Perl::Formance 2015

Steffen Schwigon
"renormalist"

http://github.com/renormalist
http://perlformance.net

Benchmarking Perl


BORING

What's the big deal?

- use Benchmark;
- timeit 5, sub { interesting_code() }

done?

NO

Doing It Right™

Rekapitulacija 2015

What is the benchmark target?

Perl

Perl 5

Doing It Right™

Workloads

Workloads

Rathole

Workloads

Workloads

Micro vs. Medium vs. Macro

Workloads

Micro vs. Medium vs. Macro

Just remember at Judgement Day!

Workloads

Micro vs. Medium vs. Macro

Just remember at Judgement Day!

Panic or Victory vs. *Indicator*

Stable numbers

- finetune runtime
- warmup - internal repeat cycles
- dedicated server
- switch-off turbo boost
- OS noise (ASLR)
- rerun 10..20 times

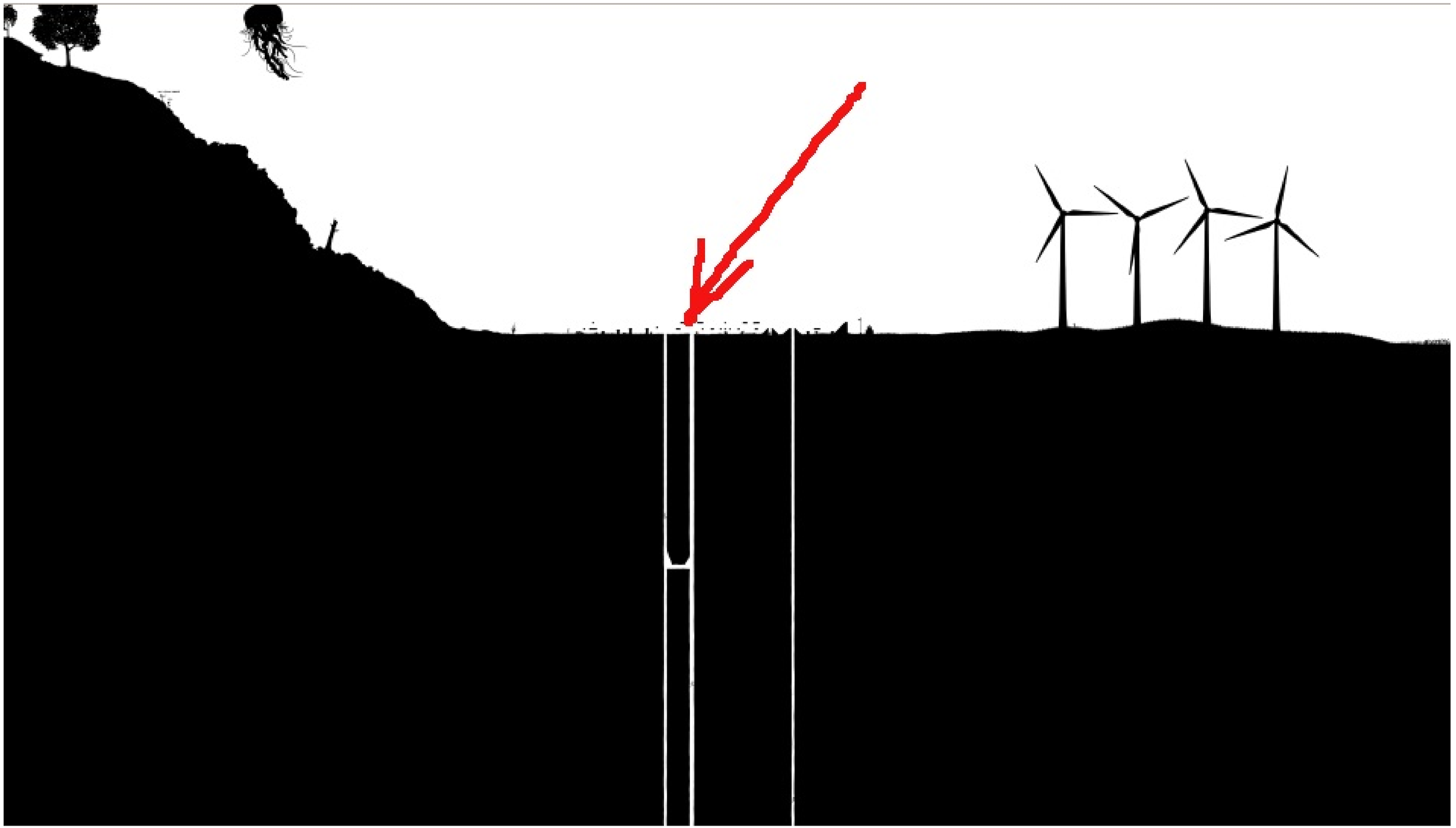
Ultimate Goal

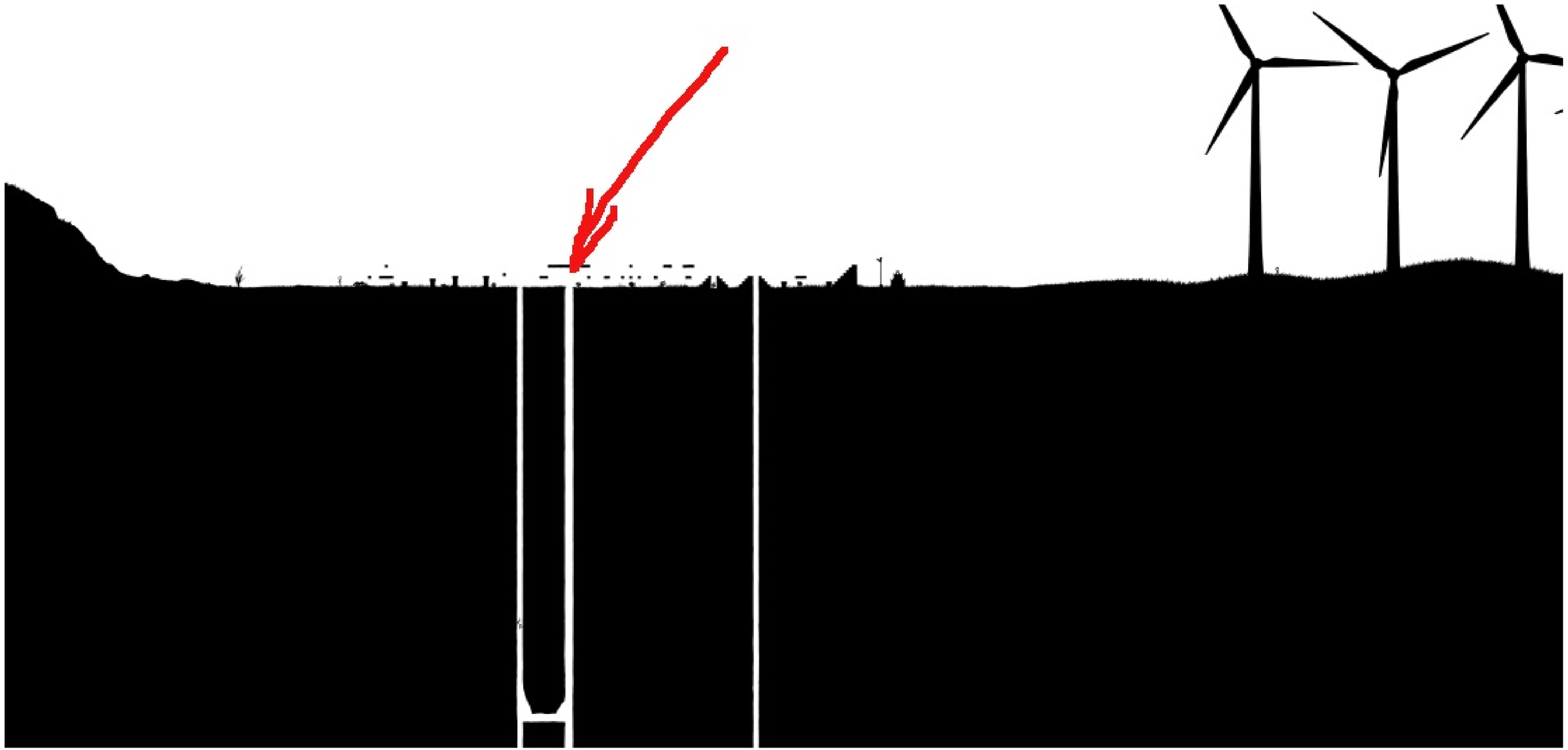
Trust

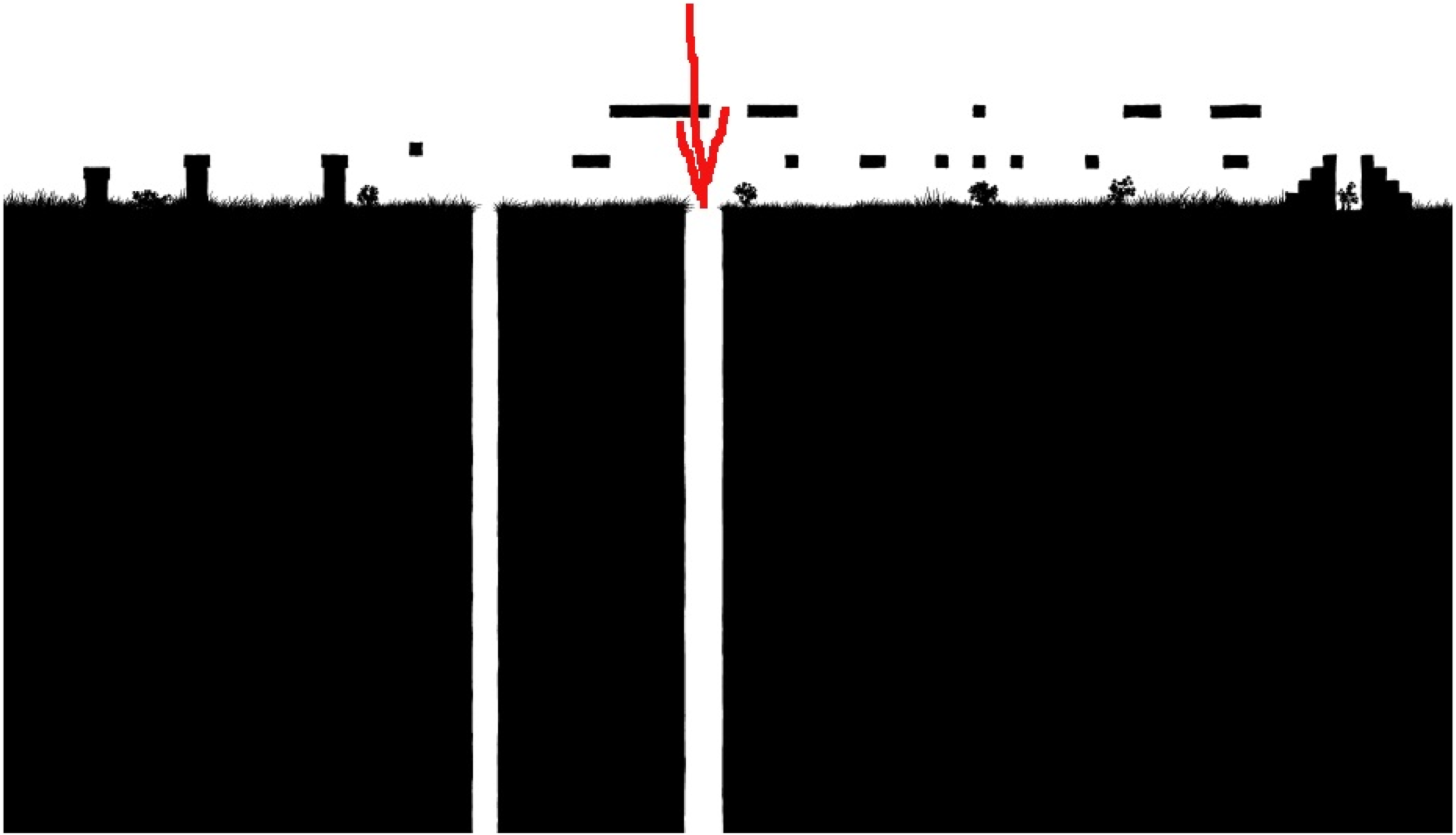
Doing It Right™

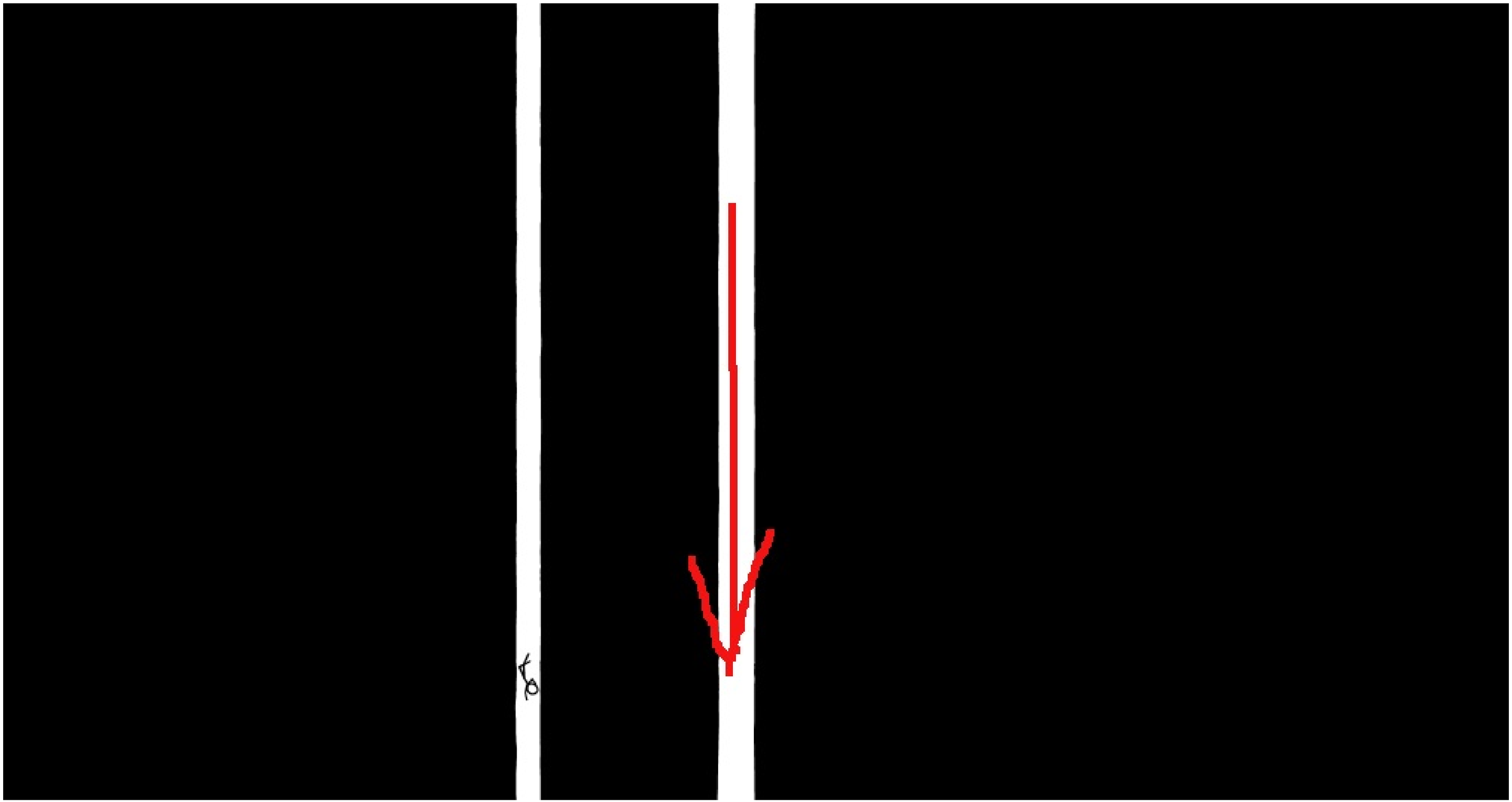
Building Perl from git

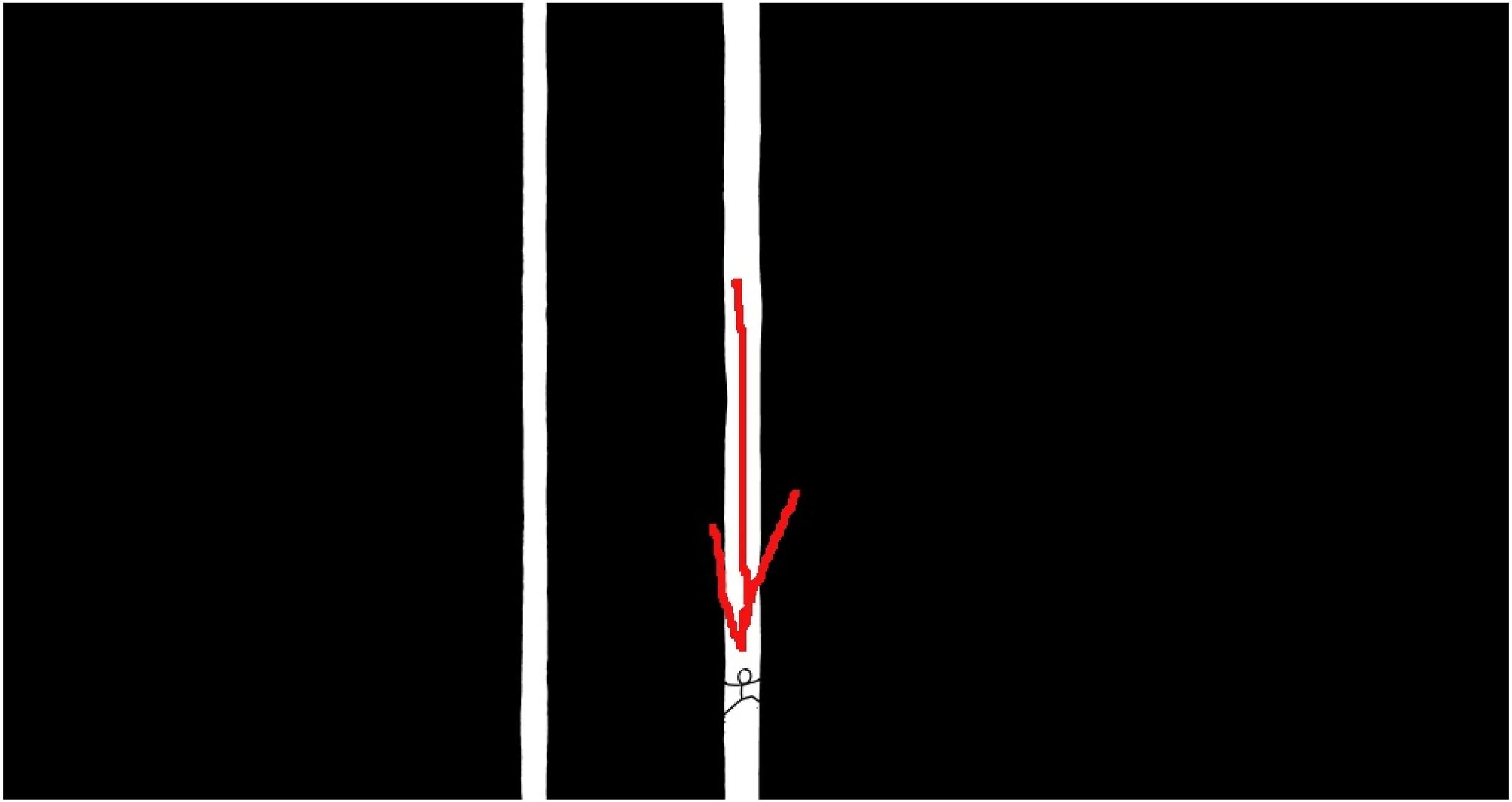
Building Perl from git

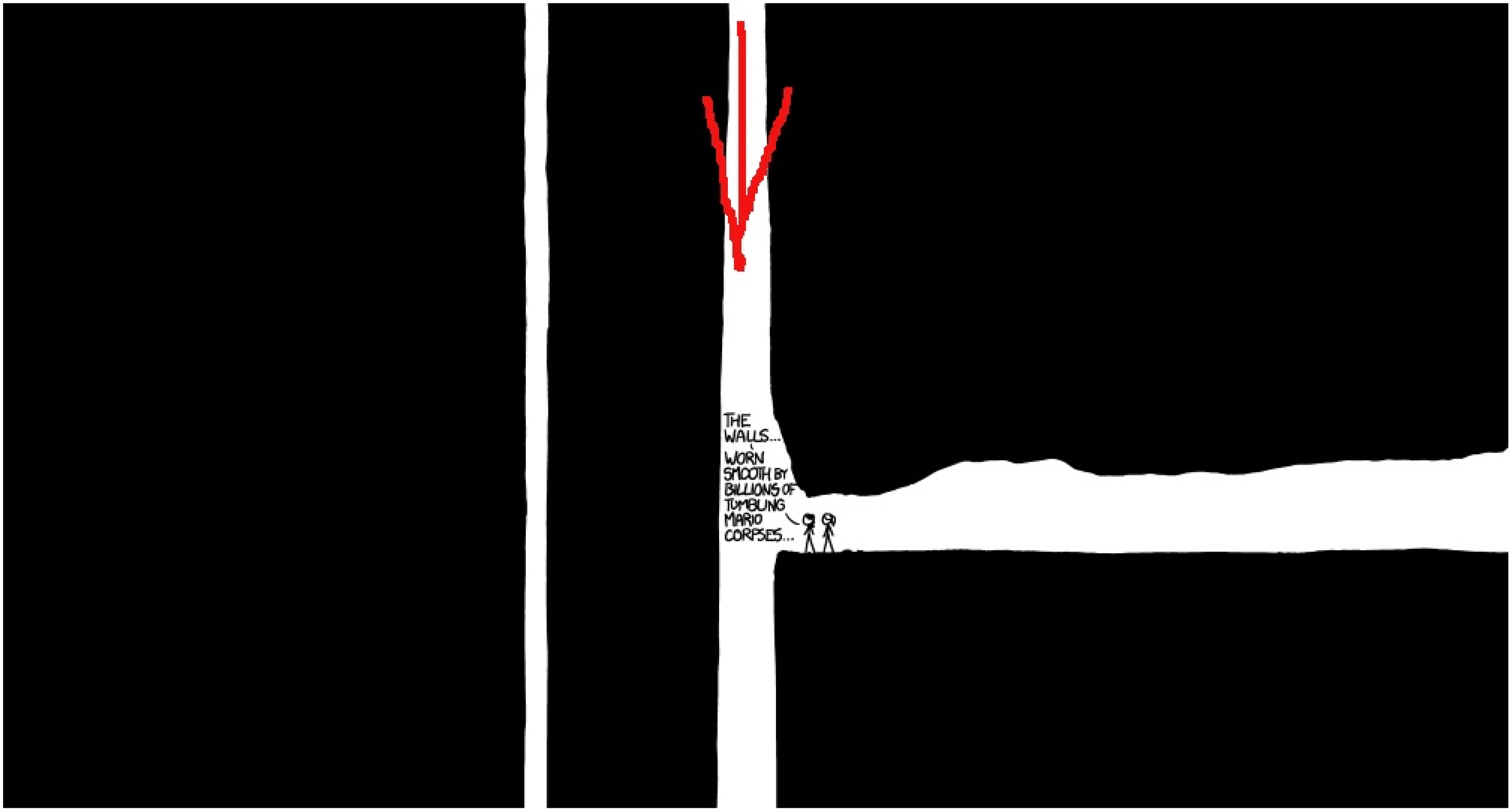


