Adaptable Accessibility Features for Mathematics on the Web

Davide Cervone
MathJax Consortium
Union College, NY
dpvc@union.edu

Volker Sorge
MathJax Consortium
University of Birmingham, UK
V.Sorge@cs.bham.ac.uk

ABSTRACT

Accessibility of mathematics is still a challenging problem and providing the right level of support to a reader depends on many factors, such as their particular assistive technology needs, their level of expertise, and the subject area they are working in. We present work towards making math accessibility more adaptable to the reader's personal needs that is implemented in the MathJax library for rendering mathematics on the web. While MathJax provided accessibility support for several years, the new version 3 has both more new features and means of personalization. In particular, it provides adaptable combinations of highlighting, colorization, and magnification techniques. Both Braille and speech output can be generated, with different speech rule sets allowing readers to flexibly change presentation and adaptation for better interpretation of formulas in different subject areas, like Physics, Chemistry, and Logic.

Keywords

STEM Accessibility, Mathematics, MathJax

1. INTRODUCTION

Assistive technology support for mathematical expressions has always been a challenging problem that is characterized by a number of often competing affordances and requirements: Multiple markup languages are used to represent formulas; even in the same markup language the same formula can be described in multiple ways; readers often have different assistive technology needs; they have different levels of expertise; or work in different mathematical subjects.

Consequently, a number of technology solutions have experimented over time with support for different aspects of the problem. From software for generating Math Braille [5], over dedicated screen readers that work on different input languages [9, 11], to tools for dyslexia support [14]. Some of these systems also support different areas of mathematics although mainly geared towards high-school curricula. For a detailed overview of attempts at math accessibility before the dominance of the web see [4].

Modern web technology offers the possibility to combine multiple features into a single medium by augmenting DOM structure with custom data on alternative format information without altering visual representation. MathJax [6], a JavaScript library for TeX-quality typesetting of Mathematics on the web, exploits this approach since version 2.7 to provide an accessibility extension that aims to give flexible support for readers with visual and print impairments (i.e., readers with cognitive disorders like dyslexia) [1]. Although MathJax can render the three most common markup language for Mathematics (LaTeX, ASCIIMath, MathML), none of these has sufficient semantic information to allow for adequate accessibility support. Consequently MathJax relies on an improved semantic interpretation of formulas, which is provided by the Speech Rule Engine (SRE) [12], a system that provides speech generation for mathematical expressions given in presentation MathML.

In this paper we present, how embedded semantics is employed to provide diverse assistive technology features in MathJax, emphasizing features new to version 3 and personalization aspects, that exploit SRE's advanced features like Braille generation, adaptations for reading style and subject areas, like Physics, Chemistry, and Logic.

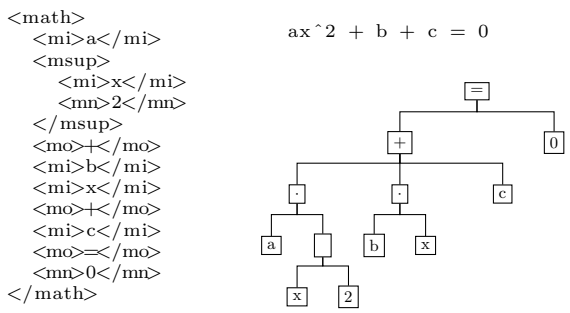

Figure 1: Quadratic equation $ax^2+bx+c=0$ in LaTeX, presentation MathML, and as a semantic term tree.

2. SEMANTIC ENRICHMENT

One of the main challenges to produce useful assistive technology support in MathJax is the lack of semantic information in most standard mathematical markup notations. Formulas are usually given in formats that are geared towards visual representations, such as LaTeX, ASCIIMath, or

W4A '19, May 13–15, 2019, San Francisco, CA, USA

DOI: https://doi.org/10.1145/3315002.3317567

```
<mjx-container class="MathJax" jax="CHTML" display="true" hasspeech="true" tabindex="1">
  <mjx-math type="relseq" role="equality" id="16" children="15,10" content="9"
            aria-label="a␣x␣squared␣plus␣b␣x␣plus␣c␣equals␣0"
            speech="a␣x␣squared␣plus␣b␣x␣plus␣c␣equals␣0">
    <mjx-mrow type="infixop" role="addition" id="15" children="12,14,8" content="4,7" parent="16"
              speech="a␣x␣squared␣plus␣b␣x␣plus␣c">
      <mjx-mrow type="infixop" role="implicit" id="12" children="0,3" content="11" parent="15"
                speech="a␣x␣squared">
        <mjx-mi class="mjx-i" type="identifier" role="latinletter" font="italic" id="0" parent="12" speech="a">
          <mjx-c c="a" />
        </mjx-mi>
        <mjx-mo class="mjx-n" type="operator" role="multiplication" id="11" parent="12" speech="times">
          <mjx-c c="2062" />
        </mjx-mo>
        <mjx-msup type="superscript" role="latinletter" id="3" children="1,2" parent="12" speech="x␣squared">
          <mjx-mi class="mjx-i" type="identifier" role="latinletter" font="italic" id="1" parent="3" speech="x">
            <mjx-c c="x" />
          </mjx-mi>
          ...
  </mjx-math>
</mjx-container>
```

Figure 2: Rendered quadratic equation in MathJax v3 with embedded semantic tree and speech.

