DVII: A TEX dvi file information utility

Adam H. Lewenberg

Abstract

dvii is a free utility written in portable C that extracts information from a TEX dvi file and displays it on the command line. The information can include paging, fonts, specials, and per-page message digests. The output is designed to be easily parsed by a text-processing language (such as Perl) to allow other kinds of summaries to be generated (such as a font difference utility, or to help dvips print only those pages containing certain kinds of \special's).

— — ∗ — —

1 Introduction

Consider a dvi file. Perhaps you created it by compiling a TEX, or maybe you received it in your e-mail. Here are some questions about the dvi file:

1. How many pages does it contain?
2. How do the "physical" pages correspond to the "TEX" pages? (For example, if the file is printed, what page number will be printed at the bottom of the seventh page out of the printer; it will not necessarily be 7.)
3. What fonts are called for? Which fonts are called for *on a specific page*?
4. Which pages contain (externally linked) figures?
5. Have the page breaks changed since the last time I compiled?
6. Has this file been corrupted?
7. When was this file compiled?[1]
8. Does this file use the same fonts as some other file?

The dvii program is a free utility that extracts and displays information from a TEX dvi file that allows us to answer all of these questions quickly and easily. For information on where to download the dvii utility, see section 10.

2 An Example

If we run the dvii utility on the file test.dvi we get the following output:

```
File size: 1188 bytes (1 K)
Comment string:  TeX output 2001.12.29:2041
Page count: 7
Number of fonts: 3
f:[50/cmr10/1200]::4bf16079
f:[23/cmbx10/1000]::1af22256
f:[0/cmr10/1000]::4bf16079
p:[1/1]
p:[2/2]
p:[3/3]
p:[4/4]
p:[5/5]
p:[6/-1]
p:[7/-3]
s:[3/3]:: A short special
s:[5/5]:: PSfile 1.eps
s:[5/5]:: PSfile 2.eps
s:[5/5]:: PSfile 3.EPS
s:[5/5]:: PSfile dog1.gif
s:[5/5]:: PSfile cat.eps
```

(To the see the source file test.tex see Appendix A.)

This output tells us several things. First of all, there is a summary of the dvi file including the file size, the comment string, and the number of fonts and pages in the dvi file. Next, there is more detailed information listing the fonts used, the pages (both the "physical" page and the TEX page), and the TEX specials.

3 Purpose and Motivation

Much of what dvii does could be done with Donald Knuth's dvitype and a text processing language like Perl. My motivation was not to make another dvi parser, but to create a utility that would extract specific sorts of information quickly from a dvi file.

I wrote dvii with the following goals in mind:

1. It should be *fast*, faster than dvitype.
2. It should be easy to use the output as a back end to Perl (or any other text processing language), enabling the easy manipulation of the data for more specific purposes.
3. It should be useful to the TEX community.
4. It should be portable.

I believe I have met most[2] of these goals, but if anyone has suggestions for improving this utility, I will be quite happy to hear them.

4 Consistency

You can check that a dvi file has not been corrupted by using the -C option.

```
dvii -C test.dvi

dvi file 'test.dvi' passed validation
check (level 1).
```

Here is an example where dvii detects a problem:

```
dvii -C bad.dvi

[dvi validation error] missing postamble id
(should be 2)
```

[1] This is not necessarily the same as the file date.

[2] However, it is hard to beat Knuth in program efficiency and speed!

5 Pages

You want to know the number of pages in a dvi file. How do you determine this? You could view it with a dvi viewer, or perhaps look at the .log file that was generated when the dvi file was originally created. But a dvi viewer is rather heavy machinery to simply find the number of pages in a dvi file, and, besides, you may be working from the command line where there is no dvi viewer handy. As for the .log file, it may no longer be around.

The dvii utility provides a page count virtually instantaneously. Furthermore, if you want to know the page numbering layout of physical vs. TEX pages, dvii will also tell you that. (By a "physical" page I mean the order of the page as printed, and by TEX page I mean the page as it appears printed in the footer or running head. For example, the third page out of the printer might have TEX page number −3, that is, iii.)[3]

From the example There are seven physical pages where the first five have TEX page number matching the physical page number. Physical pages 6 and 7 have TEX pages −1 and −3.

6 Fonts

One of the reasons that dvi files are not very portable is that the fonts that a dvi file uses are not embedded in the dvi file itself, but rather are identified by number and name. TEX leaves it to the dvi interpreter to find the proper external font. Thus, if you receive a dvi file via e-mail or download one from the web, there is a good chance it will not look the way the author intended or perhaps it will not display at all, unless you have the same fonts with the same names as the dvi file's author. So, a list of fonts that a dvi file calls for is, at times, quite useful.

