The Statusquo of MKM
Current Trends in Knowledge Buildup and Retrieval

Camelia Rosenkranz*

Research Institute for Symbolic Computation
Johannes Kepler Universität Linz
4040 Linz, Austria

Abstract

We survey the state-of-the-art in mathematical knowledge management, investigating systems that provide tools for processing mathematical formulae. We emphasize the logical questions, concentrating on the aspects of *build up*, *structure* and *retrieval* of mathematical knowledge.

1 Origin and Perspectives of MKM

The challenge of *building up, structuring and retrieving* mathematical knowledge is not new, but it becomes more and more acute as the volume of mathematical information on different subjects increases. For instance, in order to avoid extensive literature search, mathematicians often prefer reinventing lemmas, and thus lose precious time with "trivial" but tedious proofs that could be (at least partially) automated instead of spending days or months for finding the already existing proofs, hidden somewhere in the huge literature.

Thus tools for managing mathematical knowledge (e.g. providing well-structured formalized mathematical information in an electronic format that can be browsed and searched automatically) are urgently needed. The underlying software technology is already here —but the ideas and the techniques are yet to be developed.

Developed from the outset as an interdisciplinary field, MKM finds itself at the intersection of mathematics, logic, computer science, and scientific publishing. Its goal may be summarized as the discovery of new techniques—based on formal mathematics and supported by software tools—for building up repositories of mathematical knowledge and structuring them from a practical perspective: e.g. mathematicians should be helped in their *day-by-day research*, where knowledge retrieval is needed for finding slightly different formulations of an existing lemma, doing simple proofs etc.

As a research field, MKM was launched in September 2001 at the *First International Workshop on MKM* in Hagenberg, Austria (see the Special Issue [20] of the *Annals of Mathematics and Artificial Intelligence*).

*This work was supported by the Austrian Science Foundation FWF under the SFB grant F1302

In its call for papers, Bruno Buchberger defined MKM as computer-supported management of mathematical knowledge, parsing "mathematical knowledge management" as "(mathematical knowledge) management" rather than "mathematical (knowledge management)". A dedicated conference [40] now reached already its seventh year.

In this context, (MK)M should be understood as the general investigation of how one can manage *mathematical knowledge* with its very specific features: highly hierarchical structure, infinitely many variants of different logical foundations, etc.

In order to develop the new academic field, several *European projects* have been initiated: Calculemus[1], MOWGLI[2], MKMNet[3], and others. Their objective is to identify various subdomains of MKM and to clarify foundational questions. In the actual evolution of these projects, one can also see the difficulty of formalizing everyday mathematics as well as the subtly different views of the scientists involved.

In this paper, we want to present the emerging field of MKM mainly from the logical perspective, surveying various ideas for structuring, building up and retrieving mathematical knowledge.

2 Processing Mathematical Knowledge

Having formalized mathematical information in a computer-readable format is not enough for being able to use it in proving, solving or computing. One needs on top of the knowledge an organizational structure with browsing and searching capabilities. In a first approximation, we call a system with these capabilities an *MKM System*.

Even though there are various viewpoints as to what MKM encompasses [35, 55, 7, 58, 26, 54], one can identify some *characteristics* that should be part of an MKM system. Along the lines of [35], we regard as defining components:

- the *language* (first-order or higher-order logic, simply-typed or some form of polymorphic lambda calculus), having a certain *representation* (that deals with the notational issues),
- the way in which the mathematical theories are *organized* both logic-internally (i.e. by their logical structure) and logic-externally (i.e. by their segmentation in chapters).
- facilities of *access*, i.e. tools for carrying out *searches*, for performing deductions or computations in a (fully or partly) automatical way, and for displaying the contents and structure of the knowledge under consideration.

In the following subsections we present a more detailed description of these characteristics, emphasizing the logical aspects of build-up, structure and retrieval of mathematical knowledge.

2.1 Language and Representation

Formalized mathematical knowledge is expressed in a *logical language*. This can vary from classical first order logic, as in the case of Mizar[4], up

[1] Calculemus, `http://www.eurice.de/calculemus`
[2] MOWGLI, `http://mowgli.cs.unibo.it/`
[3] MKMNet, `http://monet.nag.co.uk/mkm/index.html`
[4] Mizar, `http://www.mizar.org`

to some version of higher order dependently-typed polymorphic lambda calculus, as in CoQ[5]. It may (like Isabelle/FOL[6]) or may not (like TPS[7]) include sorts, it can be based on an underlying set theory (like Mizar, Metamath[8]) or dispense with it (like Ωmega[9]).

Most people seem to prefer a formal language close to that of working mathematicians, as expressed in the concept of a "mathematical vernacular" [34, 65].

For storing mathematical knowledge one needs a *representation*. The existing representation languages differ from system to system. Some of them are application-independent, like MathML[10] and OpenMath[11]. Other systems, like Mizar and Theorema[12], have their own representation language (that can be easily translated to MathML, OpenMath or OMDoc [8, 45]).

MathML was created for presenting *mathematical formulae on the web*. Its authors described it as "an XML application for describing mathematical notation and capturing both its structure and content. The goal of MathML is to enable mathematics to be served, received, and processed on the World Wide Web, just as HTML has enabled this functionality for text." Its main parts are presentation MathML, a markup used for the rendering of mathematical content on the web (can be seen as an extension of HTML), and content MathML, the actual mathematical content that can be processable by machines. It covers "high-school mathematics" (by having around 80 specialized elements for the most common functions and individual constants) and can currently be rendered on most available web browsers with the help of plugins.

OpenMath was originally constructed for facilitating communication of mathematical knowledge between different computer algebra systems. It extends content MathML by using so-called *content dictionaries* that define the "semantics" (or "meaning") of the objects expressed by the OpenMath symbols.

While content MathML defines the *meaning* of symbols only in the sense of an informal comment directed to human readers (as it is explained in the MathML Recommandation), OpenMath aims to provide machine-readable content dictionaries, defining the meaning of mathematical concepts expressed by OpenMath symbols [42]. Up to now, what is done in establishing the "meaning" of a concept like $+$ is

- to select certain properties considered crucial (e.g. commutativity) and to formalize them in a custom-tailored language,
- to offer a common reference point for MKM-aware applications such that they can agree on one and the same concept of addition (no matter how it is represented in either application).

The context of a mathematical formula is strongly related with the way in which the mathematical information is build up and organised. As we see in the following section, languages like OMDoc[13] are a building block for the organisation of mathematical knowledge.

