Can LaTeX profiles be rendered adequately with static CSS?

William F. Hammond

Abstract

MathJax demonstrates that heavy customization of CSS with JavaScript and webfonts provides good platform-dependent rendering. The issue with MathJax is speed, not quality. There has been and continues to be intense development with CSS. One may speculate that, as CSS continues to evolve, static CSS may entirely suffice not only for HTML documents with mathematics but also for the direct online rendering of profiled LaTeX documents when presented using XML syntax.

My purpose is to report on some of what can now be done, to indicate how I would like to see things develop, and to stir interest in the LaTeX community for incorporating the ideas of CSS into print typesetting.

1 Background

This article is an elaboration of what I said in my talk at TUG 2014 (slides are available at `http://www.albany.edu/~hammond/presentations/tug2014`). About half the time in the talk was devoted to showing LaTeX profiles styled with CSS in a web browser. In fact, the slides themselves were such. Figure 1 is a screenshot of static CSS styling for math in a LaTeX profile. At this point satisfactory results are being obtained with three of the major browsers using three different wide-coverage Unicode fonts. (Corresponding to the fonts there are three different stylings that differ from each other, aside from font invocations, only in handling a few things.)

For those who have less than satisfactory interaction with the XML slides, there is an HTML version provided at the previous url. Aside from the slideshow poster, which was written directly in HTML, the slides and all of the materials linked therefrom originated as source prepared for the LaTeX profile of the GELLMU Didactic Production System, `http://www.albany.edu/~hammond/gellmu/`, which may be regarded as a base for spawning other profiles and which will be used as the profile of discourse here.

CSS [1] stands for *Cascading Style Sheets*. It is the standard design language used in presenting HTML (Hypertext Markup Language), as well as XML (eXtensible Markup Language) [2] applications, in web browsers.

A LaTeX profile [8] is a dialect of LaTeX [4] with a fixed command vocabulary, where all macro expansions must be effective in that vocabulary, having a well-defined XML shadow. LaTeX profiles are suitable domains for defining reliable translations to other profiles and, where sensible, to other markup languages.

The author recalls a conversation at a meeting in 2002 on the question of whether CSS might someday suffice for rendering segments of MathML (Mathematical Markup Language) in web pages. The participants agreed that it might be possible, but a sufficiently wide deployment of sufficiently capable CSS engines would then be the issue.

By early 2006 I had begun writing a CSS stylesheet for the XML guise of the LaTeX profile of the GELLMU didactic production system [5, 6, 7]. Especially where mathematics is concerned, the richness of the vocabulary of profiled LaTeX compared to the vocabulary of MathML makes it easier to think about[1] rendering math with CSS. The CSS rendering of profiled LaTeX in 2006 was quite limited, not remotely close in quality to the rendering of MathML in a browser like Firefox, but still of some use.

By 2010 Davide Cervone's MathJax, `http://www.mathjax.org`, if not Cervone's earlier project *jsmath*, had demonstrated that with heavy (and slow) JavaScript-based customization for available fonts, web browser, and computing platform, MathML could be rendered with CSS.

During the time that MathJax was being developed George Chavchanidze of Opera Software had been pursuing the idea of basing native browser support for MathML solely on static CSS.[2] Following that work the W3C Math Working Group in 2011 produced a W3C recommendation entitled "A MathML for CSS Profile" [9] that suggested restricted use of MathML by content generators interested in having their content rendered with CSS.

In early 2014 I learned about Frédéric Wang's idea, following the earlier work of Chavchanidze and quite apart from MathJax, of providing "fallback" rendering of MathML in web browsers lacking native support for MathML, taking advantage of relatively new ideas in CSS, particularly CSS flexible boxes [10], without heavy customization for particular circumstances.