MathML, with LaTeX being clearly prevalent on the majority of web pages. And unlike most historical systems [4] MathJax has no control over type or quality of input content. Similarly, MathJax was designed for displaying formulas, offering a number of different rendering solutions, such as HTML with CSS or SVG output. Figure 1 presents two standard ways of representing the quadratic equation $ax^2+bx+c=0$, in LaTeX on the top and in MathML on the left. Both are given in a flat structure, that is sufficient for a linear representation but in general is not enough to create good mathematical explanations and interaction support.

Consequently, the first step towards accessibility support in MathJax is by imposing a semantic interpretation on a given math expression and generating a tree representation that can be embedded into rendered MathJax expressions to ensure a similar user experience across browsers. The idea of the semantic interpretation is an extension of the heuristics implemented in the screen reader ChromeVox [13] and further developed in the context of MathJax [1], which effectively rewrites a flat expression into a term tree structure by first interpreting the basic nature of symbols, and propagating this through the expression to determines the scope of operators, relations, etc. Figure 1 presents this transformation for the example of the quadratic equation $ax^2+bx+c=0$, which is rewritten from either its MathML or LaTeX form into its semantic interpretation on the right.

The resulting semantic tree can be understood as an orthogonal view of the mathematical expression. To exploit it, we embed it using `data` attributes into the DOM elements that represent MathJax's rendering of the equation regardless of the choice of a particular rendering solution. MathJax offers several different output formats, where expressions in the DOM are collections of `div` and `span` elements, HTML5 custom elements, or SVG graphics elements. These collections often don't correspond well to the mathematical structures they represent. Figure 2 contains the MathJax's DOM output using HTML5 custom elements. `data` attributes `type`, `role`, `id`, `parent`, `children`, `content`, and `speech` are injected (here the `data-semantic-` prefix is omitted to preserve space and improve readability). The actual semantic tree structure is represented via the `id`, `parent`, `children`, and `content` attributes. Note that data attributes provide a fast and standardized means of retrieving information from the DOM that is fully consistent with HTML5 practices.

3. ACCESSIBILITY FEATURES

MathJax's assistive technology extension is mainly aimed at supporting users with reading disorders, such as dyslexia, and visual impairments. However, some aspects of it could also be used as a general aid for readers unfamiliar with the content or for learners at different levels. We summarize the main features in this section.

3.1 Speech Output

One main emphasis of the assistive technology extension is to support screen-reader users, but make them independent of their particular screen-reader's Math capabilities. MathJax exploits SRE's feature to provide aural rendering for mathematical expressions. Speech string computation is based on the embedded semantic tree. Speech strings for a formula are either generated on the fly when running MathJax in a browser client, or pre-computed by a page author, when formulas are pre-rendered and inserted into the DOM.

To expose the speech strings to a screen reader, MathJax uses two main techniques. Firstly, a description of an entire formula is given in the ARIA label at the top level of the DOM structure representing the expression. Figure 2 demonstrates the ARIA label embedding as well as the use of a custom data attribute for speech. The latter allows us to aurally render not only the entire formula but each of its sub-expressions, which is exploited during user interaction with the formula, as discussed below. For this purpose, MathJax introduces a dedicated assertive ARIA live region into the DOM and updates it with the desired speech output. Content changes in the live region are picked up by screen readers and spoken. This is a feature that is particularly in line with MathJax's mission to support mathematics rendering in all browsers and on all platforms, working with any modern screen reader that supports live regions.

3.2 Tactile Output

In addition to speech output, SRE now provides generation of Braille output using Unicode Braille symbols. It currently supports translation of expressions into Nemeth

Braille [7] only, but we plan to support additional Math Braille styles in the future. In MathJax we can make use of this feature by computing Nemeth output as a further rendering mode in parallel to visual and speech output.

Unfortunately, there is currently no easy way to get tactile output to the user during regular reading, as most screen readers translate text internally into Braille, which is not necessarily useful for Mathematics that needs specific Braille formats. In particular there is no way to inject Braille directly into alt text or aria labels for screen readers to simply pick it up and pass it on. However, during user interaction with a formula, MathJax takes control so that Braille content is exposed similar to speech: The Unicode Braille is simply pushed into a second live region allowing a screen reader to push it directly to a connected Braille display.

