APSR
AUSTRALIAN PARTNERSHIP FOR
SUSTAINABLE REPOSITORIES

Preservation of TeX/LaTeX documents

Ian Barnes
The Australian National University

Friday, 21 July 2006, 4:57:23 PM

Table of Contents

1. Introduction

TeX[1] is a mathematical typesetting markup/macro language. It is used for writing up almost all scholarship in mathematics, most computer science and physics and also for some work in linguistics.

TeX was first created by Donald Knuth in 1977 after he became dissatisfied with the quality of typesetting of his series *The art of computer programming*[2]. Knuth is one of the world's leading computer scientists and has been prominent in the field for over 40 years. The use of TeX exploded after Leslie Lamport created the LaTeX macro package[3] which made TeX much easier to use. The main difference between TeX and LaTeX is that LaTeX is a "document preparation system" that works at the level of structural elements of documents while TeX is directly concerned with visual typesetting details.

In other words, LaTeX is a layer that translates structural markup like `\chapter{Introduction}` into basic TeX commands that say (for example): flush the current page out of the system, spit out a blank page if necessary so that we're on a right-hand page, don't print running heads on this page, skip down 2cm, choose a large bold font, print the word "Chapter", increment the chapter counter and then print it, move down another 5cm, pick an even larger font, print the word "Introduction", skip down another 5cm, set running heads to use the word "Introduction" until further notice, and put a line containing the word "Introduction" and the current value of the page counter into the table of contents.

TeX and LaTeX have many variants. Plain TeX[1] is a set of user-oriented TeX macros designed by Knuth himself. It rests quite closely on the underlying typesetting code and requires authors to do a lot of formatting themselves. AMS-TeX[4] is a TeX macro package specifically for mathematical writing, developed by Michael Spivak for the American Mathematical Society. REVTeX[5] is a macro package designed for the American Physical Society and used by many prominent physics journals. There are many other major and minor macro packages[6], some that stand alone and many that build extra features on top of LaTeX. There is also a complete rewrite of the TeX processor that targets PDF directly[7]. (The original TeX program targets its own DVI "device independent" file format. Third-party processors like dvips then convert the DVI file into Postscript or other formats for printing.) For the rest of this document, I will use the term "TeX/LaTeX" to refer not only to TeX and LaTeX proper, but also to all of these many variants.

Until fairly recently, TeX/LaTeX was the only reasonable option for representing mathematics. Now there is an alternative: MathML[8]. MathML is XML[9], and can be embedded in another XML file format (like XHTML[10] or DocBook XML[11]). If support for MathML is adequate, DocBook XML + MathML would be a good archival format, for reasons discussed elsewhere[12]. Having all documents archived in essentially the same XML format would be advantageous as it would mean archivists only have to worrry about one file format rather than two.

I will address the following questions in this report:

1. Is TeX/LaTeX an adequate long-term preservation format?
2. Is there a viable process for converting TeX/LaTeX into DocBook XML + MathML?
3. How good is the support for MathML (for rendering to the screen by web browsers, and for rendering to PDF by XSL-FO processors)?

I will also cover some other related issues, including the possibility of outsourcing preservation of TeX/LaTeX documents to existing archives like arXiv[13], and using the TeX typsetting engine to produce good quality PDF renditions of XML documents, including those with mathematical content in the form of embedded MathML. I will finish with some recommendations.

2. Is TeX/LaTeX a preservation format?

2.1. Points in favour

Points that support keeping TeX/LaTeX documents in their original format for archiving, rather than attempting to convert them:

- TeX/LaTeX is a very powerful language for expressing mathematics. It is not 100% clear whether MathML is a rich enough target for conversion. It would be totally unacceptable to lose content in the preservation process. (See Section 7.1 below.)
- An enormous amount of material has already been successfully archived in TeX/LaTeX format, for example at the arXiv site[13] The tools work. Because people who write in TeX/LaTeX already have something of a programmer mentality, they seem to be able to follow the instructions and archive their work successfully there, despite all the pitfalls mentioned below.
- The TeX4ht software mentioned in Section 5 below, while it has some problems converting TeX/LaTeX to MathML, has no such problems converting TeX/LaTeX documents to HTML with embedded images for the mathematics. While this results in a very large number of images and rather large documents, it means that TeX/LaTeX documents can easily be viewed over the web via HTML as well as being rendered to PDF.
- TeX and LaTeX are relatively stable (despite the second comment in Points Against below). Compared to activity in the XML world, TeX/LaTeX is slow-moving, tried and tested. The basic TeX processor was written by Knuth, whose reputation is unparallelled, and the source code has been publicly available in a very readable form for many years[14]. Knuth gives out rewards for finding bugs, but these days they are few and far between, and all very minor. The system really works as specified (unlike FO processors for example).