[5] CoQ, `http://coq.inria.fr`
[6] Logics of Isabelle, `http://www.cl.cam.ac.uk/research/hvg/Isabelle/logics.html`
[7] TPS, `http://gtps.math.cmu.edu/tps.html`
[8] Metamath, `http://metamath.org/`
[9] Ωmega, `http://www.ags.uni-sb.de/~omega/omega/index.php`
[10] The MathML 2.0 Recomandation `http://www.w3.org/TR/MathML/`
[11] OpenMath, `http://www.openmath.org/`
[12] Theorema,`http://www.theorema.org`
[13] OMDoc,`http://www.omdoc.org/`

2.2 Organization and Buildup

In mathematics, one usually does not study a mathematical formula in isolation, but in the context of a mathematical theory. For storage and organisation of mathematics, the context where a formula occurs is therefore important. This is the premise of OMDoc [42], which is based on the following three-level hierarchy:

- mathematical formulae (coded in content MathML or OpenMath),
- various statements (like definitions, assertions and examples),
- theories (related by morphisms for expressing e.g. theory inclusion/interpretation).

One way of organising mathematical knowledge by considering the mathematical knowledge as a book, and splitting it in sections and paragraphs, being included in and/or having links to other "books". The organisation can also be logic-internal, i.e. depending on the mathematical content.

An example of organized library of mathematics is the MML library of Mizar, which is, at this moment, the biggest repository of mathematical knowledge (see chapter 3, the section on Mizar). Other libraries of mathematics include HELM [1] and MBase [41]. One may also view the content dictionaries of OpenMath itself as a primitiv e structuring tool. The ongoing efforts for reworking the theory of special functions into an online database in the DLMF project are another instance of structuring a considerable portion of mathematics. Also the authoring tools presented below have an organizational aspect.

Building mathematical knowledge is a first step necessary for storing and organizing mathematical knowledge. This process entails the generation of mathematical information by

- authoring new documents,
- supporting (or even automating) the systematic exploration of mathematical theories and
- retrodigitization (reconstructing mathematical contents from electronic files written in formats like LaTeX, or obtaining a scanned/text version out of a printed one).

There exist different tools for supporting the authoring of new documents. In Theorema, using the Label Management[57] tool, the user can segment the mathematical content in chapters, sections. She can also choose the type of information input (e.g. whether is a Definition or a Theorem). Another example of supporting authoring of mathematical documents is ActiveMath, where one can input OMDoc documents interlaced with textual information.

Systematic theory exploration [13] is a natural environment for mathematics. The buildup of mathematical knowledge is an involved process that can be carried out in various different ways [12]. One can contrast top-down versus bottom-up approaches [17]. In the latter case, one studies systematically all interactions between the concepts involved: Each time a new predicate or function is introduced, one investigates all interesting interactions to the old concepts. Thus one builds up a mathematical theory in layers, starting from elementary notions and axioms.

In the top-down approach, one starts out with a rudimentary knowledge base containing only the axioms and some fundamental theorems. Trying to prove a conjecture "higher up" in the knowledge hierarchy will typically lead to auxiliary lemmata whose proof will also guarantee the main conjecture. These lemmata will in turn produce new auxiliary lemmata, thus producing a whole *cascade* [16] of new mathematical formulae. These can all be added to the initial knowledge base—if the recursive generation of lemmata is eventually grounded in direct proofs from the rudimentary knowledge base. In an ideal setup for generating mathematical knowledge, the top-down and the bottom-up approaches should be combined.

Theory exploration can also be contrasted to *isolated theorem proving* [12], where one usually proves a single important theorem, directly from the axioms of the domain in which this theorem holds. In mathematical practice, though, theorems typically come in a context of various propositions and lemmata that can and should be used (provided they are already proved). Thus exploring a mathematical theory can help to produce shorter and clearer proofs. HR [30] and MATHsAID [50] provide tools for automatical theory exploration, SysteMathEx [32] carried case studies in systematic theory exploration. During the years, people also developed several domain-specific custom-tailored automated conjecture generators; see the survey papers [48, 47].

Retrodigitization of mathematical knowledge is a vast field on its own. It starts with the numerous retrodigitized mathematical journals available on the web (e.g. EMIS[14], JSTOR[15], WDML-Göttingen[16]), usually in pdf-format, and continues with tools like Hermes [60] that translate TEX documents into machine-processable texts (e.g. OMDoc texts).

2.3 Access

An important role in *accessing* mathematical knowledge is played by various version of searching.

2.3.1 Bibliographic searches

Bibliographic searches are based on BibTeX fields like author, title, keywords, etc. They perform online searches for mathematical information independent of the structure and content of the mathematical knowledge, taking into account only the metadata information stored in the BibTeX fields asociated with documents containing mathematical information. Some examples are Zentralblatt Math[17] or MathSciNet[18].

2.3.2 Mathematical Web Services

Mathematical web services also perform online searches for mathematical information. As explained in [7], generic "web services are software systems designed to support machine interaction over the network (usually the internet), typically exposing a programming interface based on exchange of XML documents." Mathematical web services are web services specialized for communicating mathematical information. For the latter, MSDL documents (Mathematical Service Description Language or MSDL is a standardized XML format) describe mathematical problems by their

[14] The European Mathematical Information Service, http://www.emis.de/
[15] The Scholarly Journal Archive (JSTOR), http://www.jstor.org/
[16] Göttingen Digitalisierung Zentrum, http://gdz.sub.uni-goettingen.de/
[17] Zentralblatt Math. FIZ-Karlsruhe, http://www.zentralblatt-math.org/zmath/en/
[18] MathSciNet. Mathematical Reviews on the Web, http://www.ams.org/mathscinet

input/output type and pre/post conditions [25]. As such, mathematical web services can be seen as a special case of content-based searches (in this case the content is restricted to the pre/postconditions).

Mathematical web services can be classified according to

- specific problem classes treated (for each problem type the precondition and postconditions are given),
- standardized taxonomies like GAMS[19],
- algorithms implemented by the individual services.

Examples of mathematical web services are MONET[20], MathBroker [10] and MathServe [68].

2.3.3 Formula Retrieval

We name here as *formula retrieval* a search depending on the structure and content of mathematical knowledge. This type of knowledge retrieval provides techniques for *searching* (the content of) mathematical knowledge, covering various levels of difficulty: textual search, pattern matching, simple inference steps, full-fledged theorem proving.

In our view the activities of buildup, organisation and formula retrieval are tightly interconnected: While appropriate structuring of knowledge should facilitate the retrieval process, appropriate buildup of knowledge will lead to a more natural structure of the knowledge base.