I have spent the last few months seeing how these new ideas in CSS could be used to improve the static CSS presentation of GELLMU's LaTeX profile. It is work in progress. My purpose is to report on some of what can now be done, to indicate how I

[1] (but probably not actually easier)

[2] Followers of LaTeX3 might think of this as an effort, unlike that in LaTeX3, to leverage the *design layer* without much in the way of apparent new support from the *programming layer*.

$$\cfrac{1}{1+\cfrac{e^{-2\pi\sqrt{5}}}{1+\cfrac{e^{-4\pi\sqrt{5}}}{1+\cfrac{e^{-6\pi\sqrt{5}}}{1+\cdots}}}} = \left(\cfrac{\sqrt{5}}{1+\sqrt[5]{5^{3/4}\left(\cfrac{\sqrt{5}-1}{2}\right)^{5/2}-1}} - \cfrac{\sqrt{5}+1}{2}\right) e^{2\pi/\sqrt{5}}$$

Figure 1: Static CSS styling of profiled LaTeX

would like to see things develop, and to try to stir the interest of the LaTeX community in incorporating CSS ideas into print typesetting.

2 Processing

In another talk at TUG 2014, by S. K. Venkatesan and C. V. Rajagopal, one of the slides had this poetic line:

> TeX is poured into the XML mould, and DTD is used as the sieve.

"DTD" refers to the document type definition. Document type definitions are in one-to-one correspondence with LaTeX profiles. There are at least two reasons for sifting:

1. To know that a correctly written processor will reliably produce correct results.
2. To put the LaTeX under a framework that facilitates processing by any of the many software libraries, written in various programming languages, that operate on XML.

Those interested will be able to find more information about this in many places including, for example, *The LaTeX Web Companion* [3].

For rendering a LaTeX profile with CSS a small amount of "server-side" processing (independent of fonts, browser, and platform) may be used to dress the regular XML guise of the LaTeX so that it may be more easily addressed with CSS. In this report the main concern is with mathematics, but I should note that the whole of a profiled LaTeX document must be styled with CSS for presentation in a web browser.

2.1 Source

I will illustrate briefly with a tiny example under GELLMU's LaTeX profile. This is the source `tg.glm`:

```
\documenttype{article}
\title{test}
\begin{document}

One has
\[ \Gamma(3) = 2! \]
\end{document}
```

2.2 Author-level XML

Under main track GELLMU processing, based only on syntax, not vocabulary, this resolves to the following XML instance `tg.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css"
   href="gellmuart.css"?>
<!DOCTYPE article SYSTEM "axgellmu.dtd">
<article stem="tg">
<preamble><title>test</title></preamble>
<body>
<parb>One has
<displaymath>
<Gamma/>(3)<eqs/>2<exc/>
</displaymath>
</parb>
</body>
</article>
```

The XML tags should be self-explanatory except for `<parb>`, which indicates the paragraph begun with a blank line, the empty element `<eqs/>` coming from the "=" (that makes it possible for downstream decisions to be made on the "="), and the `<exc/>` coming from the "!".

2.3 Elaborated XML

In main track processing under the GELLMU Didactic Production System this "author-level" XML that closely shadows the original source is processed, preparatory to translation toward either regular LaTeX or HTML, to the following elaborated XML instance `tg.exml`:

```
<?xml version="1.0" encoding="UTF-8">
<?xml-stylesheet type="text/css"
  href="gellmuart.css"?>
<?centralStyled?>
<!DOCTYPE article SYSTEM "uxgellmu.dtd">
<article stem="tg">
<preamble><title>test</title></preamble>
<body>
<parb>One has
<displaymath>
```

```
<Gamma/>(3)<equals/>2<exc/>
</displaymath>
</parb>
</body>
</article>
```

The only change noticeable in this very simple example is that the "=" has now become `<equals/>` because it is within a mathematical container. (Other changes that would happen with a more complicated document would be resolution of cross-references and assignment of section numbers.)

2.4 Dressed XML

To prepare for CSS it is necessary that every nugget of character data inside math zones be wrapped in a tag indicating whether the nugget is numeric, word-like, or operator-like, so that the questions of whether to use an upright or italic font and how to space may be addressed.

Such "dressing for CSS" leads to `tg-lm.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css"
  href="gellmualm.css"?>
<!DOCTYPE article SYSTEM "vxgellmu.dtd">
<article stem="tg">
<preamble><title>test</title></preamble>
<body>
<parb>One has
<displaymath mlvl="1"
   mchld="me|bal|me|mx|me">
<me name="Gamma" mlvl="2" mchld="Gamma"
   ><Gamma/></me>
<bal mlvl="2" mchld="mx"
    ><mx type="number">3</mx></bal>
<me name="equals" mlvl="2" mchld="equals"
   ><equals/></me>
<mx type="number">2</mx>
<me name="exc" mlvl="2" mchld="mx"
   ><mx type="character">!</mx></me>
</displaymath>
</parb></body>
</article>
```

