retrieval for STEM documents.

MWS comprises of a custom math search engine, which uses compressed formula representation (using substitutions) to build an in-memory index and a text engine system based on Apache Solr - ElasticSearch [Eso]. Documents are simultaneously indexed in both systems and a coordination query proxy is used. Users can search these documents using a web interface, as shown in Figure 1. We will now describe each of the system components:

1. the MWS Crawler – see Section 2.1
2. the MWS Formula Indexer – see Section 2.2
3. ElasticSearch– see Section 2.3
4. the MWS Query Proxy – see Section 2.4
5. the MWS User Interface – see Section 2.5



arXiv.org: : *Oration for Andrew Wiles*

Title: Oration for Andrew Wiles
arXiv Link: http://arxiv.org/abs/math/9807081

Fermat are beautifully documented in John Lynch's BBC Horizon documentary; I particularly like the bit at an aggressive British research department. Fermat–Wiles in three minutes Fermat's Last Theorem as the sum of two perfect nth powers. In other words, for any math1 , the equation

$$a^n + b^n = c^n$$

(1) does
curves accumulated since the time of Fermat and Euler. The deepest fact about elliptic curves, and phrase from Fermat: Hoc elogium exiguitas non caperet . In one sense, all this talk of Fermat and what's

Figure 1: MWS ArXiv frontend

Note that we will describe MWS in enough detail to make this work self contained. For further information, see [KMP12; Pro14], and to learn more about other applications, see [MWSb].

2.1 The MWS Crawler

The MWS crawler parses HTML and XHTML documents and generates XML files we call **harvests**. Harvests collect the searchable information from documents. For instance, a document starting with the passage

> Under the constraint $x \to 0$, the formula $f(x) + 2$, ...

is harvested as the XML file in Listing 1. This contains formulae encoded as Content MathML (in the expr elements), plain text versions of each document (data/text), the original math data (data/math*), as well as document metadata (data/metadata).

Data elements and Content MathML expressions are cross-linked through data identifiers (data_id). Formulae can be located in their respective documents using identifiers (in the url attribute of expr elements). As such, Content MathML with identifier 1 is represented in plain text as math1 and its full math data is located in the math element with local_id 1.

Listing 1: Example MWS Harvest

```
<harvest>
  <data data_id="0">
    <id>ntcir−11−math2−wiki/00065.html</id>
    <metadata>
      <title>Quantum Interpretation of ...</title>
      <author>John Doe</author>
      <zbl_id>0343.2322</zbl_id>
    </metadata>
    <text>Under the constraint math1, the formula ...</text>
    <math local_id="1"> ... </math>
    ...
  </data>
  <expr data_id="0" url="1">
    <m:apply>
      <m:ci>&#x2192;</m:ci>
      <m:ci>x</m:ci>
      <m:cn>0</m:cn>
    </m:apply>
  </expr>
  <expr data_id="0" url="2">...</expr>
  ...
</harvest>
```

The harvest format standardizes search data and thus abstracts from the format of the source document. Together with a front-end adequate to the domain, a special harvester is the only component needed to adapt MWS to a new domain. Besides the (X)HTML crawler described here, we have implemented crawlers for Excel documents [KPL13] and formal mathematical documents [Ian+13] like the Mizar Mathematical Library.

2.2 The MWS Formula Indexer

MWS indexes Content MathML formulae, which describe the structure of mathematical expressions using XML trees in a harvest.

Each expression is encoded as a set of substitutions based on a depth-first traversal of its Content MathML tree. For example, $f(g(a,a),b)$, which is represented as shown in Listing 2, would be encoded as:

$$\begin{aligned}
@0 &\to @1(@2,@3) && \text{// <m:apply> of arity 3}\\
&\to f(@2,@3) && \text{// <m:ci>f</m:ci>}\\
&\to f(@4(@5,@6),@3) && \text{// <m:apply> of arity 3}\\
&\to f(g(@5,@6),@3) && \text{// <m:ci>g</m:ci>}\\
&\to f(g(a,@6),@3) && \text{// <m:ci>a</m:ci>}\\
&\to f(g(a,a),@3) && \text{// <m:ci>a</m:ci>}\\
&\to f(g(a,a),b) && \text{// <m:ci>b</m:ci>}
\end{aligned}$$

At each step, exactly one variable is substituted and this path of substitutions describes the initial formula. The MWS index is a space-optimized in-memory trie of substitution paths and each leaf has an associated identifier. As such, this identifier uniquely represents the respective formula.