2.2. Points against

Points that suggest we might be better to try converting TeX/LaTeX documents into an alternative archival format:

- Because TeX/LaTeX is a macro language, many mathematicians and computer scientists feel the urge to modify it and add to it. In a quite fundamental way, a TeX/LaTeX file is not a *document*, but a *computer program*, which when run produces a typeset document. This is an extra layer of complexity over documents written in more conventional markup languages. Preservation of TeX/LaTeX documents has a great deal in common with preservation of software: in order to preserve a computer program, you need to preserve the entire environment needed to compile and run it.
- The TeX/LaTeX maintainers occasionally change the definition of the language, making old documents stop working.

  For example, an exam paper that I wrote in LaTeX in June 2004 would not run through the LaTeX processor on my desktop computer in May 2005 when I began to modify it. It turned out that my file contained a user-defined macro called `\marks` (used to indicate how many marks a question was worth, and linking up with a counter that kept track of the total). In the intervening year, the LaTeX maintainers added a new command `\marks` to the core of LaTeX, so that my definition of `\marks` caused a conflict. I have a good enough understanding of TeX/LaTeX that I was able to find and correct this (by renaming my macro to something like `\exam-marks` and replacing all references to it appropriately) but this is not the sort of thing we want happening to documents stored in our repositories.

- There are many different versions of the language, and a huge number of styles, document types and macro packages. Each package evolves over time. A document may only work with one particular version of a package.
- Although LaTeX increased the level of abstraction from Plain TeX, there is still a mix of structural and presentation markup in LaTeX documents. Also some authors write directly in Plain TeX, which is essentially presentation markup only. For long-term preservation, structural markup is preferable.

3. arXiv.org

The arXiv site[15] contains over 370,000 preprints in mathematics, physics and computer science. It stores almost all of them as TeX/LaTeX source, and serves them up as PDF or Postscript. For Postscript output, it gives readers a choice of Type1 or bitmapped fonts, and for the bitmapped fonts a choice of resolution.

The site supports OAI-PMH for metadata.

PDF and Postscript renditions of documents are not stored, but are generated on demand from the source using automated TeX tools.

The arXiv solution puts responsibility on authors to make sure that their submission goes through the automated TeX/LaTeX processing without errors. This works. I suspect that this is because authors who use TeX/LaTeX are already much more prepared (than word processor users for example) to mess around with the technical details of document production, because that is in the nature of the TeX/LaTeX system. In this sense the macro facility that makes TeX a headache for archivists paradoxically makes archiving work better.

“An aspect of preservability is the degree to which a problem is shared by a great many people, or some ‘strong’ community”[16]. The success of arXiv, together with major initiatives like CTAN (the Comprehensive TeX Archive Network) and TUG (the TeX Users Group) is clear evidence that TeX/LaTeX has this strong active user base.

4. MathML

I *think* DocBook XML + MathML is capable of expressing everything that TeX/LaTeX can. That is:

- Certainly DocBook is rich enough to handle the document structure of any TeX/LaTeX document.
- I have not yet come across any mathematics that couldn’t be expressed in MathML. (But that doesn’t mean there isn’t any. I don’t know for example, whether commuting diagrams as used in algebra and topology, and seen in Section 7.5 below, can be described in MathML. I do know that the TeX4ht conversion can’t handle them.)

In practice, the more important question is whether there is an effective conversion process. We don’t want to have to rely on people converting their documents by hand. This is partly because of the work involved, and partly because MathML is so extremely verbose that writing it by hand is almost impossible.

For example, in LaTeX the quadratic formula can be written:

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

whereas in MathML it is:

```
<math xmlns="http://www.w3.org/1998/Math/MathML"
      display="block">
  <mi>x</mi>
  <mo class="MathClass-rel">=</mo>
  <mfrac>
    <mrow>
      <mo>&#x2212;</mo>
      <mi>b</mi>
      <mo>&#x00B1;</mo>
      <msqrt>
        <mrow>
          <msup>
            <mi>b</mi>
            <mn>2</mn>
          </msup>
          <mo>&#x2212;</mo>
          <mn>4</mn>
          <mi>a</mi>
          <mi>c</mi>
        </mrow>
      </msqrt>
    </mrow>
    <mrow>
      <mn>2</mn>
      <mi>a</mi>
    </mrow>
  </mfrac>
</math>
```

Not only is the LaTeX code much more concise, to a mathematician it is also much clearer.

