Journal of Science Education for Students with Disabilities

ISLAND 2019 ARTICLE

The Accessibility of Mathematical Notation on the Web and Beyond

Jason J.G. White *
Educational Testing Service

Abstract: *This paper serves two purposes. First, it offers an overview of the role of the Mathematical Markup Language (MathML) in representing mathematical notation on the Web, and its significance for accessibility. To orient the discussion, hypotheses are advanced regarding users' needs in connection with the accessibility of mathematical notation. Second, current developments in the evolution of MathML are reviewed, noting their consequences for accessibility, and commenting on prospects for future improvement in the concrete experiences of users of assistive technologies. Recommendations are advanced for further research and development activities, emphasizing the cognitive aspects of user interface design.*

Keywords: *accessibility, Mathematical Markup Language, text to speech, braille, user interface design, Web standards*

***Corresponding Author, Jason J.G. White** (jjwhite@ets.org)
Submitted January 21, 2020
Accepted March 2, 2020
Published online April 20, 2020

SIGNIFICANCE OF THE PROBLEM

Mathematics is a beautiful and important discipline in which profound insight can be derived logically from relatively simple, foundational principles and definitions. In addition, it functions as a central component of the natural and social sciences. Thus, mathematical knowledge is fundamental to a scientific education, and to many scientific or technical professions. For this reason, access to mathematical content is a necessary element of equality of opportunity for people with disabilities in scientifically-oriented educational programs, and in associated careers.

The World Wide Web poses risks as well as opportunities for enhanced access to mathematics. The positive side consists of a growing body of mathematically-oriented documents and applications, including educational materials that are now available either on the Web itself or in electronic books, some of which are based on Web technologies and standards, such as the EPUB publishing format. These materials include books, mathematical or scientific journals, online courses, and application software used for a variety of educational and professional purposes. By making these resources intrinsically accessible, there is great potential to overcome limitations often encountered by students with disabilities in the conversion of mathematical texts into alternative formats, such as embossed braille, large print, and recorded audio (Bouck & Meyer, 2012). In principle, this access could be achieved in a timely manner and in ways that meet individuals' needs, thus approximating true equality much more closely than approaches based on conversion of print media into conventional specialized formats have been able to accomplish. Web-based materials are only actually accessible, however, to the extent that they are designed and implemented appropriately, and the required support is available from Web browsers, electronic book reading tools, and assistive technologies. Thus, for example, notation occurring in much of the vast mathematical and scientific journal literature that is now available in Portable Document Format (PDF) is not accessible to users of screen readers, in the absence of further innovation in advanced document recognition techniques that can achieve the required levels of reliability. The negative side of the story, then, acknowledges the barriers to scientific education and professional opportunities created by inadequacies in the accessibility of mathematical content appearing in Web-based materials.

The scope of this paper is confined to a single but essential aspect of the problem: the accessibility of the notation itself. As will become clear in the subsequent discussion, this includes effective access to the processes of reading, writing and manipulating mathematical expressions occurring in a wide variety of contexts and applications. The accessibility of graphics, including diagrams and charts, is not treated here, though it is recognized also as indispensable. Following a review of central needs of users with disabilities for access to mathematical notation, the role of MathML is explained. Current developments are then discussed, with commentary concerning the challenges and opportunities that remain.

ACCESS TO NOTATION: USERS' NEEDS

Before reviewing the current state of the MathML Web standard and its implementations, a valuable perspective can be gained by advancing informed hypotheses that identify

pertinent needs of users with disabilities in accessing mathematical content.

For users who are blind, a braille or spoken representation of mathematical notation is necessary. Standards for the braille encoding vary internationally. They include Nemeth Code, Unified English Braille, the Marburg mathematics code, and French mathematical braille. Although the spoken representation has not been subject to formal standardization, there are several established approaches recognized in the literature, notably MathSpeak (Isaacson et al., 2010) and, more recently, ClearSpeak (Frankel et al., 2016).