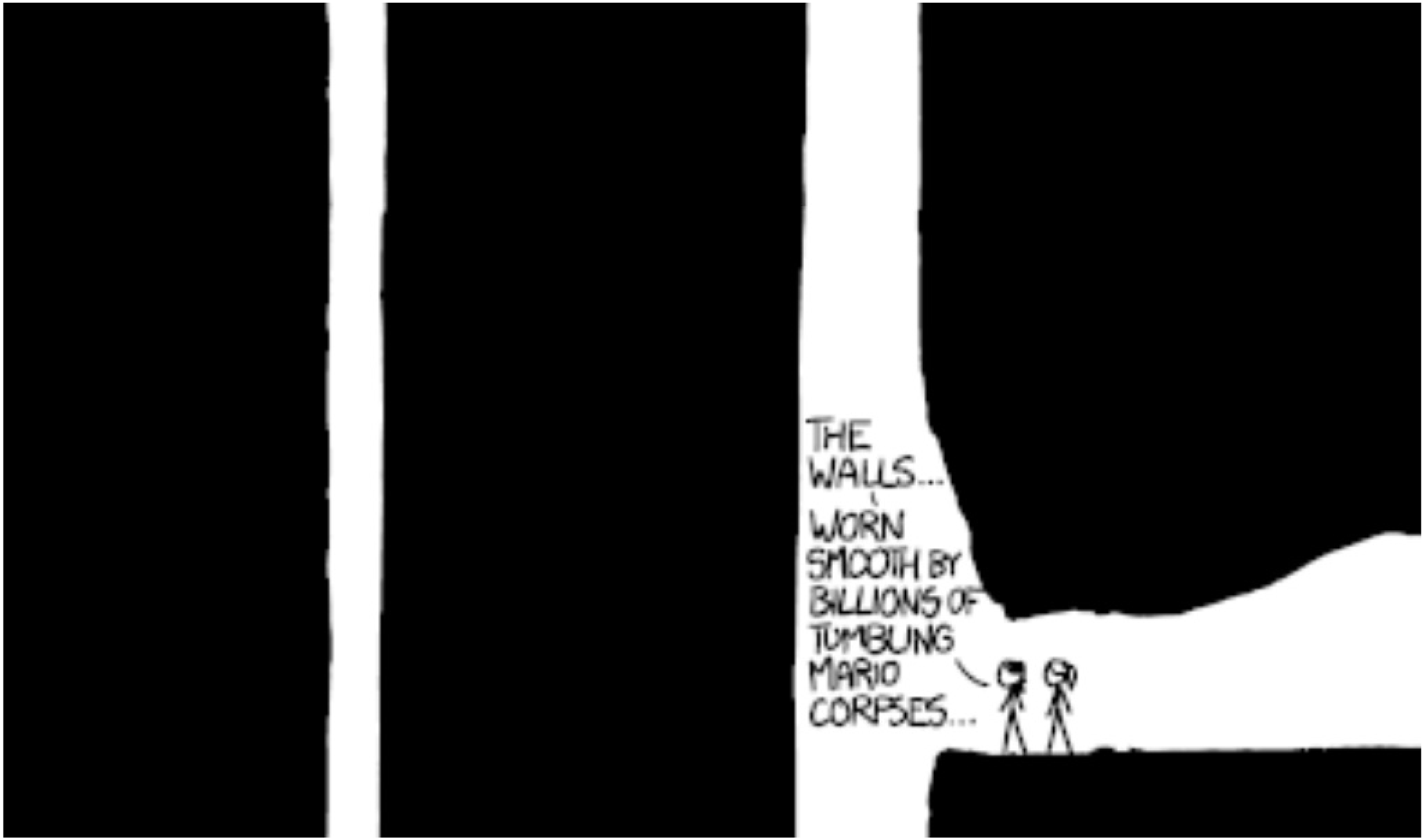









THE WALLS... WORN SMOOTH BY BILLIONS OF TUMBLING MARIO CORPSES...


THE WALLS...
WORN SMOOTH BY BILLIONS OF TUMBLING MARIO CORPSES...

Building Perl from git

Building Perl from git

- App::Bootstrap::Perl
- CPAN.pm distroprefs
- cherry picking from the future

Doing It Right™

Keeping CPAN stable

Keeping CPAN stable

Keeping CPAN stable

Keeping CPAN stable

- CPAN::Mini -->no
- CPAN::Mini + git snapshots --> no
- real CPAN mirror + git snapshots --> no
- Pinto + module patches --> nearly

Keeping CPAN stable

- CPAN::Mini --> no
- CPAN::Mini + git snapshots --> no
- real CPAN mirror + git snapshots --> no
- Pinto + module patches --> nearly

5.18 hash key order randomization!