Listing 2: ContentMathML for $f(g(a,a),b)$

```
<m:apply>
  <m:ci>f</m:ci>
  <m:apply>
    <m:ci>g</m:ci>
    <m:ci>a</m:ci>
    <m:ci>a</m:ci>
  </m:apply>
  <m:ci>b</m:ci>
</m:apply>
```

The index can be queried through a RESTful API. Queries use a Content MathML formula augmented with query variables (which can match any subterms), while responses contain the formula identifiers which match the respective formula.

For the NTCIR-11 version of the MWS formula indexer, we improved space efficiency by 81% and added persistency – the index survives process and machine restarts. Further details on these developments, as well as an evaluation of these are provided in [Pro14].

2.3 ElasticSearch

ElasticSearch [Eso] is an open-source full text search engine optimized for scalability and availability. It offers partitioning of data in shards, distribution across multiple nodes, replication and fault tolerance. Its core text search engine provides full-text queries with ranking, with the ability to define simple schemas and search using exact matches, wildcards, and range queries, among many others. Indexing and querying in ElasticSearch are done using a RESTful HTTP API which accepts JSON. It supports typed data (text, integers, floats) and key-value pairs in JSON are used to support semi-structured indexing.

To integrate MWS formula information into ElasticSearch, we developed a tool which takes MWS harvests and a MWS index and generates fixed schema documents. The schema contains the original harvest information (plain text, formulae and metadata), along with formula annotations. Each formula is annotated with its respective MWS formula identifier and each document is annotated with the formula identifiers it contains. Note that, a formula is annotated with multiple identifiers, as each subexpression has a different identifier in MWS. This allows us to perform simultaneous text and formula search:

2.4 Integration via the Query Proxy

The query proxy is a HTTP RESTful interface which provides text and formula search. It is the operational heart of the full-text search functionality that is new for the NTCIR-11 version of MWS. As we show in figure 2, its backend is a MWS formula indexer serving the formula index and an ElasticSearch daemon serving the annotated documents.

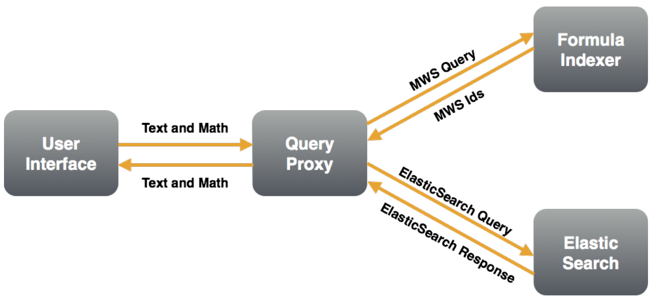


Figure 2: MWS System Architecture

Upon receiving a query, it sends the formula to the formula indexer service and receives the matching formula identifiers. Using these, along with the text query, it queries ElasticSearch for documents which contain the respective text and the formula identifiers. As these documents may be large, only their unique identifiers are retrieved and a secondary query is used to provide snippets with the relevant data.

Internally, we can configure what is considered a hit by manipulating the query we present to ElasticSearch. It understands an expressive language, allowing us to define a match as any document containing the formula, and (or) at least n or a percentage of words. In the current configuration, a document is considered a hit if it contains a formula matching the formula query and at least one of the words in the text query.