One will see that the mathematics has been greatly elaborated. The elaboration creates hooks for possible use in CSS selectors. Note, in particular, the values of the attribute `type` on the element `<mx>`. The attribute `mlvl` on almost every element inside a math zone indicates how deep that element is in the math zone (a selection issue not foreseeably addressable by CSS). The attribute `mchld` on math containers lists the names of the child elements (another selection issue not currently addressable in CSS).

2.4.1 Balancing parentheses

Another thing to notice is that

```
<Gamma/>(3)
```

has become (with some simplification for human clarity):

```
<Gamma/><bal><mx type="number">3</mx></bal>
```

That is, at this stage of processing the parentheses have been replaced with the element `<bal>...</bal>`, "bal" for "balanced", which at the source level is `\bal{...}`, corresponding to `\left(...\right)` in regular LaTeX.

2.4.2 A bit of CSS for `<mx>`

By way of example, relevant code from the linked CSS, available from `http://www.albany.edu/~hammond/webstyle/gellmualm.css`, includes:

```
mx[type="letter"] {
  font-style: italic;
}
mbox mx[type="letter"] {
  font-style: normal;
}
mx[type="number"] {
  font-style: normal;
  font-size: 0.92em;
}
[mlvl="1"]>mx[type="number"] {
  font-style: normal;
  font-size: 1em;
}
```

Many things in the style sheet are debatable, even questionable. In particular, the font-size adjustment for `mx[type="number"]` is based on a personal judgement that numbers at levels greater than 1 were too large compared to letters.

As mentioned above, the name of the dressed XML instance is `tg-lm.xml`. The pre-suffix "`-lm`" is a mnemonic reference to the "Latin Modern" font, matching the "`lm`" in the name `gellmualm.css` of the linked style sheet. Because at this time there is a certain lack of standardization in fonts used on the web, I am providing a different style sheet for each font. This must mean, of course, that the style sheet is being used to control the font, which, in turn, means that I must be serving web fonts.

3 Fonts

For most of the history of the World Wide Web the fonts used in web browsers have been the fonts found on the user's computer. Relatively recently what are called "web fonts" have appeared and have gained support in major web browsers. Web fonts are

fonts served through the web, usually from the site where an HTML document is posted. The MathJax project has made a great deal of use of web fonts. My understanding is that web fonts may be made from either OpenType fonts or TrueType fonts.

I know relatively little about fonts, whether in the world of the web or in the LaTeX world. A few of the things that I have learned include:

- Unicode fonts, i.e., fonts whose glyphs are referenced by the Unicode value for the corresponding character, are becoming the standard both on the web and for LaTeX with the new TeX engines (*xetex* and *luatex*).
- OpenType fonts seem to be becoming the standard.
- A free program by Jonathan Kew of Mozilla (the originator of *xetex*) called *sfnt2woff*[3] will convert an open type font file, say `foofont.otf`, or a true type font file, say `foofont.ttf`, to a web font (`foofont.woff`).
- A CSS `@font-face` directive is used to tie a web font on your server to a `font-family` name, possibly also with a `font-weight` and `font-style` specification.

4 Casting for glyphs

There comes the question in styling a LaTeX profile with CSS of what one is to do with the four "casting" commands: `\mathbb`, `\mathcal`, `\mathfrak`, and `\mathscr`. In LaTeX, as one knows, each of them takes as argument a Latin letter and produces, respectively, a double-struck, calligraphic, Fraktur, or script version. Apart from Fraktur, it can be argued that these are mere stylistic variations. But most mathematicians are inclined to regard the original Latin letter and these four casts as five semantically distinct symbols. While in a translation to MathML, it might be reasonable to generate appropriate Unicode points for the casts,[4] that would be a bit out of scale for styling the XML guise of a profiled LaTeX document with CSS where the author has used one of these commands.

Of course, the author could just put the appropriate Unicode points in the source. Aside from that, an off-track approach is to specify a sequence of matching old fonts such as `msbm10`, `cmsy10`, `eufm10`, and `rsfs10` for the four casts.

