A Survey on Retrieval of Mathematical Knowledge

Ferruccio Guidi and Claudio Sacerdoti Coen(✉)

Department of Computer Science and Engineering – DISI,
University of Bologna, Bologna, Italy
{ferruccio.guidi,claudio.sacerdoticoen}@unibo.it

Abstract. We present a short survey of the literature on indexing and retrieval of mathematical knowledge, with pointers to 72 papers and tentative taxonomies of both retrieval problems and recurring techniques.

1 Purpose Driven Taxonomy of Retrieval Problems

Retrieval of mathematical knowledge is always presented as the low hanging fruit of Mathematical Knowledge Management, and it has been addressed in several papers by people coming either from the formal methods or from the information retrieval community. The problem being resistant to classical content search techniques [LRG13], it is usually addressed combining a small set of new ideas and techniques that are recurrent in the literature. Despite the amount of work, however, there is not a single solution that is the clearly winning on the others, nor convincing unbiased benchmarks to compare solutions. Some authors like [KK07] also suggest that the community should first better understand the actual needs of mathematicians from an unbiased perspective to improve the MKM technology as a whole. In this paper we collect a hopefully comprehensive bibliography, and we roughly classify the papers according to novel taxonomies both for the problems and the techniques employed. The only other surveys on the same topic are [AZ04], now outdated and focused mostly on (European) research projects that contributed to the topic in the 6th Framework Programme, [ZB12], which covers less literature in much greater detail without attempting a classification, [L13], which is focused on evaluation of mathematics retrieval, and [L10], which is written in Slovak.

We begin our discussion with a purpose driven taxonomy made of three different retrieval problems that deal with mathematical knowledge. Each problem is characterised by its own set of expectations and constraints, and adopting a solution to another problem may be infeasible or yield poor results. In the next sections we classify the papers according to an encoding based taxonomy (presentation vs. content vs. semantics) and to a taxonomy of techniques employed. Finally we point to the rich literature relative to the problem of ranking, and we touch the problem of evaluation of systems. We conclude with some notes on the availability of math retrieval systems.

M. Kerber et al. (Eds.): CICM 2015, LNAI 9150, pp. 296–315, 2015.
DOI: 10.1007/978-3-319-20615-8_20

1.1 Problem 1: Document Retrieval

Objective: A *human* is interested in recalling a *set of mathematical documents* (or fragments) that are related to a particular mathematical topic. Typically it is not the case that only one document provides the correct answer; on the contrary the user may be interested in a corpora of different documents that yield different, only partially overlapping information. In [Koh14] and other papers there are attempts at a classification of the information needs of users. However, at the moment only the system described in [ZKT08] tries to use the classification to improve the user experience.

Input: The human composes a query combining keywords (e.g. for topics [ACK08]), free text and mathematical formulae. Often the mathematical formulae are intended as examples of expressions related to the topic of interest. For example, a user interested in trigonometric identities can just enter one identity to retrieve them all. Or a formula showing a particular property of a special function can be used to disambiguate the special function among the ones with similar names.

The query can be composed using a very simple, Google-inspired, single line interface, or written using an ad-hoc query language (see [AY07b, AY08b, YA07] for some proposals), or by filling in some form. The first solution is the one preferred in the literature. In [Koh14] a comparison of the behaviour of mathematicians vs. other users highlighted that the professional mathematician is more interested in the precision of the output than the effort put into the input. Therefore mathematicians may use and appreciate more complex interfaces. On the contrary, other users are likely to prefer a simple, modern search interface.

Formulae can be entered in some textual syntax (e.g. LaTeX, MathML), maybe with the help of on-the-fly formulae rendering [LSR14], or using graphical editors [MM06], or they can be acquired from hand-written snippets [AY08a]. The formula is likely to contain errors and ambiguities, for example if it is encoded at the presentational level (e.g. in Presentation MathML or LaTeX), if it is acquired from hand-written text, or if the user only remembers it partially or in a wrong way. Errors and ambiguities are not a critical problem because formulae are just used to retrieve documents that contain *similar* formulae according to some similarity criterion. In [AY08a] the authors address the problem of combining and ranking results from different queries generated from ambiguous formulae due to errors in the recognition process. See also [ZB12] for a survey on the interaction between mathematical information retrieval and mathematical document recognition. Some authors [KT09] suggest that the visual presentation of the formula may sometimes be important in the definition of similarity, whereas in other situations it is the mathematical *content* of the formula that matters. Logically equivalent formulae whose content encoding is highly different are better considered less similar.

Once the search engine returns the result, the user may be given the opportunity to enhance the query by further filtering.

Output: The output is a *ranked list* of matching documents or document fragments (e.g. a chapter of a book, or a section of an article). When the query involves mathematical formulae, the ranking is determined by the similarity relation. The user must be given the possibility to quickly determine whether the matched document is interesting or not. Therefore the problem of how to present summaries of the selected documents in the result list is of fundamental importance [LSLM11, LSR14, MG08b, WG10, You05, You06, You07, You08]. Even highlighting correctly the bits of the summary that matches the query can make a significant difference in the user experience [LSLM11, LSR14, You05, You06]. The list of results must be the starting point for further investigations by the user. At least, all results must contain hyperlinks or other ways to retrieve the original document the summary points to. A study of user requirements in [ZKT08] suggests that results should be presented after clustering them according to their resource type (research paper, tutorial, slides, course, book, etc.). For example, a student may immediately decide to skip research papers, and a researcher may skip websites and tutorials.

Constraints: A balance must be obtained between *precision* (the fraction of retrieved documents that are relevant) and *recall* (the fraction of relevant documents that are retrieved). To maximise recall, precision is affected and many out of topic documents (false positives) are retrieved, penalising performance. Too many results are overwhelming and the user is likely to give attention only to the first ones in the list. Therefore, the search engine does not have to rank and produce summaries of documents with low scores. The ranking function is ultimately the one responsible for the perceived quality of the search engine.

Since the query is intended to be issued by a human, the performance of the search engine is not a critical requirement and up to a few seconds (or even minutes in some particular situations) may be acceptable. Nevertheless, modern textual search engines like Google are extremely fast, and the user is likely to expect the queries to be solved in less than a second.