2.5 Improved User Interface

We use a separate web user interface that supports users in composing queries and presents search results. Figure 1 shows a results page for a combined formula+keywords query. The formula query can be entered in LaTeX – extended with the ? operator that marks up query variables that can be instantiated by arbitrary formulae in the search. Formula queries are sent to the LaTeXML conversion service [GSK11] which transforms it to Content and Presentation MathML. The content MathML is sent to the query proxy along with the keywords from the text field, while the presentation MathML is used to render an interactive preview[1] that allows the user to verify that the entered LaTeX conforms to her search intent – query variables are rendered in red for clarity. For browser which do not support MathML rendering, we use MathJAX, making the MathML-based user interface usable on all browsers.

The query proxy returns document snippets and metadata (title, authors, original ArXiv link), where occurrences of the query terms are highlighted. Note that snippetizing (X)HTML documents is a non-trivial task due to the presence of XML tags and MathML expressions. Text highlighting is done through XML tagging, while math highlighting is done by passing additional XML formula identifiers and XPaths [XPa10] (when subexpressions need to be highlighted). We have developed a general highlighting library for this purpose, which was also utilized in the Math-2 evaluation interface. Finally, we developed an interface that allows users to explore the substitutions – i.e. which parts of the result term are instances of which query variables – but marking up corresponding parts by corresponding colors turned out unintuitive; more research is needed on this front.

To engage new users, we added an examples section next to the search button. This provides a menu with a few examples which auto-fill text and formula queries and initiate the search.

3. RESULTS & EVALUATION

MWS participated in the NTCIR-11 Math-2 challenge. We submitted a run containing a total of 5498 document hits to 49 out of the 50 text and formula queries.

3.1 System Setup and Instrumentation

The data-set provided by NTCIR consisted of 8.3 million paragraphs of 105,539 (X)HTML documents from the arXMLiv corpus, in total 176Gbs. We started by crawling these, generating 224 Gbs of harvest data in 16 hours. The larger size is expected, since the harvests contain all the document data, along with annotations which comply to a specific format. Next, we indexed

[1]As the LaTeXML conversion has a turnaround time of ca 50 milliseconds, the formula preview directly reacts to user input.

these in the math search engine in another 20 hours. This generated an index of 15.9Gb, containing $583,718,824$ formulae, out of which $63,402,844$ unique formulae. Next, we generated annotated documents and indexed these into ElasticSearch for a total size of 51.84Gb. In this step document math elements were annotated with identifiers from the formula index. This final step took 9 hours, bringing the total indexing time to 45 hours, but once the index has been created, it can be read from memory in under 90 seconds. This is the standard setup for indexing an (X)HTML corpus into MWS.

For the NTCIR-11 challenge we created an custom script which processes the query (MathML formulae and text keywords) file. For each query containing n formulae and m keywords, we sent n queries to MWS, since our system only allows one math formula to be queried, along with arbitrary many words. Additionally, we sent another query without a formula, these results having a reduced score. Special attention was required for some of the queries which had invalid Content MathML, exhibited in the form of m:cerror elements (queries 3, 23, 39, 40, 41). Finally, our script aggregated these results and saved under the specified TSV format [Koh14c].

3.2 Results Analysis

Using the setup described above, we obtained results for 49 out of the 50 queries. On average, we obtained 112.20 hits per query. This includes high precision results (matching formula and text) and low precision results (matching only text). We obtained high precision results for only 26 queries, with an average of 32.15 high precision results per query. Since the challenge result format required exactly 1000 hits, we used randomly sampled documents to make our submission compliant.

Looking at the NTCIR evaluation sheet shown in table 1, we can see our system is precise for first page results (top 5 hits). As such, 50% of our top5 hits were considered relevant, while 79% were partially relevant. We notice excellent top5 precision when formulae have query more than two query variables (queries 6, 7, 9, 17, 19, 21, 24, 26, 34, 37, 38, 44, 49). These results found many matching formulae, and used text search only to further rank already meaningful results.