The dvii utility will list each font used in a dvi file listing its name, font number, scale factor, and checksum. The checksum is especially useful in detecting when two fonts that seem to be the same (same name and scale) are in fact different.

From the example There are three fonts, two copies of cmr10 (Computer Modern Roman 10pt) scaled at 100% (font number 0) and 120% (font number 50), and cmbx10 (Computer Modern BoldFace 10pt) scaled at 100% (font number 23).

[3] More accurately, by TEX page I mean the contents of the \count0 register which (normally) stores the page number that is printed on the page itself.

7 Specials

The TEX special is a "hook" that D. Knuth put into TEX to allow later functionality without having to rewrite TEX. To insert a special you use the \special{} command. For all practical purposes, TEX ignores specials,[4] merely copying the special text to the dvi output. It is up to the dvi driver to decide what meaning (if any) to give to a special.

Some uses that people have made of specials are to incorporate color, to help with the edit-compose-view cycle, and, most commonly, to allow external figure file inclusion, in particular the inclusion of EPS files. The graphics and graphicx packages along with most dvi-to-PostScript drivers indicate the inclusion of an EPS file by inserting a special that starts with the string PSfile and then is followed by a number of arguments including the file name and the dimensions of the figure.

The dvii utility will list all specials in a dvi file. If you want to list all specials that match some particular string, you can pipe the output of dvii through the standard grep utility.

From the example There are six specials, one on page 3, and five on page 5. The specials on page 5 appear to be included EPS files with the file name specified after the string PSfile.

8 Message Hashes

At this point, I have shown how to answer all the questions asked in section 1 except questions 5 and 8. For an answer to question 8 see section 9.1. In this section I will take up question 5.

The trickiest part of setting type is almost always page optimization, that is, fixing bad page breaks and getting floats (figures, tables, etc.) put in the right places. This part of composition is more an art than a science although TEX and LATEX do provide tools to help. This is why you always, **always**[5] wait until the last possible second before doing final page layout.

But, inevitably, after you have spent a week getting all your page breaks and floats in your 800 page book set just where you want them, someone comes along and insists on making some small change that has the potential of messing everything up. Wouldn't it be nice to have a quick and easy method to see how a change to the text affects the page layout for the entire document?

My solution to this is to take a numerical "snapshot" of each page before and after the change and

[4] Well, almost. It is possible for a special to affect page breaks.

[5] Always!

then see which pages' snapshots have changed. This snapshot is called a *message digest* or *checksum*. dvii can, if asked, calculate a 16-byte checksum for each page. This checksum will change if the contents of the page change. Thus, if you calculate the message digest before and after a change to the TEX source, you can see which pages have changed and thus where to look for possible new bad page breaks.

From the example Here is the message digest for `test.dvi`:

```
dvii -m test.dvi
[message digest: simple sum]
p:[1/1]::D8C977816A091771A3A631E7582DAD6D
p:[2/2]::284E8575505581BAA95B7CB132F1F435
p:[3/3]::16E5A31FF87F926DB1F86CAD165C5453
p:[4/4]::C72EE84EFA36764C537229D4968F8DF9
p:[5/5]::68E5A2E9D7320743CB85769CAAEAB023
p:[6/-1]::DE72BC3345FFDF7E779C8C667DCE3F97
p:[7/-3]::13B730AA855EB18E30CED11AB1D26FAF
```

Let `test2.tex` be an exact copy of `test.tex` except that we have changed the first word "This" to "Thus" (see source listing in Appendix A). Here is the resulting message digest.

```
dvii -m test2
[message digest: simple sum]
p:[1/1]::556A2538928963D8B5776E1245364DE6
p:[2/2]::284E8575505581BAA95B7CB132F1F435
p:[3/3]::16E5A31FF87F926DB1F86CAD165C5453
p:[4/4]::C72EE84EFA36764C537229D4968F8DF9
p:[5/5]::68E5A2E9D7320743CB85769CAAEAB023
p:[6/-1]::DE72BC3345FFDF7E779C8C667DCE3F97
p:[7/-3]::13B730AA855EB18E30CED11AB1D26FAF
```

If you look carefully, you will notice that the checksum for the first page has changed while the others have not.

For a large project you would not want to try to detect such changes by eye, so you would instead use a utility such as `diff` to detect the differences.

```
dvii -m test > test.md
dvii -m test2 > test2.md
diff test.md test2.md
2c2
< p:[1/1]::D8C977816A091771A3A631E7582DAD6D
---
> p:[1/1]::556A2538928963D8B5776E1245364DE6
```

The first two commands store the message digests in the files `test.md` and `test2.md`. The third command uses the `diff` command to find how the two files `test.md` and `test2.md` differ; in this case, the `diff` command shows us that they differ only in the first line.