1.2 Problem 2: Formula Retrieval

Objective: A program — more rarely a human — is interested in retrieving *all* formulae that are in some relation $\mathcal{R}$ with a query formula E. Sometimes the formula E can actually be a set of formulae. For example, E can be a goal to be proved automatically, and $T\,\mathcal{R}\,E$ when T is the conclusion of the statement of a theorem that can be instantiated to prove E. More precisely, T contains metavariables to be instantiated and $\mathcal{R}$ is one-sided unification up to some equational theory. The dual query is also used in the literature: E is a property (a statement containing metavariables), $\mathcal{R}$ is unification and the query finds all operations that satisfy the property E. By using several properties at once, the query can find all models of a given theory (e.g. all semirings in the library) [NK07], also up to renaming of constants and properties. An interesting

application presented in [GK14], that uses techniques similar to [NK07], consists in matching concepts across libraries by first computing properties of an object in one library (i.e. patterns like commutativity of a binary operator) and then looking for objects in the other library that satisfy the same properties. To be more effective, properties are extracted from all libraries and concepts are matched according to a similarity measure to identify objects that satisfy a similar set of properties. A third example is obtained by choosing logical implication for $\mathcal{R}$. The query looks for all formulae that imply E.

Input: One or more formulae E that may or may not contain metavariables to be instantiated. Rarely, additional constraints can be expressed using keywords, classifications, free text, authors, etc. Formulae are not supposed to be ambiguous or contain errors. In [AGSC+06] ambiguity is resolved before performing the query using type checking and interaction with the user.

Some dedicated query languages are proposed to specify the structure of the formulae E [Ban06, BR03, BU04, GS03, KT10, Rab12]. They are implemented on top of relational databases or ad-hoc in-memory indexes.

Output: The query is meant to retrieve a set of formulae that satisfy a certain property. When the search is performed by a program, there is no need to present summaries of the document the formula occurs in. Even when a human issued the search, an hyperlink to the document may be sufficient.

In many situations the relation $\mathcal{R}$ can be extended to a ternary relation $T\,\mathcal{R}_\rho\,E$ meaning that T is related to E with score ρ, and the results can be ranked according to ρ. For example, if $\mathcal{R}$ reduces a proof of E to a proof of T, then the T's may receive a higher score if they are judged easier to prove.

Constraints: Maximisation of the recall is fundamental. The query should return all formulae that satisfy the query, even if they rank very low. Because searches are often basic operations of complex algorithms (e.g. automatic provers), speed is also critical. In several situations, the searches need to be performed in milliseconds.

To speed up the searches or when the relation $\mathcal{R}$ is undecidable, the search engine may use a second decidable relation $\mathcal{R}'$ such that $\mathcal{R}\subseteq\mathcal{R}'$. Using $\mathcal{R}'$, the query can return false positives, i.e. formulae T such that $T\,\mathcal{R}'\,E$ but not $T\,\mathcal{R}\,E$. For example, when E is a pattern and $\mathcal{R}$ is unification, $\mathcal{R}'$ may ignore the structure of the two formulae E and T and conclude $E\,\mathcal{R}'\,T$ when all symbols in E are also in T. Example: $f(xz, y+z)\,\mathcal{R}\,f(?, ?{+}?)$ and $f(y+z, xz)\,\mathcal{R}'\,f(?, ?{+}?)$, but not $f(y+z, xz)\,\mathcal{R}\,f(?, ?{+}?)$.

1.3 Problem 3: Document Synthesis

Objective: Composing a new mathematical document assembling fragments to be retrieved from a library. The most common occurrence is in educational software where a learning object must be assembled according to the expertise of

the user, the topic of interest, etc. [BF03,LDM+08,LM06]. An unrelated example is the automatic generation of summaries and statistics for a mathematical library [BR03]. A final example is mining of formalised libraries, for example to build visual representations of the graph of dependencies over an axiom (to understand its implications) or a definition/statement (to understand the propagation of changes).

A variant is to solve a mathematical problem by composing mathematical (Web) services [CDT04]. Each service exposes metadata about the problem solved, the algorithm implemented, and its preconditions and postconditions.

Input: The query is not likely to involve mathematical formulae, and it is usually expressed using a query language over ontologies. A high level interface may hide the underlying query language. Sometimes the query is fixed once and for all, and needs to be run at regular periods.

Output: The expected output depends on the particular use case and it is usually made of a single result in place of a ranked list. The result may consist of a graph of objects and relations between them, or it may consist of the minimal information to build the expected document or solve the algorithmic problem.

Constraints: The constraints depend on the particular use case.

After an initial screening of the literature, we decided to analyse only papers about the first two problems, where formulae play a central role. Indeed, at a first glance most solutions to Document Synthesis employ standard query languages for ontologies, and only the ontologies themselves are math-specific (e.g. [CDT04, LDM+08]). Logic programming languages are also employed to represent what the user knows/ignores and the inference rules to assemble documents [BF03].

Moreover, we did not find in the literature convincing examples for the need of very expressive query languages to solve the Document Retrieval and the Formula Retrieval problems, where the kinds of queries are essentially fixed a priori. Moreover, evaluation of queries expressed in these languages are reported to be too slow to be used for Formula Retrieval. Sometimes additional techniques are employed for Document Synthesis, like semantical query reduction to relax the user provided query by allowing additional topics close to the one specified by the user [Lib13]. These techniques too seem to be very general and applicable to domains very different from that of mathematics.

Despite the strong interest of the community in the use of formulae in queries, studies on the behaviour of users [KK07,ZKT08] conclude that the added value may be low, and the finding is confirmed in [LM06,Mil13] where the logs of the DLMF and of ActiveMath search engines are analysed concluding that only few queries contain mathematical formulae, most are very simple ones, and such queries do not yield satisfactory results.

2 Encoding Based Taxonomy

Mathematical information can be encoded in a library at three different levels. The most shallow one is *presentation*. Presentation markup uses a finitary language to express the bi-dimensional layout of a formula, useful to present it to the reader. The standard XML language for presentation is *Presentation MathML*, and several tools can generate Presentation MathML from LaTeX (e.g. LaTeXML, Tralics), PDF files (e.g. Maxtract), handwritten text (e.g. InfTy Reader), digitised documents (e.g. InfTy Reader) or content markup (e.g. via XSLT stylesheets). We can therefore assume that the totality of the documents to be indexed are available in Presentation MathML.