Analyzing the mean average precisions, we immediately notice that our system is better at ranking relevant results (29%), compared to partially relevant ones (25%). Generally, we notice lower mean average precisions for relevant hits when our system did not find any formulae and had to rely solely on text (queries 2, 3, 11, 21, 33, 35). However, when the text query is highly descriptive (queries 13, 14, 18, 24, 26, 29), even without matching the formula we obtain fully relevant, high-precision hits.

We did not manage to find any solutions for query 33. We see two reasons for this. Firstly, our formula search engine is rigid, looking only for exact matches of the entire formula. Secondly, only one keyword was provided (in two different spellings) and it was a common one. The same applies to query 11, which explains why we found only partially relevant hits.

3.3 Discussion

We saw that augmenting exact formula search with text search is beneficial. Using keywords along with formulae provides some flexibility, the system being able to work around ill formed queries, which leads to higher recall. However, if the keywords are not specific enough, precision suffers. Furthermore, text search is a good disambiguation tool for humans, especially in the case of simple formulae where relevance cannot be provided by formula alone.

Considering these factors, the performance profile shown by these results meets our expectations of the MWS system: high precision provided by exact formula search combined with recall based on text search.

4. CONCLUSION & FUTURE WORK

We have presented the MATHWEBSEARCH text and formula search system, reported on the results of participating in the NTCIR-11 Math task and evaluated these results.

The MWS web service is open source software released under the GNU public license. The code is available from the developer portal [MWSa]. Search front-ends for various corpora and applications are referenced on the MATHWEBSEARCH project page [MWSb].

The performance profile shown by the NTCIR Math task has confirmed the design decisions made in the the MWS system (precision, well-defined query semantics). But it has also revealed the limitations of exactly these design decisions (inflexibility) in search practice. An instance of MWS 1.0 is running in production mode as the formula search engine for Zentralblatt Math [ZBM]; another at the MathHub portal for flexiformal Math [MH]; see [IKP14] for details.

The main improvement of MWS1.0 over the the NTCIR-10 version described in [KP13] are the integration of keyword search, scalability, and index persistence, which required considerable re-engineering of core system components. With these enhancements, we view the core of the MWS1.0 system as a stable base line for further development in interface and application areas.

As the scalability work had priority – and the NTCIR challenges do not address them – we did not prioritize the four main applicability bottlenecks identified in [KP13]. We will now discuss them in the light of the extended experience of the last 18 months.

Expressive query languages.

We want the user to be able to describe the document fragments he/she is looking for. Extensions of unification search queries can be given in the form of *restrictions of the answer substitutions*: We can restrict the set of result instances by decorating query variables with annotations. For instance the query[2]

?f:[type=$\mathbb{R} \to \mathbb{R}$]=?G:[weight<=10]!

attributes a type constraint to the query variable f that only allows function terms as answers, and a weight constraint (only 10 constants) that limits the size of the right hand side of the result equations. Somewhat surprisingly, decorated query variables can be used for generalizations of queries as well, e.g. for specifying *ranges of (similar) numbers*: For instance, a query of the form

?x:[weight=1]^n:[st=prime(n)]+?y:[weight=1]^?n =
?z:[weight=1]^?n

could be used to describe equations similar to the the Fermat equation $x^2 + y^2 = z^2$, but restricted to prime exponents.

We made first progress in this direction by implementing type restrictions in the MWS indexer [Has14], but – as so often in MIR – the greater problem is to extract type constraints for the formulae in the index. This is essentially a semantics extraction task, which is currently largely unsolved.

Similarity Search & Scoring.

Given a sufficiently expressive vocabulary, the query languages described above even subsume similarity search as we can describe the set of similar formulae declaratively via a schema with suitably

[2]Here and below we use a glossed query syntax used in the front-end (see Section 2.5) instead of extended MathML for readability.

