New Accessibility Features in MathJax

Cervone, Davide; Krautzberger, Peter; Sorge, Volker

New Accessibility Features in MathJax

Davide Cervone,

MathJax Consortium & Union College, NY

dpvc@union.edu

Peter Krautzberger,

MathJax Consortium & krautzource, UG, Germany

peter.krautzberger@mathjax.org

Volker Sorge

MathJax Consortium & University of Birmingham, UK

v.sorge@mathjax.org

Abstract

Recent changes in the landscape for assistive technology solutions for Mathematics on the web have prompted the development of MathJax into a single rendering and accessibility solution. We present our current efforts that depend on a novel semantic interpretation of Presentation MathML expressions. This allows us to introduce a new notion of responsive equations, on which we build advanced accessibility features with improved reflow of content, selective highlighting, and dynamic speech text generation, as well as an innovative interaction technique based on abstracting and intelligently summarizing sub-expressions.

Keywords

STEM Accessibility, Mathematics, MathJax

Introduction

The text-to-speech translation of mathematical expressions has always been a challenging problem and is one major obstacle for fully inclusive education. Consequently, a number of software solutions have been researched over the years (see Karshmer, Gupta, and Pontelli (664-669) for an overview). As web delivery grows, web-accessibility for mathematics is more important than ever. Although Mathematical formulas on the web can be represented in their own specialized markup language, MathML (Carlisle, Ion, and Miner), as part of the HTML5 standard (W3C), only very few major browsers implement MathML rendering natively. Consequently MathJax (MathJax Consortium) a JavaScript library for rendering Mathematics in any browser, has become a quasi-standard for displaying Mathematics on the web.

Accessibility features have been a core functionality of MathJax since its inception. But while in 2010, this meant MathPlayer (Soiffer, 205-206) compatibility, zoom, magnification, and copy&paste features, today we find ourselves in a very different assistive technology (AT) landscape for mathematics. MathPlayer has been forced into becoming a secondary library, while general screen readers like ChromeVox (Sorge, Chen, Raman, and Tseng) and VoiceOver (Apple VoiceOver) have implemented (partial) support for MathML. Nevertheless the core problem persists: browser vendors continue to show little interest in implementing MathML natively, thus damaging accessibility on a fundamental level for all non-blind users. At the same time, MathML rendering solutions on the web, such as SVG or HTML/CSS converters, cannot rely on web standards such as ARIA to provide intermediary solutions.

To overcome the challenges of this changed landscape, we have developed some advanced accessibility features for MathJax. At the core of this work lies a procedure for the

semantic interpretation and enrichment of presentation MathML that is based on the speech rule engine originally implemented as part of ChromeVox's open-source technology and that also drives the Benetech/MathMLCloud project. The semantic interpretation not only improves the structure of the presentational content, but also allows us to implement advanced accessibility features, which include the following:

1. Improved reflow of expressions for better support of small screens and magnification.
2. Structural abstraction of meaningful sub-formulas together with their interactive exploration and synchronized highlighting that can be helpful for dyslexic readers.
3. Direct generation of speech strings and their exposure to third party screen readers to allow for a uniform reading experience of mathematics for visual impaired users independent of platform or AT environment they use.

Semantic Enrichment

The semantic enrichment is a heuristic procedure that effectively constructs a term tree representing a mathematical expression by analyzing its presentation MathML elements, interpreting their type, role, and font, and turning them into semantic nodes with a parent pointer and a variable number of children. The core of the semantic information goes beyond Presentation MathML and identifies the underlying mathematical structure of an expression. It allows us to identify function applications, functional types, operator scope, limits etc., while aiming to stay faithful to the original notation, and not fixing too much of the semantics, which could lead to false interpretations. It consequently provides a much more shallow interpretation than a full-blown semantic markup language like Content MathML.

As an example, consider the quadratic equation $ax^2 + bx + c = 0$, which is

syntactically, with the exception of the x^2, a linear sequence of single expressions, and is

commonly written as such as in MathML, a single row with nine elements. The semantic interpretation will rewrite it into a tree where the top layer is an equality with two branches, where the right branch is 0, while the left branch itself is a tree representing a sum with three summands.

The heuristic approach both conservatively improves the original Presentation MathML input and extends it beyond presentational information. Consequently the semantic information can be embedded using HTML5-compliant data-* attributes. This provides flexibility for embedding the complete information while leveraging standard APIs to recover it for application developers. The resulting enhanced MathML is embedded in MathJax's internal format as well as its output so that other technologies can leverage the additional information while remaining backwards compatibility to any other MathML tool.

The conservative changes to the MathML source itself already improve the quality of the original source by fixing common authoring issues such as missing grouping or unbalanced fences. As an immediate improvement, the resulting MathML is easier to digest both visually and aurally, as well as for exploration via AT. It also improves basic reflow of mathematics by enabling line-breaking algorithms to better identify good breakpoints. This semantic enrichment is being leveraged in the new accessibility features we have developed.

