Profiling TeX input files

Do you know how TeX spends its time?

Martin Ruckert

Munich University of Applied Sciences
Department of Mathematics and Computer Science

TUG 2024, Prague

Outline

What is a Profiler?

- A profiler maps runtime to program source lines.
  Example:

```
line percent   absolut count   average file
6450  13.67% 716.40 ms  2541 281.94 us expl3-code.tex
```

What is a Profiler?

- A profiler maps runtime to program source lines.
  Example:

```
line percent   absolut count   average file
6450  13.67% 716.40 ms  2541 281.94 us expl3-code.tex
```

- A profiler maps runtime to program functions.
  In TEX macros play the role of functions.

Example (TEX runs for about 5s):

```
     time loop percent count/total macro and children
100.62 ms        1.92%      *      \@swaptwoargs
  1.02 ms        1.02%     42      \@swaptwoargs
 85.09 ms       84.56%    10/10    \@input@file@exists@with@hooks
 17.42 ms       17.31%     1/4     \loop
```

What is a Profiler?

- A profiler maps runtime to program source lines.
  Example:

```
line percent   absolut count    average file
6450  13.67% 716.40 ms  2541 281.94 us expl3-code.tex
```

- A profiler maps runtime to program functions.
  In TEX macros play the role of functions.
  - The cumulative time spent in a macro and its submacros.

  Example (TEX runs for about 5s):

```
     time loop percent count/total macro and children
100.62 ms         1.92%       *      \@swaptwoargs
  1.02 ms         1.02%     42       \@swaptwoargs
 85.09 ms        84.56%    10/10     \@input@file@exists@with@hooks
 17.42 ms        17.31%     1/4      \loop
```

What is a Profiler?

- A profiler maps runtime to program source lines.
  Example:

```
line percent    absolut count    average file
6450  13.67% 716.40 ms  2541 281.94 us expl3-code.tex
```

- A profiler maps runtime to program functions.
  In TeX macros play the role of functions.
  - The cumulative time spent in a macro and its submacros.
  - The time spent directly in the macro.

  Example (TeX runs for about 5s):

```
     time loop percent count/total macro and children
100.62 ms        1.92%      *      \@swaptwoargs
  1.02 ms        1.02%     42      \@swaptwoargs
 85.09 ms       84.56%    10/10    \@input@file@exists@with@hooks
 17.42 ms       17.31%     1/4     \loop
```

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.
- One second CPU time = 28mg CO_2 emission.
  Driving 200km = 28kg CO_2 emission.
  (assuming 200Watt peak, 500g CO_2/kWh, 2370g/l, 6l/km).

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.
- One second CPU time = 28mg CO_2 emission.
  Driving 200km = 28kg CO_2 emission.
  (assuming 200Watt peak, 500g CO_2/kWh, 2370g/l, 6l/km).
- The profiler will tell you where to look for optimizations.

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.
- One second CPU time $=$ 28mg CO_2 emission.
  Driving 200km $=$ 28kg CO_2 emission.
  (assuming 200Watt peak, 500g CO_2/kWh, 2370g/l, 6l/km).
- The profiler will tell you where to look for optimizations.
- The profiler will tell you where **not** to look for optimizations.

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.
- One second CPU time = 28mg CO_2 emission.
  Driving 200km = 28kg CO_2 emission.
  (assuming 200Watt peak, 500g CO_2/kWh, 2370g/l, 6l/km).
- The profiler will tell you where to look for optimizations.
- The profiler will tell you where **not** to look for optimizations.
- The profiler will tell you if your optimization had any effect.

Who Needs and Who does not Need a Profiler?

- You do **not** need a profiler if you are a document author.
- You might need a profiler
  - if you are a macro programmer,
  - and if you care about run time,
  - and your macros are used **very** often.
- One second CPU time = 28mg CO_2 emission.
  Driving 200km = 28kg CO_2 emission.
  (assuming 200Watt peak, 500g CO_2/kWh, 2370g/l, 6l/km).
- The profiler will tell you where to look for optimizations.
- The profiler will tell you where **not** to look for optimizations.
- The profiler will tell you if your optimization had any effect.
- Never optimize for speed without a profiler.

How does the TEX Profiler Work?

Collect Data,Write Output File, Analyze Output File

- Collect data

```
main_control(void) {
  big_switch:
    ⟨look up the time⟩
    get_x_token();
    ⟨determine current command, file, line, and macro⟩
    switch (⟨current command⟩) { ⟨execute current command⟩ }
  goto big_switch;
  main_loop:
    ⟨loop over characters, kerns, spaces, ligatures, ...⟩
  goto big_switch;
}
```

How does the TEX Profiler Work?