5. Converting TeX/LaTeX documents with TeX4ht

The TeX4ht[17] program uses the TeX/LaTeX processor itself to typeset a document, and then extracts information from the resulting `.dvi` (TeX DeVice Independent) file. This is a very clever approach in that it makes it unnecessary for the conversion tool to replicate all the intricate business of package handling, macro processing and so on. The tool can convert a TeX/LaTeX document into several target formats, including:

- DocBook XML + MathML
- XHTML + MathML
- HTML with images for all mathematics

In my experiments the tool does a good job of converting most mathematics into MathML, although there are some problems:

- It cannot handle some complex TeX/LaTeX macro definitions.
- Occasionally it gets its MathML mixed up so that the order of end tags does not match the corresponding start tags. So for example you will have:

```
<mfrac>
  ...
  <mrow>
  ...
  </mfrac>
  ...
</mrow>
```

  Naturally this causes any XML parser, including the one in the Firefox[18] browser, to reject the document. I have encountered this twice in my experiments, while converting LaTeX and AMS-TeX documents into XHTML+MathML with the `xhmlatex` variant of TeX4ht. The documents are fairly easily fixed by hand, but this defeats the purpose of an automated conversion process.

- It does not manage to convert the “title page” information: title, author names and affiliations etc, into the appropriate DocBook XML equivalent.

These problems are discussed in more detail in the Section 7 below.

6. Support for MathML

Another issue with MathML is the lack of support for it by browsers and rendering software. If MathML is to be a part of a viable alternative preservation strategy for mathematical documents, we will need to be able to render MathML content into viewing formats, specifically something that can be embedded in an HTML document and viewed in a web browser, and something that can be printed on paper, probably PDF.

Rendering preserved material into good viewing formats is not the digital *archivist’s* primary responsibility. In principle, we need to first choose the best possible preservation format, and then if necessary develop the tools needed to provide good access. In practice however, authors are much less likely to deposit documents if they can’t even view them to check that the ingestion process went through without errors, and digital repositories fail in one of their primary functions if researchers cannot access the archived resources.

6.1. Web browsers

The Firefox browser has MathML support for viewing XHTML+MathML files. Generally this requires the user to install some fonts. It also seems to only work if the file has extension `.xhtml` or `.xml`, but fails if it is `.html`. This is the case regardless of whether there is a correct XML declaration at the top of the file. Firefox is available on all major platforms (Windows, Mac OS-X, Linux).

As for other browsers, Design Science have created MathPlayer[19], which enables Microsoft Internet Explorer to display MathML. Safari on MacOS-X does not support MathML.

By contrast, if we store documents in TeX/LaTeX format, we can use the `htlatex` form of TeX4ht to convert to HTML with embedded images. In my experience this is completely reliable and works in all browsers.

There is an accessibility issue to consider here. HTML with embedded images is unlikely to work well in screen readers. The alt text that TeX4ht supplies for the images is unsatisfactory. For example, the alt text associated with the barred integral equation from Section 7.1 below is:

```
"                &#x222B;                &#x222B;
Atu(x)  :=  uQ := -- u(y)dy =  -1--  u(y) dy
                  Q           &#x2223;Q &#x2223;  Q  "
```

The question of what would be suitable alt text is an interesting one. Perhaps supplying the TeX/LaTeX source for the equation would be good enough, on the assumption that readers of mathematical documents would be familiar with TeX/LaTeX markup.

Even the XHTML+MathML version has accessibility problems, since TeX4ht only produces *presentation* MathML, while Glenn Hennesse suggests that for mathematics on the web to be accessible to the visually impaired, it needs to not only be in MathML, but in *content* MathML[20]. Presentation MathML simply describes the appearance of a piece of mathematics, including grouping. Content MathML encodes its meaning, so that software can actually work with the mathematics. For example, a formula encoded in Content MathML could be evaluated by software given values for the variables; a function could be graphed, an algebraic expression could be simplified. By contrast, something encoded in Presentation MathML can only be rendered to different media for viewing or printing[8].

6.2. PDF Rendering

Support for XSL-FO + MathML in FO processors is hard to find. There is no MathML support in the current stable release of FOP, but apparently there is a “demonstration” FOP extension module that handles MathML. I haven’t had a chance to test this, but it’s a promising development. There is no MathML support in XEP. XSL Formatter from Antenna House[21] has an add-on for rendering XSL-FO + MathML to PDF. Their sample PDF page is passable but ugly, certainly not up to the same standard of mathematical typesetting as TeX/LaTeX.