3.3 Visual Aids

MathJax offers a number of assistive techniques based on visual alteration and enhancement of formulas. While these are mainly aim at supporting readers with low vision or print impairments like dyslexia, they can also aid a general audience in understanding formulas and their structure.

Highlighting and Contrast — Math expressions generally are large collections of mostly unconnected symbols in a two-dimensional layout, which can be particularly daunting for readers with dyslexia. Reading comprehension can be supported, however, by a choice of high-contrast colors [10] and selective highlighting [3]. We realize this in MathJax by offering changes of fore- and background colors, and selective highlighting of sub-expressions. In the case of the former, the entire formula is simply colored to simulate a colored overlay to provide high-contrast in itself or to offset it from the surrounding text. For the latter, we again exploit the semantic structure of the formula to provide a more meaningful highlighting of sub-formulas. Below is a simple example of progressive shading for the quadratic equation:

$$ax^2 + \quad + \quad = 0$$

In practice, sub-expressions are highlighted when hovering over them with the mouse pointer, or by switching highlighting on permanently. This can be recursively refined for sub-expressions, e.g., hovering on the denominator or numerator of a fraction only. For permanent highlighting, the opaqueness of the background gradually increases in nested sub-expressions. Nevertheless, in large expressions, highlighting is of limited utility in getting an overview of the structure of a formula; to aid this further, we introduce a technique for structural abstraction.

Structural Abstraction — The idea of structural abstraction is to assist readers by simplifying the structure of the formula initially and letting them individually explore the equation by manually expanding selected sub-expressions. Below is an abstracted version of the quadratic equation, where the left-hand side is collapsed into a single addition symbol that can be expanded to the full sum by clicking.

$$\blacktriangleleft + \blacktriangleright = 0$$

While in this case the abstraction is fairly trivial (for some more complex examples see [1]), for large or complex formulas and equation systems, abstraction can often significantly simplify the visual appearance, leading to less cognitive load for a reader, as well as a better understanding of the basic structure of an expression. In addition, abstractions can aid in producing concise speech descriptions by summarizing formulas or parts thereof. This can be particularly important for presenting formulas inline in mathematical texts, where a reader might want to understand the whole text without having their reading flow constantly interrupted by every formula being spoken in minute detail. Visual abstraction can also be helpful in presenting mathematics on small form factor displays by adapting formulas to the page reflow to avoid the need for excessive panning in a page.

Formula Coloring — An alternative means to providing support to readers with print impairments is to heighten the contrast of elements within a formula itself. This effectively amounts to offsetting neighboring symbols as much as possible with contrasting colors. While this could be done in a linear fashion, exploiting the semantic tree structure allows for a similar effect while also offering the possibility of retaining the same or similar colors for the same or similar operations. A tree Coloring example for the quadratic equation is given below:

$$ax^2 + bx + c = 0$$

Magnification — While changing contrasts can already help low-vision users, many rely on screen magnification to read content. MathJax supports magnification in multiple ways: standard browser zoom is supported by re-rendering mathematics at the new size. MathJax also provides global scaling settings to enlarge all math elements at once. Individual math expressions can be further magnified using a "lens" that ordinarily offers a zoom window on top of an expression, with scrolling to pan across the entire zoomed expression. Exploiting the semantic markup, magnification can also be restricted to particular sub-expressions only, as in the example below.

3.4 Interactive Features

Since mathematical formulas are generally of a complex nature, and already a small formula can make for a complex utterance, just listening to an expression once is generally not enough for comprehension. Therefore, it is particularly important to provide the reader with a means of engaging with formulas interactively. MathJax offers an interface for interactive exploration of mathematical expressions that allows a reader to step through sub-expressions. Technically, this is realized by giving each math expression an ARIA role of `application`, and upon entering an expression, the user can walk it using the cursor keys.

During this exploration, many of the previously described features can be restricted to sub-expressions as well. Speech strings can be computed for all sub-expressions and voiced by updating the ARIA live region introduced in the DOM. Similarly highlighting as well as magnification can be synchronized to the exploration using CSS techniques.

It is important to note that the walking is not done in a linear "left-to-right" fashion, but instead follows the embedded semantic tree structure. Thus starting with the entire expression, the next lower level consists, for instance in the

case of the quadratic equation, of the sum on the left hand side of the equation, the equality sign and the 0 on the right. The sum can then be further explored allowing the reader to step through the single summands from left to right, and so on. Note also that navigating the semantic tree ensures that the user experience of exploring a structure is the same regardless of the specific renderer used even though their underlying DOM trees differ. In addition to this, MathJax also provides the ability to query the position of elements in a formula, means to summarize the expression currently highlighted, cursor virtualization, as well as special navigation modes for matrices and equation systems.