[3] C source available from `http://people.mozilla.org/~jkew/woff/woff-code-latest.zip` — I found it quite easy to build with `gcc`.

[4] This ignores that the Unicode standard seems to have merged calligraphic and script, and the standard provides only three incomplete "alphabets".

If the Unicode standard were amended to fill the gaps left in the three alphabets that correspond to previously assigned glyphs (not characters in the abstract sense), then in the "dressing process" one could use offsets to those alphabets for `bb`, `frak`, and `scr`.

Finally, by way of trying to nudge the guardians of Unicode, I would like to note that in the document "The STIX Package" (`stix.pdf`) accompanying the 2014 release of the STIX fonts and found in TeX Live 2014, the table in section 3 indicates support for a number of `\mathxx` commands beyond what is provided in Unicode. The author has been shown browser-private CSS properties that may be used to access alternate glyphs for Unicode points in OpenType fonts. This gives hope that eventually there will be better ways to use CSS to access such alternate glyphs.

5 Fractions

Fractions may be rendered reasonably well using CSS tables. There are, however, two important things to note. First, each of the numerator and denominator must be the sole table-cell in a table-row. The bar that separates numerator and denominator may be provided as the "collapsed" border-bottom of the numerator with the border-top of the denominator. This arrangement for the bar works with table-rows. However, table-rows cannot contain essentially arbitrary content, while table-cells can. Thus, the dressing of the profile must re-arrange things to be as if `\frac{a}{b}` had been marked up as `\frac{{a}}{{b}}`. This much is a slight inconvenience but not a problem.

The second thing to note is a problem. There does not seem to be any reasonable way at present to have horizontally adjacent fraction bars align with each other. It is not an obstruction to comprehensibility, but it leaves an appearance one does not want. It is part of a much larger concern with vertical alignment for the math in LaTeX profiles.

The author understands that vertical alignment is a subject of continuing work within the CSS community.

6 Borders as balancers

Previously (in section 2.4.1) I mentioned `\bal{...}` as the profile's version of `\left(...\right)`. At the stage of dressing when all character data in math zones is being wrapped in tags the character "`(`" is replaced with `<bal>` and the character "`)`" is replaced with `</bal>`. Aside from the fact that this is important for trapping the author error of having unbalanced parentheses because the output will not

parse correctly without balance, it is important because CSS (not unlike LaTeX) is largely about boxes. Every XML element gives rise to a box. CSS properties control that box.

One might eventually hope that perfectly sized parentheses for a given box might be provided using CSS as a border segment object. What works now is the following:

```
bal {
  align-self: center;
  display: inline-block;
  margin-left: 0.15em;
  margin-right: 0.15em;
  padding: 0.2ex 0.2em 0.2ex 0.2em;
  border-left: 0.2ex solid;
  border-right: 0.2ex solid;
  border-radius: 0.5em;
}
```

Here is a screenshot that illustrates this handling of `\bal{}`:

$$\frac{\pi}{2}\left(1+\left(\lim_{n\to\infty}\left(\log\bigl(n+1\bigr)-\left(\sum_{k=1}^{n}\frac{1}{k}\right)\right)\right)\right)$$

Notice how each pair of parentheses, allowing for CSS-specified padding of boxes, fits its box precisely. There is no limit to the number of sizes. How easy it could be for an author to omit one of the parentheses at the end if they were all of the same size as here:

$$\frac{\pi}{2}\left(1+\left(\lim_{n\to\infty}\left(\log(n+1)-\left(\sum_{k=1}^{n}\frac{1}{k}\right)\right)\right)\right)$$

Among my wishes for the future of CSS are new border-decoration properties for the four sides of a box that would enable one to have precisely fitting parentheses, braces, and brackets. Further one might wish eventually to be able to attach to a point on the border of a box a small piece of drawing, similar to a picture environment drawing in LaTeX.

Thus, for example, radicals (from `\sqrt[]{}`), which are presently rather fragile with the CSS styling now available (except for the top line that presents well when styled as the `border-top` of the radicand), would be handled better using the `border-top` of the radicand and a segment of the `border-left` of the radicand with a radical hook drawn from the bottom of the segment on the left. Failing that one can even now make Orwellian "victory radicals" consisting simply of the top and left borders of the radicand.

