

Multilingual Access to Mathematical Exercise Problems

Andreas Strotmann¹, Wanjiku Ng'ang'a², and Olga Caprotti³

¹ University of Cologne, ZAIK/RRZK

² University of Helsinki, Dept. of Linguistics

³ University of Helsinki, Dept. of Mathematics

Abstract. The Web Advanced Learning Technologies (WebALT) project, financed through the European Union's *eContent* programme, is working to provide pan-European (and eventually world-wide) multilingual and multicultural internet access to a repository of algorithmically generated exercises for students and teachers of mathematics at the secondary and tertiary education levels, building as much as possible on existing frameworks, standards, and software. The two-year WebALT project has reached its quarter mark now, and we can now report early results.

We are working on a framework in which a large percentage of undergraduate and highschool mathematics exercises can be created in the language independent form of content markup in a way that captures both the meaning of the simple sentences and the formulas embedded in them that together make up a math problem. Such content, expressed in OpenMath because it is more easily extended with the extra concepts required here, is then localized using language-specific content-to-presentation markup stylesheets for the embedded formulae, and natural language generation techniques for rendering the embedding sentences that tell the students what to do with those formulae in the language of their choice.

1 Introduction

In recent months, something quite rare happened: mathematical research results were making headlines around the world – or rather, results in mathematics education research. These results showed, not entirely surprisingly, that practice makes perfect: students need to do many exercises in order to comprehend the mathematical ideas they exemplify. Perhaps more surprisingly, they showed that even highly abstract mathematics loses its terror when students have enough opportunity to practice it.

However, practice makes perfect only when it comes with feedback, and grading exercises is a major and immensely time consuming chore for mathematics teachers who struggle to provide that crucial ingredient of the learning process to their students. Teachers are themselves becoming a more and more “endangered” resource, while their teaching is becoming more and more important to the knowledge-based society.

An obvious solution, therefore, is to support both the students of mathematics and their teachers by providing extra opportunities for doing exercises with immediate feedback, and to do so without increasing the teacher's workload significantly, by using the computer both to pose problems and evaluate solutions.

Internet Accessible Distance Learning in Mathematics – including online assessment with automatic grading – is a well-served market and has been available for many years now, but it is being served mostly in a few of the languages with the most speakers – and all too many minority students struggle with a language not their own when the teaching is not done in their native language.

1.1 State of the Art

Online Learning Environments like Blackboard,⁴ Ilias,⁵ or WebCT⁶ have been under development for more than a decade, and have reached considerable maturity by now, providing almost all of the basic features that students and teachers need to interact in a distance learning situation. This includes, in particular, on-line assessment, including automatically graded self-assessment.

In the Internet Accessible Mathematical Computing community we are today able to use computer algebra systems to automatically grade students' answers to quite complex mathematical problems. With systems like MapleTA,⁷ generating members of an open-ended class of exercise problems to pose to students is becoming possible, as is automatically providing intelligent feedback to these students.

With MathML,⁸ a standard has become available that allows us to deliver such problems to any student anywhere on the Web via a standard web browser. Yet, we still cannot, in a truly universal fashion, provide such a service to people across the globe in a form that is suitable to their own languages and cultures, even though all the important ingredients are available, in principle:

- a language-independent content markup, a truly universal language for storing mathematical content, and
- the ability to present formulae encoded in content markup to the user [6] – even audio rendering is already possible [11].

1.2 Multilingual Access

The caveat today is that we can do this only for users who are familiar with the Anglo-Saxon way of expressing mathematics and/or who speak English. In order to offer mathematical exercise material in a truly universal manner, the computer algebra techniques and webservice technologies which have been developed by

⁴ www.blackboard.com

⁵ www.ilias.uni-koeln.de

⁶ www.webct.com

⁷ www.maplesoft.com/products/mapleta/

⁸ www.w3.org/Math/

the IAMC community need to be augmented with natural language processing capabilities to obtain universal accessibility for the contents [7].

Two European Union funded R&D projects are currently exploring two separate aspects of language processing capabilities in online mathematics education:

- LeActiveMath⁹ (“language enhanced” ActiveMath) is working on modeling the interaction between an intelligent math tutoring system and the student in a multi-lingual fashion,
- WebALT¹⁰ (Web Advanced Learning Technologies [5, ?]) concentrates on modeling the exercises that form the core of these interactions in a language-independent and localizable way.

