Improving Accessibility to Mathematical Formulas: The Wikipedia Math Accessor

José Fuentes Sepúlveda
Department of Computer Science
University of Concepción
Concepción, Chile
Email: jfuentess@udec.cl

***Abstract*—Mathematics accessibility is an important topic for inclusive education. In this work, we make Wikipedia's repository of mathematical formulas accessible by providing a natural language description of its more than 420,000 formulas using a well-researched sub-language. We also contribute by targeting Spanish speakers, for whom assistive technologies, particularly domain-specific technologies like the one described here, are scarce. Our focus on the *semantics* of formulas (rather than their visual appearance) allowed us to generate verbalizations with a precision of approximately 80% of understandable descriptions, as shown in an evaluation with sighted users.**

***Keywords*-Accessibility (Blind and Visually Impaired); Natural language Interaction; Mathematical formulas**

I. INTRODUCTION

Wikipedia is the largest free encyclopedia, featuring over 18 million articles altogether, 3.7 million in English and over 830,000 articles in Spanish. Wikipedia is mostly text-based, and accessing its *textual* information through screen-readers is, relatively speaking (relative to PDFs, for instance), quite straightforward. However, Wikipedia also features images, sound and other multimedia content, that are, perhaps not surprisingly, much less accessible to certain populations such as blind and visually-impaired or deaf people.

One problematic content type, particularly for the blind community, is that of mathematical formulas. In Wikipedia, there are (in November of 2011) a total of 426,431 mathematical formulas spread over 29,374 Wikipedia articles. Unfortunately, mathematical formulas in Wikipedia are inaccessible, in part because they are displayed using rasterized images (`png` files) of LaTeX expressions embedded in the `<alt>` attribute of an `<img>` HTML tag, where for, for example,

$$E = mc^2 \qquad (1)$$

```
<img class="tex"
 alt="E = mc^2"
 src="name.png" />
```

This paper introduces to MATHACC, an Assistive Technology (AT) designed to help visually impaired people gain access to graphical representation of mathematical formulas published in Wikipedia, using their own screen-readers. Our contribution is threefold: 1) a study to identify templates for verbalization, based on the semantics of mathematical formulas; 2) the implementation of a prototype system and 3) a preliminary evaluation of this system and the generated descriptions with "blind-folded" sighted users.

II. RELATED WORK

Recently, researchers have become very interested in the problem of accessibility to mathematical formulas by the blind community, and excellent quality research has been produced [1], [2]. Among them there is the LAMBDA project [3], funded by the European Union in the Information Society Technologies. The system consists of a markup language to represent mathematical expressions, not unlike, for instance, *MathML*. This is an excellent choice for certain contexts, but today's enormous availability of and reliance on digital information makes the use of Braille an expensive and/or slow technology.

AsTeR, in turn, is a system that helps produce renderized audio of electronic documents [4]. This system takes a LaTeX document as input and parses it, generating a tree structure representation, mapping the tree's nodes onto the audible portion of the node contents that allow the sound presentation of mathematical formulas. *AsTeR* also allows navigation of the formulas, exploring the expression as a tree-based structure. There is no knowledge, however, about how this system performs on a massive dataset of formulas.

Stanley and Karshmer introduce *MathGenie* [5], a system that reads out mathematical formulas, together with their *Nemeth* code [6]. *MathGenie* was specially designed for students in the sciences who had some visual impairment. The shortcoming with this system lies in that it receives *presentation MathML* as input, focusing mostly on how formulas are presented, rather than what their semantics is.

MathPlayer is a plug-in for Microsoft Internet Explorer [7], [8] that was designed, primarily, for rendering a visualization of *MathML*. It now allows users to interact with mathematical expressions through two navigational modes: one based on text and another based on a tree representation of the formula. Unfortunately, its use is restricted only to platforms that support Internet Explorer.

In [9], Reddy and Gupta proposed a different approach: they report on a method for translating *MathML* into *voiceXML*[1]. In this way, a *voiceXML* interpreter will render the documents generating audio. Nevertheless, pure use of *presentation MathML* in translation is less reliable than the use of *content MathML* to exploit the semantics of the formulas.