7 Flexible boxes

7.1 *underset*

The LaTeX profile in use here does not, though it certainly could, provide `\lim` as a command. The limit in the previous display is generated as an *underset*. Explicitly, the corresponding source is:

```
\underset{n \rightarrow \infty}{\mbox{lim}}
```

In the XML everything needs a name. The first argument of *underset*, the "decoration", has the name *deco*, while the second argument, the "base", has the name *expr*.[5]

It would be rather heavy-handed to style an *underset* as a table. Instead it uses a new concept in CSS [10]: the flexible column.[6]

The command *underset* is treated in this segment of the CSS stylesheet:

```
underset {
  display: inline-flex;
  flex-direction: column;
  align-items: center;
  vertical-align: text-top;
  justify-content: flex-start;
}
underset > expr {
  order: +1;
  padding: 0;
  margin-top: -0.2ex;
  margin-bottom: -0.3ex;
}
underset > deco {
  font-size: 0.8em;
  padding: 0;
  order: +2;
}
```

The `order` property for the children indicates the order from top to bottom in which the children should be displayed. In this case the first child is to be displayed second and the second child first.[7] The ability to arrange the order of children is useful. (Its usefulness is made less important, however, if the XML is processed for "dress".)

This segment of CSS above is not very robust. The `vertical-align` specification may or may not be what one will ultimately want. If it is used, the idea is to align the *underset* box within the parent. It is not supposed to be effective if the parent is a flexible box (row or column), in which

[5] The argument order for MathML's corresponding *munder* is reversed.

[6] As an exercise, the reader might ponder why it would not work to style a fraction as a flexible column.

[7] With MathML's *munder* it would be first first and second second.

case a property called `align-self` with different permitted values can be used. It's not fully clear which display types of the parent are appropriate if `vertical-align` is to be effective. Moreover, it's not clear what point on a flexible column is being aligned, say, in a parent of type `inline-block`. I can report that for one browser the display changed for *underset* and *overset* with a version upgrade during July, 2014. The top and bottom margin settings for *expr* are a font-dependent attempt to adjust for the inadequacies of vertical-align.

7.2 *sum*-like operators

The sum in the previous display also involves flexible boxes, in this case two of them, one a row and one a column. In the LaTeX profile at hand, sums, integrals, and products are handled together for most purposes, and I call them the "sip" (the first letters of sum, integral, and product) elements. (There could be more of these: unions, intersections, coproducts, ..., but no others are presently in the profile.) Where in regular LaTeX the corresponding commands reference the operator symbols, in the profile they reference the whole structure. Nonetheless, the markup is almost the same as with regular LaTeX except that an explicit termination of the object of the operation is required. For the example in the display of section 6, the markup is

```
\sum_{k=1}^n\frac{1}{k}\sum:
```

This could be marked up in a manner close to its XML guise:

```
\sum{\siphead{\lower{k=1}\upper{n}}
     \sipbody{\frac{1}{k}}}
```

The CSS scheme used is that *sum* is a flexible row, while its first child *siphead* is a flexible column given that its parent is a *displaystyle* sum. At the stage of dressing an *mx* containing the summation symbol is inserted between the *upper* and the *lower*. The *sipbody*, which is the second child of the *sum*, contains the object of the summation — in this case the fraction $1/k$. A listing of the relevant CSS code follows. Note that the property `align-self` should govern vertical alignment of the *sum* in its parent in the case that the *sum* is itself inside a flexible row, while the property `vertical-align` should govern the case that the *sum* is inside an inline block (which I have taken to be the default display style for any expression more complicated than a single symbol).

```
sum, int, prod {
  align-self: center;
  vertical-align: middle;
  margin-left: 0.2em;
  margin-right: 0.2em;
  display: inline-flex;
  flex-direction: row;
  justify-content: flex-start;
  align-items: center;
}
siphead {
  vertical-align: middle;
  align-self: center;
  display: inline-flex;
  flex-direction: column;
  justify-content: flex-start;
  align-items: center;
}
siphead > upper {
  font-size: 0.6em;
  order: -1;
  line-height: 2.5ex;
  min-height: 1.5ex;
  margin-bottom: 0.15ex;
}
siphead > mx {
  order: 0;
}
siphead > lower {
  order: 1;
  font-size: 0.6em;
  line-height: 2.5ex;
  min-height: 2.5ex;
  margin-top: 0.15ex;
}
sipbody {
  display: inline-block;
  align-self: center;
  padding-left: 0.05em;
}
```

8 Why?

It is important to explore all avenues for making mathematical content fully available online in proper online formats. This includes the world of small screens as found on "smart phones" and the world of "e-books".