Collect Data,Write Output File, Analyze Output File

- Collect data

```
main_control(void) {
  big_switch:
    ⟨look up the time⟩
    get_x_token();
    ⟨determine current command, file, line, and macro⟩
    switch (⟨current command⟩) { ⟨execute current command⟩ }
  goto big_switch;
  main_loop:
    ⟨loop over characters, kerns, spaces, ligatures, ...⟩
  goto big_switch;
}
```

- Write data to a file.

Hard and Soft Problems

- Soft Problems

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts
  - Frequency scaling

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts
  - Frequency scaling
  - Performance cores and efficiency cores

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts
  - Frequency scaling
  - Performance cores and efficiency cores
  - Current accuracy: $\approx 100\mu$s

Hard and Soft Problems

- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts
  - Frequency scaling
  - Performance cores and efficiency cores
  - Current accuracy: $\approx 100\mu$s
  - Possible solution: using multiple runs

Hard and Soft Problems

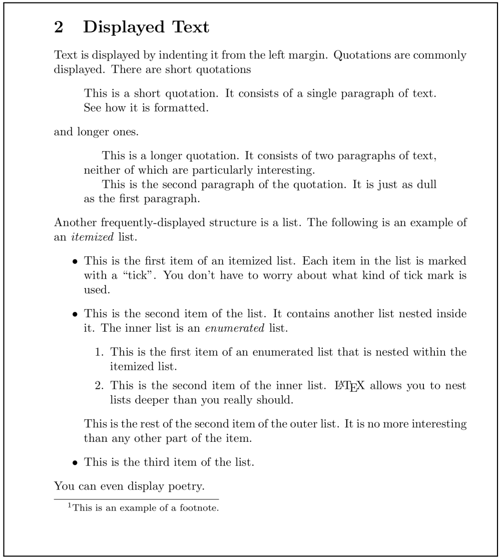
- Soft Problems
  - Determine and store file and line for each token.
  - Determine and store file, line, and name for each macro.
  - Keep track of the macro stack.
  - Keep track of expand and backup.
- Hard Problems:
  Get a reliable and consistent time source.
  - Time sharing and interrupts
  - Frequency scaling
  - Performance cores and efficiency cores
  - Current accuracy: $\approx 100\mu$s
  - Possible solution: using multiple runs
  - Possible solution: using synthetic times

Run Times

example	pages	macro use	tex	texprof	pdftex
sample2e.tex,	3	medium	95ms	75ms	110ms
large, mostly text, plain TEX	1130	low	335ms	400ms	950ms
large, mostly text, LATEX	1119	medium	1330ms	1390ms	1960ms
medium, cweb output	758	high	260ms	280ms	2265ms

Example: sample2e

- small size: 3 pages
- medium complexity
- LaTeX



2 Displayed Text

Text is displayed by indenting it from the left margin. Quotations are commonly displayed. There are short quotations

> This is a short quotation. It consists of a single paragraph of text. See how it is formatted.

and longer ones.

> This is a longer quotation. It consists of two paragraphs of text, neither of which are particularly interesting.
>
> This is the second paragraph of the quotation. It is just as dull as the first paragraph.

Another frequently-displayed structure is a list. The following is an example of an *itemized* list.

- This is the first item of an itemized list. Each item in the list is marked with a "tick". You don't have to worry about what kind of tick mark is used.
- This is the second item of the list. It contains another list nested inside it. The inner list is an *enumerated* list.
  1. This is the first item of an enumerated list that is nested within the itemized list.
  2. This is the second item of the inner list. LaTeX allows you to nest lists deeper than you really should.

  This is the rest of the second item of the outer list. It is no more interesting than any other part of the item.
- This is the third item of the list.

You can even display poetry.

[1]This is an example of a footnote.

Profiling sample2e

Running texprof

Running texprof with the LaTeX format

```
> latexprof -prof sample2e
This is texprof, Version 3.141592653-1.0 (preloaded format=latexprof)
entering extended mode
(/usr/local/texlive/2023/texmf-dist/tex/latex/base/sample2e.tex
LaTeX2e <2023-11-01> patch level 1
L3 programming layer <2024-02-20>
(/usr/local/texlive/2023/texmf-dist/tex/latex/base/article.cls
Document Class: article 2023/05/17 Standard LaTeX document class
(/usr/local/texlive/2023/texmf-dist/.../size10.clo))
(/usr/local/texlive/2023/texmf-dist/.../l3backend-dvips.def)
(./sample2e.aux) (/usr/local/texlive/2023/.../omscmr.fd)
[1] [2] [3] (./sample2e.aux) )
Output written on sample2e.dvi (3 pages, 7548 bytes).
Transcript written on sample2e.log.
```

Profiling sample2e

Running tprof

- Running `texprof` with the LaTeX format

```
> latexprof -prof sample2e.tex
```