Sebastian Rahtz, the creator of the TEI XSLT stylesheets, has also created PassiveTeX[22], a TeX-based XSL-FO processor. PassiveTeX is a set of TeX macros that allows the TeX processor to understand XSL-FO + MathML natively. In my experiments, the output never looked quite right, so this is not a viable tool at present. Unfortunately development of PassiveTeX is frozen at present[23]. If the project continues, it could become an important component in an XML-based preservation strategy for TeX/LaTeX documents. Having a good-quality FO processor that understands embedded MathML would allow us to store mathematical text as DocBook XML + MathML and render it to PDF. Viewing DocBook XML + MathML online can be done now, by processing the document to XHTML with the Norm Walsh DocBook XSL stylesheets, possibly fixing the header to say XHTML + MathML rather than just XHTML, and then viewing the result with Firefox.

Another possibility for rendering MathML to PDF is to create custom XSLT stylesheets for doing the backward conversion from DocBook XML + MathML to LaTeX. I have some experience writing XSLT stylesheets that produce LaTeX files (not from DocBook but from a very simple custom XML language). This did not include converting mathematical content, but with a bit of time and a competent XSLT programmer who understands TeX/LaTeX, we could produce a solution. This would be an interesting and worthwhile project. It would also allow “round-tripping” of LaTeX documents to and from DocBook + MathML, improving user confidence in the process, and allowing the archival XML to be the reference *version of the document for storing in version control systems and collaboration.*

7. Case studies

The following examples mainly illustrate problems with the TeX4ht processor or with MathML support in web browsers. This is an arbitrary selection based on my experiments. I want to emphasise at this point that these tools and processes work correctly a lot of the time.

7.1. The barred integral

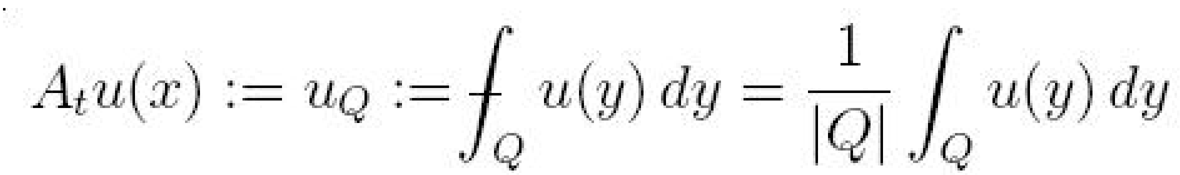
This is from an important mathematics paper entitled *Quadratic estimates and functional calculi of perturbed Dirac operators*[24]. The original LaTeX file was given to me by one of the authors, Alan McIntosh of the ANU's Centre for Mathematics and its Applications. You can see a PDF version of this on the ANU Mathematics web site at http://wwwmaths.anu.edu.au/~alan/papers/AKMcDirac.pdf.

In the document, the authors define a macro `\barint` as follows:

```
\def\barint_#1{\mathchoice
  {\mathop{\vrule width 6pt height 3 pt depth -2.5pt
    \kern -8.8pt \intop}\nolimits_{#1}}%
  {\mathop{\vrule width 5pt height 3 pt depth -2.6pt
    \kern -6.5pt \intop}\nolimits_{#1}}%
  {\mathop{\vrule width 5pt height 3 pt depth -2.6pt
    \kern -6pt \intop}\nolimits_{#1}}%
  {\mathop{\vrule width 5pt height 3 pt depth -2.6pt
    \kern -6pt \intop}\nolimits_{#1}}}
```

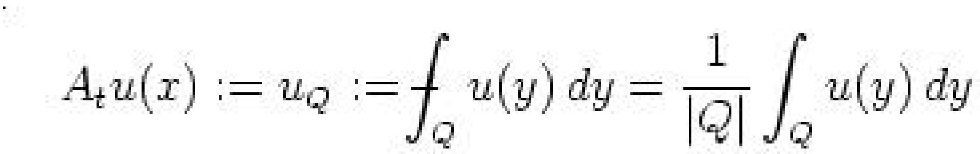
This defines an integral sign with a line through it, representing an average quantity. The different options selected using the `\mathchoice` and `\mathopt` macros (which work more or less like a `switch` statement in C or Java) are about getting the little bar to be the right size and in the right position depending on the size of the integral sign.