For users with low vision, the mathematical notation can be enlarged by applying commonly implemented magnification techniques. The inclusion of success criterion 1.4.10 in the *Web Content Accessibility Guidelines (WCAG) 2.1* standard (World Wide Web Consortium, 2018) may, however, lead to greater demand to optimize the spatial layout of mathematical content automatically to ensure that it is wrapped within the view-port, thereby avoiding the need for the user to scroll the display horizontally to read an entire expression.

There is suggestive evidence (Lewis et al., 2010) that a spoken presentation of mathematical expressions, with synchronized highlighting, can benefit students with learning disabilities. Although further empirical investigation of the efficacy of a spoken presentation and of appropriate modes of delivery (e.g., speaking styles) is warranted, the potential of speech output to enhance mathematical learning beyond the population of students who are blind or visually impaired whom it has traditionally served is noteworthy.

Working effectively with mathematical notation demands more than having the ability to read it. One must also be able to 'do mathematics'—that is, to understand, manipulate, and write symbolic expressions in order to solve a mathematical problem or to develop a mathematical proof. By analyzing a detailed example drawn from elementary algebra, Stöger et al., 2004 have argued that for a person who is blind and who uses a single-line refreshable braille display (i.e., hardware that is currently available commercially), simplification of a mathematical expression can impose considerable demands on working memory. In addition, frequent shifting of the display is necessary between the given expression that is being manipulated, and the new expression that is under construction. In the strictly serial modality of speech, it is reasonable to suppose that the same difficulties would emerge to an equal or greater degree than in braille. As the authors acknowledge, and as remains largely the case today, the development of tools to support nonvisual manipulation of mathematical expressions has not received sustained research attention, unlike efforts to enable reading via a spoken or braille presentation, or indeed means of navigating the logical structure of the notation to enhance comprehension.[1] However, due to the fundamental importance of symbolic manipulation in mathematical learning and practice, it is clear that the development of well researched approaches to facilitating such tasks in nonvisual modalities is desirable. In sum,

[1]On the latter point, it may be noted that a user interface for interactively reading and navigating the structural components of spoken mathematical notation was introduced as early as Raman, 1994, together with variable substitution as a means of summarizing complex expressions to aid the reader's understanding.

adequate access to mathematical notation entails effective means of reading, navigating, manipulating, and writing it, in modalities suited to the needs of each individual, and without creating unnecessary cognitive load.

REPRESENTING MATHEMATICS ON THE WEB

Overview of MathML

The Mathematical Markup Language (MathML), first standardized by the World Wide Web Consortium in 1998 and now in its third major version (World Wide Web Consortium, 2014), is the established format for representing mathematical notation on the Web. Expressions written in MathML can be included in Hypertext Markup Language (HTML) documents directly, for consumption by tools such as Web browsers, or in formats based on the Extensible Markup Language (XML) that may be used by publishers to produce HTML-based or typeset material. The MathML standard provides for two distinct means of representing mathematical expressions—Presentation MathML, and Content MathML. Presentation MathML is overwhelmingly the more common form used in practice. It represents mathematical notation suitably for visual rendering, but without necessarily preserving its underlying meaning. Content MathML, however, represents notation as an expression tree in which the mathematical meaning is given unambiguously, for example by applying operators to operands. The chief limitation of Content MathML is that it is constrained in the types of mathematical subject-matter which it can convey, as specifying the meaning of notation used in the entire discipline would be infeasible.[2] In deciding whether to implement Content MathML or Presentation MathML in any particular application, developers are thus confronted with a trade-off between the unambiguous semantics of the former, and the more comprehensive repertoire of mathematical notations that can be represented by the latter. In addition to having interesting potential to enhance accessibility, as will be noted subsequently in this paper, Content MathML is also suitable for processing by general-purpose software, such as symbolic algebra systems that depend on a representation of the underlying meaning of mathematical expressions. The fragments of code in Figure 1 illustrate the contrast between the Presentation MathML and the Content MathML representations of the same elementary linear equation.