Both projects cooperate closely.

This paper concentrates on one of the tasks of the WebALT project: enabling truly universal authoring and delivery of high-quality mathematical content, to be show-cased by an internet accessible repository of mathematical exercise problems.

2 Putting the Pieces Together the WebALT Way

The WebALT project is developing a framework for creating a database of undergraduate and highschool mathematics exercises that are authored and stored in a language independent form and presented to the student in a form that is suitable to the language and culture that the student is in.

To do this, WebALT uses a language-independent content markup that captures the meaning of an exercise – both the meaning of the simple sentences and that of the formulas embedded in them that together make up a typical math problem. This mathematical content is expressed using OpenMath¹¹ [1, 4, 3], extended by Content Dictionaries needed for the extra concepts required to handle the natural language sentence fragments in a cross-linguistic manner.

The OpenMath-encoded mathematical content is to be embedded in XML markup that provides a standard interface to distance learning environments – SCORM,¹² MathQTI,¹³ or MathDox[8] are among the candidate emerging XML standards we are considering. These provide the web interface for exercises in general, and they are typically available localized to a student’s locale by the distance learning environment vendor.

In the WebALT framework, when an exercise problem is presented to a student, the OpenMath markup is rendered into that typical mix of natural language and mathematical formula that characterizes most of mathematics. Natural language generation technology produces the sentences that tell the students what

⁹ www.leactivemath.org

¹⁰ www.webalt.net

¹¹ www.openmath.org

¹² www.adlnet.org/scorm/

¹³ www.maths.ed.ac.uk/mathqti/

to do with the formulae embedded in them. For the embedded formulae, locale-dependent content-to-presentation markup stylesheets are to be used in order to take into account differences in notational and typographical conventions in different locales.

Together, these two interwoven techniques (content-markup to natural language and to presentation markup rendering) are surprisingly powerful – we have succeeded in generating even an exercise problem such as

Calculate $f'(1)$, where f is the inverse of the polynomial $x^2 + 1$.

with a rather complicated syntactical structure where formal mathematics is expressed in natural language text, from an appropriately decorated OpenMath object representing:

```
(lambda [f]. (diff(f))(1)) (inverse(lambda [x]. plus(power(x,2),1)))
```

Just as for English, the technique allows us to generate equivalent text in a number of other languages and cultural contexts.

In addition, the linguistic part of this technology is general enough to allow us to create a fully expanded natural language version from mathematical content markup, which presumably makes it ideal for creating content suitable for screen readers, thus enhancing the accessibility of this kind of mathematical computing on the Internet considerably, e.g. for the visually impaired.

There are a number of advantages to this approach to the problem of making mathematical computations more “accessible” over the internet (in the sense of making it possible to reach a wider audience over the internet). First of all, by using a semantically rich representation of the mathematical content, it supports actual computations rather than just static text with embedded static formulae, with the added ability to provide a huge number of algorithmically generated exercises with automated feedback. Second, by relying on natural language technology, renderings of such content in different cultural or linguistic contexts are guaranteed to be equivalent, thus paving the way for delivering standardized assessment tests independently of the language. Third, almost all of the components needed for this approach to work are available today, including a good natural language generation solution that is both well suited to this type of compositional transformation and supports most of the languages targeted at this early stage of the project.

3 Language-Independent Markup of Mathematics

Internet accessibility of mathematical content is largely dependent on the possibility of representing formulae in a language independent format that is very close to the abstract symbolic formalism used in computational backends. This abstract language however needs to be presented to the reader using specific notation and typographical conventions that often require language-dependent customization (e.g. the symbols for the greatest common divisor or least common multiple are localized to abbreviations of these terms in the local languages)

[7]. Having noticed this duality, markup languages for mathematics, such as OpenMath or MathML, make a distinction between content versus presentation markup. While content markup allows to create formulae in a language independent, abstract manner, presentation markup provides a framework for creating language and culture dependent presentations of the mathematical content.