- This creates `sample2e.tprof`

Profiling sample2e

Running tprof

- Running `texprof` with the LaTeX format

```
> latexprof -prof sample2e.tex
```

- This creates `sample2e.tprof`
- Running `tprof sample2e.tprof` shows a summary

```
> tprof sample2e
Total time measured:          25.50 ms
Total number of samples:      13204
Average time per sample:       1.94 us
Total number of files:           10
Total number of macros:       22746
Maximum stack nesting depth:    126
```

Profiling sample2e

Running tprof -T

Running `tprof -T sample2e` shows the top ten lines

```
> tprof -T sample2e
The top ten lines:
  file  line     percent    absolut  count    average    file
   0        0    94.03%   24.05 ms    330   72.89 us    unknown
system  linebrk    1.04%  265.37 us     39    6.80 us    system
system  shipout    0.90%  229.50 us      3   76.50 us    system
system  buildpg    0.20%   52.27 us    155  337.00 ns    system
   9        4      0.14%   34.88 us      1   34.88 us    sample2e.aux
   4      150      0.10%   25.41 us      1   25.41 us    article.cls
   5      170      0.09%   22.72 us      1   22.72 us    size10.clo
   7        4      0.09%   22.57 us      1   22.57 us    sample2e.aux
   4      275      0.07%   18.79 us      1   18.79 us    article.cls
   3       91      0.06%   14.75 us      1   14.75 us    sample2e.tex
```

Profiling sample2e

Modifying sample2e.tex

- Modifying sample2e.tex to load LaTeX from the file

Profiling sample2e

Modifying sample2e.tex

- Modifying sample2e.tex to load LaTeX from the file
- Switch on profiling using the `\profileon` primitive after loading the LaTeX format

Profiling sample2e

Modifying sample2e.tex

- Modifying sample2e.tex to load LaTeX from the file
- Switch on profiling using the `\profileon` primitive after loading the LaTeX format
-
```
\let\dump=\relax
\input latex.ltx\relax
\profileon
% This is a sample LaTeX input file.(Version of 12 August 2004.)
%
% A '%' character causes TeX to ignore all remaining text on
% the line, and is used for comments like this one.
\documentclass{article}      % Specifies the document class
...
```

Profiling sample2e

Running texprof

Running texprof with the LaTeX format

```
> texprof -ini -etex -ltx sample2e.tex
This is texprof, Version 3.141592653-2.6-1.1.0 (INITEX)
entering extended mode
(./sample2e.tex (/usr/local/texlive/2023/texmf-dist/.../latex.ltx
(/usr/local/texlive/2023/texmf-dist/tex/latex/base/texsys.cfg)
...
(/usr/local/texlive/2023/texmf-dist/.../l3backend-dvips.def)
(./sample2e.aux) (/usr/local/texlive/2023/texmf-dist/.../omscmr.fd)
[1] [2] [3] (./sample2e.aux) )
Output written on sample2e.dvi (3 pages, 7548 bytes).
Transcript written on sample2e.log.
```

Profiling sample2e

Running tprof -T

Running `tprof -T sample2e`

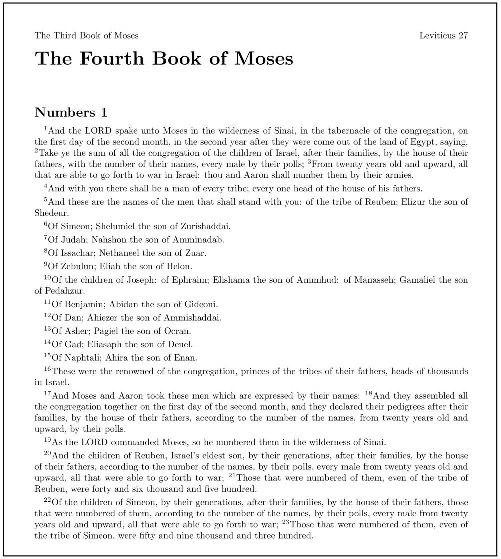
Total runtime about 5s, time measured 37ms.

