Automatic Tagging of Formulae in PDF Documents and Assistive Technologies for Visually Impaired People: The LaTeX Package `axessibility` 3.0

Dragan Ahmetovic[1], Tiziana Armano[2], Cristian Bernareggi[1], Anna Capietto[2], Sandro Coriasco[2], Boris Doubrov[3], Alexandr Kozlovskiy[4], and Nadir Murru[2]
[1] Universit`a degli Studi di Milano, Department of Computer Science, Italy
[2] Universit`a degli Studi di Torino, Department of Mathematics G. Peano, Italy
[3] Dual Lab, Belgium
[4] Dual Lab Bel, Belarus

Introduction

Assistive technologies for visually impaired people (e.g., screen readers, Braille displays, magnifiers) work well with digital documents containing structured text. On the other hand, when digital documents contain mathematical formulae, there are still many issues concerning the accessibility that should be addressed. In the recent years, many improvements have been achieved, but a comprehensive solution is still far to be obtained.

For instance, different multimodal systems to write and read scientific documents through nonvisual tools have been developed. One of the most used tools is the LAMBDA editor [2], that allows blind people to write and process text and mathematical formulae through Braille display and speech output. However, LAMBDA is not a mainstream tool to produce accessible scientific content by sighted people. Another way for allowing the reading of digital scientific documents by visually impaired people involves the use of MathML in web pages, also through MathJax (see, e.g, [5]). Indeed, MathML, being a markup language intended to the writing of formulae, can be interpreted by most common screen readers to generate a verbal description of the formula [3, 16]. Moreover, MathPlayer, a web browser plug-in for rendering MathML on the screen, through speech output and on Braille devices, enables hierarchical navigation of mathematical formulae, including bi-dimensional notations used, e.g., for matrices [14]. MathJax can be embedded in web pages to enable adaptable accessibility features for representing and navigating formulae (e.g., LaTeX, ASCIIMath or CSS representation; [6, 7]). However, MathML is not used for authoring documents but only for displaying.

The LaTeX language can overcome the above issues, because it is widely used by the academic community for writing scientific documents and producing PDF documents. Several works [8-10], [13], [15], [18] exploit LaTeX in different ways for improving the accessibility of scientific documents, both for the writing and the reading. Unfortunately, since these tools are produced for a small community, due to the rapid evolution of technology, they often incur in maintenance and compliance issues. Therefore, in general, the PDF documents obtained from LaTeX were not accessible, because a tagged structure is missing and the formulae are not readable at all by screen readers. It may be possible to add accessibility features to mathematical content as alternate text and to tag manually the structure of the obtained PDF documents. It can be specified manually using, for example, a proprietary editor such as Adobe Acrobat. Guidelines have been produced to create accessible PDF according to this procedure [17] with a focus on mathematical content [11, 12], [4]. However, this approach requires the availability of a suitable editor, and it entails additional labor from the document author. Furthermore, alternate text most often does not carry the same semantic value as the original mathematical content.

In a previous work [1], we presented a preliminary version of a new LaTeX package that allows to produce accessible formulae in the PDF documents by automatically adding a textual replacement corresponding to the LaTeX commands that generate the formulae. This prototype provided just a

first, partial solution to the problems illustrated above: only some environments for inserting formulae were managed there, and no tagged structure was generated. Moreover, the solution leveraged undocumented proprietary features of PDF readers in order to work. In this paper, we present the last updated version of this package, now named `axessibility.sty`, available on the CTAN repository and also present in the current TeXLive distributions. The formulae are now marked and described using both the `/Alt` and `/ActualText` attributes in the PDF document, and many more environments are considered. In particular, also multiline structures are now managed. Furthermore, we are also able to produce some tagged structures in the PDF document, and developing additional functionalities, to be implemented in subsequent versions.

