

Deliverable 3.3:
Comparison of semantic bases of formal tools

Mathematical Knowledge Management Network
MKMnet, IST-2001-37057

Summary

The present document is the result of **Task 3.3: *Comparison of semantic bases of formal tools***, of the **Work Package 3: Varied Presentations of Mathematics and Varied User Needs**.

The purpose of this task is to compare the different approaches to the semantic representation of mathematical knowledge and to try to unify them in a common semantic basis of primitive notions.

The sections have been compiled by various partners in the project, using information from other groups when necessary.

The complete report has been collected and integrated by Fairouz Kamareddine, Heriot-Watt University, Edinburgh.

1 Developments and Comparison of Semantic Representations

Clearly, a single representation of mathematical knowledge is not sufficient. Tools that give access to mathematical knowledge need to be adapted to the needs of different types of user. An integral part of any tool giving access to mathematical knowledge bases must translate into the language/conventions of the user whilst providing access to the different types and levels of knowledge available. However, in order to ensure consistency and unambiguous communication, the tools will need to work from a common semantic basis for the mathematical knowledge. A lot of work has already been done, for example by designers of mathematical provers¹ to identify basic concepts at the root of their own systems. The third objective of this work package is to compare the different approaches and to try to unify them in a common semantic basis of primitive notions. This work can be seen as an extension of the OMDoc project to provide ways of working with ordinary document structures (chapters, sections, etc.). In this document we report on the work carried out by members of the network on this task. In particular, we describe in the following sections, comparison studies and semantic extensions achieved under this task.

1.1 Some Statistical Considerations on the COQ library [10]

Among the many interesting opportunities offered by a content, machine-understandable encoding of mathematical information, one of the most fascinating is the possibility to perform a systematic, automated investigation of the concrete structure of mathematical notions and developments. Summarizing the problem in a provocative way, logic teaches us what a theorem is, but what is a *good* theorem? Is there any hope to answer to this question in a completely internal way, by the study of the structure and dependencies of mathematical entities and developments? We are of course very far from this goal, but the mathematical and statistical investigation of mathematics appears in any case as a natural and major underpin for Mathematical Knowledge Management (how can we manage an information we know so little about?)

Currently, the biggest problem in this direction is still the lack of large repositories of machine understandable mathematical information. The only, modest exception is provided by the libraries of tools for the mechanizations

¹see Automath, Mizar, Theorema, FOC, Impls, etc.

of mathematics and automation of formal reasoning. These libraries are typically of limited dimension, just covering basic, elementary fields of mathematics; moreover one could argue that a formal development may have in any case a sensibly different structure and nature than a mathematically rigorous but informal one² (again, how can we answer to this doubt without a statistical investigation?)

On the other side, a statistical investigation of these libraries may have in this case a direct and beneficial impact on the tools themselves, pointing out possible idiosyncrasies of the systems or suggesting improved techniques for indexing, retrieving, and possibly automatically applying theorems.

The work is a preliminary report on some statistical investigations performed by the HELM group in Bologna on the library of the Coq Proof Assistant, developed at INRIA, France. The concrete outcome of the work is the validation of a restricted set of metadata [10] for indexing and retrieving statements in large repositories of content based mathematical knowledge.

1.2 Assertion level proof representation with under-specification [1]

One aspect of mathematical knowledge management that is now in its first stage of development, is tutorial computer assistance for teaching students how to prove mathematical theorems. The paper argues that under specification is the main feature which arises in the context of user-input in tutoring systems and has to be resolved in the representation format before the proof is shipped to an Automated Theorem Prover (ATP). The paper goes on to describe ongoing work, within the DIALOG project, in the representation of under-specified proofs at the assertion level arising from human interaction with a tutoring system. Proofs are analysed in terms of granularity, style, detail of specification and a proof representation format is described that is rich enough to enable automated processing. The paper is based on the experiments carried out by the group at Saarlands within one of their projects: the DIALOG project. The authors concentrate on the design of a proof language and of proof checking rules meant to support a dialog for proof management –the dialog being performed between student and machine. Because of the desired human-orientation, the presented proofs are originally in the form of assertions based on natural language. Moreover, there is natural occurrence of under-specification, since students will only mention some of the arguments and/or references that they use, omitting others (e.g., the

²Also, does each foundational system induce its own, sensibly different development of traditional fields of mathematics?

observations which are obvious in their opinion, or instantiations of context variables in definitions, lemmas and theorems).