The next level is *content*. At the content level, the structure of the formula is described, and symbols and operators appearing in it are linked to their entry in an ontology, called *content dictionary* in OpenMath terminology. The markup language is finitary, but the ontology is not since new mathematical entities can always be defined. The relation between content and presentation is one-to-many: the same presentation markup may represent different content expressions (*ambiguity*), and a content expression can be given different presentations according to the conventions of the community of readers, the language of the reader, but also for purely aesthetic reasons like constraints on the size of the formula. OpenMath and Content MathML are the two standard XML language for formulae at the content level. OMDoc is an attempt at standardising at the content level whole mathematical documents, comprising proofs. Content markup is currently mostly used for the exchange of formulae between systems, in particular CAS. There are no significant examples of large libraries of documents natively written using content markup. Nevertheless, there are tools like SnuggleTeX based on heuristics to semantically enrich (annotated) LaTeX documents or even MathML Presentation documents to content.

The last level is *semantics* and it is specific to libraries of formalised mathematical knowledge. The semantics level refines the content level by picking for every content level object one particular definition in a given logic. The definition chosen embeds the object with additional properties, e.g. computational properties. For example, addition over natural numbers can be defined in the Calculus of Inductive Constructions as a non-computable ternary predicate in logic programming style, or as a recursive function on the first argument — such that $0 + x$ and x become logically indistinguishable — or as a recursive function on the second argument — so that $0 + x$ and x are not indistinguishable, but only provably equal. Interactive theorem provers often provides an XML dump of their internal semantics representation.

Formula Retrieval is always formulated either on semantics markup or on content markup. Even when the semantic markup is available, it may be convenient to convert the library to content level by identifying alternative definitions of the same mathematical notion. In this way, it becomes possible to retrieve useful theorems on mathematically equivalent definitions, in the hope to reuse them after conversion to the definitions in use. One application of this technique is reuse of libraries across different systems based on the same logic.

The Document Retrieval problem is formulated in a way that is agnostic of the encoding. However, the user is likely to enter formulae in the query using a presentation language (mostly LaTeX, even if MathML starts to be used [LSLM11,LSR14,MG08b]). Some authors have provided evidence that precision is improved when exploiting parallel markup, even when the content part is automatically generated from the presentation part [NKTA14]. See [MY08] for motivations against content/parallel markup and in favour of a more lightweight encoding of content information in Presentation MathML. Other authors claim that precision can be lost by embracing content because sometimes the actual layout used in a presentation or the name used for variables are significant. Reference [GPBB14] in retrospect also described the choice of using Content MathML as a bad decision. Other authors dismiss indexing of Content MathML because of the non-availability of libraries or because of conversion from presentation to content being approximative and unreliable. Finally, [NKTA14] reports that automatic conversion of large formulae from Presentation to Content may be computationally unfeasible, and propose to limit the conversion to small ones.

Recently, the debate on presentation only vs. parallel markup seems to be solved in favour of the latter. For example, the system that scored better at the last NTCIR task reports better scores when applied to content markup generated from LaTeX w.r.t. presentation only markup [RSL14]. The authors second this observation already in [LSR13].

Moreover, several works in the literature that deal with presentation markup enrich it — in the document itself or in the indexes — with additional annotations to make explicit additional semantics that is latent in the library [Cai04] or in the text surrounding the formulae [GPBB14,KTHA14]. For example, in [KTHA14] artificial intelligence is applied to the whole document to recover from the text surrounding the formulae the name associated to the mathematical entities in the formula (e.g. "posterior probability", "derivative of f"). Reference [GPBB14] uses a cheaper approach by considering only one sentence around a formula, but it later observes that one sentence is often not sufficient and many relevant results are therefore missed. Another example is an analysis of co-occurrence of symbols in the corpus to identify related ones. It is shown that these techniques are important to augment recall or, sometimes, precision. In our view, like the heuristic based presentation-to-content translation, these are *attempts to infer and store partial semantics of mathematical expressions*. It may be questioned (see for example [MY08]) if the current content markups (OpenMath and Content MathML) are the right instruments to augment presentation markup with partial, approximate semantics, and if such additional semantics makes only sense in the indexes of search systems, or it may be serialised to an XML format for being reused by third parties.

Systems based on Content MathML, parallel markup or semantics appear in [Ban06,BR03,BU04,GK14,HS13,HKP14,KP13,L13,LSLM11,LSR13,LSR14,MG08a,MG08b,MM06,NCH12,NK07,Rab12,SLM13,YA09,ZY14].

3 Taxonomy of Techniques for Mathematical Retrieval

Implementations of solutions to mathematical search problems can be obtained combining one of the main techniques that will be presented in Sect. 3.2 with a choice of modular enhancement techniques from Sect. 3.1 used to improve precision, recall or both.

3.1 Modular Enhancement Techniques

The following techniques are general enough to be applied to solve both the Document and the Formula Retrieval problems, and by analysing the literature it seems that every system eventually applies all of them.

Segmentation. A preliminary step to indexing is segmentation of documents into chunks. Chunks are the unit of information to be returned to the user, with pointers to the parent document. Segmentation is trivial on formal mathematical documents, hard on web-pages, and intermediate on other resources like books or papers. See, for example, [ZKT08] for a discussion. Several systems implement segmentation; however, the last NTCIR competition has provided a data set of already segmented documents [AKO14] and that may hinder the study of segmentation techniques in the future.

Normalisation. To improve recall, both formulae in the query and the formulae in the library are put in normal form before indexing them. Having the same normal form is an equivalence relation $\equiv$, and the query retrieves formulae up to $\equiv$. For Formula Retrieval it is necessary that $\equiv\mathcal{R}\equiv \subseteq \mathcal{R}$. For Document Retrieval the $\equiv$ relation must be compatible with the similarity and ranking functions. When this is not the case, precision can be critically lowered.

Uses of normalisation include: repairing of broken XML/MathML generated by automatic conversion tools [MM07] (e.g. when the structure imposed by $<$ `mrows` $>$ is not compatible with the mathematical structure); removal of information that does not contribute to the semantics like comments, layout elements (spaces, phantoms and linebreaks), XML/MathML attributes (color, font, elements in other namespaces) [FLRS12, HHN08, MM07]; picking canonical representations of the same presentation/content when different MathML encodings are possible (e.g. `msubsup` vs. `msup` and `msub`, `mfenced` vs. use of two parentheses, applications of trigonometric functions with/without using parentheses, etc.) [AY07a, FLRS12, MM07]; replacing names of bound variables with unique numerical indexes (e.g. De Brujin indexes) to search up to α-conversion [MM07, NK07]; ignoring parentheses and ordering of arguments of associative/commutative operators [AY07a, MG08a, MG08b, NK07, SY07, SL11, YS06]; expressing derived notions exposing the derivation (e.g. replacing $x \geq y$ with $y \leq x$, $x \not\leq y$ with $\neg(x \leq y)$, arcsin with $\sin^{-1}$, etc.) [AY07a, MM07]; capturing logical equivalence/type isomorphisms (e.g. writing formulae in prenex normal form, currification of functions) [Del00, GK14, NK07].