Task	R MAvg	PR MAvg	Top5 R	Top5 PR
Math-1	0.47	0.26	0.80	1.00
Math-2	0.00	0.05	0.00	0.20
Math-3	0.00	0.33	0.00	0.80
Math-4	0.50	0.14	0.20	0.40
Math-5	0.06	0.23	0.00	1.00
Math-6	0.33	0.22	1.00	1.00
Math-7	0.20	0.32	0.20	0.80
Math-8	0.54	0.48	0.60	1.00
Math-9	0.00	0.39	0.00	0.80
Math-10	0.14	0.17	0.20	0.80
Math-11	0.00	0.17	0.00	1.00
Math-12	0.30	0.20	0.60	0.80
Math-13	0.59	0.33	1.00	1.00
Math-14	0.42	0.43	0.60	1.00
Math-15	0.36	0.28	0.80	0.80
Math-16	0.40	0.24	0.80	0.80
Math-17	0.47	0.40	1.00	1.00
Math-18	0.18	0.16	1.00	1.00
Math-19	0.19	0.14	0.80	0.80
Math-20	0.59	0.53	1.00	1.00
Math-21	0.39	0.35	0.60	1.00
Math-22	0.26	0.16	1.00	1.00
Math-23	0.15	0.14	0.60	1.00
Math-24	0.27	0.31	0.60	1.00
Math-25	0.45	0.47	0.40	0.80
Math-26	0.55	0.38	0.80	1.00
Math-27	0.38	0.31	0.40	0.80
Math-28	0.21	0.16	1.00	1.00
Math-29	0.15	0.25	0.40	1.00
Math-30	0.13	0.12	0.80	0.80
Math-31	0.55	0.53	0.60	1.00
Math-32	1.00	0.38	0.40	1.00
Math-33	0.00	0.00	0.00	0.00
Math-34	0.26	0.17	0.40	0.40
Math-35	0.06	0.08	0.00	0.00
Math-36	0.50	0.34	1.00	1.00
Math-37	0.48	0.26	0.80	0.80
Math-38	0.02	0.05	0.20	0.40
Math-39	0.11	0.05	0.40	0.40
Math-40	0.44	0.39	0.40	1.00
Math-41	0.05	0.32	0.00	1.00
Math-42	0.19	0.08	0.20	0.40
Math-43	0.00	0.06	0.00	0.40
Math-44	0.24	0.13	0.60	0.60
Math-45	0.69	0.41	1.00	1.00
Math-46	0.46	0.45	0.60	0.80
Math-47	0.34	0.25	0.60	0.80
Math-48	0.11	0.12	0.20	0.80
Math-49	0.00	0.11	0.00	0.40
Math-50	0.11	0.25	0.40	1.00
Average	**0.29**	**0.25**	**0.50**	**0.79**

Table 1: NTCIR Results for MathWebSearch: Mean average and Top 5 precision, as judged for relevant and partially relevant hits

restricted query variables. The only missing piece to a flexible similarity search regime with well-defined semantics is an equally flexible and declarative system of scoring. We conjecture that query variable restriction vocabularies for similarity search can be expressed in terms of metric relations so that we can give the intended score of the query by an arithmetic expression. For instance a query of the form

```
∀ ?x.?f(?x)=?y |> ed(?f,"sin")/(1+#occ(?x,?y))
```

uses the edit distance ed and the number of occurrences #occ (both are metrics) to compute the score on the left of the operator |>. This particular query looks for universally quantified equations whose left hand side contains a function that is similar to sin and whose right hand side may mention the quantified variables, but multiplicity is penalized.

The upshot of this treatment of similarity search is that instead of letting the MIR system decide on the notion of similarity, we believe that the user should be able to; in particular if we allow suitable front-ends to generate queries and hide query language complexity from the end user.

A low-hanging fruit in this domain would be to extend the indexer to deal with "ranges" of numbers. For instance, we could query for ?t:[range:37−38]^\circ C} to search for temperatures between $37^\circ C$ and $38^\circ C$. In contrast to the case of typed search, this does not need additional semantics extraction, only that numbers are recognized as such during harvests. This approach could be extended to other "literals", e.g. to token elements in , where we could query for presentations from a give list.

Novel approaches to Ranking.