Keeping CPAN stable

- CPAN::Mini -->no
- CPAN::Mini + git snapshots --> no
- real CPAN mirror + git snapshots --> no
- Pinto + module patches --> nearly

Keeping CPAN stable

- CPAN::Mini -->no
- CPAN::Mini + git snapshots --> no
- real CPAN mirror + git snapshots --> no
- Pinto + module patches --> nearly
- **Just use a real CPAN mirror + occasional sync** --> YES

Doing It Right™

A benchmark storage

A benchmark storage

- Forget the monitoring tools!
- I want to query and slice data points by arbitrary criteria

http://benchmarkanything.org

Doing It Right™

Doing It Right™

Benchmarking Perl

Benchmarking Perl

- That's what I actually started this year at the Perl QA Hackathon 2015 in Berlin
- lots of data points and meta information

Benchmarking Perl

- Runtime! Underestimated!
- 260 versions - takes **days** to run all

- 5.[10..23].[0..10]
- (no)usethreaded
- (no)use64bit
- some long-running benchmarks
- some actual speed regressions!

Evaluate results

- Currently transform into google charts

`http://perlformance.net/charts/`

Results

Understanding the charts

- measuring time, unit=seconds
- smaller is better
- "/threads" or /nothreads" means
  Perl was compiled with `-Dusethreads`
- average multiple points per version
  stable versions: 10-20 data points
  devel versions: only 2 data points,
  but just running more

Understanding the charts

`http://perlformance.net/charts/raw-numbers.txt`

...for more details like

- average
- confidence interval
- standard deviation
- data point count

Understanding the benchmark

Read the source code:

- https://goo.gl/pJgOCg
  (metacpan Benchmark-Perl-Formance)

Understanding the changes

What does x% less time mean?

- 10% - arthouse

Understanding the changes

What does x% less time mean?

- 10% - arthouse
- 20% - blockbuster

Understanding the changes

What does x% less time mean?

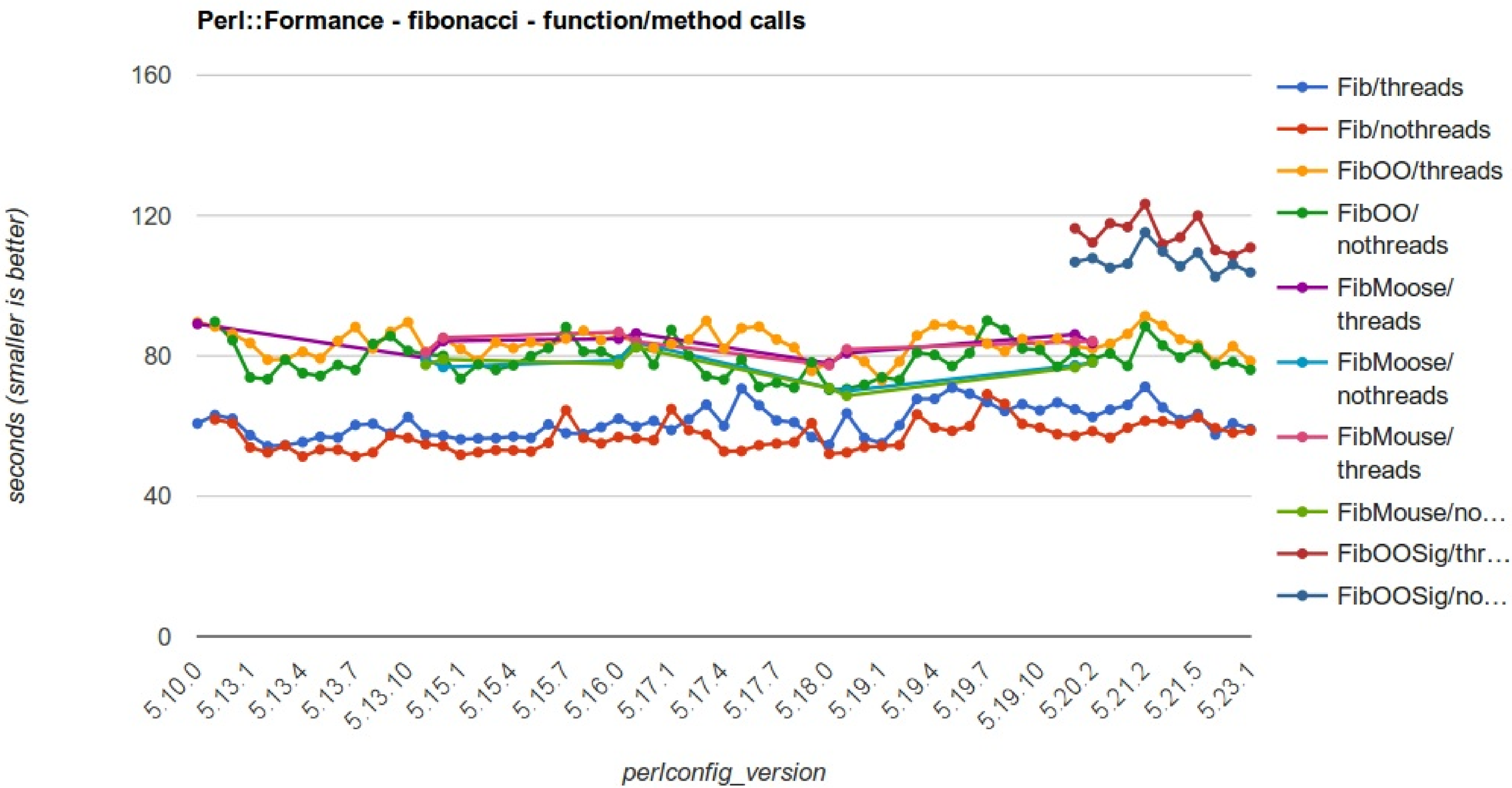
- 10% - arthouse
- 20% - blockbuster
- 30% - Apocalypse Now