In the optimal case, a complete exploration saturates a given knowledge base by adding all properties that were "difficult" to prove (typically using domain-specific reasoners). Then *retrieval* of a mathematical formula will only involve "easy" inference steps [14], yet a bit more than textual search and pattern matching: also formulae that are "obviously equivalent" to it should be found. More precisely, we want to find any formula that follows "easily" from the knowledge base by an efficient variant of proving that may be called "symbolic computation reasoning", "rewrite-style inferences", etc. [15].

Processing mathematical knowledge is tightly connected to *formal proofs*. For their creation, one can see a wide spectrum of tools, ranging from proof checkers (given an assertion and a proof, check whether the proof establishes the assertion) through proof assistants (given an assertion, build up a proof interactively) to theorem provers (given an assertion, generate a proof automatically).

Of course, there is often a *blurred transition* between these cases (and the terminology is not always consistent). For example, some proof checkers may offer special modes ("auto tactics") for automatically generating simple proof fragments, while an automated theorem prover is often too weak to prove an interesting theorem without breaking it up into numerous auxiliary lemmata.

The existing tools dealing with *retrieval* of mathematical knowledge depend to a larger extent on some representation of the mathematical knowledge involved, and not so much on its logical meaning.

Most of the tools for searching (available in ActiveMath[21], DLMF[22], MathWebSearch[23] etc.) are strongly influenced by the Semantic Web Co-

[19]NIST Guide to Available Mathematical Software(GAMS), `http://gams.nist.gov/`
[20]MONET, `http://monet.nag.co.uk/cocoon/monet/index.html`
[21]ActiveMath, `http:\\www.activemath.org`
[22]DLMF, `http://dlmf.nist.gov/`
[23]MathWebSearch, `http://www.mathweb.org/mws/`

munity, considering the retrieval of mathematical formulae as a special case of information retrieval.

Searches operating on mathematical formulae are reduced to database queries working on metadata (certain crucial properties of the formulae). In this view, search is ultimately textual retrieval, reaching at most some form of pattern matching. The advantage, though, is a relatively fast retrieval process.

3 Groups and Projects

In the following paragraphs, we present (in alphabetical order) *systems* that exemplify the current status in mathematical knowledge management.

ActiveMath is an *interactive learning environment* for mathematics [49], based on an extension of OMDoc. It is web-based and user-adaptive. Its building blocks are content items holding both mathematical expressions and text (that can contain hyperlinks); they are annotated with mathematical and pedagogical attributes and relations (i.e. metadata). For managing, referencing and searching mathematical information, it mainly uses the OMDoc level of definitions, examples and theorems. Accesing mathematical information starts by opening a *course* containing several content items provided by an author. Unknown concepts or symbols can be linked by the authors to their corresponding content item (e.g. their definition) and are thus readily available to the user. Alternatively, the user can search by the name of a content item. The courses visible to a user are automatically adapted to his learning goals, his knowledge and the previous scenarios. Checking the user exercises is done by evaluating the mathematical input (by a computer algebra system connected to it) and providing correspondigly feedback.

LeActiveMath,LeActiveMathis a *language-enhanced* version of ActiveMath that improves among many things the search techniques not only for English, but also for German and Spanish. Another improvement is the use of the interactive concept mapping tool iCMap [52].

iCMap is an interactive visual tool, helping students discover relations between mathematical concepts. Students can play with notions borrowed from the existing ActiveMath courses, creating links between them, and validating them (after the validation, the program colors the correct edges in green, and the incorrect ones in red). The user can construct concept maps and use the validation for all his edges (global verification) or only for one (local verification). The verification is made by a small set of inferences upon the knowledge base and upon the (independent) authored information existing in the exercise. The user input is basically matched with the existing ontologies. E.g., if the user draws an edge between a and b , iCMap checks whether there is a relation between the object associated with a and the object associated with b. A relation can:

- be added by the author of the document, as a template edge,
- relations existing in the underlying OMDoc source (the OMDOC element theory has the purpose of gathering notions, definitions and theorems into theories),
- deductive relations, like is_equivalent, belongs_to. Two definitions relating to the same symbol are linked by the is_equivalent edge (are

considered equivalent). Also transitivity rules are build in for several types of edge (like is_a or belongs_to).

In case of failure, explanations are given to the user like "This edge has a wrong type", "This edge is correct, but subsumes several steps. Please elaborate'".

Search queries in ActiveMath can be combinations of:

- *Text queries* that allow exact phrase searches or fuzzy searches. The latter are based on phonetic and edit-distance fuzziness.
- *Attribute queries* that check whether the mathematical item has certain characteristics. For example, one can search (in the current course) for all items of type "definition", or for all items having the difficulty "very easy".
- *Mathematical expression queries* that are entered by the WIRIS[24] input editor.

Mathematical expressions are converted into text (linearizing the tree expression by giving depth indication for the components) and then indexed by Lucene[25], a full-featured text-searching library. The actual *search* in ActiveMath treats both the formal and the informal part of an OMDoc document in the same way. By its ranking heuristic, it sorts the results that the user sees as a list of titles (hyperlinks to the real matches).

COQ is a formal proof management system [11]. Its logical framework is the Calculus of Inductive Constructions [36].

It provides the following *features*:

- Definition of functions and predicates, statement of mathematical theorems, specification of software.
- Interactive construction of formal proofs that can be checked mechanically by a small certification kernel.
- Automated generation of certified programs from proofs of their specifications.
- Execution of functional programs on appropriate pieces of input.

Certified mathematical knowledge is stored in knowledge files but can only be used by explicitly referring to the relevant knowledge items through their names (the only search mechanisms available are `grep` and similar text-based tools). A recent example of a large knowledge library named C-Corn, building up portions of constructive algebra and analysis, has been developed at Nijmejen [33].

The object-oriented language FoCAL, previously known as FoC, follows a somewhat different approach [59]: Algorithms in computer algebra are verified by coding proofs in a specialized language that is then translated to COQ for *verification*.

DLMF or the *Digital Library of Mathematical Functions* [53, 66] intends to be the (online and printed) successor of Abramowitz and Stegun's "Handbook of Mathematical Functions" [3], a standard reference for engineers throughout several decades. The project, headed by the National Institute of Standards and Technology (NIST), creates documents coded

[24]WIRIS, `http://www.wiris.com/content/view/20/3/lang,en/`
[25]Lucene, `http://lucene.apache.org`

in LaTeX, using special macros for producing a handbook on paper as well as a dynamic website.