1.3 Algorithm Synthesis by Lazy Thinking: Examples and Implementation in Theorema [3]

The authors of this paper demonstrate their idea for automatically developing algorithms from formal specifications using 3 detailed case studies. The idea is nice: Have a database of general algorithm schemes (essentially either a concrete algorithm or the abstract top level of an algorithm without lower level details, e.g. the divide and conquer strategy), choose one to start the process and use a theorem prover to try to prove that the scheme. When the prover fails (as it must because the lower levels of the algorithm don't exist yet, analyse what the prover was trying to prove about the missing elements when it failed in order to create a formal specification for them. Then the process is applied recursively. If the process completes successfully, then the lowest level specifications are short and simple enough that, presumably, concrete algorithms are available in the database from which the final algorithm is constructed. The lazy thinking method consists in, firstly, describing data by properties (in the spirit of abstract data types), secondly, describing by a formula and auxiliary names the specification of an algorithm manipulating this data. The names are those of other algorithms, introduced to simplify the expression of the property. Thus, the idea is very reminiscent of abstract data types. Then, an attempt, in a backward manner, to prove that the algorithm satisfies some properties leads to some hints on the introduced names. These hints are considered as a part of a specification of these names. The approach is bottom-up, with synthesis of the needed lemmata and is fully mechanized. Induction principles on data are managed by the tool.

1.4 Adapting Mainstream Editors for Semantic Authoring of Mathematics [4]

This article deals with the idea of adding structure to a text by using the definitions, lemmas, proofs, and such other markers within the OMDoc framework. This is work in progress and needed to add more structure and eventually more meaning and hence semantic to a text.

1.5 Logic and Type Theory in Theorem Provers [12]

Logic is an important (though not the only) way to add semantic meaning to a text. In this article, the author Seldin compares two way of representing logic in type theory: one is to use propositions-as-types and the other, called here the Frege representation, is to encode a proposition as an object of type `Bool`, `A` being provable iff `A = True:Bool`. In particular, the paper deals with the problem of representing the HOL theorem prover in the Calculus of Constructions. The interest lies in the fact that HOL contains features that would make the Calculus of Constructions proof irrelevant if added to the latter. Proof irrelevance is a highly undesirable property if we intend to have a propositions-as-types representation of logics, since it implies `True = False`. To overcome the problems raised by a propositions-as-types representation, the author proposes to use the Frege style representation. There are other interesting aspects to the paper, like the version of the Diaconescu's theorem and the discussions about the Frege style representation.

1.6 MathLang: Experience-driven new mathematical language [9]

In this paper we report on the design of a new mathematical language driven by the encoding of mathematical texts. MathLang should provide support for checking basic well-formedness of mathematical text without requiring the heavy and difficult-to-use machinery of full type theory or other forms of full formalization. At the same time, it should allow the addition of fuller formalization to a document as time and effort permits. MathLang should, ultimately, be useful in providing better software support for authoring mathematics, reading mathematics, and organizing and distributing mathematics. The language presented in this paper is intended only for machine manipulation and for debugging of the design of MathLang. MathLang attempts to and partially succeeds at representing the structure of mathematical texts as designed by the mathematician himself. The design of MathLang aims at allowing a smooth passage from the mathematician's text to the symbolised representation of it and thereafter to filling in the logical structure as well as fully machine checking the text.