The top ten lines:

```
  file    line    percent    absolut  count    average  file
system  initrie   39.91%   12.31 ms      1   12.31 ms  system
     9     1536    7.20%    2.22 ms    150   14.80 us  expl3-code.tex
     4     9536    2.81%  866.93 us     29   29.89 us  latex.ltx
     9     1546    2.42%  747.89 us     86    8.70 us  expl3-code.tex
     4     4591    1.99%  614.37 us     40   15.36 us  latex.ltx
     9     2101    1.98%  612.14 us     39   15.70 us  expl3-code.tex
     9     4759    1.81%  557.97 us     12   46.50 us  expl3-code.tex
     9     1554    1.53%  471.79 us     23   20.51 us  expl3-code.tex
     4    16430    1.49%  461.15 us      9   51.24 us  latex.ltx
     9     2105    1.41%  434.11 us    312    1.39 us  expl3-code.tex
```

Example: bible

- large size: 1130 pages
- use of macros and macro complexity is low
- plain TEX



The Third Book of Moses Leviticus 27

The Fourth Book of Moses

Numbers 1

[1]And the LORD spake unto Moses in the wilderness of Sinai, in the tabernacle of the congregation, on the first day of the second month, in the second year after they were come out of the land of Egypt, saying, [2]Take ye the sum of all the congregation of the children of Israel, after their families, by the house of their fathers, with the number of their names, every male by their polls; [3]From twenty years old and upward, all that are able to go forth to war in Israel: thou and Aaron shall number them by their armies.

[4]And with you there shall be a man of every tribe; every one head of the house of his fathers.

[5]And these are the names of the men that shall stand with you: of the tribe of Reuben; Elizur the son of Shedeur.

[6]Of Simeon; Shelumiel the son of Zurishaddai.

[7]Of Judah; Nahshon the son of Amminadab.

[8]Of Issachar; Nethaneel the son of Zuar.

[9]Of Zebulun; Eliab the son of Helon.

[10]Of the children of Joseph: of Ephraim; Elishama the son of Ammihud: of Manasseh; Gamaliel the son of Pedahzur.

[11]Of Benjamin; Abidan the son of Gideoni.

[12]Of Dan; Ahiezer the son of Ammishaddai.

[13]Of Asher; Pagiel the son of Ocran.

[14]Of Gad; Eliasaph the son of Deuel.

[15]Of Naphtali; Ahira the son of Enan.

[16]These were the renowned of the congregation, princes of the tribes of their fathers, heads of thousands in Israel.

[17]And Moses and Aaron took these men which are expressed by their names: [18]And they assembled all the congregation together on the first day of the second month, and they declared their pedigrees after their families, by the house of their fathers, according to the number of the names, from twenty years old and upward, by their polls.

[19]As the LORD commanded Moses, so he numbered them in the wilderness of Sinai.

[20]And the children of Reuben, Israel's eldest son, by their generations, after their families, by the house of their fathers, according to the number of the names, by their polls, every male from twenty years old and upward, all that were able to go forth to war; [21]Those that were numbered of them, even of the tribe of Reuben, were forty and six thousand and five hundred.

[22]Of the children of Simeon, by their generations, after their families, by the house of their fathers, those that were numbered of them, according to the number of the names, by their polls, every male from twenty years old and upward, all that were able to go forth to war; [23]Those that were numbered of them, even of the tribe of Simeon, were fifty and nine thousand and three hundred.

Profiling bible.tex: The Top Ten Lines

Running tprof -T bible

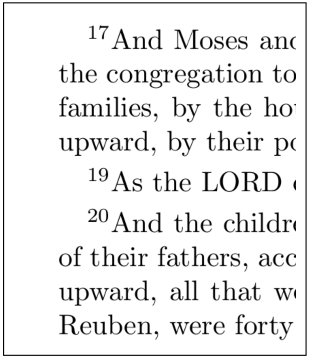
- Running `tprof -T bible` shows the top ten lines

```
  file     line   percent     absolut  count     average  file
   3        29     18.79%   135.14 ms  54649     2.47 us  bible.tex
system  shipout    14.96%   107.58 ms   1130    95.20 us  system
system  linebrk    11.89%    85.48 ms  25778     3.31 us  system
system  buildpg     1.68%    12.11 ms  55190   219.00 ns  system
   3        56      0.97%     6.98 ms   4750     1.47 us  bible.tex
   3        15      0.69%     4.95 ms   6183   799.00 ns  bible.tex
   5       555      0.53%     3.79 ms   8549   443.00 ns  plain.tex
   5      1204      0.27%     1.97 ms   3390   580.00 ns  plain.tex
system  initrie     0.26%     1.87 ms      1     1.87 ms  system
   5      1203      0.24%     1.75 ms   2258   774.00 ns  plain.tex
```

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```



^{17}And Moses anc
the congregation to
families, by the ho
upward, by their pc
^{19}As the LORD c
^{20}And the childr
of their fathers, acc
upward, all that w
Reuben, were forty

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- "by" is an optional keyword

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- "`by`" is an optional keyword
- "`1`" is scanned as a character sequence and converted to an integer.

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- "`by`" is an optional keyword
- "`1`" is scanned as a character sequence and converted to an integer.
- Math mode requires expensive processing, just to raise a box and use a small font.