A normalised formula can be quite different from the original one, and that can be a symptom that the formula is not significant. Therefore in [RSL14]

normalised formulae are weighted according to their similarity to the initial one, and weights are considered during the ranking phase with great results.

Approximation. Normalisation does not lose information, converting a document to an equivalent one. Many papers call "normalisation" an approximation phase where subformulae are replaced with constrained placeholders to allow the formula to be matched by similarity. For example: names of variables or constants can be replaced by a single name [GWHT14]; all numeric constants by a single identifier [GWHT14,MM07,SL11]; subformulae may be replaced by their type. For example, in [HKP14] type information is used to retrieve formulae by sorted unification, i.e. by constraints with type placeholders in patterns. Approximation improves recall. To limit the loss of precision, systems that approximate index both the original and the approximated formula (or even several instances at different levels of approximation). The effects of approximation are similar to those of query reduction, but approximation is more efficient because it works at indexing time.

Enrichment. Enrichment works on the library or on the query to augment the information stored/looked for in the index by inferring new knowledge from existing one. It can contribute to the solution of both the Document Retrieval and the Formula Retrieval problems. Typical examples of enrichment are: heuristically generating and storing content metadata from Presentation MathML [MY08]; automatic/interactive disambiguation of formulae in the queries to perform a precise query at the content or semantics level [AGSC+06,Ban06,BR03,BU04]; automatic inference of metadata from context analysis or usage analysis (latent semantics) [Cai04,KTHA14,WG10].

The most impressive application of enrichment is presented in [HQ14]. The aim is to search for geometrical constructions that are described using a procedural language (e.g. draw the segments connecting A with B, B with C, and A with C). Enrichment consists in replacing the procedural with a declarative description (e.g. ABC is a triangle). The same declarative description can be obtained by multiple procedural ones, and thus recall is greatly improved. The technique can also be seen as a form of normalisation (see Sect. 3.1) where the normal form is not unique (e.g. it may be the case that by analysing the hypothesis one could deduce that ABC is also an equilateral triangle even if that is not stated in the procedural description).

Query Reduction. Query reduction trades precision for recall by selectively dropping or weakening some of the constraints present in the query. Results obtained from reduced queries can be ranked after results from precise queries. In the literature it occurs in many forms in solutions to both the Formula Retrieval and the Document Retrieval problems: a constant can be weakened to other constants that co-occur frequently with the given one; constants that occur too frequently can be dropped from the queries; a formula may be required to match only the toplevel structure of the formula given as a query.

3.2 Main Techniques

The following techniques are mutually exclusive. Moreover, each technique performs better on only one of the two problems.

Reduction to Full-text Searches. The technology to perform full-text searches is very advanced and there are popular open software implementations with good performance like Apache Lucene/Solr and ElasticSearch. The benefits of reducing search for formulae to full-text searches are speed of execution of the queries and the combination of formula based and textual searches almost for free. The main drawback is that the precise structure of a formula is partially lost in the translations proposed in the literature, and that it is impossible or very hard to capture precisely the kind of relations $\mathcal{R}$ used for Formula Retrieval, unless $\mathcal{R}$ is approximated by a much coarser relation $\mathcal{R}'$. Therefore the technique has been successfully applied so far only to Document Retrieval [ACK08, GWHT14, GPBB14, HHN08, KTHA14, LM06, LSLM11, LSR14, Mil13, MY03, MM07, MG08a, MG08b, PZ14, MM06, SL11, You05, You07, You08].

All the proposals employ vectors to represent features, and compare features with weighted cosine distance. The usual approach consist in turning an expression into a (large) set of "sentences" that partially describe the formula. For example, in [KTHA14, TKNA13] a sentence is the set (ordered or not) of symbols found in either a path from the root of the formula to a leaf, or as children of the same node. Matching is then performed by a disjunctive query and results are ranked using TF-IDF and length normalisation. As the authors claim, the system "is *too* flexible: it is difficult to say where the relevant results stop and random matches begin; thus we predict higher recall but lower precision rates than exact match systems". Other authors extract sentences or n-grams that capture the formula more precisely. As a general remark, the clear impression we got from the literature is that the fewer features extracted, the lower the precision. All kinds of techniques can be used to extract the features, comprising regular expressions [ACK08] and finite state automata [NCH12].

Some systems cluster documents at indexing time (e.g. [ACK08]), and retrieve documents comparing the feature vector of the query with the centroid of the cluster. For example, documents about trigonometric functions are likely to be automatically clustered together. However most systems do not seem to cluster in advance, and prefer the flexibility of weights to capture similarity of features (e.g. similarity of occurrences of trigonometric functions).

According to the set of features extracted, the weighting function used, and the other modular techniques used in combination, the accuracy achieved by systems based on this technique range from extremely low to extremely high (see, for example, [AKO14]).

Structure-Based Indexing via Tries/Substitution Trees. Formula Retrieval can be solved with the data structures developed for automatic theorem proving to store libraries of lemmas and quickly retrieve formulae up to instantiation/generalisation. Pointers to all the statements are stored in the leaves of

a tree that precisely encodes in its paths the statements. To match a formula, the tree is recursively traversed using the formula to drive the descent. The relations $\mathcal{R}$ that can be captured are only instantiation and generalisation of whole formulae. MathWebSearch [HKP14,KP13,KT10] are based on this approach. Retrieval of formulae is very fast, assuming that the index can be entirely stored in main memory.

This approach consistently maximises precision but presents poor recall. To accept larger relations, or to be applied to Document Retrieval, or to cope with too rigid queries, the technique needs to be integrated with other ideas. For example: to match subtrees of formulae in the library, every subtree of a lemma needs to be stored as well in the index; to solve unification problems up to an equational theory that admits normal forms, all formulae are normalised; to allow queries that use keywords or free text, a free-text search engine must be run in parallel and the results need to be combined in the ranking phase [HKP14,LDM+08].