This website will provide mathematical formulae, 3D interactive graphics, methods of computation, references, links to software, and an *equation search capability.* The latter is understood as a layer on top of textual search. Mathematical information is preprocessed in the following way [67]:

- *Textualization*: The mathematical information is transformed into structured text. For example, $x^a + y_b - t$ is encoded as `x super a endsuper plus y sub b endsub minus z`.
- *Flattening and scoping*: The scope of quantifiers (linguistic constructs that bind variables) is made explicit and their (typically two-dimensional) syntax is linearized as in the previous step. For example, $\int_0^1 y\,dy$ becomes `integral sub 0 endsub super 1 endsuper integrand y endintegrand dy`.
- *Normalization*: The mathematical equation is brought into a normal form with respect to certain algebraic and notational rules. For example, $ab^{-1}cd^{-1}$ has the normal form `frac{ac}{bd}`, and `x_2^3` and `x^3_2` both have the normal form `x sub 2 endsub super 3 endsuper`.

The preprocessed mathematical information is then indexed by the underlying text search system. A *query* (input in a simplified LaTeX-like query language) goes through the same process as above and triggers a search in the index. Besides this equation search, one can also search for keywords like "Bessel Function". An experimental version of the whole library is available at `http://dlmf.nist.gov/`, but the search engine is not yet online.

HELM stands for the *Hypertextual Electronic Library of Mathematics.* The system "aims to study and develop a technological infrastructure for the creation and maintainance of a virtual, distributed, hypertextual library of formal mathematical knowledge" [1]. HELM imports libraries from COQ [11] and (experimentally) from NuPRL [26], transforms the representation into an intermediate MathML content representation, and makes it available on the internet for browsing and searching.

The *search engine* of HELM is called Whelp [6, 4] and is located at `http://helm.cs.unibo.it`; it has its own syntax, which in most kind of queries is disambiguated into suitable terms of Coq's Calculus of Inductive Constructions.

Whelp is based on the *metadata* model described in [39, 27, 5]. Metadata is used for indexing mathematical notions and marking input/output types as well as the components in the body (hypotheses, conclusion, consistency proof of the notion). For the metadata, only the constant symbols of the original mathematical formula are taken into account. For example, from

$$\forall n : nat.\forall m : nat.\forall p : nat.n * (m + p) = n * m + n * p,$$

the automatically extracted metadata contains only the constants nat, $=$, $*$, $+$ and their position (hypothesis/conclusion) in the formula. Search is then not performed on the stored knowledge, but only on the metadata. Thus one may view the metadata as taking the role of indexes in a database, allowing quick searches. The *search facilities* are as follows:

[26] NuPRL

Locate is the simplest search. The user introduces a string regular expression, and the tool outputs the list of all formulae that have this string as "name".

Hint retrieves all theorems that can be applied to derive the current goal. This means, given a theorem $t_1 \rightarrow t_2 \rightarrow ... \rightarrow t_n \rightarrow t$ and a goal g, it checks whether there exists a substitution θ such that $t\theta = g$. This is simulated in HELM by metadata filtering.

Match amounts to reverting the operation of indexing, looking for terms matching the metadata set, which are automatically computed from the user input. The query

$$\forall x, y, z : nat.x * (y + x) = x * y + x * z$$

gives answers that have a type α-convertible with the input, like *cic:/Coq/Arith/Mult/mult_plus_distr_l.con* having the type

$$\forall n : nat.\forall m : nat.\forall p : nat.n * (m + p) = n * m + n * p,$$

as well as *cic:/Coq/Arith/Mult/mult_plus_distr_r.con* (a „fake match") with type

$$\forall n, m, p : nat.(n + m) * p = n * p + m * p,$$

which also proves to be useful, being a symmetric analog of distributivity.

Elim gives the induction axioms for a domain. It is used for inductive proofs of properties of functions and relations over algebraic types. For example, giving `nat` as input, we obtain

$$\forall P.(P\ 0) \Rightarrow (\forall n.(P\ n) \Rightarrow (P\ (n + 1))) \Rightarrow (\forall n.(P\ n)),$$

the usual induction priciple on natural numbers.

Also Semantic Web techniques custom-tailored for *mathematical content* [4, 6], are used in HELM. The difference between HELM and OpenMath-based platforms is in their different use of XML. Whereas HELM uses XML only for the management of libraries by proof assistants, OpenMath employs it for communication between applications.

For manipulating *mathematics on the web*, an interactive prover called Matita [2] and the mathematical web service called MONET[27] were implemented in the same framework. The associated project MOWGLI[28] has developed system- (and reasoner-) independent tools for presenting mathematical information on the Internet.

HR invents concepts and theorems in mathematical domains like finite algebras/groups/rings, graph theory and number theory [30]. The underlying idea is the formation of new conjectures, which are then passed to a resolution theorem prover (usually Otter[29]): If the proof goes through, the conjecture is adopted as a theorem; otherwise, a model finder (often MACE[30]) is called for drawing counterexamples from it.

Each concept is an operation or a relation on the carrier set of a certain model of interest (e.g. a fixed finite group), represented by a combination

[27] MONET, `http://monet.nag.co.uk/cocoon/monet/index.html`

[28] MOWGLI, `http://mowgli.cs.unibo.it/`

[29] Otter, `http://www-unix.mcs.anl.gov/AR/otter/`

[30] MACE, `http://www.cs.unm.edu/~mccune/mace4/`

of a "data table" (also called Cayley table in the case of groups) and a formula with an indication of its free variables (the "arguments" of the operation or relation). For example, the concept of multiplication in $\mathbb{Z}_6$ is described by a table containing rows like $[4, 2, 2]$ and $[3, 3, 5]$ together with the formula $[n, a, b]\colon a|n \wedge b|n \wedge a \times b = n$.

Concept formation works as follows:

- *Initial concepts* are given axiomatically (like the multiplication in a group) or by an explicit definition (like divisibility in number theory). The corresponding data tables are either supplied by the user as key examples or generated by MACE.
- *Production rules* [24] are applied to the existing concepts. For example, universal quantification (with respect to the first argument, say) produces an $(n-1)$-ary predicate from an n-ary one.
- *Measures of interestingness* are used for ranking the concepts on a heuristic basis [29], for instance "parsimony" by measuring the relative size of its data table. Note that these measures are recomputed at various stages in the concept/theory formation cycle.
- *Filtering* the eventually accepted concepts works by computing a total score computed from various measures of interestingness; these are subsequently used for creating conjectures and new concepts.

Examples of concepts invented in group theory include: Abelian group, cyclic group, order of elements, central elements.