One hopes that the design concepts in CSS[8] for online content eventually become adequate for handling mathematics — they are not far from that now — and that those concepts come to be supported in all major web browsers.

With fully robust static CSS for LaTeX profiles the gain for the online presentation of mathematics could be:

1. Faster browsing of math online.

[8] general concepts not particularly tied to mathematics

2. The potential for greater control in the presentation of math online.
3. Elimination of the need for special handling of math in the online world.

If this future is realized, there are two other things to note:

- A LaTeX profile can be robustly translated (on the server side) to HTML with MathML. In that process the output can be "dressed" so that it can be styled with static CSS in a way that has an almost identical presentation in web browsers to that of the profile itself presented with its static CSS. There should be no need to observe the restrictions of "MathML for CSS" [9].
- Web-served HTML with MathML will have the advantage over web-served LaTeX profiles of having mathematical expressions that can be "pasted" into a computer algebra system.

References

[1] Bert Bos, Tantek Çelik, Ian Hickson, & Håkon Wium Lie, *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification* World Wide Web Consortium Recommendation, 7 June 2011, http://www.w3.org/TR/2011/REC-CSS2-20110607.

[2] Tim Bray, Jean Paoli, C.M. Sperberg-McQueen, Eve Maler, & François Yergeau, *Extensible Markup Language (XML) 1.0 (Fifth Edition*, World Wide Web Consortium Recommendation, 26 November 2008, http://www.w3.org/TR/2008/REC-xml-20081126.

[3] Michel Goossens and Sebastian Rahtz et al., *The LaTeX Web Companion*, Addison-Wesley, 1999.

[4] Leslie Lamport, *LaTeX: A Document Preparation System*, 2nd edition, Addison-Wesley, 1994.

[5] William F. Hammond, *The GELLMU Manual*, 2007, http://mirror.ctan.org/support/gellmu/doc/glman.pdf, or http://mirror.ctan.org/support/gellmu/doc/glman.xhtml (XHTML+MathML).

[6] William F. Hammond, "GELLMU: A Bridge for Authors from LaTeX to XML", *TUGboat*, vol. 22 (2001), pp. 204–207; also available online at http://www.tug.org/TUGboat/tb22-3/tb72hammond.pdf.

[7] William F. Hammond, "Dual presentation with math from one source using GELLMU", *TUGboat*, vol. 28 (2007), pp. 306–311; also available online at http://www.tug.org/TUGboat/tb28-3/tb90hammond.pdf. A video recording of the presentation at TUG 2007, July 2007, in San Diego is available at http://river-valley.zeeba.tv/conferences/tug-2007/.

[8] William F. Hammond, "LaTeX profiles as objects in the category of markup languages", *TUGboat*, vol. 31 (2010), pp. 240–247; also available online at http://www.tug.org/tugboat/tb31-2/tb98hammond.pdf. A video recording of the presentation at TUG 2010, June 2010, in San Francisco is available at http://river-valley.zeeba.tv/conferences/tug-2010/.

[9] Bert Bos, David Carlisle, George Chavchanidze, Patrick D. F. Ion, & Bruce Miller, "A MathML for CSS Profile", World Wide Web Consortium Recommendation, 7 June 2011, http://www.w3.org/TR/mathml-for-css/.

[10] T. Atkins, fantasai, & Rossen Atanassov, ed., "CSS Flexible Box Layout Module Level 1", World Wide Web Consortium, last call working draft (work in progress), March 25, 2014, http://www.w3.org/TR/2014/WD-css-flexbox-1-20140325/, (latest: http://www.w3.org/TR/css-flexbox-1/).

◇ William F. Hammond
University at Albany, Albany, New York
and San Diego, California
hammond (at) albany dot edu
http://www.albany.edu/~hammond/