Reduction to SQL or ad-hoc Queries. The third approach consists in approximating formulae via relations to be stored in a relational DB [AS04, GS03]. An alternative consists in storing the relations in ad-hoc indexes in memory, and it is employed when the indexes already exists for other purposes (typically in libraries of formalised knowledge) [Ban06,BR03,BU04]. The technique is applied to Formula Retrieval and the database can be reused for Document Synthesis without modifications. Approximated queries up to generalisation/instantiation can be made efficient [AS04] without requiring an index stored in main memory for Structure-Based Indexing (see page 10). Recall can be maximised by relaxing the representation of formulae as relations or by employing normalisation. Ad-hoc inverted indices for paths and to map each Content MathML node to its parent have also been used in [HS13]: the search engine is very fast, but the precision obtained is low.

Reduction to XML-based Searches. Some systems [AY07b,AY08b,YA07] that index MathML documents at the content level, base their searching capabilities on the existing XPath/XQuery technology. The system described in [SLM13], which is based on Stratosphere, is batch oriented, trades flexibility with performance, and it is essentially math-unaware (for example, it does not normalise the input in any way). Other systems [CDT04,LDM+08], that deal with ontologies indexed in the Ontology Web Language, rely on third-party OWL search engines implementing graph matching.

4 Ranking

Because users only inspect the first results returned by a query, precision when solving Document Retrieval is strongly determined by the ranking function. Ranking is also of paramount importance for Formula Retrieval: when the search retrieves the candidates for progressing in a proof, correctly ranking the results

may dramatically cut the number of wrong proof attempts and backtracks. The ranking criterion for the two Problems is, however, very different: for the first problem similarity of formulae in the query and in the results should contribute significantly to the score; for the second problem the score should be determined by the intended use of the results. For example, a lemma L_1 that exactly matches the goal to prove and has no premise should always score better than a lemma L_2 that also exactly matches the goal, but that has hypotheses to be proved later.

Ranking according to the intended use for Formula Retrieval has received very little interest in the literature we examined. On the other hand, several papers explicitly address ranking for Document Retrieval. The consensus seem to be that a good ranking function needs to be sophisticated and that the usual metrics induced by reduction to textual searches are completely inadequate (see, for example, [You07]). All the proposed ranking techniques are strongly based on heuristics and, unfortunately, most of them are incomparable and hard to combine.

One class of metrics takes into consideration also the structure of the formulae involved and the enriched semantics, when available. For example, [ZY14] heavily exploits Content MathML to rank results by considering the taxonomic distance of constants (where close is approximated to being defined in the same content dictionary), the data type hierarchical level (matching a function is more significant than matching a numerical constant), matching depth (partially matching the formula at the top level is more significant than matching a deeply nested subformula), coverage (percentage of formula matched), kind of matched expression (formula vs. term). All this information needs to be computed and amalgamated employing some kind of heuristic algorithm. A second paper [SYM+14] confirmed that each one of the listed similarity feature factors significantly improves the ranking, but the last one, that still contributes, has lower relevance.

In [SL11] ranking is determined by the weights used during matching, and the authors claim that each document base and scientific field should have its own weighting function. Nevertheless, they "tried to create a complex and robust weighting function that would be appropriate to many fields".

In [You07] the author proposes a parameterised ranking function that works on mathematical documents (not only formulae), that seems applicable to enriched presentation and that weights a lot of additional information including keywords, the number of cross-references and their kind (e.g. definitional vs. propositional). Ranking employs a hybrid of scalarisation and vectorisation.

In [KT13] the authors propose to adopt tree edit distance to measure similarity of formulae. Most of the paper is about optimisations to improve efficiency of ranking because tree edit distance is hard to compute. The final proposal combines some clever memoisation and a procedure to quickly prune documents bounding their similarity scores with a lightweight computation. The paper also shows benchmarks comparing the processing time and success rate of most search and ranking algorithm in the literature, reimplemented by the authors and run on the same dataset. From the benchmarks the method proposed seems to be

superior, but the implementations do not exploit relevant enriched information like cross-references, semantic proximity of definitions, etc. The benchmarks are therefore non conclusive.

In [NCH12] the authors employ a continuous learning ranking model after having extracted features from Content MathML mathematical formulae using a finite state automata. Benchmarks show their ranking to be superior than the ones used in classical ranking of textual documents. However, they do not compare with [KT13] or [You07] (that work on Presentation MathML).

Simpler approaches to ranking can be found in [YA09] (based on Subpath Set, reported to work well only on "simplified" Content MathML) and [KT09] that works on Presentation MathML and measures similarity as a function of the size of subtrees in common. The ranking metric used in [ACK08] is a TF-IDF modified with weights to assign more importance to some operators, but the details given to determine the weights are insufficient.

Ranking algorithms can be too complex to be incorporated in the search phase, for example when using Lucene technology. Moreover, they are typically slower than the search phase. Therefore several authors suggest to re-rank only the first results of the query, that employs a simpler ranking measure to determine the interesting candidates to be ranked more accurately [KT13, You07].

An algorithm to automatically categorise documents is presented in [ZKT08], where it is argued that clustering documents according to their category greatly improves the usefulness of the tool for the user.

5 Evaluation of Math Information Retrieval

Several papers present benchmarks on the systems proposed, and rarely compare them with reimplementations of the algorithms found in the literature (e.g. [KT13]). The significance of most of these benchmarks is unclear, because conflicting results are found in the literature, most techniques are not presented in sufficient details in the papers to be exactly reproduced, and systems are very sensitive to the kind of queries examined. The only alternative is to compare different tools on unbiased, standard benchmarks that are currently lacking.

The main issue is not to come up with large corpora of documents: at least for Document Retrieval on enriched Presentation MathML documents, a large corpus can be easily obtained converting documents from ArXiV, DLMF, PlanetMath, Wikipedia, etc. For Formula Retrieval, the existing libraries of interactive theorem provers, like Mizar and Coq, can be directly used after conversion. The problem is to determine large sets of *real world, interesting queries*, and to evaluate the results. Automatic evaluation is particularly hard in the domain of mathematics, whereas manual evaluation is limited to a tiny number of queries and runs. Formulating good sets of queries is also complex, because users with different mathematical background and motivations are likely to issue different queries. Moreover, what makes a query hard can just be the use of non standard mathematical notations, errors in the encoding of formulae, or formulation at the wrong level of abstraction. Reference [L13] discusses the problem

at length and reviews the state of the art of evaluation of Math Information Retrieval before 2013, including the experience of the MIR workshop at CICM 2012 were two systems were compared on about ten hard queries proposed by the judges, and the conclusion was that the systems were too sensitive to the formulation of the query.

