The PracTEX Journal, 2010, No. 1
Article revision 2010/06/06

Some PDF/A tricks

Claudio Beccari

Email claudio dot beccari at gmail dot com
Address Villarbassse (TO), Italy

Abstract This short contribution explains how to fix some font problems when creating PDF/A documents, the new standard for archival PDF documents.

1 Introduction

At the beginning of December 2008 the package pdfx was released on CTAN. During its gestation the developer, Hàn Thế Thành, set up a web site[1] where he described the work in progress and the solution to some of the encountered problems.

First of all, what is a PDF/A document? It is an electronic document that can be archived for an unlimited period of time while being readable exactly as when it was first typeset, in spite of the fact that, say, 50 years later the reading software that existed at the time of creation does not exist any more; the new software that will be developed shall conform to the ISO 19005-1:2005 regulation, and will ensure that the document has been typeset in conformance with the same regulation.

It's the existence of this regulation that assures readability from now on. This regulation therefore has to put some restrictions on the way the document is typeset, on how the fonts used (only outline fonts are accepted) are embedded into the document, on the kind of pictures included in it, and on the format of the *metadata* the document should contain, in order to be usable as an archivable object.

[1] http://support.river-valley.com/wiki/index.php?title=Generating_PDF/A_compliant_PDFs_from_pdftex

2 Problems with some fonts

Fonts pose some problems, especially the Computer Modern ones and those that derive from this collection. Specifically, it's the cmsy*xx* fonts that cause some problems.

The ISO regulation requires that all glyphs used in the document have a non-vanishing width; it is surprising to find that some glyphs pertaining to the cmsy*xx* fonts have zero width; specifically the glyph corresponding to \not and the tail of the special arrow created with the command \mapsto; this tail is made up of a zero width vertical small bar, \mapstochar, joined to a normal right-pointing arrow.

It's understandable that the \not glyph has zero width, as it has to be superimposed onto the other binary relation operators in order to negate them; the same does not appear (to me) to be true for the \mapstochar glyph. In any case, this is the situation and this situation must be corrected because, even if the documents print and/or display in a perfect way, it violates the ISO regulation and becomes non-archivable.

Since a PDF/A document may be generated by running ghostscript in a special way, with suitable options and configuration files, the font problem cannot be circumvented even by resorting to this auxiliary program.

3 A possible solution

Hàn Thế Thành proposes to solve this problem by creating suitable virtual fonts that map the cmsy*xx* fonts (or any other font with the same features) onto itself but with a different metric file that uses a non-vanishing width, say, 0.001 em; this width is non-zero and is acceptable to the ISO regulation. However, at this point another problem arises because the metric information is not consistent with what is stored in the scalable Type 1 font with the extension .pfb; this requires another adjustment of the .pfb file, in order to store the same width used in the metric file.

Hàn Thế Thành used this technique to overcome the described problems. I believe that fiddling with back and forth transformations of binary and ASCII files, metric and real or virtual property list files, is error-prone and may not be suitable for users who are not able to make modifications at that level. Therefore, I propose the following macros that do not fiddle with anything.

My macros rely on a zero-width box that contains a non-zero-width glyph; the zero-width box is `\rlap`, a Plain TeX control sequence more or less equivalent to `\makebox[0pt][l]`; and this box overlaps its contents on top of whatever happens to be on its right.

The non-zero-width glyph replacing the `\not` oblique bar is the slash taken from the `cmmmi`*xx* fonts, the ones that contain the normal math italic letters. The non-zero-width glyph replacing the `\mapstochar` one is a rule that uses the LaTeX command `\rule`.

Since typesetting mathematics implies four different styles, the correct style must be chosen for inserting the appropriately-sized replacement glyph in every mode; to this end, the primitive TeX command `\mathchoice` can be used and the trick is complete.

The two macros are the following:

```
% New \not command
\renewcommand*\not{\mathrel{\mathchoice
  {\rlap{$\displaystyle
       \mkern2.5mu\mathnormal{/}$}}%
  {\rlap{$\textstyle
       \mkern2.5mu\mathnormal{/}$}}%
  {\rlap{$\scriptstyle
       \mkern2.5mu\mathnormal{/}$}}%
  {\rlap{$\scriptscriptstyle
       \mkern2.5mu\mathnormal{/}$}}%
}}
%
% New \mapstochar command
\renewcommand\mapstochar{%
\mathrel{\mathchoice
   {\rlap{\rule[.05ex]{.1ex}{1ex}}\mkern-.5mu}%
   {\rlap{\rule[.05ex]{.1ex}{1ex}}\mkern-.5mu}%
   {\rlap{\rule[.035ex]{.08ex}{.75ex}}\mkern-.5mu}%
   {\rlap{\rule[.025ex]{.065ex}{.55ex}}\mkern-.5mu}%
}}
```

Of course the “magic” numbers inserted in either macro should be verified with any font different from the assumed `cmsy`*xx* and `cmmmi`*xx* fonts, although the values should remain approximately the same.

With these macros in place it is not necessary to touch any font file, either the `.tfm` metric one or the `.pfb` binary one.

What I hope is that the small corrections indicated by Hàn Thế Thành be introduced into the original fonts distributed with every TeX distribution, so that my macros become superfluous, and that the indicated problems with the `cmsy`*xx* fonts in connection with the archivable PDF/A format vanish at their very origin.