Theory formation [30, 31] is built on top of concept formation. HR takes the newly obtained concept and forms conjectures about it:

- If the set of models is empty, a *non-existence conjecture* is made (i.e. the concept definition is inconsistent with the axioms). For instance, the concept of non-trivial idempotent elements in groups does not have a model, thus the conjecture $\nexists a(a \neq \mathrm{id} \wedge a * a = a)$ is generated.
- If the sets of models coincide for the newly created concept and an already existing one, an *equivalence conjecture* is made. In the case of idempotent elements, HR notices that for all group examples it has, the idempotent elements are the identity, thus generating the conjecture $a * a = a \leftrightarrow a = \mathrm{id}$.
- If neither of the above two cases applies, HR adds the concept to the theory and looks for *subsumption conjectures*. These are asserted about concepts whose set of models is contained in the set of models of the new concept. For each such case, HR conjectures that the definition of the new concept is implied by the definition of an old one.

The theories produced in this fashion will include concepts, theorems that relate concepts and proofs of these theorems.

Recent *improvements* of HR allow using Otter as a filter for newly obtained theorems [28], inputting initial concepts in Maple format, and substituting Otter and MACE by humans. HR is currently connected to the MathWeb software bus [69].

MATHsAID [50] is a system for *automated discovery* of "interesting" mathematical theorems. It automatically discovers and proves theorems (lemmas, corollaries, etc.) from a given set of axioms and definitions (supplied by the user). Its main components are the automatic hypothesis generator, the theorem generator and the theorem filter [50].

Given the set of axioms and definitions, the *hypothesis generator*(HG) builds up:

- a finite sequence $\{H_i\}_{i=1..n}$ where H_i is a set of hypotheses and
- a selection of axioms and terms of interest corresponding to each H_i.

This allows the system to concentrate on local aspects of the theory, and thus (in the following components) to build theorems in layers, rather than build them all at once. The HG also looks at the existing axioms and theorems for finding converses. If such a converse can exist, it is passed over to the theorem generator that will attempt to prove it.

The *theorem generator* takes as input a set of hypotheses H_k, asserts them and applies a forward-chaining process, using the build-in first-order theorem prover and deriving all conclusions. Conclusions satisfying certain criteria are then asserted, the process starts over, until no more new conclusions can be found. The resulting conclusions are passed over to the theorem filter. The most important criteria a conclusion has to satisfy is that it should not be a conclusion of an axiom, the "trivial conclusion of a theorem or "trivially derived from the existing knowledge (axioms, theorems). A more detailed description of triviality is found in [50].

The *theorem filter* takes these conclusions, and runs them through a number of tests, like irredundancy (checking whether all hypothesis from H_k are needed to prove the conclusion) or simplicity (based on a heuristic detailed in [51]). All conclusions failing a test are eliminated. All remaining conclusions are coupled back with H_k, and recorded as theorems.

The system was recently extended towards automated discovery of inductive theorems [61].

MathWebSearch is an online *search engine* for mathematical expressions developed by the KWARK team at Bremen [44, 43]. It contains

- knowledge bases obtained by harvesting the internet for content representations of mathematical expressions (currently the OpenMath and the content MathML formats are supported),
- an indexing mechanism using substitution trees [38] that creates index terms for all mathematical expressions (i.e. for each `math` element in MathML and more or less for each `apply` element in OpenMath) and
- a query language for mathematical expressions, realized as a generic extension mechanism for XML-based representation formats.

The repositories of mathematical knowledge that are harvested include the ActiveMath repository mentioned above, the CONNEXIONS corpus[31] and the MBase system mentioned below. The mathematical expressions are interpreted by the indexing mechanism as first-order terms.

The *query language* is obtained by adding new attributes and tags to the XML-based languages. For example, the attribute `mq:generic` is used for matching any subterm in the index, the attribute `mq:anyorder` for specifying an arbitrary order of the parameters of the current node. The user can input the desired search query in OpenMath or MathML

[31] http://www.rhaptos.org

format. In the former case, one may also take advantage of the WIRIS[32] OpenMath Editor (a user-friendly Java plug-in for editing mathematical expressions).

The standard *term retrieval algorithm* for substitution trees [37] is used as a search algorithm. The system can find an expression (occurring as a subexpression in any statement) up to α-conversion by adding the `mq:generic` attribute to every bound variable in the search query. Searches in informal text are not possible.

A related OMDoc tool also developed by the KWARK team is a *semantic wiki* (a wiki with Semantic Web Capabilities) called SWiM [46].

MBase [33] is a structured and distributed repository of mathematical knowledge [41]. It is implemented as a *mathematical service* of the MathWeb mathematical software bus, which is in turn built on top of OMDoc. MathWeb connects a wide range of mathematical services (including interfaces to Omega, Bliksem, Otter, SPASS, RDL, Mace, λ-Clam, Maple, CoCoA).

The online demo of MBase contains several preloaded theories. These are imported from the OMDoc libraries of Omega and TPS, and from OpenMath content dictionaries in OMDoc form. The demo allows browsing through the theories and *retrieval facilities* like

- textual search on all mathematical elements available in the knowledge base (called QuickSearch),
- textual search restricted to names of theories, axioms, symbols, definitions and theorems,
- pattern search for terms and formulae. The pattern matching mechanism (applicative higher-order) is inherited from Oz, the underlying programming language.

As announced in [41], there are plans to improve the system by adding inference procedures like higher-order matching and (logic-internal) structuring mechanisms like theory morphisms.

MIZAR strives to transform mathematical ideas into journal articles that are readable both by humans and machines, warranting consistency of the definitions and correctness of proofs [62]. The mathematical objects in Mizar are theorems, definitions and schemes (i.e. theorems with second-order variables). The main components of the system are the Mizar Verifier and the *Mizar Mathematical Library* (MML).

The Mizar Verifier takes a *Mizar article* (consisting of definitions, theorems/schemes and proofs, written according to Mizar syntax) as input and checks the file for logical errors. It is based on classical first-order logic with the added capability of forming second-order schemes.

MML is the largest library of *formalized mathematics* available in the world at this time. It aims at the reconstruction of the core of mathematics. Built on the axioms of Tarski-Grotendieck set theory, its current version includes 942 articles, written by 182 authors, comprising 42694 theorems, 7957 definitions, 728 schemes, 6942 registrations, 5879 symbols, and 1903 keywords. MML articles contain mathematical theorems like Birkhoff's Variety Theorem, the Fundamental Theorem of Algebra,

[32] `http://www.wiris.com/content/view/20/3/lang,en/`
[33] `http://mbase.mathweb.org:8080/mbase/`

the Fundamental Theorem of Arithmetic, Gödel's Completeness Theorem, the Jordan Curve Theorem, the Reflection Theorem, Rolle's Theorem.