The LaTeX package `axessibility` 3.0

In the first version of the package, our approach required the `accsupp.sty` package, which was used in order to inject PDF `/ActualText` commands for (inline and displayed) formulae into the output file. A subsequent version expanded this functionality to multiline displayed formulae environments. In this most recent update we added the option of using instead the `tagpdf.sty` package, through which each inline or displayed formula in the source LaTeX document is wrapped into a marked content sequence. In addition, the original formula is added to this marked content sequence as `/ActualText` and `/AltText`. These properties are read by screen readers and braille displays instead of the ASCII representation of the formula, which is often incorrect. Additionally, the package adds a tagged PDF structure. This includes, at the moment, the top level document structure element, to mark the beginning and the end of the document, and the P (paragraph) tag for each formula.

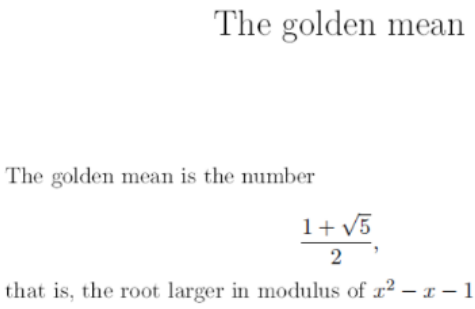
To create an accessible PDF document for visually impaired people, the authors just need to include the `axessibility.sty` package into the preamble of their LaTeX project. The supported mathematical environments will then automatically produce the `/ActualText` and `/AltText` contents and include them in the produced PDF file. Formulae will also be automatically tagged, as well as the document environment. The tagging of other text tokens (paragraphs, sections, etc.), at the moment, has to be inserted manually, under the guidelines of the `tagpdf.sty` package.

The environments for writing formulae which are presently supported are `\(`, `\[`, `equation*`, `equation`, and all the environments present in the `amsmath.sty` package for multiline formulae. Hence, any formula inserted using one of these environments is accessible and tagged in the corresponding PDF document. The click-copy of the formula LaTeX code from the PDF reader, to be pasted elsewhere, works if the screen reader is active. In Figure 1, we report the use of the `axessibility.sty` package in a simple LaTeX document, together with the corresponding source code of the PDF output.

Additional Tools and Features

In addition to the `axessibility.sty` package, we developed additional software to address two use cases: preprocessing scripts to support the application of the package on existing documents, and screen reader dictionaries for natural language reading of formulae made accessible with `axessibility.sty`. Inline and displayed mathematical modes activated by the old syntaxes `$...$` and `$$...$$` are not supported by the `axessibility.sty` package (as in the previous versions). An additional issue lies in the usage of userdefined macros in the LaTeX code. While this is a common practice to avoid code repetitions and simplify document authoring, it can limit the accessibility of formulae with `axessibility.sty`. Indeed, `axessibility.sty` is transparent to commands used in math environments, which means that it will include standard LaTeX as well as custom macros within the replacement text.

```
\documentclass[a4paper,11pt]{article}

\usepackage{axessibility}

\title{The golden mean}
\author{}
\date{}

\begin{document}

\maketitle

\tagstructbegin{tag=P}\tagmcbegin{tag=P}
The golden mean is the number
\tagmcend\tagstructend

\[\frac{1 + \sqrt{5}}{2},\]

\tagstructbegin{tag=P}\tagmcbegin{tag=P}
that is, the root larger in modulus of
\tagmcend\tagstructend \auxiliaryspace
\( x^2 - x - 1. \)

\end{document}
```




```
%PDF-1.5
%ÐÔÅØ
11 0 obj
<<
/Length 2504
>>
stream
BT
/F27 17.2154 Tf/F52 1 Tf( )Tj/F27 17.2154 Tf 236.053 660.524 Td [(The)]TJ/F52 1 Tf( )Tj/F27 17.2154 Tf 33.247 0
Td [(golden)]TJ/F52 1 Tf( )Tj/F27 17.2154 Tf 51.215 0 Td [(mean)]TJ
ET
/P <</MCID 0 >> BDC
BT
/F26 10.9091 Tf/F52 1 Tf( )Tj/F26 10.9091 Tf 134.765 588.53 Td [(The)]TJ/F52 1 Tf( )Tj/F26 10.9091 Tf 22.296 0
Td [(golden)]TJ/F52 1 Tf( )Tj/F26 10.9091 Tf 34.352 0 Td
[(mean)]TJ/F52 1 Tf( )Tj/F26 10.9091 Tf 28.925 0 Td [(is)]TJ/F52 1 Tf( )Tj/F26 10.9091 Tf 10.911 0 Td
[(the)]TJ/F52 1 Tf( )Tj/F26 10.9091 Tf 18.683 0 Td [(n)28(u)-1(m)28(b)-28(er)]TJ
ET
EMC
/P <</MCID 1
/Alt <FEFF0020005C00660072006100630020007B00310020002B0020005C00730071007200740020007B0035007D007D007B0032007D002C>
/ActualText <FEFF0020005C00660072006100630020007B00310020002B0020005C00730071007200740020007B0035007D007D007B0032007D002C>
>> BDC
BT
/F15 10.9091 Tf/F52 1 Tf( )Tj/F15 10.9091 Tf 278.974 564.088 Td [(1)-222(+)]TJ/F48 10.9091 Tf/F52 1 Tf( )Tj/F48 10.9091
Tf 18.788 9.024 Td [(p)]TJ
ET
```