You can see the first use of this in Section 5 of the paper (in the PDF this is on page 19, 2nd line from the top).



$$A_t u(x) := u_Q := -\!\!\!\!\!\!\int_Q u(y)\,dy = \frac{1}{|Q|}\int_Q u(y)\,dy$$

The document runs through LaTeX and PDFLaTeX without any problems. It also runs through `htlatex` just fine, producing HTML with images for the mathematics, although it takes quite a while for it to generate the hundreds of PNG images required, and they take up significant disk space. That same equation looks like this in the HTML version:



$$A_t u(x) := u_Q := -\!\!\!\!\!\!\int_Q u(y)\,dy = \frac{1}{|Q|}\int_Q u(y)\,dy$$

You can see that the integral sign with the bar through it doesn't look quite right, but it's clearly still what it's meant to be.

The problems come when we try to convert the mathematics into MathML. I ran the `xhmlatex` program to convert it to XHTML+MathML. There are at least three problems here:

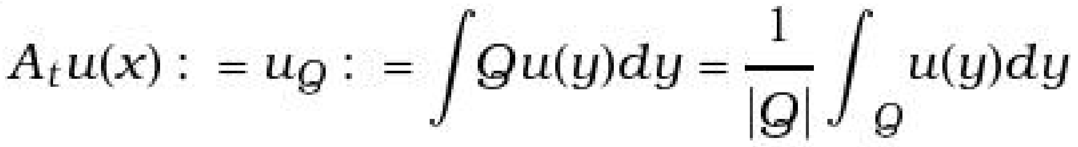
1. The converter can’t handle the use of the underscore character in that macro definition. It crashes attempting to process it.

   This problem is documented, and I was able to make a very simple change to the source file that fixed this.

2. The converter messes up some of the mathematics (not in this particular equation, oddly enough), mismatching a pair of MathML start and end tags as described in Section 5 above. This means that the resulting file is not well-formed XML, and the Firefox browser simply displays an error message.

   After I fixed this in the XHTML file *by hand*, the document parses correctly and Firefox can display it.

3. The conversion doesn’t understand the definition of *\barint*, and important information in the document is lost.

   $$A_t u(x) : = u_Q : = \int Q u(y)dy = \frac{1}{|Q|}\int_Q u(y)dy$$

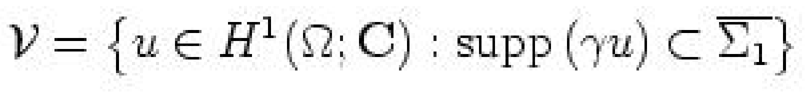
   Comparing this image with those above, you can see that the bar through the integral sign has disappeared, and that the following *Q* has become part of the integrand rather than being placed at the foot of the integral sign where it represents the region of integration.

This is unacceptable. Anyone competent to read the paper would be able to figure out that something was wrong there, and probably correct it, but they shouldn’t have to. While the first two errors would be immediately apparent to anyone attempting to convert the paper, this third error is silent and simply changes its content without warning.

7.2. The missing script capital V

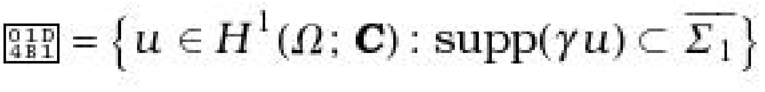
In another paper, the same authors[25] use a script capital V to represent an important quantity. This passes through all the processing steps without problems, but even after installing all the relevant fonts, I still find that Firefox can’t display it.

Here is what it looks like in the HTML version with embedded images:



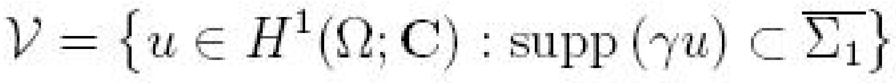
$$\mathcal{V} = \left\{u \in H^1(\Omega; \mathbf{C}) : \mathrm{supp}\,(\gamma u) \subset \overline{\Sigma_1}\right\}$$

Contrast this with the XHTML+MathML version:



$$\boxed{\substack{01D\\4B1}} = \left\{u \in H^1(\Omega; \mathbf{C}) : \mathrm{supp}(\gamma u) \subset \overline{\Sigma_1}\right\}$$

Clearly I don’t have a font with character *x01D4B1*. Comparing with the PDF version:



$$\mathcal{V} = \{u \in H^1(\Omega; \mathbf{C}) : \operatorname{supp}(\gamma u) \subset \overline{\Sigma_1}\}$$

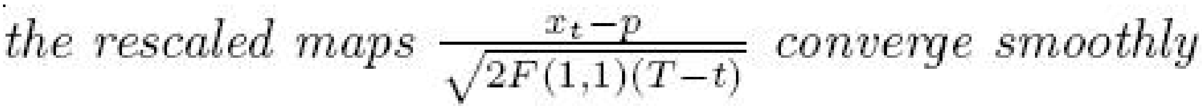
shows that apart from that missing character, both conversions are quite satisfactory.