MMLQuery [9] is a fast *query language* for MML. A set of indices (which is their version of metadata) is extracted from the library, and the user can retrieve a theorem by combining various filters on these indices. Some examples are:

- Getting the names and links to all articles containing `NAT` in their identifier:

  `list of articles |grep -i NAT`

- Obtaining all theorems referring to the mode `GROUP_2:mode 1`:

  `list of th from (GROUP_2 : mode 1  article);`

- Items which are neither notations nor registrations:

  `list of item | [ not notat and not reg];`

- Finding all items refering to all constructors from the theorem labelled `FUNCT_1:70`:

  `at least * ( FUNCT_1:th 70 ref )`

Most of Mizar Matches (MoMM) is another search tool for mathematical databases, optimized for Mizar [63]. It is based on Stephan Schulz's prover E. Its main goal is to eliminate redundant formulae. The tool translates parts of the MML into its own clausal-like format and eliminates the ones that can be subsumed by the others, using a kind of typed subsumption.

The *Journal of Formalized Mathematics*[34] gathers the articles accepted in MML. It is a compromise between a formal-language and natural-language postprocessor. It has been published since 1989 and is now available on the web. MML articles can be exported to Prolog-based formats for theorem proving via the MPTP system [64]. Importing facilities for parts of Mizar are provided by ISAR and HOL.

TH∃OREM∀ strives to support the whole cycle of mathematical activity [18], going along the "creativity spiral"[22]:

- creating new notions and *conjectures*,
- turning the conjectures into *theorems* by computer-supported verification (using automated proofs as much as possible),
- extracting *algorithms* (or at least the algorithmic parts) from proofs,
- running the algorithms on suitable input data for producing relevant new *facts*, which may in turn lead to new conjectures.

It is an interactive system based on Mathematica[35] (inheriting all its type-setting facilities e.g two dimensional syntax). It is an environment for writing papers and doing proofs/computations at the same time. One can apply various simplifiers, solvers, and provers from the predefined libraries.

For supporting theory exploration, several reasoners [21, 19] and organizational tools have been developed. The most recent of these organisational tools is *Label Management*[56, 57]. It is used for managing composite, hierarchical labels for mathematical knowledge. It also allows simple search facilities:

[34] `http://www.mizar.org/JFM`
[35] `http://www.wolfram.com`

- Search by formula label. For instance, looking for all formulae that have "associative" as a postfix for their label looks like

  $$\texttt{LabelLookup}[\text{"}\texttt{*associative}\text{"}, \texttt{"BN:Tuples"}].$$

- Applicative higher-order pattern matching. For example, if we want to search all the formulae in the theory "BN:Tuples" that use the concatenation operation, $\asymp$, on the left-hand side of the equal sign we can express this as

  $$\texttt{FormulaLookup}[\,\_ \asymp \_ \; = \; \_\,,\ \{\texttt{"BN:Tuples"}\}].$$

  This will yield solutions like $\forall_{\overline{x},\overline{y}}(\langle\overline{x}\rangle \asymp \langle\overline{y}\rangle = \langle\overline{x},\overline{y}\rangle])$.

- Variants of textual search (searching after the constants in a formula). For instance, calling

  $$\texttt{SymbolLookup}[\{\texttt{is-empty}, \smile\}, \{\texttt{"BN:Tuples"}\}, \texttt{FilterType} \rightarrow \texttt{"All"}]$$

  returns all formulae in which the constant symbols `is-empty` and $\smile$ occur.

A methodology for systematic theory exploration was introduced in [13]. The SysteMathEx subproject carries out case studies of systematic explorations of mathematical domains (e.g. the theory of natural numbers [32]).

4 Conclusion

In the six years that have passed since MKM was initiated in the 2001 conference, one of its main threads was concerned with the *representation* of mathematical knowledge for communicating it between systems or for presentation on the web. For searching in such knowledge repositories textual searches turned out to be insufficient, as already noted in [23]: more and more systems evolve through pattern matching and unification towards simple instances of proving.

Another main thread is the question of *integrating* theorem provers and computer algebra systems, as promoted and pursued in the Calculemus community. Also older repositories of mathematical knowledge like Mizar are being incorporated into MKM systems. Such systems should provide an integrated environment for storing, accessing and using mathematical knowledge. While this is partially realized for special-purpose fields and math education, an overall tool for the working mathematician is yet to be developed.

The vision of such systems is not far-fetched, however. There are efforts for standardizing the representation of mathematical expressions on the web, for facilitating a distributed comunication of mathematical information between systems with moderately different logical frameworks and different languages. The difficulties of *searching* mathematical information become clearer and more explicit.

There are still several *challenges* that MKM has to overcome. The communication between groups working in MKM is made difficult by differencies of terminology (e.g. terms like "semantics" and "retrieval" have different meanings in different groups). The repositories of formalized mathematics (still) cover very little from the vast domains of mathematics.

Acknowledgements

I thank Prof. Tudor Jebelean, Prof. Bruno Buchberger and Florina Piroi for very fruitful discussions and comments.

References

[1] C. Sacerdoti Coen A. Asperti, L. Padovani and I. Schena. An Hypertextual Electronic Library of Mathematics. INRIA, Roquencourt, 06 2000.

[2] E. Tassi A. Asperti, C. Sacerdoti Coen and S. Zacchiroli. User Interaction with the Matita Proof Assistant. *Journal of Automated Reasoning, Special Issue on User Interfaces for Theorem Proving*, 39/2:109 – 139, August 2007.

[3] Milton Abramowitz and Irene A. Stegun, editors. *Handbook of Mathematical Functions*. Dover, New York, 1965.

[4] A. Asperti, F. Guidi, C. Sacerdoti Coen, E. Tassi, and S. Zacchiroli. A Content Based Mathematical Search Engine: Whelp. In *Types for Proofs and Programs International Workshop (TYPES 2004)*, volume 3839 of *Lecture Notes in Computer Science (LNCS)*, pages 17–32, 2006.

[5] A. Asperti and M.Selmi. Efficient Retrieval of Mathematical Statements. In *Proceedings of the Third International Conference of MKM*, volume LNCS 31119, 2004.

[6] A. Asperti, L. Padovani, C. Sacerdoti Coen, and I. Schena. Content-centric Logical Environments. Short presentation at LICS'2000, June 26-28 2000. Santa Barbara, California, USA.

[7] A. Asperti and S.Zacchiroli. Searching Mathematics on the Web: State of the Art and Future Developments. In *New Developments in Electronic Publishing*, 2004.

[8] Grzegorz Bancerek and Michael Kohlhase. *From Insight to Proof. Festschrift in honour of A. Trybulec*, chapter Towards a Mizar Mathematical Library in OMDoc Format, pages 265–277. University of Bialystok,10 (23), 2007.