Fig.1: A short LaTeX document, employing the package `axessibility`; below, the source of the typeset PDF document, where it is highlighted the hexadecimal code corresponding to the LaTeX command `\frac{1 + \sqrt{5}}{2}`, automatically generated by the package.

However, custom commands used by an author may bear no meaning for other readers. To (partially) overcome these issues, we developed *Axesscleaner*, an external script written in Python and Perl, through which it is possible to substitute unsupported environments with suitable replacements. It also replaces user defined macros with their content, in order to only contain standard LaTeX code within the replacement text. An alternative approach requires the typesetting with LuaLaTeX (see the package documentation).

Mathematical formulae included as replacement text using `axessibility.sty` are easy to read by LaTeX proficient users, using either a screen reader or a braille display. However, for novice users, the LaTeX code read by a screen reader may be difficult to comprehend. To address this problem, we also provide dictionaries for NVDA and JAWS screen readers, which convert LaTeX commands contained within the replacement text created by `axessibility.sty` into their natural language counterparts (e.g., '\frac{1}{2}' becomes "one half").

All the above tools, including `axessibility.sty`, are available at our github repository `https://github.com/integr-abile`

Finally, we highlight that our package can be used for uploading accessible papers on arXiv. In particular, it is sufficient to add our package, selecting the `accsupp` option, and the auxiliary file 00README.XX just containing the text `nostamp` (this allows to avoid errors in the production of the corresponding PDF file).

Future Work

We are currently working on a new update of the package, in order to

1. provide the automatic tagging of all paragraphs, section headers, etc.
2. convert the LaTeX code into MathML and embed it in the PDF document
3. automatically manage the environments that are not currently supported

Moreover, we are currently developing additional scripts for NVDA, using sophisticated natural language processing techniques, to personalize formula reading considering their complexity and context. In addition, these scripts will enable an interactive navigation of formulae, allowing to move between elements of the formula with hotkeys. The scripts will be developed for the NVDA based on Python 2.X, and will be updated for the NVDA version based on Python 3.X, when the latter will be more stable.