Responsive Equations

The first application focuses on visual representation of mathematics on small screens. Responsive web design and progressive enhancements are the cornerstone of modern web design. For mathematics, however, reflow is extremely difficult as mathematical presentation is usually two-dimensional and predominantly table-oriented, especially as most mathematics is still primarily designed for print layout. To overcome this, we have developed a rendering mode that is fundamentally different from traditional layout. Leveraging the approach of (math) accessibility tools, this technique visually summarizes a math fragment rather than forcing iterated line breaks (which quickly destroy legibility on small screens). Instead, sub-expressions are collapsed into icon-like representations to fit the equation to the screen width. This approach leverages the semantic enhancement for identifying sub-expressions and estimating their size as well as for providing a suitable iconic representation that summarizes the collapsed sub-expression (e.g., we use + to indicate a sum, an integral symbol for a collapsed integral, etc.)

While initially hidden from view, thus not interfering with the reading flow and page layout for large text magnification or small screens, hidden sub-expressions can be manually expanded in an explorative fashion either by clicking the summary symbol with the mouse or alternatively using pinch and zoom gestures on touch-screen devices. Similarly, expressions can be collapsed by clicking on the relevant parts of a sub-expression.

As an example, consider the identity of Ramanujan in Fig.1, which describes a continued fraction. On the left we have the fully expanded version, while on the right is its maximally collapsed counterpart, where the product in the denominator on the left-hand side of the equation and the continued fraction on the right-hand side are hidden and abbreviated by a dot multiplication symbol and a fraction slash, respectively.

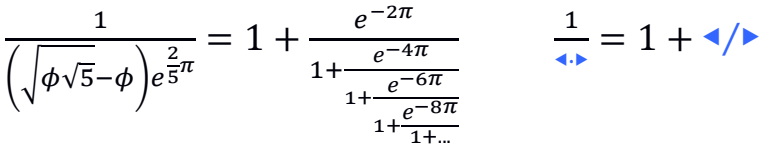

Fig. *1*. A complicated expression and its maximally collapsed version.

Structural Abstraction and Synchronized Highlighting

Responsive equations also serve as a foundation for a new approach to assisting users with learning disabilities. Mathematical expressions are generally large collections of mostly unconnected symbols in two-dimensional layout and can therefore be particularly daunting for readers with dyslexia. MathJax provides support in particular aimed at dyslexic readers by

1. simplifying the structure of formulas using the collapsing technique described in the previous section,
2. enabling interactive exploration of semantically relevant sub-expressions, and
3. providing synchronized, customizable highlighting for sub-expressions.

For this purpose we have implemented a web interface that allows users to explore the content using click, keyboard, or touch events. As opposed to the purely responsive equation approach, however, where the initial state of collapse of an equation depends on many, not necessarily user controlled, factors like screen size, page size, or zoom factor, this interface offers the user fine-grained control over the structural exploration of an expression.

Initially the structure of a formula is maximally simplified as demonstrated in Fig. 1 above and lets users individually explore the equation by manually expanding selected sub-expressions. The sub-expression elements are nested so that only the next level of the collapse is revealed, and the collapsed content is then identified by a mathematical symbol representing the

semantic content of the underlying expression as before. The basic idea of this approach is to present a user with an outline of the formula, visually summarizing the most important components, and letting them choose time and order in which to dive deeper into an expression, thus avoiding the cognitive overload of having to deal with too many symbols at once.

As formula exploration can be done either by point and click or via the keyboard, we offer a number of highlighting options to further support readers with learning disabilities. For the former, we have two modes of highlighting: semantic components can be highlighted when hovering over them, thus also indicating that they can be further abstracted, or alternatively, all hierarchical composition of semantic sub-expressions can be shown by underlaying them with a background color of incrementally increasing intensity.

Keyboard-driven exploration is implemented using a simple navigation model that allows readers to select a formula and explore it using the arrow keys. We again exploit the semantically enriched structure by traversing sub-formulas in terms of semantic components rather than their syntactic structure. For example, when traversing the quadratic equation $ax^2 + bx + c = 0$, the first level would consist of three elements: the left-hand side of the equality, the equality sign, and 0. When diving into the left-hand side, the next level would consist of the three summands and their plus signs. During traversal, sub-expressions can be expanded or collapsed using the enter key. Visually, keyboard navigation is supported by synchronized highlighting of traversed expressions. All highlighting options are fully customizable, offering users choices of fore- and background colors, to combine their own preferred high-contrast colors, as research on optimal color selection for dyslexic readers is still inconclusive (see Rello and Baeza-Yates).