Finally, Pontelli and Abu Doush [10] propose a framework that allows navigation of mathematical expressions using two modalities: one based on linear access, and the other based on hierarchical access, using a tree as the underlying data structure. In their system, the implementation is a plug-in for the FireFox web browser, on top of *FireVox*[2], a FireFox screen-reader system. This system also takes *presentation MathML* as input. Besides, *FireVox* ceased to be updated in 2008.

[1] http://www.voicexml.org/
[2] http://www.firevox.clcworld.net/

None of these papers, however, have attempted to, given some semantics (as encoded in *content MathML*), provide linguistic descriptions to a *massive* repository of formulas such as the ones in Wikipedia; nor have they done a thorough investigation on the *language* used to talk about formulas. Notice as well, that none of these systems actually work with the Spanish language natively. These are our contributions.

III. SYSTEM DESCRIPTION

A. MATHACC *architecture*

The system consists of three sub-systems: one devoted to the detection of mathematical formulas in Wikipedia pages, together with a process that translates from LaTeX to a more structured, less ambiguous representation written in *MathML* (specifically *content MathML*), a template-based natural language generation system like the one in [11], and finally a module that restructures the Wikipedia page to incorporate the linguistic description of the formula. In what follows, we detail the workings of these different modules.

1) Detecting and cleaning mathematical expressions: This first module of the system is in charge of capturing, parsing and cleaning the HTML code to retrieve all mathematical expressions in it. In Wikipedia articles, mathematical expressions are shown as raster images, using the `<img>` tag, with the attribute `<class="tex">`, while the attribute `<alt>` contains the LaTeX definition, as shown in the introduction (see Equation 1). Note that MATHACC does not do image processing itself, but rather works with the semantically undetermined LaTeX string that Wikipedia exposes in the source HTML.

Once the LaTeX formulas were curated, the next step was to generate a more structured, less ambiguous representation. LaTeX is semantically ambiguous, in the sense that not all information about a formula is explicit in its representation. To clean the *semantics* of the formulas, we chose to translate them into the content markup of *MathML*, see Figure 1 for an example. Mathematical Markup Language (*MathML*) is an application of XML for describing mathematical notations and capturing both its structure and content. To do this, we used *SnuggleTex*, a Java library that does part of the translation between LaTeX and *MathML* automatically[3].

```
<math>                <math>
  <apply>               <apply>
   <inverse/>             <power/>
    <ci> f </ci>            <ci> f </ci>
  </apply>                  <cn> -1 </cn>
</math>                   </apply>
                        </math>
```

Figure 1. Example of *content MathML*. The left XML snippet belongs to the inverse function, while the one on the right corresponds to the "variable f" to the power of -1.

2) The Sub-Language of the Generation Module: Perhaps the largest contribution of this paper is the design of the language to be used with each *MathML* operator. We present a template-based natural language generation (NLG) system [12], [13], [14], [15], and thus each *MathML* operator will be associated with a template. To find out what the best template was for each operator, we carried out a simple experiment that consisted in having participants look at a formula, and provide the best natural language description they could think of. In the experiment, we showed formulas formed by 21 operators, that correspond to the most used operators in Wikipedia. For example, the template obtained of the formula $\frac{x+1}{x-1}$ was ``$OP1$ dividido por $OP2$'' and the template obtained of $\sqrt[n]{a}$ was ``raíz $GRADO$ de OP'' (for more detail see [16]).

3) The Language Generation Module: The backbone of the NLG system used in MATHACC has been successfully used in other applications, most notably in IGRAPH-LITE [17], [11], [18]. The version we are working on here, however, has a new engine that allows for stack-based generation that is vital given the nesting-based nature of mathematical expressions. In what follows, we give a detailed run of the system to explain its workings. To do this, we use, allegedly, one of the most beautiful (and universal) equations of all time: $E = mc^2$.

