

PostScript and the Printer System

Richard J. Mathar*
(Dated: May 23, 2009)

I. DO NOT PRINT

Most paper is wasted because stuff is sent to the printer that is not intended to be used. (Look at the paper trays in this room with output not picked up.) Before sending your pages a second time suspecting the first submission “has been lost”, use

```
lpq ; lpstat -p
netscape http://maas:631/jobs
to check if the first attempt is still on the way. Use
cancel -u $user
```

before pressing the printer’s cancel buttons to terminate output prematurely. If you use **a2ps**: disable automated flushing of all **a2ps** outputs to the printer by adding the lines

```
Options: --output=-
DefaultPrinter: file | cat
to the file ~/.a2ps/a2psrc. If you use enscript:2 dis-
able automated flushing of all outputs to the printer by
adding the line
DefaultOutputMethod: stdout
to the file ~/.enscriptrc. This allows examination of
the output with
gv myfile.ps
first. Never use the Unix command print to print
PostScript.
```

II. SUBSETS OF PAGES

A subset of PostScript documents with many pages may be selected by (i) the Unix command **psselect** or (ii) marking pages within **gv** that are eventually saved or printed or (iii) using the **-pp** option of **dvips**. These approaches do not work if the PostScript file does not contain the Document Structuring Comments¹ (eg, if generated with the **-N** option of **dvips**).

III. AREA REDUCTION

A first form of area reduction is the use of **a2ps** or **enscript**² to transform ASCII files (program listings included) into intermediate PostScript files which usually results in more information put on the pages than the standard 60-lines-per-page interpretation by our spooling system. A second form is to save the contents of web pages into text files which removes most of the advertisement and graphical elements, and to continue with the ASCII texts. A third form starts from available PostScript files and uses **psnup** to put two or more of the original pages onto one page of the new file. This turns

out to be useful for preprints and similar documents with large-size fonts, and works also for PostScript files generated with

```
acroread -toPostScript < myfile.pdf | psnup -2 >
myfile.ps
```

from PDF files. These methods may be cascaded. **psnup** usually fails for files generated from MS Word documents with **Pscript.dll**. A rewriting with

```
gs -q -dNOPAUSE -dBATCH -sDEVICE=pswrite
```

```
-sOutputFile=tmp.ps-f myfile.ps
```

```
psnup -2 tmp.ps >newfile.ps
```

may help to overcome this restriction.

IV. POSTSCRIPT CONFORMANCE

A frequent cause of nuisance is files that do not conform with the well-defined PostScript standards.¹ Note that the very first two characters in the file must read **%!**. If they do not, the output will look just as if you look at the original file with a line editor (**vi**, **emacs**, **pg**, **more**,...). You may check this point with

```
file myfile.ps
```

which *must* declare this file to be PostScript, or extract the first line with

```
head -1 myfile.ps
```

which *must* start with the percent-bang. Displaying the file with **gs** or **gv** is an insufficient check for that matter, because the underlying PostScript interpreter is handling this kind of erroneous input with more tolerance than a general printer spooler system can do. (This is a bug in the programs you used to create the file, not a result of bad system administration.) In cases where just some special line printer escape sequences have been wrapped around the file (Microsoft Word’s **prn** files are well-known candidates) or the file has been sent by e-mail, it suffices to delete a few lines at the beginning and the end of the file to end up with something that prints as intended. One may try to abuse **gs** in this case like

```
gs -q -sDEVICE=pswrite -sOutputFile=tmp.ps
-dNOPAUSE myfile.ps
```

```
to create an intermediate file tmp.ps, or the blunt filter
dos2unix -ascii < myfile.prn
```

```
| sed -n '/^%\!/;/~%EOF/p' >myfile.ps
```

The filter program **fixps**² also does this job.

If **gv** reports errors in the PostScript file, there is hardly any chance of correct interpretation by the printer.

Conformable PostScript files *may* use the full range of 8-bit characters and therefore may get corrupted by e-mail transmission (depending on **sendmail** configura-

tions) or unfortunate combinations of `vi` and `stty` settings.

Newer versions of `gv` automatically decode compressed PostScript files³ on the fly. Our spooling system does not, and you need to decode them on your own *before* printing them with `lpr` or `lp`. Note also that `xv` calls `gs` for PostScript files; so apologies of the kind “but with `xv` everything looks ok” will not be accepted either.

V. ENCAPSULATED POSTSCRIPT

Encapsulated PostScript files contain a single page of information which is supposed to be encapsulated in larger documents. The final PostScript command `showpage` is therefore often missing (as it ought to) and the result is no output at all if the file is sent to the printer. Because `gs` and `gv` show the progress of building pages online, a valid result of this check on the X screen does not mean that anything has to appear in print. (This is not a bug in programs nor bad system administration but your mis-interpretation of Encapsulated PostScript.)

Previewers of some text preparation systems⁴ display the simple, embedded raster graphic image of an EPSI file, not an interpretation of the PostScript code. Their ability to display the image does not proof correctness of the PostScript contents.

VI. 2-SIDED

Sending the output to printers that can print 2-sided creates half the weight to carry around! Change the Unix environment variables `PRINTER` and `LPDEST` to let one of them become your default printer, and add the `SetPageDevice: Duplex:true` option to `~/enscriptrc` if necessary.

There is a known bug in CUP⁵ based printer spoolers that accept only 2-sided output if the file claims to be `%!PS-Adobe-3.0`. In these cases the explicit chain of options

```
lpr -o raw -o=two-sided-long-edge myfile.ps
seems to be necessary to bypass the spooler's decision to
discard the file.
```

VII. PROENGINEER AND L^AT_EX

Some PostScript files generated with ProEngineer seem to “erase themselves” when they are included in L^AT_EX scripts like

```
\usepackage{epsfig}
\epsfig{file=myfile.ps,scale=0.8}
```

The cure seems to be to remove a change of some entries in the PostScript dictionary stack found early (about line 10) in the PostScript file. Disable these four lines by either erasing them or placing a percent-sign in front of them:

```
%systemdict /setpagedevice known
%{
% <</PageSize[612 792]>> setpagedevice
%} if
```

This error does not show up if one looks at the original PostScript file with `gs` or `gv`.

A similar behaviour is found if some PostScript files that use a `Adobe_WinNT` kind of driver are bound into L^AT_EX. In these cases heuristics shows that one ought handle the line

```
PageSV restore
```

at about the sixth but last line of the PostScript file in the same fashion (erase or comment).

VIII. MISCELLANY

The annoying pop-up windows when some MS Word documents created with `AdobePS5.dll` are displayed with `gv` may be avoided by removing the single line `statusdict begin (%[ProductName:) print product pr` in the PostScript file or making it a comment by placing a `%` in front of it. Just search for the second occurrence of `flush` in the file.

* URL: <http://www.strw.leidenuniv.nl/~mathar>

¹ PostScript Language Reference Manual, <http://partners.adobe.com/asn/developer/pdfs/tn/PLRM.pdf>

² Not necessarily implemented here.

³ even if the file name suffix does not indicate the com-

pression, as with preprints downloaded from of the LANL archive

⁴ like IBM's publisher

⁵ <http://localhost:631> and `/usr/share/cups/model/postscript*`