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- "by" is an optional keyword
- "1" is scanned as a character sequence and converted to an integer.
- Math mode requires expensive processing, just to raise a box and use a small font.

```
\newcount\1 \1=1 \newdimen\3 \3=3.6pt
\def\Verse{%
\advance\vcount\1\leavevmode\raise\3\hbox{\sevenrm\the\vcount}}
```

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- “by” is an optional keyword
- “1” is scanned as a character sequence and converted to an integer.
- Math mode requires expensive processing, just to raise a box and use a small font.

```
\newcount\1 \1=1 \newdimen\3 \3=3.6pt
\def\Verse{%
\advance\vcount\1\leavevmode\raise\3\hbox{\sevenrm\the\vcount}}
```

- Avoid optional syntax.

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- “by” is an optional keyword
- “1” is scanned as a character sequence and converted to an integer.
- Math mode requires expensive processing, just to raise a box and use a small font.

```
\newcount\1 \1=1 \newdimen\3 \3=3.6pt
\def\Verse{%
\advance\vcount\1\leavevmode\raise\3\hbox{\sevenrm\the\vcount}}
```

- Avoid optional syntax.
- Use registers for constants.

Optimizing bible.tex

Line 29 contains the Verse macro

```
\def\Verse{\global\advance\vcount by 1${}^{\the\vcount}$}
```

- `\global` is not necessary. `\Verse` is used on the top level only.
- “by” is an optional keyword
- “1” is scanned as a character sequence and converted to an integer.
- Math mode requires expensive processing, just to raise a box and use a small font.

```
\newcount\1 \1=1 \newdimen\3 \3=3.6pt
\def\Verse{%
\advance\vcount\1\leavevmode\raise\3\hbox{\sevenrm\the\vcount}}
```

- Avoid optional syntax.
- Use registers for constants.
- Use math for mathematics.

Profiling bible.tex: The Top Ten Lines

Running tprof -T bible

Before optimization

```
  file     line   percent    absolut  count    average  file
   3        29     18.79%  135.14 ms  54649    2.47 us  bible.tex
system  shipout    14.96%  107.58 ms   1130   95.20 us  system
system  linebrk    11.89%   85.48 ms  25778    3.31 us  system
system  buildpg     1.68%   12.11 ms  55190  219.00 ns  system
   3        56      0.97%    6.98 ms   4750    1.47 us  bible.tex
```

Optimization yields about 2% savings (from 18.79% to 2.19% + 14.56%)

```
system  shipout    15.14%  104.49 ms   1130   92.47 us  system
   3        29     14.56%  100.45 ms  60839    1.65 us  bible-opt.tex
system  linebrk    12.09%   83.41 ms  25778    3.23 us  system
   5       666      2.19%   15.13 ms  55847  270.00 ns  plain.tex
system  buildpg     1.74%   12.02 ms  55190  217.00 ns  system
   3        56      0.87%    5.97 ms   3552    1.68 us  bible-opt.tex
```

Profiling bible.tex: The Call Graph

Running tprof -G bible

Before optimization

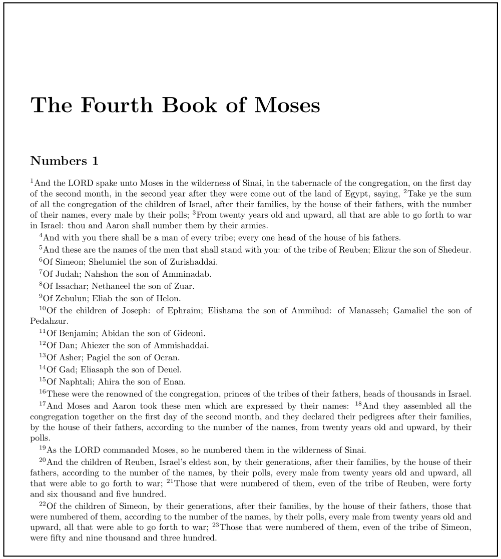
```
\Verse
   195.88 ms          27.24%        *          \Verse
   141.31 ms          72.14%    31011          \Verse
    54.56 ms          27.86%     498/1130      \output
```

After optimization, switching to horizontal mode moved to `\leavevmode`.

```
\Verse
   173.76 ms          25.18%        *          \Verse
   100.74 ms          57.98%    31011          \Verse
    73.02 ms          42.02%   31011/31011     \leavevmode
\leavevmode
    73.02 ms          10.58%        *          \leavevmode
    20.04 ms          27.44%    31011          \leavevmode
    52.98 ms          72.56%     499/1130      \output
```

Example: labible

- large size: 1119 pages
- low complexity
- LaTeX



The Fourth Book of Moses

Numbers 1

1 And the LORD spake unto Moses in the wilderness of Sinai, in the tabernacle of the congregation, on the first day of the second month, in the second year after they were come out of the land of Egypt, saying, 2 Take ye the sum of all the congregation of the children of Israel, after their families, by the house of their fathers, with the number of their names, every male by their polls; 3 From twenty years old and upward, all that are able to go forth to war in Israel: thou and Aaron shall number them by their armies.