Understanding the changes

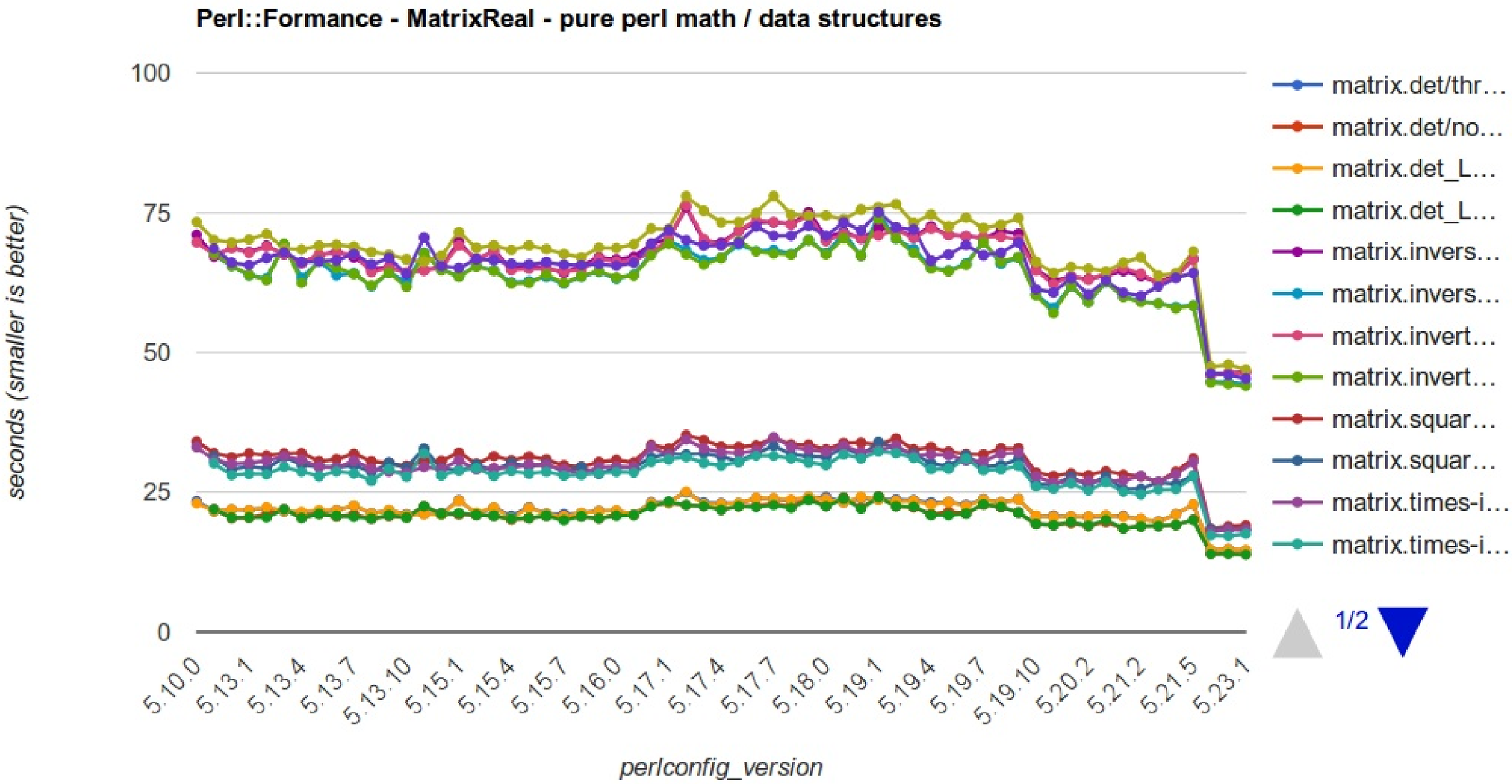
What does x% less time mean?

- 10% - arthouse
- 20% - blockbuster
- 30% - Apocalypse Now
- 50% - ...