The situation is improving since 2013 with the creation of a math oriented task in the NTCIR initiative [AKO13, AKO14] that is attracting a small, but increasing number of participants [GWHT14, GPBB14, HS13, HKP14, KP13, KTHA14, LRG13, LAP+14, LSR13, PZ14, RSL14, SLM13, SYM+14, TKNA13]. The initiative is too young to come to definite conclusions and the current choice of tasks and queries is not granted yet to have significant coverage and to be unbiased. For example, in [KTHA14] the authors report that despite several improvements to the tool (quantified via NTCIR-11 runs), their tool scored lower than in NTCIR-10. They justify the phenomenon by noticing that "in NTCIR-11, query variables get much bigger emphasis, most topics feature complete and very particular formulae, and sub-formulae matching is not nearly as useful as before". Indeed, as reported in [AKO14] "the design decision ... to exclusively concentrate on formula/keyword queries and use paragraphs as retrieval units ... has also focused research away from questions like result presentation and user interaction. ... few of the systems has invested into further semantics extraction from the data set. ... We feel that this direction should be addressed more in future challenges". An effect of the bias towards search up to unification w.r.t. search up to similarity is observable in [PZ14] too: the system proposed works very well even if it works on Presentation MathML only and the set of features extracted is very simple (bag-of-symbol-pairs model, where a pair is made of two symbols in a father-son relation). The reason why it works well is that the authors also index approximated pairs where the child is a wildcard, and in the future works they are thinking at improving even more the handling of wildcards to score better. In comparison, most other systems based on feature vectors just replace wildcards in the query with $< \texttt{m} : \texttt{ci} >$ identifiers. The emphasis on formula queries is also to be evaluated considering the already cited works that conclude that users do not see (yet?) much value in them [KK07, ZKT08].

Finally, the NTCIR task does not cover distinctly the Document Retrieval and the Formula Retrieval problems, but only Document Retrieval with an emphasis on exact pattern matching of formulae that should be more distinctive of Formula Retrieval.

6 Availability of Math Retrieval Systems

Most of the systems described in the literature are research prototypes, and the majority of them are no longer working or no longer accessible. At the time this paper was written, the only ones for Document Retrieval with a running Web interface or code that can be downloaded are: (1) Design Science's MathDex (formerly MathFind) (2) NIST DMLF (3) MathWebSearch (4) MiAS (Math Indexer and Searcher), also used to search the EuDML (5) the system described

in [PZ14]. In addition to those, the following commercial systems are also accessible: (a) Springer LaTeXSearch (b) Wolfram Alpha. Systems (a), (1), (2) and (5) are based on Presentation MathML; systems (3) and (4) can use either Presentation MathML or Content MathML/parallel markup, but work better on the latter; finally system (b) actually generates on the fly most of the result of the query, for example by plotting functions, computing their Taylor expansion, etc. It does not really qualify then as a search engine.

Most interactive theorem provers also have their own implementation of a search engine to solve Formula Retrieval. Most of the time, the implementation is embedded in the system and does not work on the whole library at once, with the exception of MML Query for Mizar.

7 Conclusions

Mathematical knowledge retrieval, the low hanging fruit of Mathematical Knowledge Management, is still far from being grasped. Despite the significant amount of work dedicated to the topic in the last 12 years, only a few systems are still available, and their precision and recall scores compared to other knowledge retrieval fields are low. Moreover, usability and user requirement studies suggest that queries containing formulae — the main focus of the majority of papers — are perceived by users as not very useful (yet?).

The main contributions of this paper have been providing an hopefully comprehensive bibliography on the subject, and presenting taxonomies for both mathematical retrieval problems and techniques. We believe that our purpose driven taxonomy can be useful in classifying papers, in clarifying the scope of application of techniques and in the much needed development of unbiased benchmarks for mathematical retrieval.

References

ACK08. Adeel, M., Cheung, H.S., Khiyal, S.H.: Math GO! prototype of a content based mathematical formula search engine. J. Theor. Appl. Inf. Technol. **4**(10), 1002–1012 (2008)

AGSC+06. Asperti, A., Guidi, F., Coen, C.S., Tassi, E., Zacchiroli, S.: A content based mathematical search engine: Whelp. In: Filliâtre, J.-C., Paulin-Mohring, C., Werner, B. (eds.) TYPES 2004. LNCS, vol. 3839, pp. 17–32. Springer, Heidelberg (2006)

AKO13. Aizawa, A., Kohlhase, M., Ounis, I.: NTCIR-10 math pilot task overview. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 654–661 (2013)

AKO14. Aizawa, A., Kohlhase, M., Ounis, I.: NTCIR-11 math 2 task overview. In: Proceedings of 10th NTCIR Conference, Tokyo, Japan, pp. 88–98 (2014)

AS04. Asperti, A., Selmi, M.: Efficient retrieval of mathematical statements. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 17–31. Springer, Heidelberg (2004)

AY07a. Altamimi, M.E., Youssef, A.: A more canonical form of content MathML to facilitate math search. In: Proceedings of Extreme Markup Languages (2007)

AY07b. Altamimi, M.E., Youssef, A.S.: Wildcards in math search, implementation issues. In: Proceedings of the ISCA 20th International Conference on Computer Applications in Industry and Engineering, CAINE 2007, San Francisco, California, USA, 7–9 November, pp. 90–96 (2007)

AY08a. Ahmadi, S.A., Youssef, A.: Lexical error compensation in handwritten-based mathematical information retrieval. In: Proceedings of Towards Digital Mathematics Library, DML 2008, Birmingham, UK, 27 July, pp. 43–54. Masaryk University, Brno (2008)

AY08b. Altamimi, M.E., Youssef, A.S.: A math query language with an expanded set of wildcards. Math. Comput. Sci. **2**(2), 305–331 (2008)

AZ04. Asperti, A., Zacchiroli, S.: Searching mathematics on the web: state of the art and future developments. In: Karlsruhe, F (ed.) Proceedings of New Developments in Electronic Publishing of Mathematics, pp. 9–18 (2004)

Ban06. Bancerek, G.: Information retrieval and rendering with MML query. In: Borwein, J.M., Farmer, W.M. (eds.) MKM 2006. LNCS (LNAI), vol. 4108, pp. 266–279. Springer, Heidelberg (2006)

BF03. Baumgartner, P., Furbach, U.: Automated deduction techniques for the management of personalized documents. Ann. Math. Artif. Intell. **38**(1–3), 211–228 (2003)

BR03. Bancerek, G., Rudnicki, P.: Information retrieval in MML. In: Asperti, A., Buchberger, B., Davenport, J.H. (eds.) Mathematical Knowledge Management, pp. 119–132. Springer, Bertinoro (2003)