4 And with you there shall be a man of every tribe; every one head of the house of his fathers.

5 And these are the names of the men that shall stand with you: of the tribe of Reuben; Elizur the son of Shedeur.

6 Of Simeon; Shelumiel the son of Zurishaddai.

7 Of Judah; Nahshon the son of Amminadab.

8 Of Issachar; Nethaneel the son of Zuar.

9 Of Zebulun; Eliab the son of Helon.

10 Of the children of Joseph: of Ephraim; Elishama the son of Ammihud: of Manasseh; Gamaliel the son of Pedahzur.

11 Of Benjamin; Abidan the son of Gideoni.

12 Of Dan; Ahiezer the son of Ammishaddai.

13 Of Asher; Pagiel the son of Ocran.

14 Of Gad; Eliasaph the son of Deuel.

15 Of Naphtali; Ahira the son of Enan.

16 These were the renowned of the congregation, princes of the tribes of their fathers, heads of thousands in Israel.

17 And Moses and Aaron took these men which are expressed by their names: 18 And they assembled all the congregation together on the first day of the second month, and they declared their pedigrees after their families, by the house of their fathers, according to the number of the names, from twenty years old and upward, by their polls.

19 As the LORD commanded Moses, so he numbered them in the wilderness of Sinai.

20 And the children of Reuben, Israel's eldest son, by their generations, after their families, by the house of their fathers, according to the number of the names, by their polls, every male from twenty years old and upward, all that were able to go forth to war; 21 Those that were numbered of them, even of the tribe of Reuben, were forty and six thousand and five hundred.

22 Of the children of Simeon, by their generations, after their families, by the house of their fathers, those that were numbered of them, according to the number of the names, by their polls, every male from twenty years old and upward, all that were able to go forth to war; 23 Those that were numbered of them, even of the tribe of Simeon, were fifty and nine thousand and three hundred.

Profiling labible: The Top Ten Lines

The LaTeX version

Profiled time 1.89s instead of 0.87s.

```
  file    line   percent    absolut    count    average  file
   3        29     8.97%  169.65 ms  110697    1.53 us  labible.tex
  10      3482     7.35%  138.91 ms   10106   13.74 us  expl3-code.tex
system shipout     6.30%  119.17 ms    1119  106.49 us  system
system linebrk     4.96%   93.74 ms   25711    3.65 us  system
   5      7305     3.96%   74.82 ms   25711    2.91 us  latex.ltx
  10      2101     3.34%   63.19 ms    3408   18.54 us  expl3-code.tex
   5      7312     2.99%   56.58 ms   77131  733.00 ns  latex.ltx
   5     15014     2.50%   47.33 ms    1119   42.30 us  latex.ltx
   5     16672     2.38%   45.00 ms    2238   20.11 us  latex.ltx
   5      7294     1.72%   32.57 ms   25711    1.27 us  latex.ltx
```

Profiling labible: The Call Graph

The LaTeX version

The most expensive macros:

```
      time      loop     percent    count/total  child
\output
    665.34 ms            35.19%          *       \output
      4.07 ms             0.61%      1192        \output
    651.31 ms            97.89%     1119/1119    \@opcol
      7.09 ms             1.07%     1119/1119    \@makecol
      2.51 ms             0.38%     1119/1119    \@startcolumn
    371.04 us             0.06%       73/73      \@specialoutput
\Verse
    659.22 ms            34.87%          *       \Verse
    174.99 ms            26.54%     31011        \Verse
    453.76 ms            68.83%   24336/25708    \everypar [5,7275]
     30.47 ms             4.62%   31011/31011    \everymath
      7.67 us             0.00%        1/3       \everypar [5,7282]
```