Currently, MWS only has a very simple-minded notion of search results ranking (preferring formula hits over keyword hits, ordering formula hits by the size of substitutions, and then falling back on the ranking provided by Elastic Search). This has worked satisfactorily in the absence of other ranking criteria. Ranking by a scoring regime as practiced by other MIR systems in the NTCIR may be a partial solution, but cannot distinguish top-scored results – of which there can be many in the presence of query variables. In the Zentralblatt Math instance of MWS, we simply use the publication date or ordering, which is what users are accustomed to in this context.

Before we can make real progress here, we need to do user studies to find out what criteria mathematical practitioners have for preferring hits. In collaboration with Zentralblatt Math, we ran a series of repertory grid interviews (see e.g. [CICM1414; Koh14a; Koh14b]) on mathematicians to elicit preconceptions and cognitive constructs of mathematicians.

A way to support user-determined ranking intuitions (and to delegate the problem to the user) would be to allow uses to specify a declarative ranking formula as part of the query, log these, and mine them for innovative ideas.

References

[AKO13] Akiko Aizawa, Michael Kohlhase, and Iadh Ounis. "NTCIR-10 Math Pilot Task Overview". In: *NTCIR Workshop 10 Meeting*. Tokyo, Japan, 2013, pp. 1–8. URL: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings10/pdf/NTCIR/OVERVIEW/01-NTCIR10-OV-MATH-AizawaA.pdf.

[Aus+10] Ron Ausbrooks et al. *Mathematical Markup Language (MathML) Version 3.0.* W3C Recommendation. World Wide Web Consortium (W3C), 2010. URL: http://www.w3.org/TR/MathML3.

[CICM1414] Andrea Kohlhase. "Search Interfaces for Mathematicians". In: *Intelligent Computer Mathematics 2014*. Ed. by Stephan Watt et al. Lecture Notes in Computer Science. Springer, 2014, pp. 153–168. URL: http://arxiv.org/abs/1405.3758.

[Eso] *Elastic Search*. Feb. 20, 2014. URL: http://www.elasticsearch.org/ (visited on 02/20/2014).

[GSK11] Deyan Ginev, Heinrich Stamerjohanns, and Michael Kohlhase. "The LaTeXML Daemon: Editable Math on the Collaborative Web". In: *Intelligent Computer Mathematics*. Ed. by James Davenport et al. LNAI 6824. Springer Verlag, 2011, pp. 292–294. ISBN: 978-3-642-22672-4. URL: https://svn.kwarc.info/repos/arXMLiv/doc/cicm-systems11/paper.pdf.

[Has14] Daniel Hasegan. "Sorted Unification in MathWebSearch". Bachelor's Thesis. Computer Science, Jacobs University, Bremen, 2014.

[HY14] Hoon Hong and Chee Yap, eds. *Mathematical Software - ICMS 2014 - 4th International Congress*. Vol. 8592. Lecture Notes in Computer Science. Springer, 2014. ISBN: 978-3-662-44198-5. DOI: 10.1007/978-3-662-44199-2.

[Ian+13] Mihnea Iancu et al. "The Mizar Mathematical Library in OMDoc: Translation and Applications". In: *Journal of Automated Reasoning* 50.2 (2013), pp. 191–202. DOI: 10.1007/s10817-012-9271-4.

[IKP14] Mihnea Iancu, Michael Kohlhase, and Corneliu-Claudiu Prodescu. "Representing, Archiving, and Searching the Space of Mathematical Knowledge". In: *Mathematical Software - ICMS 2014 - 4th International Congress*. Ed. by Hoon Hong and Chee Yap. Vol. 8592. Lecture Notes in Computer Science. Springer, 2014, pp. 26–30. ISBN: 978-3-662-44198-5. DOI: 10.1007/978-3-662-44199-2_5.