3.1 Mixing Natural Language Context with Formal Mathematics

Mathematical markup languages such as OpenMath and MathML focus on the representation of mathematical objects and as such do not support markup for specifying the various contexts in which mathematics is used. Because of this, a variety of markup languages has been designed to cater for specific purposes: for example, OMDoc [9] and MathDox [8] to capture mathematical documents and libraries of mathematical software, MSDL [2] to describe mathematical web services, MathQTI to describe question and tests in mathematics.

All of these allow authors to mix natural language text with mathematics markup, effectively ending up with content that is not anymore language independent because the text is written in one particular language. To exemplify, the markup for an exercise would say something like "solve $\langle\text{math}\rangle$ " , "integrate $\langle\text{math}\rangle$ " typically using verbs in English like *solve* or *integrate* or their equivalent in some other language in order to give instructions to the students.

The reason for this is explained by an overly simplified example problem:

Integrate x .

The closest MathML-Content representation is, for example:

```
<apply>
  <int/>
  <bvar><ci>x</ci></bvar>
  <ci>x</ci>
</apply>
```

which reads as the noun phrase *the integral of x wrt. x* , while the natural language expression gives a command that specifies an action to be taken by the reader using a verb phrase in imperative mood. MathML is thus missing the ability to mark the crucial difference between noun and verb phrases and between different sentence moods.

3.2 Representing Both Text and Formula Using Content Markup

On the other hand, the natural language expression above also misses crucial information: it fails to specify explicitly the variable of integration that is obligatory in Content-MathML in this case, and necessary for automatic grading, say. However, MathML-Presentation has the concept of "invisible" rendering of mathematical content, and this concept clearly generalizes to this particular instance of rendering "invisibly" the phrase *with respect to x* .

The WebALT approach is different: the mathematical markup itself is decorated with hints on how to verbalize the content. These hints are given in an abstract, cross-linguistic way so that a natural language generation tool is able to render the mathematical object in a variety of European (and eventually other) languages using the appropriate jargon for mathematical text. Our approach works well with exercise texts which consist of short sentences.

4 Techniques for Localizing Mathematical Exercises

4.1 Categories of Mathematical Exercise Problems

A survey that we conducted on an existing database of several hundred undergraduate Calculus exercises suggested a classification of the types of these exercises from the natural language processing perspective into “short” versus “long” problems [7]. The category of “short” problems, which comprises the overwhelming majority of problems, is characterized by very simple sentences with embedded formulas, e.g. *Integrate <some formula>* or *What is the derivative of <another formula> with respect to <a variable>?*.

The much smaller category of “long” problems subdivided linguistically into a subcategory of “explorations” and a subcategory of “story problems”. The first of these, “explorations”, consisted of several steps, each step linguistically of the same general form as a “short” problem: *The function f has zeroes 0, 2, and 5. Draw f and f' (and so on).*

The second subcategory, “story problems”, was characterized by complex verbalizations with comparatively few or small embedded formulae, typically involving some references to real-world applications instead of purely abstract mathematics.

As it turns out, the overwhelming majority of mathematical exercise problems is therefore linguistically quite simple and regular in structure, and it is quite natural to try computational linguistics techniques to exploit this regularity in order to localize the exercises to a wide variety of languages and cultures from a single source.

4.2 The Software Engineering Approach

The LeActiveMath project has been investigating a method for handling multi-lingual variants of a mathematical exercise using traditional software engineering techniques. In this approach, the text fragments that are part of a mathematical exercise are translated by hand, and the presentation markup version of mathematical formulae embedded in the text may also need to be adjusted to notational differences in different cultures.

Handling several parallel versions of a mathematical exercise in this manner presents its own challenges, but from our perspective, the biggest drawback is the need to translate manually.

4.3 The Babelfish Approach

A first approach to automatic translation of mathematical exercises that might come to mind to a non-linguist would be to follow the lead of, say, Google's translation service¹⁴ or Altavista's Babel Fish.¹⁵ As in the previous case, problems would be authored in English, say, and automatic translation systems would be trained to handle the mathematics-exercises fragment of English to translate them into many other languages.