[9] Grzegorz Bancerek and Piotr Rudnicki. Information Retrieval in MML. In A. Asperti, B. Buchberger, and J.H. Davenport, editors, *Proceedings of the Second International Conference on Mathematical Knowledge Management*, volume 2594 of *LNCS*, pages 119–132. Springer, 2003.

[10] Rebhi Baraka. *A Framework for Publishing and Discovering Mathematical Web Services*. PhD thesis, RISC, August 2006. PhD Thesis.

[11] Y. Bertot and P.Casteran. Coq'Art. Available online at `http://www.labri.fr/Perso/~casteran/CoqArt` (last check on 18.12.07).

[12] B. Buchberger. Theory Exploration Versus Theorem Proving. In *Proceedings of the Calculemus '99 Workshop*, volume 23-3 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 1999.

[13] B. Buchberger. Theory Exploration with Theorema. *Analele Universitatii Din Timisoara, Seria Matematica-Informatica*, XXXVIII(2):9–32, 2000. Selected papers of the 2nd International

Workshop on Symbolic and Numeric Algorithms in Scientific Computing, Oct. 4-6, 2000, Timisoara, Romania (T. Jebelean, V. Negru, A. Popovici eds.).

[14] B. Buchberger. Mathematical Knowledge Management in TH∃OREM∀. In B. Buchberger and O. Caprotti, editors, *First International Workshop on Mathematical Knowledge Management (MKM 2001)*, pages –, September 24-26 2001.

[15] B. Buchberger. Algorithm Retrieval: Concept Clarification and Case Study in Theorema. SFB report 44, Research Institute for Symbolic Computation, Johannes Kepler Univ ersity, October 2003.

[16] B. Buchberger. Algorithm Synthesis by Lazy Thinking: Examples and Implementation in Theorema. In *Proc. of the Mathematical Knowledge Management Workshop*, volume 93 of *Electronic Notes in Theoretical Computer Science*, pages 24–59, 25 November 2003.

[17] B. Buchberger. Algorithm Supported Mathematical Theory Exploration: A Personal View and Stragegy. In B. Buchberger and John Campbell, editors, *Proceedings of AISC 2004 (7 th International Conference on Artificial Intelligence and Symbolic Computation)*, volume 3249 of *Springer Lecture Notes in Artificial Intelligence*, pages 236–250. Copyright: Springer, Berlin-Heidelberg, 22-24 September 2004.

[18] B. Buchberger, A. Craciun, T. Jebelean, L. Kovacs, T. Kutsia, K. Nakagawa, F. Piroi, N. Popov, J. Robu, M. Rosenkranz, and W. Windsteiger. Theorema: Towards Computer-Aided Mathematical Theory Exploration. *Journal of Applied Logic*, pages –, 2005. To appear.

[19] B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The Theorema Project: A Progress Report. In M. Kerber and M. Kohlhase, editors, *Symbolic Computation and Automated Reasoning (Proceedings of CALCULEMUS 2000, Symposium on the Integration of Symbolic Computation and Mechanized Reasoning)*, pages 98–113. St. Andrews, Scotland, Copyright: A.K. Peters, Natick, Massachusetts, 6-7 August 2000.

[20] B. Buchberger, G. Gonnet, and M. Hazewinkel, editors. *Mathematical Knowledge Management*, volume 38 of *Special Issue of Annals of Mathematics and Artificial Intelligence*. Kluwer Academic Publisher, Dordrecht, Netherlands, May 2003.

[21] B. Buchberger, T. Jebelean, F. Kriftner, M. Marin, E. Tomuta, and D. Vasaru. A Survey of the Theorema project. In W. Kuechlin, editor, *Proceedings of ISSAC'97 (International Symposium on Symbolic and Algebraic Computation, Maui, Hawaii, July 21-23, 1997)*, pages 384–391. ACM Press, 1997.

[22] Bruno Buchberger. Mathematica: A System for Doing Mathematics by Computer? Technical Report 93-50, RISC Report Series, University of Linz, Austria, September 1993. Invited Talk at DISCO'93, Gmunden, Austria, September.

[23] Bruno Buchberger. Theorema and Mathematical Knowledge Management, 2002.

[24] A. Bundy, S. Colton, and T. Walsh. HR - Automated concept formation in finite algebras. In *Proceedings of the Machine Discovery Workshop, ECAI 98, Brighton.*, 1998. Also Research Report RP920, Division of Informatics, University of Edinburgh.

[25] Stephen Buswell, Olga Caprotti, and Mike Dewar. Mathematical Service Description Language:Final Version. Monet Deliverables, March 2003.

[26] O. Caprotti, J.H. Davenport, M. Dewar, and J. Padget. Mathematics on the (Semantic) NET. In *Proceedings of ESWS 2004, First European Semantic Web Symposium*, number 3053 in LNCS, pages 213–224, 2004.

[27] C. Sarcerdoti Coen. *Knowledge Management of Formal Mathematics and Interactive Proving*. PhD thesis, University of Bologna, 2004.

[28] S. Colton. Making Conjectures about Maple Functions, 2002. appeared later in JSC!

[29] S. Colton, A. Bundy, and T. Walsh. On the Notion of Interestingness in Automated Mathematical Discovery. *IJHCS: International Journal of Human-Computer Studies*, 53, 2000.

[30] Simon Colton, Roy McCasland, Alan Bundy, and Toby Walsh. Automated Theory Formation for Tutoring Tasks in Pure Mathematics, 2002.

[31] Simon Colton, Pedro Torres, Paul Cairns, and Volker Sorge. Managing Automatically Formed Mathematical Theories.

[32] A. Craciun and M. Hodorog. Decompositions of Natural Numbers: From A Case Study in Mathematical Theory Exploration. In D. Petcu, V. Negru, D. Zaharie, and T. Jebelean, editors, *Proceedings of the 9th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC07)*, pages 1–8, 26-29 September 2007. to appear as IEEE volume.

[33] Lus Cruz-Filipe, Herman Geuvers, and Freek Wiedijk. C-CoRN, the Constructive Coq Repository at Nijmegen.

[34] N.G. de Bruijn. The mathematical vernacular, a language for mathematics with typed sets. In P. Dybjer et al., editor, *Proceedings of the workshop on Programming Languages*, 1987. Marstrand, Sweden.

[35] W.M. Farmer. MKM: A New Interdisciplinary Field of Research. *SIGSAM Bulletin*, 38(2):47–52, June 2004.

[36] E. Gimenez and P. Casteran. A Tutorial on Recursive Types in Coq.