The syntactic verbosity of MathML (evident in Figure 1) entails that it is difficult to write or edit by hand. Instead, either a graphical, mathematical editor may be used, or the MathML may be generated via conversion from another format, such as the linear textual notation used in the TEX typesetting system.[3]

Review of Relevant Implementations of MathML

Whereas MathML has been implemented and deployed for a variety of purposes, the focus of this section lies in those applications which are most directly relevant to enhancing access for people with disabilities. Historically significant research projects intended to improve the accessibility of mathematics to people who are blind, some of which are based on presentation MathML, have

[2]The authors note that '[t]he base set of content elements is chosen to be adequate for simple coding of most of the formulas used from kindergarten to the end of high school in the United States, and probably beyond through the first two years of college, that is up to A-Level or Baccalaureate level in Europe'. (World Wide Web Consortium, 2014, § 4.1.2).

[3]Some tools and techniques for creating accessible content in MathML are documented in Michigan State University, n.d., 2019.

been reviewed elsewhere in the literature (Archambault, 2009; Karshmer et al., 2007). For the most part, these projects took the form of specialized tools for reading and editing mathematical content nonvisually. The narrow scope of such software, which focused on working with mathematical notation, and its separateness from the applications and assistive technologies already familiar to users, may in part explain its lack of success in making the transition from completed research projects to widespread use in practice. As discussed in a detailed exposition by Soiffer and Noble, 2019, developments in recent years have taken a somewhat different turn, emphasizing the integration of support for reading and navigating mathematical notation directly into assistive technologies themselves, especially screen readers.

```
<mrow>
   <mi>a</mi> <mi>x</mi>
   <mo>=</mo> <mi>b</mi>
   < / mrow>
```

(a) The equation $ax = b$, given in Presentation MathML.

```
<apply>
<eq/>
<apply>
            <times/>
            <ci>a</ci>
            <ci>x</ci>
</apply>
<ci>b</ci>
</apply>
```

(b) The equation $ax = b$, given in Content MathML.

Figure 1. Contrasting examples of Presentation MathML and Content MathML. In the former case, the multiplication is implicit in the algebraic expression, whereas it is explicit in the latter as an operator applied to two arguments.

Thus, in the popular Microsoft Windows environment, both the JAWS for Windows and NVDA screen readers enable reading and interactive navigation of presentation MathML, which may be presented simultaneously in spoken form and in Nemeth Code braille via a refreshable display. In the case of NVDA, the MathPlayer plug-in (Soiffer, 2005, 2009) is required, whereas the developers of JAWS chose to implement support for MathML independently. Likewise, the VoiceOver screen reader in Apple's iOS operating system can render presentation MathML in speech or as Nemeth Code braille. Spoken rendering is also available from the VoiceOver screen reader in Mac OS, whereas the braille representation in this environment presently falls short of offering Nemeth or other mathematical codes. Also significant is the implementation of spoken rendering and interactive navigation of Presentation MathML in the ChromeVox screen reader (Sorge et al., 2014), which led to further development of the Speech Rule Engine originally created for ChromeVox as a stand-alone software project.

A notable advance in nonvisual entry and editing of mathematical notation occurred with the development of an equation editor capable of converting Content MathML simultaneously to Presentation MathML for visual rendering, and to Nemeth Code braille, while also converting Nemeth Code entered via the keyboard of a refreshable braille display to Content MathML (Dooley et al., 2016). Together, these features enable real-time display of interactively edited mathematical expressions on screen and in braille, with input taken from either a qwerty keyboard or a braille keyboard. By adopting Content MathML as the authoritative internal format in which expressions are represented by the program, inaccuracies of conversion that could result from ambiguities in the notation are avoided. The equation editor is implemented as a

Web-based application, thus furthering the trend toward incorporating accessibility of mathematics into general software environments rather than building special-purpose tools.

To assist students and professionals with editing notation in documents prepared for use by the popular LATEX typesetting system, Sorge, 2016 demonstrated an extension to the Emacspeak audio desktop environment (Raman, 1997) that enables spoken presentation and interactive navigation of mathematics occurring in LATEX content.[4] This extension enables the rich facilities for working with LATEX documents already available within the Emacs editing environment to be fully utilized, while providing a spoken rendering of the mathematical notation that is significantly superior to reading the syntax of the LATEX markup directly with a screen reader. In addition, a mechanism is provided to ease the detection and correction of errors in the LATEX syntax via the auditory interface.