BU04. Bancerek, G., Urban, J.: Integrated semantic browsing of the mizar mathematical library for authoring mizar articles. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 44–57. Springer, Heidelberg (2004)

Cai04. Cairns, P.: Informalising formal mathematics: searching the mizar library with latent semantics. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 58–72. Springer, Heidelberg (2004)

CDT04. Caprotti, O., Dewar, M., Turi, D.: Mathematical service matching using description logic and OWL. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 73–87. Springer, Heidelberg (2004)

Del00. Delahaye, D.: Information retrieval in a *Coq* proof library using type isomorphisms. In: Coquand, T., Nordström, B., Dybjer, P., Smith, J. (eds.) TYPES 1999. LNCS, vol. 1956, pp. 131–147. Springer, Heidelberg (2000)

FLRS12. Formánek, D., Líška, M., Růžička, M., Sojka, P.: Normalization of digital mathematics library content. In: Davenport, J., Jeuring, J., Lange, C., Libbrecht, P. (eds.) Joint Proceedings of the 24th OpenMath Workshop, the 7th Workshop on Mathematical User Interfaces (MathUI), and the Work in Progress Section of the Conference on Intelligent Computer Mathematics. CEUR Workshop Proceedings, vol. 921, pp. 91–103. Neuveden, Aachen (2012)

GK14. Gauthier, T., Kaliszyk, C.: Matching concepts across HOL libraries. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 267–281. Springer, Heidelberg (2014)

GPBB14. Pinto, J.M.G., Barthel, S., Balke, W.-T.: QUALIBETA at the NTCIR-11 math 2 task: an attempt to query math collections. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 103–107 (2014)

GS03. Guidi, F., Schena, I.: A query language for a metadata framework about mathematical resources. In: Asperti, A., Buchberger, B., Davenport, J.H. (eds.) Mathematical Knowledge Management, pp. 105–118. Springer, Bertinoro (2003)

GWHT14. Gao, L., Wang, Y., Hao, L., Tang, Z.: ICST math retrieval system for NTCIR-11 math-2 Task. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 99–102 (2014)

HHN08. Hashimoto, H., Hijikata, Y., Nishida, S.: Incorporating breadth first search for indexing MathML objects. In: IEEE International Conference on Systems, Man and Cybernetics, SMC 2008, pp. 3519–3523, October 2008

HKP14. Hambasan, R., Kohlhase, M., Prodescu, C.: MathWebSearch at NTCIR-11. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 114–119 (2014)

HQ14. Haralambous, Y., Quaresma, P.: Querying geometric figures using a controlled language, ontological graphs and dependency lattices. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 298–311. Springer, Heidelberg (2014)

HS13. Hagino, H., Saito, H.: Partial-match retrieval with structure-reflected indices at the NTCIR-10 math task. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 692–695 (2013)

KK07. Kohlhase, A., Kohlhase, M.: *Re*examining the MKM value proposition: from math web search to math web *Re*Search. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 313–326. Springer, Heidelberg (2007)

Koh14. Kohlhase, A.: Search interfaces for mathematicians. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 153–168. Springer, Heidelberg (2014)

KP13. Kohlhase, M., Prodescu, C.: MathWebSearch at NTCIR-10. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 675–679 (2013)

KT09. Kamali, S., Tompa, F.W.: Improving mathematics retrieval. In: Proceedings of Towards Digital Mathematics Library, DML 2009, Grand Bend, Ontario, Canada, 8–9 July, pp. 37–48. Masaryk University, Brno (2009)

KT10. Kamali, S., Tompa, F.W.: A new mathematics retrieval system. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 1413–1416. ACM, New York (2010)

KT13. Kamali, S., Tompa, F.W.: Structural similarity search for mathematics retrieval. In: Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. (eds.) CICM 2013. LNCS, vol. 7961, pp. 246–262. Springer, Heidelberg (2013)

KTHA14. Kristianto, G.Y., Topić, G., Ho, F., Aizawa, A.: The MCAT math retrieval system for NTCIR-11 math track. In: Proceedings of the 11th NTCIR Conference, Tokyo, Japan, pp. 120–126 (2014)

L10. Líška, M.: Searching Mathematical Texts (2010)

L13. Líška, M.: Evaluation of Mathematics Retrieval (2013)

LAP+14. Lipani, A., Andersson, L., Piroi, F., Lupu, M., Hanbury, A.: TUW-IMP at the NTCIR-11 Math-2. In: Proceedings of the 11th NTCIR Conference, Tokyo, Japan, pp. 143–146 (2014)

LDM+08. Libbrecht, P., Desmoulins, C., Mercat, C., Laborde, C., Dietrich, M., Hendriks, M.: Cross-curriculum search for intergeo. In: Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F. (eds.) AISC 2008, Calculemus 2008, and MKM 2008. LNCS (LNAI), vol. 5144, pp. 520–535. Springer, Heidelberg (2008)

Lib13. Libbrecht, P.: Escaping the trap of too precise topic queries. In: Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. (eds.) CICM 2013. LNCS, vol. 7961, pp. 296–309. Springer, Heidelberg (2013)

LM06. Libbrecht, P., Melis, E.: Methods to access and retrieve mathematical content in ActiveMath. In: Iglesias, A., Takayama, N. (eds.) ICMS 2006. LNCS, vol. 4151, pp. 331–342. Springer, Heidelberg (2006)

LRG13. Ray R. Larson., Chloe J. Reynolds., Fredric C. Gey.: The Abject Failure of Keyword IR for Mathematics Search: Berkeley at NTCIR-10 Math. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 662–666 (2013)

LSLM11. Líška, M., Sojka, P., Líška, M., Mravec, P.: Web interface and collection for mathematical retrieval: WebMIaS and MREC. In: Proceedings of Towards Digital Mathematics Library, DML 2011, Bertinoro, Italy, 20–21 July, pp. 77–84. Masaryk University, Brno (2011)

LSR13. Líška, M., Sojka, P., Růžička, M.: Similarity search for mathematics: Masaryk university team at the NTCIR-10 math task. In: Proceedings of the 10th NTCIR Conference, Tokyo, Japan, pp. 686–691 (2013)

LSR14. Líška, M., Sojka, P., Růžička, M.: Math indexer and searcher web interface. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 444–448. Springer, Heidelberg (2014)

MG08a. Misutka, J., Galambos, L.: Mathematical extension of full text search engine indexer. In: 3rd International Conference on Information and Communication Technologies: From Theory to Applications, ICTTA 2008, pp. 1–6, April 2008