4. PERSONALIZATION

While MathJax always supported personalization of the accessibility extension, for example by individual choice of fore- and background colors for highlighting or determining the size of magnification, the new version 3 will have a greater emphasis on adaptation to personal needs and content. We will summarize a few:

Pick and Mix — One goal is to provide compatible accessibility components so they can be flexibly combined. For example, users can have both speech and Braille output or combine goal-directed magnification with exploration and highlighting, where either can be done in a different contrasting colors. This will allow for a pick and mix strategy that a reader can tailor towards their individual needs.

Altering Speech — Another features is to switch alternative speech output even during interaction. We particular concentrate on two alternatives: MathSpeak [8] and ClearSpeak [2] provided by SRE. As example, consider the quadratic formula, i.e., the solution to the quadratic equation we have used as example so far:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Its MathSpeak translation is *"x equals StartFraction negative b plus-or-minus StartRoot b squared minus 4 a c EndRoot Over 2 a EndFraction"*, while the corresponding ClearSpeak is *"x equals the fraction with numerator negative b plus or minus the square root of b squared minus 4 a c and denominator 2 a"*. We see that ClearSpeak provides a more natural speech experience, while MathSpeak uses more disambiguating language, making it more suitable for large, complex expressions. Swapping the way an expression is spoken will allow readers multiple views to ease comprehension.

Domain Specific Semantics — A second way to improve speech and understanding is to enable more domain specific interpretations for different mathematical areas and subjects like Physics, Chemistry and Logic. This is achieved in two ways: Automatically, by MathJax pushing relevant information it has obtained from LaTeX input formulas to SRE, which can exploit this information with domain specific heuristics. Since these heuristics are often incompatible they would normally not fire unless explicitly chosen by a user by manually setting a domain.

For example consider the following physics expression:

$$\langle \phi | A | \psi \rangle$$

The formula is in so called bra-ket notation, which expresses vectors and linear functionals in quantum mechanics. Regular MathSpeak output would be *"left angle bracket phi StartAbsoluteValue upper A EndAbsoluteValue psi right angle bracket"*, which in the context of physics is wrong, as A is a matrix and ϕ, ψ are vectors. If this expressions stem from a LaTeX formula using `physics` or `braket` packages, or the physics heuristics are explicitly switched on SRE will produce *"the linear operator A acting on bra phi and on ket psi"* instead. However, using the heuristic in a non-physics can lead to misinterpretation of the meaning and nesting structure of angle brackets and vertical bars.

5. CONCLUSIONS

The presented novel accessibility features of MathJax are currently in beta testing, but will be available as an option with the full version 3 release. One particular emphasis is on ease of personalization by providing flexibly combinable assistive technology components. A second, equally important aspect of our work is to enable better support for advanced notation to help practitioners in many mathematical fields. This is an area that has often been neglected due to an emphasis on high-school mathematics in many approaches. But the possibility to better exploit content authored in semantically meaningful LaTeX in MathJax and complement with appropriate heuristics and speech in SRE should help us to make progress on this problem.

Acknowledgments

We thank the Simons Foundation for support on advanced semantic recognition, the Big Ten Academic Alliance for funding Nemeth Braille in SRE, and the Sloan foundation for supporting the initial accessibility extension in v2.7.

6. REFERENCES

[1] D Cervone, P Krautzberger, V Sorge. Towards universal rendering in MathJax. *13th W4A*. ACM, 2016.

[2] L Frankel, B Brownstein, N Soiffer, E Hansen. Development and initial evaluation of the clearspeak style for automated speaking of algebra. *ETS Research Report Series*, 2016(2):1–43, 2016.

[3] R Jones. Strategies for reading comprehension: Selective underlining, 2008.

[4] A Karshmer, G Gupta, E Pontelli. Mathematics and Accessibility: A survey. *9th ICCHP*. Springer, 2007.

[5] S Maddox. Mathematical equations in braille. *MSOR connections*, 7(2):45–48, 2007.

[6] MathJax v2.7, 2016. www.mathjax.org.

[7] A Nemeth. *The Nemeth Braille code for mathematics and science notation*. Library of Congress, 1972.

[8] A Nemeth. Mathspeak. gh-mathspeak.com, 2005.

[9] TV Raman. AsTeR: Audio system for technical readings. *Inform. Technology & Disabilities*, 1, 1994.

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

[11] N Soiffer. Mathplayer: web-based math accessibility. *Computers and accessibility*, p. 204–205. ACM, 2005.

[12] V Sorge. Speech Rule Engine version 3, 2018. github.com/zorkow/speech-rule-engine.

[13] V Sorge, C Chen, TV Raman, D Tseng. Towards making mathematics a first class citizen in general screen readers. *11th W4A*. ACM, 2014.

[14] TextHelp. EquatIO. www.texthelp.com/equatio.