Unfortunately, there are insurmountable problems with this approach: minor or even major translation errors garble the results of such a translation. In the context of mathematical exercises, however, even minor glitches in translation are unacceptable since they may distort the intended meaning of the exercise. The student has to be certain that the automatically translated version of the exercise is faithful to what was intended by the teacher. This is especially important when grading of problems is fully automated since there is no way to catch this kind of linguistic mishap.

4.4 Disambiguated Natural Language Text

Such linguistic mishaps are almost always due to a fundamental problem that computational linguistics needs to handle: sentence structures and words in natural language are highly ambiguous, a problem that in computational mathematics we are well aware of with mathematical notations [7].

To avoid such ambiguities, more sophisticated linguistic techniques require an unambiguous representation of the target content. For mathematical formulas, for example, this usually means using some form of content markup instead of presentation markup. For natural language text, it is possible to add markup that disambiguates sentence structures or the meaning of a word or phrase. Such disambiguated text is then more easily parsed and transformed into the syntactic and lexical requirements of a target language.

This technique works the better, the more closely related the languages involved in the translation process are to each other. Since one language pair that the WebALT project is aiming at is Finnish and English, two languages with no known genetic relationship to each other, this approach has an a priori problem, and given that Europe, our target market, has many languages from several language families – and within each family, from several groups within that family – it is not clear that it is possible to retain sufficient quality when translating between them in this way.

In addition, no tools are available that would do this for an interesting set of languages, and writing translators from scratch for a significant set of languages is a formidable task, although a familiar one to computational linguists.

¹⁴ www.google.com/language_tools

¹⁵ world.altavista.com

4.5 Natural Language Generation from Content Markup

To a mathematician, another approach to creating disambiguated mathematical content presents itself readily.

Consider, for example, the task of rendering an existentially quantified predicate encoded in MathML Content in a way that is appropriate to, say, a middle school student. Quantifier symbols would be incomprehensible to such a student, but by expressing them in plain-text English, e.g. using a template such as *We can find a <something> that <predicate>*, even elementary school student could gain an intuitive understanding of the intended meaning of that quantifier.

Just as that natural-language phrase has some formal mathematical meaning, so do those phrases that accompany formulae in a typical mathematical exercise problem – or at least those that correspond to problems that we can both create and grade automatically using a computer algebra system.

As we saw before, the innovative approach to localizing educational content being taken by the WebALT project can be characterized as follows:

- Exercise problems are created or generated in a content markup form that represents the meanings of both the mathematical formula and the simple sentence it is embedded in (i.e. both the “ x ” and the “*Integrate...!*” in one of our sample exercises),
 - either created individually
 - or generated using a tool like MapleTA.
- Content markup is rendered to a presentation form in two ways:
 - formulae are rendered to appropriate presentation markup localized to the student’s requirements [10], and
 - embedding sentences are generated to take into account the student’s language environment.

Luckily, in recent years, linguists have been working on the kind of technology that would be required to bring this scenario to life. Aarne Ranta’s Grammatical Framework (GF) [12] allows linguists to specify the general features of the grammar, morphology, and fundamental parts of the vocabulary of a natural language, while (in our case) a mathematics teacher could extend it with the words and phrases that are peculiar to (again, in our showcase) simple exercises in this particular area of mathematics. In addition, GF is specifically geared towards a functional view of grammar, a view that meshes perfectly with the compositional nature of mathematical content markup [13, 14] to enable a completely compositional natural language generation process using mathematical content markup as input.

Perhaps even more importantly, the modular nature of GF is also geared to utilizing cross-linguistic generalizations when defining the general grammar of a language, with the result that an impressive range of European languages – our initial target market – is covered at a level that we can use immediately, and others well enough advanced that they can be added even within the fairly short development cycle available for this project. Initially, our target languages include English, Finnish, Spanish, and Catalan (the latter as a European minority

language), with support for Basque, Dutch, French, German, and Swedish being planned soon after. Of these, English, Finnish, French, German, Spanish, and Swedish are already available for GF, and thus Catalan and Dutch not hard to add.

5 Authoring Language-Independent Content

This leaves just one more problem: how to create the kind of content markup that will allow us to utilize these technologies to their full effect. Creating it by hand is out of the question: even within our group, which includes several OpenMath experts, only a small core is able to do this at this point, and the evolving need to mix linguistic and mathematical markup makes it particularly hard to find people who are skilled in both.