The whole stack trace of the generation system is shown in Figure 2. Once the curated *MathML* expression is input into the generation module, the XML snippet is parsed and the tree data structure we obtain is then traversed in depth-first manner, adding *MathML* elements to a stack. Every time the `apply` node is found in the tree, an `apply` node is added to the stack, signaling the start of a new operator. Likewise, when a `ci` or `cn` node is found on the tree, only their children are added to the stack (see `m` on line 10 in Figure 2). In all other situations, the node is added to the stack without any modifications.

```
1.  []
2.  ['apply']
3.  ['apply', 'eq']
4.  ['apply', 'eq']
5.  ['apply', 'eq', 'E']
6.  ['apply', 'eq', 'E']
7.  ['apply', 'eq', 'E', 'apply']
8.  ['apply', 'eq', 'E', 'apply', 'times']
9.  ['apply', 'eq', 'E', 'apply', 'times']
10. ['apply', 'eq', 'E', 'apply', 'times', 'm']
11. ['apply', 'eq', 'E', 'apply', 'times', 'm']
12. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply']
13. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power']
14. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power']
15. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power', 'c']
16. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power', 'c']
17. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power', 'c', '2']
18. ['apply', 'eq', 'E', 'apply', 'times', 'm', 'apply', 'power', 'c', '2']
19. ['apply', 'eq', 'E', 'apply', 'times', 'm', ' (pausa) c elevado a 2 (pausa) ']
20. ['apply', 'eq', 'E', ' (pausa) m por  (pausa) c elevado a 2 (pausa)']
21. ['(pausa) E igual a (pausa) m por  (pausa) c elevado a 2 (pausa)']
```

Figure 2. Stacktrace of the NLG system for $E = mc^2$.

Every time the XML `apply` sub-tree is completely traversed (which means all the operands are now known, together with the operand itself), we start "popping" the stack until the first `apply` node to be found. The popped nodes of the stack are "pushed" into another temporary stack to be input into the subsystem that applies the templates. Once the temporary stack has been built, the templates are applied by "popping" the top of the temporary stack, and searching a template dictionary for the appropriate one to use. For instance, if the temporary stack is `['2', 'c', 'power']`, the dictionary search yields the binary operator template for `power`: "VAR elevado a VAR", where each of `2` and `c` are assigned, in "first out" order, to each "VAR". The system finally adds a

[3]http://www2.ph.ed.ac.uk/snuggletex

(pausa) string at the extremes of the verbalization. Pauses are added as markers to separate semantically self-contained units in the verbalization to help text-to-speech grouping (see [4]). The process is repeated for each apply until the stack is empty.

Although it is not a part of the generator itself, the next step is to modify the Wikipedia page adding the verb attribute with the verbalization within the <img> tag that contains the processed LaTeX snippet.

B. User's view

Although the MATHACC system has been designed, primarily, with the blind computer user in mind, the interaction with the system may be analyzed from two different points of view: the publisher of mathematical content, and the consumer of that content. In the case of the content producer, it is only necessary to add the LaTeX codification of a certain mathematical expression to the alt attribute to the <img> tag. As we will see, the system will be in charge of generating a natural language description of it.

In the case of the content seeker, and screen-reader user, the only requirement is to make the interface read the supplied verb attribute of math formulas in Wikipedia LaTeX.

IV. EVALUATION

A. Study 1: Coverage of MATHACC

Before the usability studies can be carried out, there was the need to know how comprehensive the generation stage was. In other words, it was important to see whether too few formulas from Wikipedia were generated, or whether the generation was wrong. To get a feeling of just how MATHACC was performing, we tested the generation engine on a subset of 100, 500, and 1000 random formulas from Wikipedia.

On a first pass, we counted the number of times the generator could not assign a description to a formula. For the sets of 100, 500 and 1000 test formulas, the generator provided a linguistic description, for 66%, 65.5% and 66.1%, respectively. This is without any fine tuning, just taking the LaTeX formula, translating it into *MathML* and finally going through the generator. The reasons for the failure to generate were varied but they all had to do with how *SnuggleTex* would convert from LaTeX to *MathML*.

On a second pass, for the 66 formulas that were given a description in the 100 random formula file, we counted the number of times the description provided was wrong. Since the percentages were so close, we hypothesize that the other test cases (500 and 1000 formula files) will be similar. Of the 66 formulas that were given a description, 6 (˜10%) were wrong, reaching 60% of correct verbalizations . This is not a bad result considering that there has been no optimization done to the translation between LaTeX to *MathML*.