MG08b. Mišutka, J., Galamboš, L.: Extending full text search engine for mathematical content. In: Proceedings of Towards Digital Mathematics Library, DML 2008, Birmingham, UK, 27 July, pp. 55–67. Masaryk University, Brno (2008)

Mil13. Miller, B.R.: Three years of DLMF: web, math and search. In: Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. (eds.) CICM 2013. LNCS, vol. 7961, pp. 288–295. Springer, Heidelberg (2013)

MM06. Munavalli, R., Miner, R.: Mathfind: a math-aware search engine. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 735–735. ACM (2006)

MM07. Miner, R., Munavalli, R.: An approach to mathematical search through query formulation and data normalization. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 342–355. Springer, Heidelberg (2007)

MY03. Miller, B.R., Youssef, A.: Technical aspects of the digital library of mathematical functions. Ann. Math. Artif. Intell. **38**(1–3), 121–136 (2003)

MY08. Miller, B.R., Youssef, A.M.: Augmenting presentation MathML for search. In: Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F. (eds.) AISC 2008, Calculemus 2008, and MKM 2008. LNCS (LNAI), vol. 5144, pp. 536–542. Springer, Heidelberg (2008)

NCH12. Nguyen, T.T., Chang, K., Hui, S.C.: A math-aware search engine for math question answering system. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 724–733. ACM (2012)

NK07. Normann, I., Kohlhase, M.: Extended formula normalization for ϵ-retrieval and sharing of mathematical knowledge. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 356–370. Springer, Heidelberg (2007)

NKTA14. Nghiem, M.-Q., Kristianto, G.Y., Topić, G., Aizawa, A.: Which one is better: presentation-based or content-based math search? In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 200–212. Springer, Heidelberg (2014)

PZ14. Pattaniyil, N., Zanibbi, R.: Combining TF-IDF text retrieval with an inverted index over symbol pairs in math expressions: the tangent math search engine at NTCIR 2014. In: Proceedings of the 11th NTCIR Conference, Tokyo, Japan, pp. 135–142 (2014)

Rab12. Rabe, F.: A query language for formal mathematical libraries. In: Campbell, J.A., Jeuring, J., Carette, J., Dos Reis, G., Sojka, P., Wenzel, M., Sorge, V. (eds.) CICM 2012. LNCS, vol. 7362, pp. 143–158. Springer, Heidelberg (2012)

RSL14. Růžička, M., Sojka, P., Líška, M.: Math indexer and searcher under the hood: history and development of a winning strategy. In: Proceedings of the 11th NTCIR Conference, Tokyo, Japan, pp. 127–134 (2014)

SL11. Sojka, P., Líška, M.: The art of mathematics retrieval. In: Proceedings of the 11th ACM Symposium on Document Engineering, pp. 57–60. ACM (2011)

SLM13. Schubotz, M., Leich, M., Markl, V.: Querying large collections of mathematical publications: NTCIR10 math task. In: Proceedings of 10th NTCIR Conference, Tokyo, Japan, pp. 667–674 (2013)

SY07. Shatnawi, M., Youssef, A.: Equivalence detection using parse-tree normalization for math search. In: Proceedings of the Second IEEE International Conference on Digital Information Management (ICDIM), Lyon, France, 11–13 December, pp. 643–648 (2007)

SYM+14. Schubotz, M., Youssef, A., Markl, V., Cohl, H.S., Li, J.J.: Evaluation of similarity-measure factors for formulae based on the NTCIR-11 math task. In: Proceedings of 10th NTCIR Conference, Tokyo, Japan, pp. 108–113 (2014)

TKNA13. Topić, G., Kristianto, G.Y., Nghiem, M.-Q., Aizawa, A.: The MCAT math retrieval system for NTCIR-10 math track. In: Proceedings of 10th NTCIR Conference, Tokyo, Japan, pp. 680–685 (2013)

WG10. Wolska, M., Grigore, M.: Symbol declarations in mathematical writing. In: Proceedings of Towards Digital Mathematics Library, DML 2010, Paris, France, 7–8 July, pp. 119–127. Masaryk University, Brno (2010)

YA07. Youssef, A.S., Altamimi, M.E.: An extensive math query language. In: Proceedings of the 16th International Conference on Software Engineering and Data Engineering (SEDE-2007), 9–11 July, Imperial Palace Hotel Las Vegas, Las Vegas, Nevada, USA, pp. 57–63 (2007)

YA09. Yokoi, K., Aizawa, A.: An approach to similarity search for mathematical expressions using MathML. In: Proceedings of Towards Digital Mathematics Library, DML 2009, Grand Bend, Ontario, Canada, 8–9 July, pp. 27–35. Masaryk University, Brno (2009)

You05. Youssef, A.: Search of mathematical contents: issues and methods. In: Proceedings of the ISCA 14th International Conference on Intelligent and Adaptive Systems and Software Engineering, 20–22 July, Novotel Toronto Centre, Toronto, Canada, pp. 100–105 (2005)

You06. Youssef, A.M.: Roles of math search in mathematics. In: Borwein, J.M., Farmer, W.M. (eds.) MKM 2006. LNCS (LNAI), vol. 4108, pp. 2–16. Springer, Heidelberg (2006)

You07. Youssef, A.S.: Methods of relevance ranking and hit-content generation in math search. In: Kauers, M., Kerber, M., Miner, R., Windsteiger, W. (eds.) MKM/CALCULEMUS 2007. LNCS (LNAI), vol. 4573, pp. 393–406. Springer, Heidelberg (2007)

You08. Youssef, A.S.: Relevance ranking and hit description in math search. Math. Comput. Sci. **2**(2), 333–353 (2008)

YS06. Youssef, A., Shatnawi, M.: Math search with equivalence detection using parse-tree normalization. In: The 4th International Conference on Computer Science and Information Technology (2006)

ZB12. Zanibbi, R., Blostein, D.: Recognition and retrieval of mathematical expressions. Int. J. Doc. Anal. Recogn. (IJDAR) **15**(4), 331–357 (2012)

ZKT08. Zhao, J., Kan, M.-Y., Theng, Y.L.: Math information retrieval: user requirements and prototype implementation. In: Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 187–196. ACM (2008)

ZY14. Zhang, Q., Youssef, A.: An approach to math-similarity search. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) CICM 2014. LNCS, vol. 8543, pp. 404–418. Springer, Heidelberg (2014)