Nevertheless, we are proposing to offer the capability to create language-independent mathematical content to authors and teachers across the globe eventually, and across Europe very soon. Very few of these authors will know how to read or write either mathematical content markup or any linguistic markup we may need to add, much less both at the same time. Indeed, this is a special case of a problem that has been identified as one of the major stumbling blocks and, consequently, a major research area for the ambitious Mathematical Knowledge Management community:¹⁶ creating semantically sound content markup in an author-friendly manner.

Again, we are lucky that our showcase does not require a solution to this problem in its full generality, so that one of our partners has now demonstrated that it is possible to provide a reasonably user friendly solution, in the form of a generalization of the ideas behind a content-markup formula editor.¹⁷ The idea here is to support the creation of the formal mathematical content of the exercise problem as a mix of natural language text and mathematical formula by allowing the user to either type in one particular form of the problem or to use a palette to compose it. This form will be very restrictive in its linguistic coverage, both with respect to its vocabulary and with respect to the range of grammatical constructs covered: enough to unambiguously author a range of exercise problems, but with little regard for covering any linguistic niceties. Using a small number of stock phrases, chosen to cover the targetted area being taught as closely and simply as possible, a number of options become available for easing the chore of the content creator because real-time parsing and sanity-checking of partially constructed content becomes possible.

The specific linguistic form that an exercise problem then takes on the way to the reader can be chosen by the author in a separate toolbar: whether it is to take the form of a question or that of a command, which parts are to be rendered as formula and which parts as text, and so on. By providing this choice to the author, we can evade another feasibility trap for our project: essentially, the user manually performs detailed text and sentence planning for the exercise in this

¹⁶ www.mkm-ig.org

¹⁷ www.wiris.com/overview/products/wiris-editor.html

way, so that this particularly difficult problem of natural language generation from computer-generated data can be largely ignored in our showcase.

Our first example above nicely demonstrates this: by introducing the local variable f in the concrete form of the exercise, expressed in the form of a *where* construct in our example, the input content markup expression already provides the full semantic structure of a sentence that is arguably considerably easier to read and understand by a student than, say, a sentence that just says:

Calculate the derivative at 1 of the inverse of the function that maps x to $x^2 + 1$.

or than the purely formal exercise

Calculate $((x \mapsto x^2 + 1)^{-1})'(1)$.

6 Conclusions

By design, the WebALT project is being funded as a software development project and not a research project. The reason for this is that we believe that all the main ingredients are now available that we need to build a fully functional and immensely useful repository of exercise problems equally Internet accessible in any European language:

- the language-independent form of mathematical content markup combined with cross-language linguistic markup
- editors that enable the creation of mathematical content markup by non-expert users
- markup and protocol standards for integrating such a repository into distance learning and online assessment platforms
- multi-lingual compositional natural language generation technology
- mathematical content-to-presentation markup stylesheets
- algorithmic exercise problem generation systems that can output content markup
- systems that allow automatic grading of problems that are created in this way.

Very few extra ingredients need to be developed in addition to this:

- the concrete translation of the combined linguistic and mathematical markup to natural language
- the adaptation of content-to-presentation stylesheets to different language environments
- a generalization of the mathematical content markup editor to cover linguistic variations
- the code to integrate all the ingredients into a cohesive whole.

This is true only because we carefully chose our showcase in such a way that we can avoid a number of hard and as yet insufficiently resolved problems:

- Exercise problems are much simpler in their semantic structure than, say, full scale mathematical proofs, for which natural-language generation technology has yet to reach acceptable levels of quality and which are very hard to author in a fully formal fashion as required in our approach.
- Among the exercise problems, we focus on the vast majority of linguistically “simple” problems that are readily amenable to our approach; story problems, while feasible in principle using these techniques, would require considerable research to author and generate in this way.
- The rich compositional nature of the content markup that we store is able to mirror quite complex sentence and phrase structures that are commonly used to make exercise problems more easily comprehensible to students, so that it is possible to enable the author to pre-structure the problem in such a way that it will render appropriately, and the complex task of text and sentence planning or of structure recognition and encoding can largely be avoided.
- The nature of mathematics is such that purely abstract exercises can cover a vast majority of problems students will need to tackle to do those rote exercises that will make them comfortable enough with the material that they can then explore the material more deeply on their own.