[37] Peter Graf. Substitution Tree Indexing. Technical Report MPI-I-94-251, Saarbruecken, 1994.

[38] Peter Graf. *Term indexing*. Number 1053 in LNCS. Springer Verlag, 1996.

[39] F. Guidi and C.S. Coen. Querying Distributed Digital Libraries of Mathematics. Calculemus 2003, 2003.

[40] M. Kauers, M. Kerber, R. Miner, and W. Windsteiger, editors. *Towards Mechanized Mathematical Assistants. Proceedings of the 14th Symposium, Calculemus 2007, 6th International Conference, MKM 2007*, volume 4573 of *LNCS*. Hagenberg, Austria, Springer, June 27-30 2007.

[41] M. Kohlhase and H. Franke. MBase: Representing Knowledge and Content for the Integration of Mathematical Software Systems. *Journal of Symbolic Computation*, 32(4):365 – 402, 2001.

[42] Michael Kohlhase. *OMDoc - An Open Markup Format for Mathematical Documents [version 1.2]*, volume 4180 of *Lecture Notes in Computer Science*. Springer, 2006.

[43] Michael Kohlhase and Ioan Şucan. A Search Engine for Mathematical Formulae. In Tetsuo Ida, Jacques Calmet, and Dongming Wang, editors, *Proceedings of Artificial Intelligence and Symbolic Computation, AISC'2006*, number 4120 in LNAI, pages 241–253. Springer Verlag, 2006.

[44] Michael Kohlhase and Ioan Şucan. System Description: MathWebSearch 0.3: A Semantic Search Engine. preprint, 2007.

[45] Temur Kutsia and Koji Nakagawa. An Interface between Theorema and External Automated Deduction Systems. In Steve Linton and Roberto Sebastiani, editors, *Proceedings of 9th Symposium on the Integration of Symbolic Computation and Mechanized Reasoning (Calculemus'01)*, pages 178–182, Siena, Italy, Jun 21–23 2001.

[46] C. Lange. SWim - Technical Report.

[47] C. E. Larson. An Updated Survey of Research in Automated Mathematical Conjecture-Making.

[48] C.E. Larson. A Survey of Research in Automated Mathematical Conjecture-Making. Proceedings of the DIMACS Workshop on Automated Discovery in Chemistry and Mathematics, 2005.

[49] Paul Libbrecht and Erika Melis. Methods for Access and Retrieval of Mathematical Content in ActiveMath. In Nobuki Takayama, Andres Iglesias, and Jaime Gutierrez, editors, *Proceedings of ICMS-2006*, number 4151 in LNCS. Springer Verlag GmbH, sep 2006.

[50] Roy L. McCasland and Alan Bundy. MATHsAiD: A Mathematical Theorem Discovery Tool. In *Proceedings of the Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06)*, pages 17–22. IEEE Computer Society Press, 2006.

[51] Roy L. McCasland, Alan Bundy, and Patrick F. Smith. Ascertaining Mathematical Theorems. *Electronic Notes in Theoretical Computer Science*, 151(1):21–38, March 2006.

[52] Jrg M.Haake, Ulrich Lucke, and Djamshid Tavangarian, editors. *Interactive Concept Mapping in ActiveMath (iCMap)*, volume LNI, volume 66, pages 247–258, Rostock Germany, September 2005.

[53] B. Miller and A. Youssef. Technical Aspects of the Digital Library of Mathematical Functions. *Ann. Math. Artificial Intelligence*, 2002.

[54] Robert Miner and Paul Topping. Math on the web status report. Design Science, Inc., 2003. `http://www.dessci.com/en/reference/webmath/status/`.

[55] Mathematical Knowledge Management Network (MKMNet). Survey of Existing Tools for Formal MKM. Deliverable 4.1.

[56] F. Piroi and B. Buchberger. An Environment for Building Mathematical Knowledge Libraries. In Wolfgang Windsteiger and Christoph Benzmueller, editors, *Proceedings of the Workshop on Computer-Supported Mathematical Theory Development, Second International Joint Conference (IJCAR)*, pages 19–29, Cork, Ireland, 4-8 July 2004.

[57] F. Piroi and B. Buchberger. Label Management in Theorema. In M. Kohlhase, editor, *Informal proceedings of the 4th International Conference on Mathematical Knowledge Management*, pages –, July 2005. Bremen, Germany.

[58] F. Piroi, B. Buchberger, C. Rosenkranz, and T. Jebelean. Organisational Tools for MKM in Theorema. Technical Report 07-11, RISC Report Series, University of Linz, Austria, 2007.

[59] Virgile Prevosto. Certified Mathematical Hierarchies: the FoCal System. In *Mathematics, Algorithms, Proofs*, 2005.

[60] R.Anghelache. Hermes - a reliable conversion from tex to mathml. In *New Developments in Electronic Publishing of Mathematics (4-th European Congress of Mathematics)*, Stockholm, Sweden, June 25 - 27 2004.

[61] Alan Bundy Roy McCasland and Serge Autexier. *From Insight to Proof: Festschrift in Honor of A. Trybulec*, chapter Automated discovery of inductive theorems. University of Bialystok, 2007.

[62] Piotr Rudnicki and Andrzej Trybulec. Mathematical Knowledge Management in Mizar. In B. Buchberger and O. Caprotti, editors, *Proceedings of MKM01*, 2001.

[63] J. Urban. MOMM - Fast Interreduction and Retrieval in Large Libraries of Formalized Mathematics. *International Journal on Artificial Intelligence Tools*, 2005.

[64] Josef Urban. MPTP - Motivation, Implementation, First Experiments. *Journal of Automated Reasoning*, 2005.

[65] F. Wiedijk. The mathematical vernacular. `http://www.cs.kun.nl/~freek/notes/mv.ps.gz`, 2000. Unpublished paper.

[66] A. Youssef. Advanced Math Search: Issues and Techniques. IMA, 2004.

[67] Abdou Youssef. Search Systems for Math Equations. invited talk at the IMA Workshop on "Special Functions in the Digital Age" , University of Minnesota, Minneapolis, July 2002.

[68] Jürgen Zimmer and Serge Autexier. The MathServe System for Semantic Web Reasoning Services. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR'06)*, volume 4130 of *LNAI*, pages 140–144, Seattle, USA, August 2006. Springer Verlag.

[69] Jürgen Zimmer, Andreas Franke, Simon Colton, and Geoff Sutcliffe. Integrating HR and tptp2X into MathWeb to Compare Automated Theorem Provers. In *Proceedings of the CADE'02 Workshop on Problems and Problem sets*, 2002.