7.3. Problems with AMS-TeX

In AMS-TeX[4] (the old AMS-TeX, *not* LaTeX with the AMS article document style) the topmatter gets all messed up by the conversion process, even the simplest version converting to HTML with embedded images. The actual content goes through OK, but any structural understanding of it is lost. It appears in the HTML file as plain text before the start of the HTML document, as seen in this example from Ben Andrews[26].

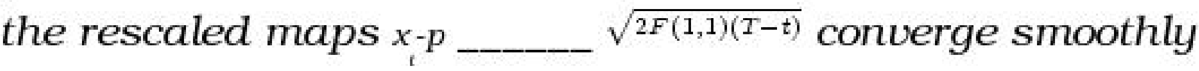
```
MOVING  SURFACES  BY  NON-CONCAVE
                    CURVATURE  FUNCTIONS
                            BEN ANDREWS
                Centre for Mathematics and its Applications
                     Australian National University
     ABSTRACT. A convex surface contracting by a strictly monotone,
homogeneous degree one function of curvature remains smooth until it
contracts to a point in finite time, and is asymptotically spherical
in shape.  No assumptions are made on the concavity of the speed as a
function of principal curvatures.

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html >
<head>
_________
    1991 Mathematics Subject Classification.  35K55, 35B65.
<title>2DsurfN.html</title>
```

Also in AMS-TeX, fractions created using `\over` don’t work. Those created using `\frac{numerator}{denominator}` seem to work fine. Compare this:



the rescaled maps $\frac{x_t - p}{\sqrt{2F(1,1)(T-t)}}$ *converge smoothly*

from the PDF version of the same paper, with this:



the rescaled maps x_t-p ______ $\sqrt{2F(1,1)(T-t)}$ *converge smoothly*

from the HTML version with embedded images.

7.4. Problems with DocBook

When using the dbmlatex variant of TeX4ht to convert from LaTeX to DocBook + MathML, the title, authors’ names etc don’t get captured correctly. For example, this is from the beginning of the article mentioned above in Section 7.1[24], as converted to DocBook XML:

```
<article>
```

```
  <section role="maketitle">
    <title />
    <para role="title">
      QUADRATIC ESTIMATES AND FUNCTIONAL
      CALCULI OF PERTURBED DIRAC OPERATORS
    </para>
    <para role="authorgroup">
      <author>
        <personname>
          <othername>
            ANDREAS AXELSSON, STEPHEN KEITH,
            AND&#x00A0;ALAN McINTOSH
          </othername>
        </personname>
      </author>
    </para>
    <note role="abstract">
      <title>ABSTRACT. </title>
      <para>We prove quadratic estimates...
```

This should have ended up looking something like this:

```
<article>
  <articleinfo>
    <title>Quadratic estimates and functional
      calculi of perturbed Dirac operators</title>
    <authorgroup>
      <author>
        <firstname>Andreas</firstname>
        <surname>Axelsson</surname>
      </author>
      <author>
        <firstname>Stephen</firstname>
        <surname>Keith</surname>
      </author>
      <author>
        <firstname>Alan</firstname>
        <surname>McIntosh</surname>
      </author>
    </authorgroup>
    <abstract>
      <title>Abstract</title>
      <para>We prove quadratic estimates...
```

While this sort of muddle doesn’t look as if it would be too hard to fix, it’s a sign that the technology is not yet mature enough to use for automated conversion to preservation formats on a large scale.

7.5. Commuting diagrams

It’s not all bad news, however. Even typesetter’s nightmares like commuting diagrams in algebra and topology convert perfectly to HTML with embedded images[27]. First, here’s a diagram for the definition of projective:

.

$$\begin{tikzcd}
 & P \arrow[d, "f"] \arrow[dl, dashed, "g"'] & \\
A \arrow[r, "\epsilon"] & B \arrow[r] & 0
\end{tikzcd}$$

Now a much more complicated diagram:

.

$$\begin{tikzcd}
A \arrow[rr, "\phi"] \arrow[dr, "\alpha"] \arrow[dd, "f"'] & & B \arrow[dr, "\beta"] \arrow[dd, "g"] & \\
 & A' \arrow[rr, "\phi'"] \arrow[dd, "f'"'] & & B' \arrow[dd, "g'"] \\
C \arrow[rr, "\psi"] \arrow[dr, "\gamma"] & & D \arrow[dr, "\delta"] & \\
 & C' \arrow[rr, "\psi'"] & & D'
\end{tikzcd}$$