The adoption of MathML on the Web at large has been constrained by a history of inadequate an inconsistent implementations of the standard in Web browsers. As a result, it has become the practice among some publishers and Web site developers to avoid delivering MathML to the user's browser, opting instead to serve notation directly as a rasterized image. Unless visually hidden MathML is also included in the markup, this practice effectively prevents assistive technologies from processing the structure and content of the notation, and therefore from supporting flexible presentation and interaction features valuable to users. These features include synchronized highlighting, simultaneous spoken and braille presentation, or interactive navigation of subexpressions. Overcoming the technical grounds for such publishing strategies is thus of great import in improving accessibility in practice. The problematic situation regarding the implementation of MathML in Web browsers has prompted two initiatives of significance to the future of mathematics accessibility.

First, tools have been implemented in JavaScript that can render MathML or other representations of mathematical notation graphically in a browser. The best known of these systems is MathJax, which can process notation given in Presentation MathML, TEX, or ASCIIMath format. The current release of MathJax implements a variety of accessibility-related features that the user can activate and control via a context menu. These capabilities include spoken representations in any of several speech styles, as well as Nemeth Code braille, achieved by using Speech Rule engine to generate labels and WAI-ARIA (World Wide Web Consortium, 2017) live regions for processing by a screen reader. Interactive structural navigation of the notation is supported, as is selective highlighting of subexpressions, and enlargement of the visual presentation (Cervone et al., 2016; Cervone & Sorge, 2019). As authors of MathJax acknowledge, however, limitations of WAI-ARIA preclude seamless integration of braille output into the user's reading experience (Cervone & Sorge, 2019).

Second, there have recently emerged renewed efforts to improve the quality and consistency of support for MathML in open-source Web browsers, including the development of an implementation for browsers based on the Chromium project, notably, Google Chrome

[4]This is achieved by converting the LATEX representation of the notation to Presentation MathML via MathJax.

and, more recently, Microsoft Edge (Igalia, S.L., 2019). These implementations are being developed in parallel with work by the recently formed MathML Refresh Community group on the MathML Core specification (W3C MathML Refresh Community Group, 2019), a draft document which aims to define the visual rendering requirements for a subset of Presentation MathML precisely, and to integrate it better with fundamental technologies of the Web. If successful, these initiatives can be expected to overcome the principal limitations of previous browser-based implementations, leading to high performance in Web-based applications and to delivering a better quality of visual rendering. Some of the implications of this work for accessibility are noted in the next section.

COMMENTARY AND PROSPECTS

As is apparent, software developers in recent years have pursued different architectural options in implementing mathematics on the Web, and in striving to make it more accessible to users with disabilities. First, there is a choice between implementing the visual rendering of MathML directly in the browser, and implementing it elsewhere—for example, on a server, or in scripts loaded by Web pages. This decision has important consequences for the way in which accessibility-related features such as highlighting and enlargement are implemented, in particular, the need for assistive technologies that function at the operating system level in providing this support. A related architectural decision is whether to deliver the underlying mathematical expressions, as MathML, to the browser, or whether to forward only the graphical representation. If only the typeset output were provided, a mechanism other than MathML would need to be devised and standardized to carry the structure and content of the notation to any assistive technology that required it, whether functioning within the browser or at the operating system level.