1.7 A semantic basis for presentation on the Web

A semantic basis for presentation on the Web is provided by OMDoc and its extensions. The ActiveMath-Group at the DFKI has developed a structure and RDF-schemata for mathematics in the Mowgli project. Moreover, sev-

eral extensions of OMDoc, in particular structures and metadata which are necessary for mathematics education have been developed by this group [11]. In a nationally funded project (MMISS) the group extended the structure also for documents on secure software and software verification [2]. Notably, a new structure/encoding for exercises has been presented in [5].

Those XML-encodings include mathematical formulae and symbols as OpenMath or MathML-content. The latter has been added just recently. From this encoding the course generator and a presentation engine produce user-adaptive Web-presentations. One big problem is that most authors are still used to focus on presentation and therefore have problems to author purely semantic content. One task for the near future will be to investigate how presentation information can be added modularly to OMDoc without corrupting the semantical basis and still allowing for different (adapted) presentation and re-usability of content.

2 Stimulating and disseminating research on semantic basis

Research on this task was disseminated in the following publication venues:

- The articles [1, 3, 4, 9, 12] described above.
- A special issue [7] just being completed in the journal of Applied Logic which contains articles dealing with this area (e.g., see the description of the article of Seldin [12] below).
- A special issue [8] in the Electronic Notes series of Theoretical Computer Science which contains articles dealing with this area (e.g., see the description of the article of Autexier etal [1] below).
- The Mathematical Knowledge Management Symposium held at Heriot-Watt in Edinburgh, Scotland at which each various talks on all deliverables were presented. See <http://www.macs.hw.ac.uk/~fairouz/mkm-symposium03/>

References

- [1] Autexier, Benzmueller, Fiedler, Horacek and Vo. Assertion level proof representation with under-specification. In [8].

- [2] B Krieg-Brückner and D. Hutter and Ch. Lüth and E. Melis and A. Poetsch-Heffter and M. Roggenbach and J-G . Schmaus and M. Wirsing. Towards MultiMedia Instruction in Safe and Secure Systems. WADT-2003.
- [3] Bruno Buchberger and Adrian Craciun. Algorithm Synthesis by Lazy Thinking: Examples and Implementation in Theorema. In [8].
- [4] G. Goguadze and A. Gonzalez Palomo. Adapting Mainstream Editors for Semantic Authoring of Mathematics. Talk presented at the MKM symposium [13].
- [5] G. Goguadze and E. Melis and C. Ullrich and P. Cairns. Problems and Solutions for Markup for Mathematical Examples and Exercises. In *International Conference on Mathematical Knowledge Management, MKM03*, A. Asperti and B. Buchberger and J.H. Davenport (editors), LNCS 2594, Pages 80-93, 2003.
- [6] Guidi, F., Sacerdoti Coen, C., “Querying Distributed Digital Libraries of Mathematics” In *Calculus 2003*, Aracne Editrice S.R.L., Therese Hardin & Renaud Rioboo editors, ISBN 88-7999-545-6, pp. 43–57.
- [7] F. Kamareddine (editor). Special issue on Logics and Examples for Automation. Special issue in the *Journal of Applied Logic*, Elsevier , North-Holland. To appear. 2004.
- [8] F. Kamareddine (editor). *Mathematical Knowledge Management*, Electronic Notes in Theoretical Computer Science, MKM-symposium03. ENTCS, Elsevier. To appear. 2004.
- [9] F. Kamareddine, M. Maarek, and J.B. Wells. MathLang: Experience-driven new mathematical language. In [13]
- [10] A.Asperti, F.Guidi, L.Padovani, C.Sacerdoti Coen. “The Science of Equality. Some statistical considerations on the COQ library”. Draft, University of Bologna, 2003.
- [11] E. Melis and G. Goguadse, P. Libbrecht, C. Ullrich. Wissensmodellierung und -nutzung in activemath. The Journal of KI, number 1, pages 12-18, 2003.
- [12] J. Seldin. Logic and Type Theory in Theorem Provers. In [7].
- [13] Mathematical Knowledge Management Symposium. <http://www.macs.hw.ac.uk/~fairouz/mkm-symposium03/>