A pullback diagram:

.

$$\begin{tikzcd}
Y \arrow[dr, "\theta"] \arrow[drr, "\phi"] \arrow[ddr, "\psi"'] & & \\
 & X \arrow[r, "p^*f"'] \arrow[d, "f^*p"] & E \arrow[d, "p"] \\
 & A \arrow[r, "f"] & B
\end{tikzcd}$$

A morphism of pullbacks:

.

$$\begin{tikzcd}
X_P \arrow[rrr] \arrow[dr, "\phi_X"] \arrow[ddd] & & & E_P \arrow[dl, "\phi_E"'] \arrow[ddd] \\
 & X_Q \arrow[r] \arrow[d] & E_Q \arrow[d] & \\
 & A_Q \arrow[r] & B_Q & \\
A_P \arrow[rrr] \arrow[ur, "\phi_A"] & & & B_P \arrow[ul, "\phi_B"']
\end{tikzcd}$$

Finally, a triangular diagram:

$$\begin{tikzcd}
A \arrow[rr, "\phi_A"] \arrow[dr, "\phi"'] & & \mathcal{O}(\mathrm{Spec}\ A) \arrow[dl, "\mathcal{O}(f)"] \\
& \mathcal{O}(X) &
\end{tikzcd}$$

8. Summary and recommendations

8.1. Summary

I believe that MathML is the way of the future. It is XML, which is a big advantage. Having all documents in DocBook XML (or another XML format like TEI) with embedded MathML for mathematics would bring word processing and mathematical work together. This would be an advantage for repositories in that they would only have to support one file format rather than two.

Browser support for MathML is reasonable now. Users of Internet Explorer will need to purchase an add-on. Firefox users need to install some fonts. This is not perfect, but will clearly improve over time.

Rendering MathML to PDF is more of a problem. Even expensive commercial solutions cannot match the output quality available for free with TeX/LaTeX.

Converting TeX/LaTeX documents to DocBook (or other) XML with embedded MathML is also problematic. The best system available at the moment is TeX4ht, but this introduces some errors, which is unacceptable.

The arXiv site is a successful archive with hundreds of thousands of preprints in TeX/LaTeX format. This proves that the TeX/LaTeX toolchain works for automated processing in large-scale operations.

8.2. Proposed preservation strategy for TeX/LaTeX documents

1. For now, TeX/LaTeX documents should be kept in their original form.

2. University archives could outsource preservation of TeX/LaTeX documents to arXiv at essentially no cost and with little risk. Perhaps it would be good to provide some sort of front end for searching, hiding from users that the documents aren’t stored locally. One way to do this would be for the institutional repository to store document metadata, but in place of the actual document, give a link to arXiv.

3. At the moment arXiv does not do an automated conversion from TeX/LaTeX source to HTML for onscreen viewing in a web browser. Adding this capability would be a worthwhile project. This could either be proposed to arXiv as an additional service on their site, or institutional repositories could retrieve the TeX/LaTeX source from arXiv, convert it to HTML with embedded images (or XHTML+MathML) on the fly, and serve the result to users.

4. For long-term preservation, we should seek eventually to convert all documents to a common XML format. At the moment DocBook XML with embedded MathML looks like a good candidate. (TEI + MathML is

another possibility.) This is not feasible now. The TeX4ht automated conversion still loses information and occasionally produces MathML that is not well-formed. Browser support for MathML is not too bad, but rendering to PDF is still unsatisfactory.

5. In the future, as the software progresses, this will become the best way to archive mathematical works. Authors will probably continue to write using TeX/LaTeX, simply for the convenience and control it gives them, and because it has become part of the culture in mathematics, physics and computer science.

   One possibility is that repositories will do TeX/LaTeX to XML conversion on ingest. Another is that an environment like the Digital Scholar's Workbench could be extended to do conversion and rendering of TeX/LaTeX documents on the fly in the same way as is currently done for word processing documents. In this scenario, the author's work cycle would be: edit the TeX/LaTeX input file, save, go to the browser and hit Refresh, and the workbench application would take the TeX/LaTeX source, convert it to DocBook XML + MathML, and on to HTML for immediate viewing and checking. The DocBook + MathML file would be under version control and would be the definitive version of the document. Authors could do a reverse conversion (round-tripping) to regenerate the TeX/LaTeX source (perhaps for different versions of TeX/LaTeX) from the XML.

9. Acknowledgments

This work was funded by the Australian Commonwealth Department of Education, Science & Training, through the Australian Partnership for Sustainable Repositories, which is part of the Systemic Infrastructure Initiative, part of the Commonwealth Government's "Backing Australia's Ability—An Innovative Action Plan for the Future".