Second, there is a choice of whether to implement nonvisual rendering of mathematical expressions within assistive technologies (screen readers and read-aloud systems), or whether to do so in scripts executed by the browser—delivering only the final speech or braille output to the assistive technology. The former option enables assistive technology developers to integrate the reading and navigation of mathematical expressions most effectively into the user interface of each software product, leading to a more consistently designed experience from the user's perspective, while also supporting mathematical notation occurring outside the context of the Web, for instance in desktop-based word processors. Consistency of keyboard commands and touch gestures, navigation functions, and respect for users' preferences within and across applications are among the advantages that can be assured by this approach. This option also places responsibility for the quality of the nonvisual rendering firmly in the hands of assistive technology creators. Anecdotal experience and informal experiments known to the author indicate that, among screen readers specifically, the quality of implementations of Presentation MathML currently varies considerably. The availability of high-quality, specialized software components that may be used by assistive technology to render mathematical notation, such as Speech rule Engine and MathPlayer, could, if further adopted, ameliorate this difficulty.

With regard to the latter option, the extent to which mathematical rendering can be effectively left to scripts operating within the browsing environment is limited by the features of current Web standards, notably WAI-ARIA. A proposal to allow braille-specific labels in WAI-ARIA has recently been advanced. In addition, the W3C is currently exploring technical approaches to enabling spoken presentation, including pronunciation and pauses, to be specified directly in Web-based content—a capability that could be used, among other purposes, in the spoken rendering of mathematical notation. Although this would enable improvement in the quality of nonvisual rendering and interaction provided by tools such as MathJax, it would not address the challenge of user interface consistency and of effective integration with the user's chosen assistive technologies. On the other hand, rendering of the notation within the browser's scripting environment circumvents the limitations and inconsistencies of the processing of MathML by assistive technologies, thus offering the potential to deliver a presentation of higher quality and with greater consistency across platforms.

Although it is possible for both browser-based and assistive technology-based nonvisual rendering solutions to coexist, only one can be applied to the presentation of any given content to the user. This offers the user an awkward and potentially confusing choice of rendering technology, while dividing standardization and software development efforts between pursuit of two distinct architectures. In the absence of a resolution in favor of either alternative, it is likely that both approaches will continue to evolve in parallel. This presents users and developers of accessible Web applications with trade-offs that may complicate decision-making, and ultimately runs the risk of imposing technical constraints that compromise the usability of interfaces needed for tasks involving the reading and manipulation of mathematical notation, which are already cognitively demanding in their own right.

Thus, these contrasting architectural approaches have resulted in a fragmentation of resources and efforts toward enhancing the accessibility of mathematical notation. Whereas the earlier era of development was characterized by a plurality of stand-alone software projects for making notation accessible, the current period reflects a corresponding diversity of solutions and implementations of varying quality among browser-based tools and assistive technologies. The extent to which MathML is effectively and consistently implemented in open-source browsers during the coming years may largely determine the dominant architectural approach taken in the future to providing basic accessibility to reading and navigation across the graphical, speech, and braille presentational modalities. It is also possible that, as has already occurred with user interface components more generally through the development of WAI-ARIA, the reflection of MathML or of other markup representing mathematical notation in the application programming interfaces used by assistive technologies in each operating system may be subject to standardization. These interfaces are principally used by screen readers, although they are intended to serve assistive technologies more widely. Such standardization would contribute to the accuracy and completeness with which screen readers can process MathML, but it would not, by itself, overcome inconsistencies in the quality of different implementations.

A further line of work that can proceed independently of the overarching architectural

concerns seeks to enrich the semantic clarity of expressions written in Presentation MathML, therefore enhancing accessibility, in particular the quality of spoken rendering. In current practice, heuristic strategies must be relied upon to discern the mathematical meaning of Presentation MathML. For example, the expression |X| could be understood, according to context, as signifying the absolute value of a real number, the norm of a vector, or the cardinality of a set, each of which interpretations could be given a different spoken presentation.[5] Similarly, enhanced spoken rendering of chemistry texts could be obtained by distinguishing formulae that contain chemical symbols from those that should be interpreted as algebraic expressions. The MathML Refresh Community Group has discussed several alternative approaches to enabling authors of mathematical and scientific material to disambiguate Presentation MathML, any of which could be put forward for standardization. These possible mechanisms include the use of parallel Presentation MathML and equivalent Content MathML,[6] as well as the creation of new disambiguating markup that could be used to clarify the underlying semantics of notation written in Presentation MathML. The first of these solutions is subject to the limitations of what is expressible in Content MathML. The second approach would require the development of a new markup standard, the scope and nature of which would need to be determined, and the purpose of which would be to clarify the meaning of semantically ambiguous notation.