[KMP12] Michael Kohlhase, Bogdan A. Matican, and Corneliu C. Prodescu. "MathWebSearch 0.5 – Scaling an Open Formula Search Engine". In: *Intelligent Computer Mathematics*. Ed. by Johan Jeuring et al. LNAI 7362. Berlin and Heidelberg: Springer Verlag, 2012, pp. 342–357. ISBN: 978-3-642-31373-8. URL: http://kwarc.info/kohlhase/papers/aisc12-mws.pdf.

[Koh14a] Andrea Kohlhase. "Design of Search Interfaces for Mathematicians". In: *MathUI, OpenMath, PLMMS, and ThEdu Workshops and Work in Progress at the Conference on Intelligent Computer Mathematics*. Ed. by Matthew England et al. CEUR Workshop Proceedings 1180. Aachen, 2014. URL: http://ceur-ws.org/Vol-1186/paper-02.pdf.

[Koh14b] Andrea Kohlhase. "Math Web Search Interfaces and the Generation Gap of Mathematicians". In: *Mathematical Software - ICMS 2014 - 4th International Congress*. Ed. by Hoon Hong and Chee Yap. Vol. 8592. Lecture Notes in Computer Science. Springer, 2014, pp. 586–593. ISBN: 978-3-662-44198-5. DOI: 10.1007/978-3-662-44199-2_88.

[Koh14c] Michael Kohlhase. *Formats for Topics and Submissions for the Math2 Task at NTCIR-11*. Tech. rep. NTCIR, 2014. URL: http://ntcir-math.nii.ac.jp/wp-content/blogs.dir/13/files/2014/05/NTCIR11-Math-topics.pdf.

[KP13] Michael Kohlhase and Corneliu Prodescu. "MathWebSearch at NTCIR-10". In: *NTCIR Workshop 10 Meeting*. Tokyo, Japan, 2013, pp. 675–679. URL: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings10/pdf/NTCIR/MATH/04-NTCIR10-MATH-KohlhaseM.pdf.

[KPL13] Michael Kohlhase, Corneliu Prodescu, and Christian Liguda. "XLSearch: A Search Engine for Spreadsheets". In: *Symp. of the European Spreadsheet Risks Interest Group (EuSpRIG 2013)*. 2013. URL: http://kwarc.info/kohlhase/papers/eusprig13-xlsearch.pdf.

[LTX] Bruce Miller. *LaTeXML: A LaTeX to XML Converter*. URL: http://dlmf.nist.gov/LaTeXML/ (visited on 03/12/2013).

[MH] *MathHub.info: Active Mathematics*. URL: http://mathhub.info (visited on 01/28/2014).

[MWSa] *Math Web Search*. URL: https://trac.mathweb.org/MWS/ (visited on 01/08/2011).

[MWSb] *MathWebSearch – Searching Mathematics on the Web*. URL: http://search.mathweb.org (visited on 04/26/2013).

[Ntc] *NTCIR Workshop 10 Meeting*. Tokyo, Japan, 2013.

[NTM] *NTICR Pilot Task: Math Task*. URL: http://ntcir-math.nii.ac.jp/ (visited on 11/11/2012).

[Pro14] Corneliu C. Prodescu. "Text and Formula Search on ArXiv Documents". M. Sc. Thesis. Jacobs University Bremen, 2014.

[Sta+10] Heinrich Stamerjohanns et al. "Transforming large collections of scientific publications to XML". In: *Mathematics in Computer Science* 3.3 (2010): *Special Issue on Authoring, Digitalization and Management of Mathematical Knowledge*. Ed. by Serge Autexier, Petr Sojka, and Masakazu Suzuki, pp. 299–307. URL: http://kwarc.info/kohlhase/papers/mcs10.pdf.

[Tre] *Text Retrieval Conference Homepage*. Aug. 22, 2014. URL: http://trec.nist.gov/ (visited on 08/22/2014).

[XPa10] *XPath Reference*. 2010. URL: http://www.w3.org/TR/xpath/ (visited on 06/05/2010).

[ZBM] *Zentralblatt MATH*. URL: http://www.zentralblatt-math.org/zbmath/ (visited on 06/12/2012).