In the particular showcase for the innovative approach to cross-lingual authoring and delivery of mathematical content that we develop in the WebALT project, we therefore aim to enable the creation of immensely useful content – exercise material that may be useful to many millions of students world wide – in a form that makes it accessible to almost all who may have a need for it, within the constraints of a short development time and a fairly small project on the theory that 90% of the task requires 10% of the work.

Nevertheless, we are also confident that the particular way that we have succeeded in combining existing technologies generalizes quite well to a considerably wider range of applications, especially as more and more of the hard problems that we mentioned above are being resolved through research and development.

Acknowledgments

This paper is based on joint work by the authors. We gratefully acknowledge contributions from the other members of the WebALT team, especially Lauri Carlson, Ramon Eixarch, Anni Laine, Daniel Saludes, and Mika Seppälä.

References

1. Abbott, J., van Leeuwen, A., Strotmann, A.: OpenMath: Communicating mathematical information between co-operating agents in a knowledge network. *J. of Intelligent Systems*, 1998 special issue: Improving the Design of Intelligent Systems: Outstanding problems and some methods for their solution **8** (1998)

2. Buswell, S., Caprotti, O., Dewar, M.: Mathematical Service Description Language: Final Version. Deliverable D14.IST-2001-34145, The MONET Consortium (2003). URL: monet.nag.co.uk/cocoon/monet/publicdocs/monet-msdl-final.pdf.
3. Buswell, S., et al.: The Open Math Standard, Version 2.0. The Open Math Society (2004).
4. Caprotti, O., Carlisle, D., Cohen, A. (eds.): The OpenMath standard – version 1.0. Technical report, Esprit Project OpenMath (2000). URL: www.nag.co.uk/projects/OpenMath/omstd/.
5. Caprotti, O., et al.: Web Advanced Learning Technologies for Assessment in Mathematics. Recent Research Developments in Learning Technologies (2005).
6. Carlisle, D.: MathML on the Web: Using XSLT to Enable Cross-Platform Support for XHTML and MathML in Current Browsers. In A. Diaz et al. (eds.): MathML International Conference (2002). URL: www.mathmlconference.org/2002/presentations.html.
7. Carlson, L., Saludes, J., Strotmann, A.: State of the Art in Multilingual and Multicultural Creation of Digital Mathematical Content. Deliverable 1.2.WebALT-EDC-22253 (2005). URL: webalt.math.helsinki.fi/content/results/docs/
8. H. Cuypers, A. Cohen, and E.R. Barreiro: MathDox: Mathematical Documents on the Web. URL: www.win.tue.nl/hansc/mathdox.pdf.
9. Kohlhase, M.: OMDoc: An Open Markup Format for Mathematical Documents (Version 1.2). (2005).
10. Manzoor, S., et al.: Authoring Presentation for OpenMath. In: Proceedings of the Fourth International Conference on Mathematical Knowledge Management, Bremen (2005).
11. Raman, T.V.: Audio System for Technical Readings. Doctoral dissertation, Cornell University (1994). URL: www.cs.cornell.edu/home/raman.
12. Ranta, A.: Grammatical Framework. Journal of Functional Programming Volume 14 , Issue 2 (March 2004) , 145 - 189.
13. Strotmann, A., Kohout, L.J.: OpenMath: Compositionality achieved at last. ACM SIGSAM Bulletin Volume 34, Special Issue 2 "OpenMath" (2000).
14. Strotmann, A.: Content Markup Language Design Principles. Dissertation, The Florida State University, Tallahassee, Florida (2003). URL: etd.lib.fsu.edu/theses/available/etd-09042003-181524/.
15. Strotmann, A., Seppälä, M.: Web Advanced Learning Technologies for Multilingual Mathematics Teaching Support. In: Milena Dobрева and Jan Engelen (eds.): From Author to Reader: Challenges for the Digital Content Chain: Proceedings of the 9th ICCO International Conference on Electronic Publishing held at Katholieke Universiteit Leuven in Leuven-Heverlee (Belgium), 8-10 June 2005 (2005).