References

1. Armano T., Capietto A., Coriasco S., Murru N., Ruighi A., Taranto E. (2018) An Automatized Method Based on LaTeX for the Realization of Accessible PDF Documents Containing Formulae. In: Miesenberger K., Kouroupetroglou G. (eds) Computers Helping People with Special Needs. ICCHP 2018. Lecture Notes in Computer Science, vol 10896. Springer, Cham. https://doi.org/10.1007/978-3-319-94277-3_91
2. Bernareggi C. (2010) Non-sequential Mathematical Notations in the LAMBDA System. In: Miesenberger K., Klaus J., Zagler W., Karshmer A. (eds) Computers Helping People with Special Needs. ICCHP 2010. Lecture Notes in Computer Science, vol 6180. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14100-3_58.
3. C. Bernareggi and D. Archambault. Mathematics on the web: emerging opportunities for visually impaired people. In Conference on Web accessibility. ACM, 2007.
4. M. Borsero, N. Murru, and A. Ruighi. Il LaTeX come soluzione al problema dell'accesso a testi con formule da parte di disabili visivi. ArsTeXnica, 2016.
5. D. Cervone. Math jax: A platform for mathematics on the web. Notices of the American Mathematical Society, (59):312–316, 2012.
6. D. Cervone, P. Krautzberger, and V. Sorge. Towards Universal Rendering in MathJax. In Proceedings of the 13th Web for All Conference, W4A '16, pages 4:1–4:4, New York, NY, USA, 2016. ACM.
7. D. Cervone and V. Sorge. Adaptable Accessibility Features for Mathematics on the Web. In Proceedings of the 16th Web For All 2019 Personalization - Personalizing the Web, W4A '19, pages 17:1–17:4, New York, NY, USA, 2019. ACM.
8. A. Manzoor, S. Arooj, S. Zulfiqar, M. Parvez, S. Shahid, and A. Karim. ALAP: Accessible LaTeX Based Mathematical Document Authoring and Presentation. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI '19, pages 504:1–504:12, New York, NY, USA, 2019. ACM.
9. A. Manzoor, M. Parvez, S. Shahid, and A. Karim. Assistive Debugging to Support Accessible LaTeX Based Document Authoring. In Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS '18, pages 432–434, New York, NY, USA, 2018. ACM.

10. Melfi G., Schwarz T., Stiefelhagen R. (2018) An Inclusive and Accessible LaTeX Editor. In: Miesenberger K., Kouroupetroglou G. (eds) Computers Helping People with Special Needs. ICCHP 2018. Lecture Notes in Computer Science, vol 10896. Springer, Cham. https://doi.org/10.1007/978-3-319-94277-3_90
11. R. Moore. Ongoing efforts to generate tagged PDF using pdfTEX. TUGboat, Vol.30, No 2, 2009.
12. R. Moore. PDF/A-3u as an Archival Format for Accessible Mathematics. In Watt. CICM, 2014.
13. Pepino A., Freda C., Ferraro F., Pagliara S., Zanfardino F. (2006) "BlindMath" a New Scientific Editor for Blind Students. In: Miesenberger K., Klaus J., Zagler W.L., Karshmer A.I. (eds) Computers Helping People with Special Needs. ICCHP 2006. Lecture Notes in Computer Science, vol 4061. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11788713_169
14. N. Soiffer. Mathplayer: web-based math accessibility. In Conference on Computers and Accessibility. ACM, 2018.
15. V. Sorge. Supporting Visual Impaired Learners in Editing Mathematics. In Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS '16, pages 323–324, New York, NY, USA, 2016. ACM. Title Suppressed Due to Excessive Length 7
16. V. Sorge, C. Chen, T. V. Raman, and D. Tseng. Towards Making Mathematics a First Class Citizen in General Screen Readers. In Proceedings of the 11th Web for All Conference, W4A '14, pages 40:1–40:10, New York, NY, USA, 2014. ACM.
17. Uebelbacher A., Bianchetti R., Riesch M. (2014) PDF Accessibility Checker (PAC 2): The First Tool to Test PDF Documents for PDF/UA Compliance. In: Miesenberger K., Fels D., Archambault D., Peňáz P., Zagler W. (eds) Computers Helping People with Special Needs. ICCHP 2014. Lecture Notes in Computer Science, vol 8547. Springer, Cham. https://doi.org/10.1007/978-3-319-08596-8_31
18. Yamaguchi K., Komada T., Kawane F., Suzuki M. (2008) New Features in Math Accessibility with Infty Software. In: Miesenberger K., Klaus J., Zagler W., Karshmer A. (eds) Computers Helping People with Special Needs. ICCHP 2008. Lecture Notes in Computer Science, vol 5105. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-70540-6_134