I thank the Australian National University's Digital Resource Services program, part of the Division of Information, for hosting me while I did this work: Program Leader Peter Raftos, APSR Project Leader Adrian Burton and my colleagues Chris Blackall, Leo Monus, Scott Yeadon, Margaret Henty & Fiona Nelson Campbell. I also thank the ANU Department of Computer Science for releasing me from teaching while I did this work.

I thank Ben Andrews, Alan McIntosh, Brendan McKay and Don Barnes for allowing me to play with the TeX/LaTeX sources of some of their work.

Bibliography

[1] Donald E. Knuth, *The TeXbook*, Addison-Wesley (1984). URL: http://www-cs-faculty.stanford.edu/~uno/abcde.html.

[2] Donald E. Knuth, *The art of computer programming (Volumes 1-6)*, Addison-Wesley (1968-1996). URL: http://en.wikipedia.org/wiki/The_Art_of_Computer_Programming.

[3] Leslie Lamport, *LaTeX: A document preparation system*, Addison-Wesley (1984, 1994). URL: http://en.wikipedia.org/wiki/LaTeX.

[4] Michael Spivak, *The joy of TeX: A gourmet guide to typesetting with the AMS-TeX macro package*, The American Mathematical Society (1990). URL: http://www.ams.org/tex/amstex.html.

[5] American Physical Society, *REVTeX 4* (2001). URL: http://authors.aps.org/revtex4/.

[6] The CTAN Team, *The comprehensive TeX archive network* (1992-2006). URL: http://www.ctan.org/.

[7] Han The Thanh, *The pdfTEX Program*, in Proceedings of EuroTeX 1998 (1998). URL: http://www.gutenberg.eu.org/pub/GUTenberg/publicationsPDF/28-29-han.pdf.

[8] World-Wide Web Consortium, *Mathematical Markup Language (MathML) specification* (2003). URL: http://www.w3.org/TR/MathML2/.

[9] World-Wide Web Consortium, *Extensible Markup Language (XML)* (1996-2003). URL: http://www.w3.org/XML/.

[10] World-Wide Web Consortium, *XHTML 1.0: The Extensible HyperText Markup Language specification* (2002). URL: http://www.w3.org/TR/xhtml1/.

[11] Norman Walsh, *DocBook* (1999-2006). URL: http://www.docbook.org/.

[12] Ian Barnes, *Preservation of word processing documents* (2006).

[13] Cornell University Library , *arXiv.org e-Print archive* (1996?-2006). URL: http://arxiv.org/.

[14] Donald E. Knuth, *Computers & Typesetting, Volume B: TeX: The Program*, Addison-Wesley (1986). URL: http://www.amazon.com/gp/product/0201134373/103-8916860-9425469?v=glance&n=283155.

[15] Cornell University Library, *arXiv.org e-Print archive* (1996?-2006). URL: http://arxiv.org/.

[16] Peter Raftos, Personal communication (2006).

[17] Eitan M. Gurari, *TeX4ht: LaTeX and TeX for Hypertext* (2005). URL: http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html.

[18] Mozilla Corporation, *Mozilla Firefox 1.5* (2005-6). URL: http://www.mozilla.com/firefox/.

[19] Design Science, *MathPlayer* (1996-2006). URL: http://www.dessci.com/en/products/mathplayer/.

[20] Glenn Hennessee, *MathML Information and Demos* (Date unknown). URL: http://chem2.chem.ncsu.edu/~hennesse/.

[21] Antenna House, *XSL Formatter V4.0 (with MathML option)* (1996-2006). URL: http://www.antennahouse.com/.

[22] Sebastian Rahtz, *PassiveTeX* (2005). URL: http://www.tei-c.org.uk/Software/passivetex/.

[23] Sebastian Rahtz, Personal communication (2006).

[24] Andreas Axelsson, Stephen Keith & Alan McIntosh, *Quadratic estimates and functional calculiI of perturbed Dirac operators* (2005). URL: http://wwwmaths.anu.edu.au/~alan/papers/AKMcDirac.pdf.

[25] Andreas Axelsson, Stephen Keith & Alan McIntosh, *The Kato square root problem for mixed boundary value problems* (2005). URL: http://wwwmaths.anu.edu.au/~alan/papers/katomixedbvp.pdf.

[26] Ben Andrews, *Moving surfaces by non-concave curvature functions* (2004). URL: http://arxiv.org/pdf/math.DG/0402273.

[27] Donald W. Barnes, Personal communication (2006).