Screen Reader Support

To aid readers who rely on screen reading technology, MathJax now also provides new accessibility features for voicing of mathematics, particularly aimed at visually impaired and blind users. While some screen readers already exploit MathJax's ability to expose MathML underlying a rendered mathematical formula to either voice mathematics directly (e.g., in the case of ChromeVox or VoiceOver), or by using a third-party library like MathPlayer (Soiffer, 205-206) for speech translation, users still have to depend on their screen reader's ability to understand MathML for mathematics support. In order to make users independent of a screen reader's math capabilities and to provide a platform independent, uniform user experience when reading mathematics, MathJax provides an aural rendering engine by exploiting the direct access to the speech rule engine that generates the semantic tree transformation in the first place.

This allows us to directly generate speech text, not only for the mathematical formula itself, but for all its component sub-expressions as well. MathJax offers options for text generation by currently providing two rule sets: MathSpeak (Nemeth) and the ChromeVox rule set (Sorge, Chen, Raman, and Tseng). Both have a choice of reading styles, pertaining mainly to their verbosity, for example, verbose, brief, and super-brief in the case of MathSpeak.

Generated text can be embedded either as alt-text, alongside the semantic data in the MathML and DOM structure via data attributes, or computed incrementally on the fly. Text is exposed to screen readers using ARIA technology, which allows us to integrate speech generation with the features we have already presented in this paper.

Summarization

The collapsing methodology used for visually responsive equations provides a natural basis for generating text summaries of mathematical formulas in lieu of their full description. These summaries can be presented more intelligently in speech. For example, a collapsed sub-expression that consists of a sum of four elements would only have a visual collapse indicating summation but can have an aural description of "Sum of four elements". As a consequence, summaries provide a quick overview of a formula or its parts and most importantly are generally considerably shorter then the full textual description.

Continuous Reading

Summarization is therefore the foundation for a unique MathJax feature to improve reading comprehension of full mathematical texts. As fully voicing even small formulas can be quite long-winded and time consuming, thus often distracting from the actual content of a publication, we exploit the summarization feature to provide an improved reading experience for mathematical documents. Instead of exposing the full mathematical formula to a screen reader, only the summary for the most collapsed version of that formula is exposed via an ARIA label.

For example, voicing the Ramanujan identity from before in MathSpeak's verbose mode leads to a string of 66 words (435 characters), whereas the corresponding summarized version is “StartFraction 1 Over collapsed product with 2 factors EndFraction equals 1 plus collapsed continued fraction”, a string with 15 words (108 characters). Even in superbrief mode, the full expression is 66 words (298 characters), while the summary is 12 words (80 characters), only.

This approach aims to emulate a casual reading style, in which a reader is more interested in getting the overview of the content of a publication and only will dive deeper into a particular

expression if necessary or interested. This drastically reduces the time needed to skim through a publication using a screen reader.

In-depth Inspection

Finally, a screen-reader user can still inspect each formula by engaging with it interactively. This is implemented via the previously explained exploration interface where speech strings for sub-expressions that the reader is traversing are computed and exposed to screen readers via ARIA Live regions. In addition, positional information of expressions is exposed, such as numerator, denominator, base, or superscript. Finally, the ability to collapse and expand certain sub-expressions lets a reader switch between summary descriptions and full verbose voicing for expressions on the fly.

Conclusions

We have presented an AT extension that turns MathJax from a purely visual rendering library for mathematics on the web into a universal rendering solution. The accessibility features aim to provide a uniform user experience regardless of their choice of platform, browser, or screen reader, or indeed how the mathematics is rendered in a page. The extension is open source and freely available from github; it can be integrated directly by web authors, or is offered as an option in MathJax's context menu for users to select.

Acknowledgements

The work was supported by the Alfred P. Sloan Foundation.

Works Cited

Apple VoiceOver, https://www.apple.com/accessibility/osx/voiceover/.

Carlisle, D., Ion, P., and Miner, R., "MathML version 3.0", *W3C Recommendation*, 2010, http://www.w3.org/TR/MathML3.

Karshmer, A., Gupta, G., and Pontelli, E., "Mathematics and accessibility: a survey", *Proc 9th Int. Conference on Computers Helping People*, vol. 3118, 2007, pp. 664-669.

Nemeth, A., "Mathspeak", http://www.gh-mathspeak.com/, 2005.

MathJax Consortium, "MathJax version 2.5", 2014, http://www.mathjax.org

Rello, L. and Baeza-Yates, R., "Optimal colors to improve readability for people with dyslexia," *Text Customization for Readability*, 2012.

Soiffer, N., "Mathplayer: web-based math accessibility", *Proc 7th International SIGACCESS Conference on Computers and Accessibility*. ACM, 2005, pp. 204-205.

Sorge, V., Chen, C., Raman, T.V., and Tseng, D., "Towards making mathematics a first class citizen in general screen readers", *11th Web for All Conference*, 2014. ACM.

World Wide Web Consortium, "HTML version 5.0," *W3C Candidate Recommendation*, 2013, available at http://www.w3.org/TR/html5.