B. Study 2: Verbalization understandability

To test if the generation engine worked correctly, 20 engineers and engineering students, between 20 and 30 years of age, participated in the study. We created two web pages, each with fifteen formulas extracted from Wikipedia. Each formula was coded directly in *content MathML* to eliminate the constraints inherent in the translation between LaTeX and *MathML*. The formulas were built using the 21 most used operators of Wikipedia, and three others were added: factorial (!), tangent (tan) and subset ($\subset$), to allow for more diversity. The formulas that made up the first web page were used to evaluate prosodic indicators (only pauses). On the other hand, the second web page was used to evaluate lexical indicators. For more information about prosodic and lexical indicators see [1].

Formulas were hidden from participants using Cascading Style Sheets (or CSS), to recreate the lack of sight as factor, which might influence the results. Each web page was verbalized with MATHACC and then read to the participants through *WebAnywhere* [19]. Participants had to listen to the verbalization of the formulas and write on paper the expressions that they believed they had heard, in mathematical notation. They could hear and navigate the expressions as many times as was necessary. Below, the responses of the participants are contrasted against the stimuli expressions present in the web page. Only matches were counted as correct answers.

The results show 76% of effectiveness to prosodic indicators and 79.3% of effectiveness to lexical indicators. The participants also had the opportunity to comment at the end of the evaluation, which highlighted two issues: (1) verbalized expressions using prosodic indicators were easier to transcribe correctly, because the verbalization was more similar to how people read expressions. However, participants made more mistakes, caused mainly by the difficulty of capturing the nesting property of some expressions and (2) verbalization with lexical indicators led to an increased cognitive load. However, it gave participants more confidence in their answer.

V. GENERAL DISCUSSION, CONCLUSIONS AND FUTURE WORK

We have demonstrated that it is possible to provide rich linguistic verbalizations to *sui generis* mathematical formulas in Wikipedia, in the Spanish language, and using *content MathML* as the input to the generator (previous curation of the Wikipedia LaTeX conventions). Using the semantics of mathematical formulas, through *content MathML*, results in less ambiguous verbalizations, compared with verbalizations based on the syntax of mathematical formulas. Furthermore, appropriate use of the indicators, whether prosodic, lexical or extralinguistic, together with verbalizations based on semantics of mathematical formulas, helps to generate unambiguous descriptions.

A few things can be improved: there is obviously the need for a better LaTeX$\rightarrow$*content MathML* translator, considering the excellent contributions by [20] or taking advantage of the context of Wikipedia's pages to disambiguate some mathematical formulas. In order to avoid the translation being a source of mistakes, it would be necessary for mathematical formulas of Wikipedia to be encoded in *content MathML*.

It would also be of interest to go deeper into the semantics of formulas, and provide a context for variables: thus, if the expression is $F = mg$, to preprocess the formula, taking into account the context it appears in, to contain the fact that m denotes mass and g denotes gravity, while F is force.

The use of indicators is still naive, making some verbalizations ambiguous. For this reason, a more rigorous study about the state of the art in the use prosodic, lexical and extralinguistic indicators, and the combination of them, should be done.

REFERENCES

[1] A. Karshmer, G. Gupta, and E. Pontelli, "Mathematics and accessibility: A survey," University of Texas at Dallas., Tech. Rep., 2007, available from: http://www.utdallas.edu/~gupta/mathaccsurvey.pdf.

[2] C. Jayant, "A survey of math accessibility for blind persons and an investigation on text/math separation," 2006, available from: http://www.cs.washington.edu/homes/cjayant/papers /MathAccessFinal.pdf.

[3] W. Schweikhardt, C. Bernareggi, N. Jessel, B. Encelle, and M. Gut, "Lambda: A european system to access mathematics with braille and audio synthesis," in *Computers Helping People with Special Needs*, ser. Lecture Notes in Computer Science, K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer, Eds. Springer Berlin / Heidelberg, 2006, vol. 4061, pp. 1223–1230, 10.1007/11788713_176. [Online]. Available: http://dx.doi.org/10.1007/11788713_176

[4] T. V. Raman, "Audio system for technical readings," Ithaca, NY, USA., Tech. Rep., 1994, available from: http://www.cs.cornell.edu/home/raman/aster/aster-thesis.ps.

[5] P. Stanley and A. Karshmer, "Translating mathml into nemeth braille code," in *Computers Helping People with Special Needs*, ser. Lecture Notes in Computer Science, K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer, Eds., vol. 4061. Springer Berlin / Heidelberg, 2006, pp. 1175–1182, 10.1007/11788713_170. [Online]. Available: http://dx.doi.org/10.1007/11788713_170

[6] A. Nemeth, *The Nemeth Braille code for mathematics and science notation: 1972 revision*. Produced in braille for the Library of Congress, National Library Service for the Blind and Physically Handicapped by the American Printing House for the Blind., 1972.