Such clarifying markup, as presently conceived, could specify the mathematical subject-matter of the expression (for example, algebra, analysis, or geometry) to indicate its meaning, or could be used to specify the meaning of the entire expression or of a subexpression, based on a list of roles defined by identifying notational ambiguities occurring in different mathematical subdisciplines. This markup could be defined directly as an extension of the syntax of Presentation MathML, or it could be introduced via the WAI-ARIA mechanism by proposing an ARIA module for mathematical content. The utility of supplying metadata to specify the subject-matter of an entire document may also be explored, as doing so could be sufficient in some cases to determine accurately the meaning of notation used in the text.

The current period of software development in this field also continues to focus on solving problems of reading and structural navigation, with some attention being devoted to the writing and editing of notation, but without a deeper investigation of how best to support the manipulation and rewriting of expressions necessary to the practice of 'doing mathematics'. Features of nonvisual user interfaces that may reduce the cognitive demands associated with editing and manipulating notation have been proposed, and in some cases implemented. In particular, the use of tabular structures, which may be represented spatially as well as in a linear format, has been investigated as a means of facilitating such tasks as polynomial long division, solving inequalities, and editing automata specified as adjacency matrices (Bernareggi, 2010). The ability to mark individual algebraic terms and to return to marked positions has also been implemented (Flores & Archambault, 2014). More extensive support for performing operations on algebraic expressions has been suggested in a somewhat ill-defined proposal by Alajarmeh et al., 2011.

[5]This example is due to participants at the Web Accessibility of Mathematics Workshop.
[6]Figure 1 illustrates the essence of this approach.

However, the efficacy of these strategies and their consequences for the cognitive load encountered in mathematical problem solving remain poorly understood from either an empirical or a theoretical point of view.[7] Nor have such proposals been developed further in any of the assistive technologies or Web-based applications that enjoy widespread use.

A recently issued accessibility standard (International Organization for Standardization and International Electrotechnical Commission, 2019, § 8.2.3) helpfully distinguishes three successive levels of 'accessibility experience': 'technical', 'effective and efficient', and 'satisfying'.[8] Whereas 'technical' accessibility is achieved by meeting guidelines such as WCAG and other technical requirements, 'effective and efficient' accessibility refers to the ability of users to accomplish tasks using the technology successfully and completely, with appropriate expenditure of time or other resources relative to the accomplished objectives. At the third level, the experience is designed to be 'satisfying/enjoyable'—that is, eliciting positive emotional responses from users. Since the aim of ensuring equality of access to mathematical content is to support full participation in mathematics-related educational and career opportunities, all three levels of accessibility merit attention. Importantly, the cognitive demands of understanding and working with mathematical content imply that effective and efficient performance of a user's tasks, as is essential for success in education or in the workplace, require approaches to user interface design which minimize extraneous cognitive load. Reaching the 'satisfying' level of accessibility, moreover, can be expected to contribute to sustaining users' interest in their work and, at least to some extent, to enhance well-being.

Thus, to direct future software development toward user interface designs that are likely to support users in completing mathematical tasks with effectiveness and efficiency, there is a need for greater understanding of the cognitive implications of alternative design choices. The limited evidence presently available indicates that the differences in cognitive demands resulting from alternative designs can be considerable. By developing task and keyboard-level cognitive models of the best-case performance of screen reader users in solving problems that involved inspection of quadratic equations, da Paixão Silva et al., 2017 demonstrated significant differences in minimum completion times, which were attributable to the user interface choices made in supporting structural navigation within formulae by three different screen readers. Although the details of the authors' findings are dated due to changes in the screen readers themselves,[9] and the simple cognitive models created do not yield much insight into questions of cognitive load, this work illustrates the value of investigating the demands imposed on the user according to different design choices made in supporting

[7]The strategies chosen by students who are blind in performing algebraic manipulation have, however, been compared experimentally with those of sighted students (Fajardo Flores & Archambault, 2012). No significant differences of strategy attributable to vision were found, and demands on working memory were judged to be similar between the two populations.