Profiling labible: The Call Graph

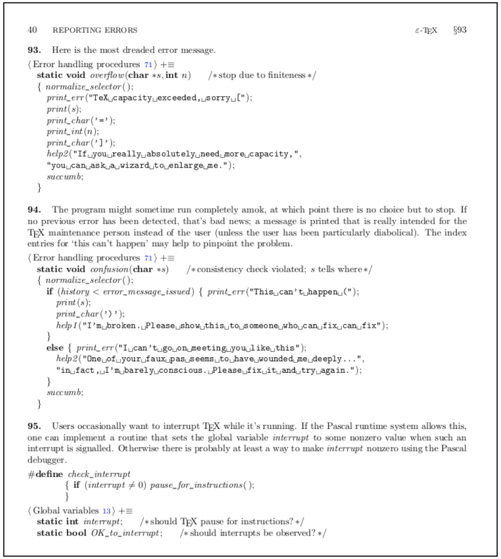
The LaTeX version

The most expensive macros (continued):

```
       time         loop    percent   count/total  child
\@opcol
    651.31 ms               34.45%         *       \@opcol
    610.04 us                0.09%     1119        \@opcol
    354.22 ms               54.39%    1119/1119    \@outputpage
    294.33 ms               45.19%    1119/1119    \@expl@@@mark@update@singlec
\use_i:nn
    641.58 ms               33.94%         *       \use_i:nn
    169.08 ms               26.35%    57501        \use_i:nn
    210.67 ms   74.22 ms    32.84%     463/1192    \output
    121.55 ms   51.23 ms    18.95%    1119/1119    \__mark_update_structure:nn
     75.73 ms  453.25 ms    11.79%   25711/25711   \mode_if_inner:F [1,1]
     26.87 ms   16.18 ms     4.19%    1119/6761    \seq_map_inline:Nn
```

Example: texprof

- medium size: 758 pages
- use of macros and macro complexity is high
- plain TeX



40 REPORTING ERRORS ε-TeX §93

93. Here is the most dreaded error message.

```
⟨Error handling procedures 71⟩ +≡
  static void overflow(char *s, int n)    /* stop due to finiteness */
  { normalize_selector();
    print_err("TeX␣capacity␣exceeded,␣sorry␣[");
    print(s);
    print_char('=');
    print_int(n);
    print_char(']');
    help2("If␣you␣really␣absolutely␣need␣more␣capacity,",
    "you␣can␣ask␣a␣wizard␣to␣enlarge␣me.");
    succumb;
  }
```

94. The program might sometime run completely amok, at which point there is no choice but to stop. If no previous error has been detected, that's bad news; a message is printed that is really intended for the TeX maintenance person instead of the user (unless the user has been particularly diabolical). The index entries for 'this can't happen' may help to pinpoint the problem.

```
⟨Error handling procedures 71⟩ +≡
  static void confusion(char *s)    /* consistency check violated; s tells where */
  { normalize_selector();
    if (history < error_message_issued) { print_err("This␣can't␣happen␣(");
      print(s);
      print_char(')');
      help1("I'm␣broken.␣Please␣show␣this␣to␣someone␣who␣can␣fix␣can␣fix");
    }
    else { print_err("I␣can't␣go␣on␣meeting␣you␣like␣this");
      help2("One␣of␣your␣faux␣pas␣seems␣to␣have␣wounded␣me␣deeply...",
      "in␣fact,␣I'm␣barely␣conscious.␣Please␣fix␣it␣and␣try␣again.");
    }
    succumb;
  }
```

95. Users occasionally want to interrupt TeX while it's running. If the Pascal runtime system allows this, one can implement a routine that sets the global variable *interrupt* to some nonzero value when such an interrupt is signalled. Otherwise there is probably at least a way to make *interrupt* nonzero using the Pascal debugger.

```
#define check_interrupt
        { if (interrupt ≠ 0) pause_for_instructions();
        }
⟨Global variables 13⟩ +≡
  static int interrupt;    /* should TeX pause for instructions? */
  static bool OK_to_interrupt;    /* should interrupts be observed? */
```

Profiling texprof.tex

texprof pretends to be hitex/pdftex

- Runtimes:

Runtimes:	tex:	270ms
	texprof:	280ms
	texprof -prof:	410ms
	pdftex:	2315ms
	pdftex –draftmode:	1610ms
	hitex:	1610ms

Profiling texprof.tex

texprof pretends to be hitex/pdftex

- Runtimes:

tex:	270ms
texprof:	280ms
texprof -prof:	410ms
pdftex:	2315ms
pdftex –draftmode:	1610ms
hitex:	1610ms

- texprof pretending to be hitex

  include:

```
\def\HINTversion{1.2}
\def\HINTdest#1 #2{}
\def\HINTcontents#1#2#3{#3}
\def\HINToutline goto #1 #2 depth #3 #4{}
\def\HINTstartlink goto num #1 #2{#2}
\def\HINTendlink{}
```

Profiling texprof.tex

texprof pretends to be hitex/pdftex

- Runtimes:

tex:	270ms
texprof:	280ms
texprof -prof:	410ms
pdftex:	2315ms
pdftex –draftmode:	1610ms
hitex:	1610ms

- texprof pretending to be hitex

  include:

```
\def\HINTversion{1.2}
\def\HINTdest#1 #2{}
\def\HINTcontents#1#2#3{#3}
\def\HINToutline goto #1 #2 depth #3 #4{}
\def\HINTstartlink goto num #1 #2{#2}
\def\HINTendlink{}
```