[7] N. Soiffer, "Mathplayer: web-based math accessibility," in *In Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility*. ACM Press, 2005, pp. 204–205.

[8] ——, "Mathplayer v2.1: web-based math accessibility," in *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, ser. Assets '07. New York, NY, USA: ACM, 2007, pp. 257–258. [Online]. Available: http://doi.acm.org/10.1145/1296843.1296900

[9] H. Reddy and G. Gupta, "Dynamic aural browsing of mathml documents with voicexml," in *Human-computer interaction*. Lawrence Erlbaum and Associates, 2005.

[10] I. Abu Doush and E. Pontelli, "Building a programmable architecture for non-visual navigation of mathematics: Using rules for guiding presentation and switching between modalities," in *UAHCI '09: Proceedings of the 5th International Conference on Universal Access in Human-Computer Interaction. Part III*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 3–13.

[11] L. Ferres, P. Verkhogliad, G. Lindgaard, L. Boucher, A. Chretien, and M. Lachance, "Improving accessibility to statistical graphs: the igraph-lite system," in *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, ser. Assets '07. New York, NY, USA: ACM, 2007, pp. 67–74. [Online]. Available: http://doi.acm.org/10.1145/1296843.1296857

[12] E. Reiter, "Nlg vs. templates," in *In Proceedings of the Fifth European Workshop on Natural Language Generation*, May 1995, pp. 95–106.

[13] G. Wilcock, "Pipelines, templates and transformations: Xml for natural language generation," in *Proceedings of the 1st NLP and XML Workshop*, 2001, pp. 1–8.

[14] K. Van Deemter, E. Krahmer, and M. Theune, "Real versus template-based natural language generation: A false opposition?" *Comput. Linguist.*, vol. 31, pp. 15–24, March 2005. [Online]. Available: http://dx.doi.org/10.1162/0891201053630291

[15] S. Siprada and F. Gao, "Summarizing dive computer data: A case study in integrating textual and graphical presentations of numerical data," in *Proceedings of Workshop on Multimodal Output Generation*, vol. volume CTIT Proceedings of the Workshop on Multimodal Output Generation, 2007, pp. 149–157.

[16] L. Ferres and J. Fuentes Sepúlveda, "Improving accessibility to mathematical formulas: the wikipedia math accessor," in *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*, ser. W4A '11. New York, NY, USA: ACM, 2011, pp. 25:1–25:9. [Online]. Available: http://doi.acm.org/10.1145/1969289.1969322

[17] L. Ferres, A. Parush, S. Roberts, and G. Lindgaard, "Helping people with visual impairments gain access to graphical information through natural language: The igraph system," in *Computers Helping People with Special Needs*, ser. Lecture Notes in Computer Science, K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer, Eds. Springer Berlin / Heidelberg, 2006, vol. 4061, pp. 1122–1130, 10.1007/11788713_163. [Online]. Available: http://dx.doi.org/10.1007/11788713_163

[18] L. Ferres, G. Lindgaard, and L. Sumegi, "Evaluating a tool for improving accessibility to charts and graphs," in *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, ser. ASSETS '10. New York, NY, USA: ACM, 2010, pp. 83–90. [Online]. Available: http://doi.acm.org/10.1145/1878803.1878820

[19] J. P. Bigham, C. M. Prince, and R. E. Ladner, "Webanywhere: a screen reader on-the-go," in *Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)*, ser. W4A '08. New York, NY, USA: ACM, 2008, pp. 73–82. [Online]. Available: http://doi.acm.org/10.1145/1368044.1368060

[20] E. Pontelli and B. Palmer, "Translating between formats for mathematics: Current approach and an agenda for future developments," in *Computers Helping People with Special Needs*, ser. Lecture Notes in Computer Science, K. Miesenberger, J. Klaus, W. Zagler, and D. Burger, Eds. Springer Berlin / Heidelberg, 2004, vol. 3118, pp. 627–627, 10.1007/978-3-540-27817-7_92. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-27817-7_92