9 Some applications based on dvii

The output of dvii has been designed to make it easy for text processing programs to manipulate and provide further useful information. Because of its near ubiquitousness, ease of use, and low cost (free), I use Perl. If Unix is your computing environment you probably already have Perl installed. If you use Windows, then you can download a free version. See section 10 for more information on where to get Perl.

In what follows I describe two such scripts based on Perl.[6]

9.1 `fontdiff.pl`

The Perl script `fontdiff.pl` finds the font differences between two `dvi` files, that is, it lists which fonts are present in one and not the other. Here is an example using the two `dvi` files `a.dvi` and `b.dvi`:

```
perl fontdiff.pl -l a b

Fonts in a.dvi NOT in b.dvi:
NOTE: fonts marked with * are in BOTH files
------------------------------
  f:[NN/cmbx10/1000]::1af22256
* f:[NN/cmr10/1000]::4bf16079
------------------------------

Fonts in b.dvi NOT in a.dvi:
NOTE: fonts marked with * are in BOTH files
------------------------------
  f:[NN/cmti10/1000]::fd00273a
  f:[NN/cmsl10/1000]::70ae304a
* f:[NN/cmr10/1000]::4bf16079
------------------------------
```

9.2 `specials.pl`

This Perl script creates the command-line option that works with Tom Rockicki's `dvips` so that just those pages of a `dvi` file containing specials get printed. Recall from Section 2, the `dvi` file `test.dvi` has `\special`'s on pages 3 and 5. To print just those pages using `dvips` you would type `dvips -pp3,5 test`. If you run `specials.pl` on the file `test.dvi` we get

```
perl specials.pl test
-pp3,5
```

Although this example is short, you can see how useful it would be to generate this page list automatically if your `dvi` file had hundreds of pages and dozens of figures.

Observe that in the above example page 3 was listed even though page 3 does not have a *figure* special. To list only those pages that have a `\special` matching some string, you can use the `--grep` option. For example, most TEX and LATEX graphics inclusion packages indicate an included figure by starting the `\special` with the string `PSfile`, so to list

[6] Both Perl scripts, as well as more information on their use, are available at the dvii home page; see Section 10.

only those pages which have a special that contain the string `PSfile` you would type

```
perl specials.pl --grep PSfile test
-pp5
```

10 Where and how to get it

The dvii utility has its home page at `http://www.macrotex.net/dvii/dvii.html`. At this site you can download the C source code which consists of a single file and should compile with any standard C compiler. No special libraries are required. There are no licensing restrictions on the use of this utility.

If you do not wish to, or are unable to compile the source code, there are binaries for Windows, Linux, Solaris, and DOS.[7] To install, download the appropriate executable file and put it somewhere in your path.

You can also download a manual for dvii which explains in detail all the options. This manual is available in HTML, `dvi`, and PDF formats.

If you prefer, you can download from CTAN the source code, manual (in PDF), and a DOS/Windows executable which you will find in the `dviware/dvii` directory.

Perl is available on nearly every computing platform for no cost. If you work on a Unix or Unix-like platform Perl is probably already installed. For more information on obtaining Perl, go to the Comprehensive Perl Archive Network (CPAN) at `www.cpan.org`.

The `diff` and `grep` utilities are even more likely to be already available on Unix and Unix-like systems. If they are not, you can get GNU versions at `www.gnu.org`. If you run Windows, you can get them at the Cygwin home at `http://sourceware.cygnus.com/cygwin/`.

11 Acknowledgements

Heiko Oberdiek made significant suggestions on improving the performance of the code, and pointed out several errors. Tom Kacvinsky helped make the code work on 64-bit machines.

[7] If you can compile the code for other platforms, I would be happy to post them there. I am especially looking for a Macintosh version (does Macintosh *have* a command line?).

A The source for `test.tex`

```
% This is test.tex
This is page 1/1.
\eject
This is page 2/2.
\eject
This is page 3/3 with a short special.
Also, a rule.
\vrule width1cm depth1cm height 1cm\relax
\special{A short special}
\eject
This is page 4/4 {\bf without} any specials.
\eject
This is page 5/5 with 5 specials.
\special{PSfile 1.eps}
\special{PSfile 2.eps}
\special{PSfile 3.EPS}
\special{PSfile dog1.gif}
\special{PSfile cat.eps}
\eject
\pageno = -1
\font\a=cmr10 scaled 1200
{\a This is page 6/-1 with font cmr10
scaled 1200.}
\eject
\pageno = -3
This page is nothing special.
\eject\eject\eject\eject
\bye
```

References

[1] Donald E. Knuth. *The TeXbook*. Addison-Wesley, Reading, Massachusetts, October 1990.

◇ Adam H. Lewenberg
211 Paddock Drive East
Savoy, Illinois 61874
adam@macrotex.net