[8]The purpose of the standard is to institutionalize development and procurement of accessible technologies in the policies and practices of organizations, preferably in ways that lead to outcomes beyond the 'technical' level.

[9]NVDA supports interactive, structural navigation within mathematical expressions (NV Access Limited, 2019, § 7.1), overcoming a limitation identified by the authors as significant.

interactive reading of mathematical expressions by an assistive technology. Future research carried out in conjunction with developments in Web browsers and assistive technologies could profitably investigate the cognitive demands of user interfaces that support reading, navigation, and manipulation of mathematical notation, leading to designs that better support users in completing tasks with efficiency and accuracy. Ultimately, satisfaction is clearly a desirable design objective, attainment of which would constitute true equality for people with disabilities.

CONCLUSIONS

Currently, the needs of users with disabilities in accessing mathematics are partially met at a technical level by the available Web browsers, assistive technologies, and associated tools. Support for reading and interactively navigating mathematical notation has improved among assistive technologies—particularly screen readers—in recent years, although the quality of implementations remains variable, as does the extent of implementation of MathML in Web browsers. These shortcomings in software development have prompted competing architectural responses that remain to be resolved in the evolution of browsers and of Web standards. The challenge of doing mathematics effectively, especially in nonvisual modalities, can best be met by refining user interfaces for applications and assistive technologies through processes that take into consideration the cognitive demands associated with different design options. There is a risk that the costs of technical compromises made in the further development of browsers, assistive technologies, Web standards, and mathematically-oriented applications will rest upon users with disabilities—specifically, that usability will be inadequate, and that accessibility will thus progress only so far as the technical level, without achieving effectiveness, efficiency, or satisfaction. Thoughtful investment in appropriate research and development activities, however, holds the promise of greatly improved outcomes, and of overcoming barriers to equality of access to mathematical notation delivered via the technologies of the Web.

ACKNOWLEDGMENTS

The author gratefully acknowledges insights gained from participants in the Web Accessibility of Mathematics workshop hosted by the American Institute of Mathematics in 2018. He also acknowledges participants in the Chemistry on the Web Community Group, the MathML Refresh Community Group, and the DIAGRAM Center, for contributions at teleconferences held in 2019 in which issues of standardization relevant to the paper were discussed. The author's colleagues at Educational Testing Service, Mark Hakkinen, Heather Buzick, Klaus Zechner, and Shrirang Sahasrabudhe, thoughtfully reviewed the manuscript.

REFERENCES

Alajarmeh, N., Pontelli, E., & Son, T. (2011). From "reading" math to "doing" math: A new direction in non-visual math accessibility, In *International conference on universal access in human-computer interaction*. Springer.

Archambault, D. (2009). Non visual access to mathematical contents: State of the art and prospective, In *Proceedings of the weims conference.*

Bernareggi, C. (2010). Non-sequential mathematical notations in the lambda system, In *International conference on computers for handicapped persons.* Springer.

Bouck, E. C., & Meyer, N. K. (2012). Etext, mathematics, and students with visual impairments: What teachers need to know. *Teaching Exceptional Children*, 45(2), 42–49.

Cervone, D., Krautzberger, P., & Sorge, V. (2016). Towards universal rendering in mathjax, In *Proceedings of the 13th web for all conference*. ACM.

Cervone, D., & Sorge, V. (2019). Adaptable accessibility features for mathematics on the web, In *Proceedings of the 16th web for all 2019 personalization-personalizing the web.* ACM.

da Paixão Silva, L. F., de Faria Oliveira, O., Freire, E. R. C. G., Mendes, R. M., & Freire, A. P. (2017). How much effort is necessary for blind users to read web-based mathematical formulae?: A comparison using task models with different screen readers, In *Proceedings of the xvi brazilian symposium on human factors in computing systems*. ACM.