- Runtime: texprof –prof: 1600ms

Profiling texprof.tex

texprof pretends to be hitex/pdftex

The top ten lines:

```
 file    line  percent   absolut   count   average  file
   7      156   19.27% 361.93 ms  173966   2.08 us  cwebacromac.tex
   7      157   14.34% 269.25 ms  133574   2.02 us  cwebacromac.tex
   7      158    9.92% 186.26 ms  134424   1.39 us  cwebacromac.tex
   7      159    6.48% 121.72 ms  110146   1.10 us  cwebacromac.tex
   0        0    4.46%  83.83 ms  196011 427.00 ns  unknown
   7      172    4.15%  77.88 ms   15808   4.93 us  cwebacromac.tex
   7      173    3.64%  68.34 ms   37002   1.85 us  cwebacromac.tex
system  shipout  2.82%  52.95 ms     777  68.15 us  system
system  linebrk  2.70%  50.64 ms   27368   1.85 us  system
   7      152    2.38%  44.61 ms   26960   1.65 us  cwebacromac.tex
```

Profiling texprof.tex

Adding links to the index

The four lines that account for 50% of the runtime

```
\def\addtokens#1#2{\edef\addtoks{\noexpand#1={\the#1#2}}\addtoks}
\def\poptoks#1#2|ENDTOKS|{\let\first=#1\toksD={#1}%
 \ifcat\noexpand\first0\countB=`#1\else\countB=0\fi\toksA={#2}}
\def\maketoks{\expandafter\poptoks\the\toksA|ENDTOKS|%
```

... *Define* `\next` *based on the next character either as* `\maketoks` *or* `\maketoksdone`

```
  \next
}
```

Profiling texprof.tex: The Call Graph

Recursive loop

The most expensive macros:

```
        time      loop  percent   count/total    child
\pdfnote [7,152]
     1.25 s             66.76%        *          \pdfnote [7,152]
    24.91 ms             1.99%     8473          \pdfnote [7,152]
     1.20 s             95.89%    8473/8473      \maketoks [7,159]
    15.05 ms             1.20%   24230/28737     \pdflink [7,145]
    11.45 ms             0.91%    4507/4507      \[ [5,334]
    88.13 us             0.01%      80/80        \ETs [5,177]
    59.87 us             0.00%      57/57        \ET [5,176]
   328.00 ns             0.00%       1/3         \glob [3,167]
```

Profiling texprof.tex: The Call Graph

Recursive loop

The most expensive macros (continued):

```
        time     loop  percent   count/total   child
\maketoks [7,159]
     1.20 s             64.02%       *         \maketoks [7,159]
     4.37 ms             0.36%    8473         \maketoks [7,159]
     1.18 s             97.87%    8473/125093  \next [7,159]
    13.48 ms             1.12%    8473/133566  \poptoks [7,157]
     7.71 ms             0.64%    8473/173958  \addtokens [7,156]
```

Profiling texprof.tex: The Call Graph

Recursive loop

The most expensive macros (continued):

```
        time      loop  percent   count/total   child
\next [7,159]
     1.18 s              62.66%        *        \next [7,159]
    51.93 ms              4.41%   125093        \next [7,159]
     0.00 ns    1.15 s   97.90%   51084/125093  \next [7,159]
   490.86 ms             41.71%    5676/173958  \addtokens [7,156]
   441.73 ms             37.54%   59557/133566  \poptoks [7,157]
   180.71 ms             15.36%   28737/28737   \makenote [7,172]
    11.49 ms              0.98%    8473/8473    \next [7,174]
```

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Outlook

- If there is substantial demand, there is room for improvements.

Summary

Summary

- The TeX profiler is a specialized tool for macro writers.
- The TeX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Outlook

- If there is substantial demand, there is room for improvements.

Questions and Discussion

- Should the profiler go into TeX Live?

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Outlook

- If there is substantial demand, there is room for improvements.

Questions and Discussion

- Should the profiler go into TEX Live?
- Is parsing input files necessarily slow?

Summary

Summary

- The TEX profiler is a specialized tool for macro writers.
- The TEX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Outlook

- If there is substantial demand, there is room for improvements.

Questions and Discussion

- Should the profiler go into TEX Live?
- Is parsing input files necessarily slow?
- Do we need special TEX primitives to speed up common tasks?

Summary

Summary

- The TeX profiler is a specialized tool for macro writers.
- The TeX profiler is open source and available on GitHub: //github.com/ruckertm/HINT/
- It is not perfect, but gives useful information.
- It will tell you where there is a chance for optimization.
- It will tell you if there is no need or little chance of optimization.

Outlook

- If there is substantial demand, there is room for improvements.

Questions and Discussion

- Should the profiler go into TeX Live?
- Is parsing input files necessarily slow?
- Do we need special TeX primitives to speed up common tasks?
- Thank you for your attention!