Dooley, S. S., Osterhaus, S., Brown, D., Lozano, E., & Park, S. H. (2016). Online nemeth braille input/output using content mathml, In *Proceedings of the 13th web for all conference*. ACM.

Fajardo Flores, S., & Archambault, D. (2012). Understanding algebraic manipulation: Analysis of the actions of sighted and non-sighted students, In *The international workshop on digitization and e-inclusion in mathematics and science*. Nihon University Tokyo, Japan.

Flores, S. F., & Archambault, D. (2014). Multimodal interface for working with algebra: Interaction between the sighted and the non sighted, In International conference on computers for handicapped persons. Springer.

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

Igalia, S.L. (2019). *Mathml and browsers*. Retrieved October 22, 2019, from https://mathml.igalia.com/news/2019/08/28/mathml-and-browsers/

International Organization for Standardization and International Electrotechnical Commission. (2019). Iso/iec 30071-1:2019 information technology—development of user interface accessibility—part 1: Code of practice for creating accessible ict products and services.

Isaacson, M. D., Schleppenbach, D., & Lloyd, L. (2010). Increasing stem accessibility in students with print disabilities through mathspeak. *Journal of Science Education for Students with Disabilities*, 14(1), 3.

Karshmer, A., Gupta, G., & Pontelli, E. (2007). Mathematics and accessibility: A survey, In *Proc. 9th international conference on computers helping people with special needs.*

Lewis, P., Noble, S., & Soiffer, N. (2010). Using accessible math textbooks with students who have learning disabilities, In *Proceedings of the 12th international acm sigaccess conference on computers and accessibility*. ACM.

Michigan State University. (n.d.). *Mathtype for equations*. Retrieved February 27, 2020, from https://webaccess.msu.edu/Tutorials/mathtype.html

Michigan State University. (2019). *Msu math portal test server*. Retrieved February 27, 2020, from http://www.msumathonline.com/

NV Access Limited. (2019). *Nvda 2019.2.1 user guide.* Retrieved November 18, 2019, from https://www.nvaccess.org/files/nvda/documentation/userGuide.html#InteractiveNavigation

Raman, T. (1994). *Audio system for technical readings* (Ph.D. dissertation). Cornell University.

Raman, T. (1997). *Auditory user interfaces: Toward the speaking computer.* Kluwer Academic Publishers.

Soiffer, N. (2005). Mathplayer: Web-based math accessibility, In *Acm sigaccess conference on assistive technologies: Proceedings of the 7th international acm sigaccess conference on computers and accessibility.*

Soiffer, N. (2009). A flexible design for accessible spoken math, In *International conference on universal access in human-computer interaction.* Springer.

Soiffer, N., & Noble, S. (2019). Mathematics and statistics. In S. Harper & Y. Yesilada (Eds.), *Web accessibility—a foundation for research* (pp. 417–443). Springer.

Sorge, V. (2016). Supporting visual impaired learners in editing mathematics, In *Proceedings of the 18th international acm sigaccess conference on computers and accessibility.* ACM.

Sorge, V., Chen, C., Raman, T., & Tseng, D. (2014). Towards making mathematics a first class citizen in general screen readers, In *Proceedings of the 11th web for all conference*. ACM.

Stöger, B., Miesenberger, K., & Batušić, M. (2004). Mathematical working environment for the blind motivation and basic ideas, In *International conference on computers for handicapped persons.* Springer.

W3C MathML Refresh Community Group. (2019). *Mathml core* (Editor's Draft). https://mathml-refresh.github.io/mathml-core/

World Wide Web Consortium. (2014). *Mathematical markup language (mathml) version 3.0 2nd edition* (W3C recommendation). http://www.w3.org/TR/2014/REC-MathML3-20140410/

World Wide Web Consortium. (2017). *Accessible rich internet applications (wai-aria) 1.1* (W3C Recommenda- tion). https://www.w3.org/TR/wai-aria-1.1/

World Wide Web Consortium. (2018). *Web content accessibility guidelines (wcag) 2.1* (W3C Recommendation). https://www.w3.org/TR/2018/REC-WCAG21-20180605/