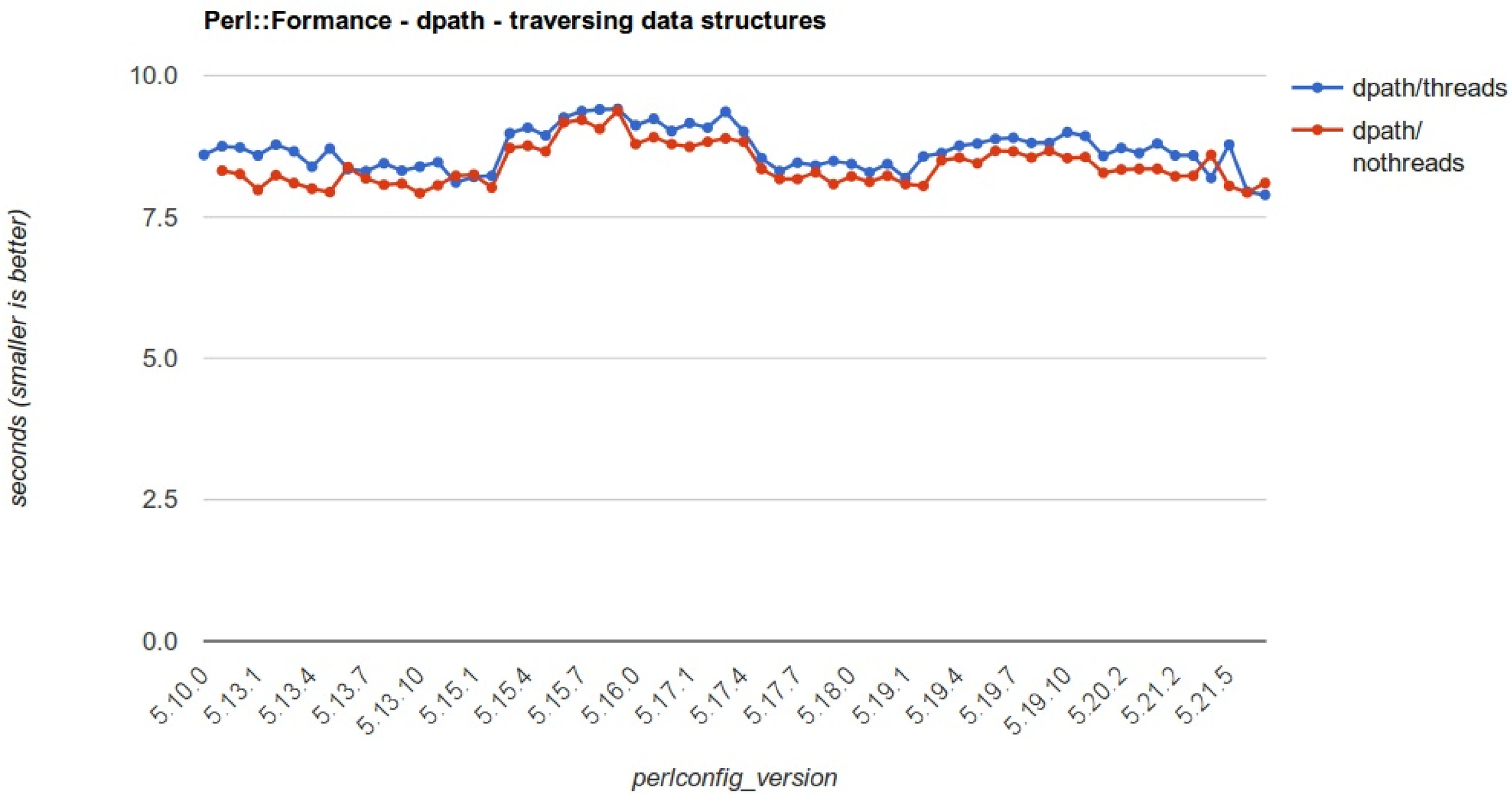
Fibonacci - function/method calls



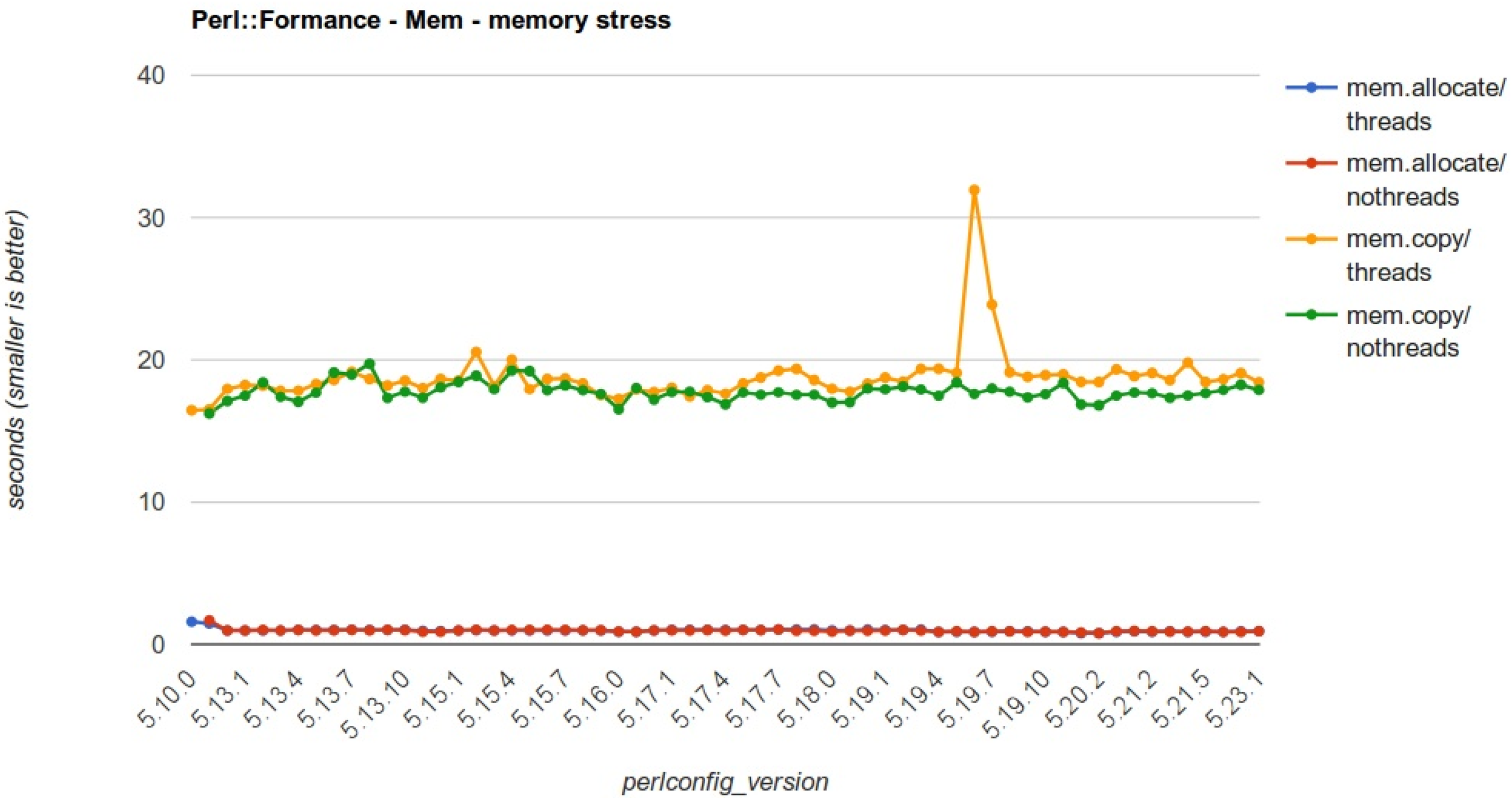
MatrixReal - pure perl math / data structs



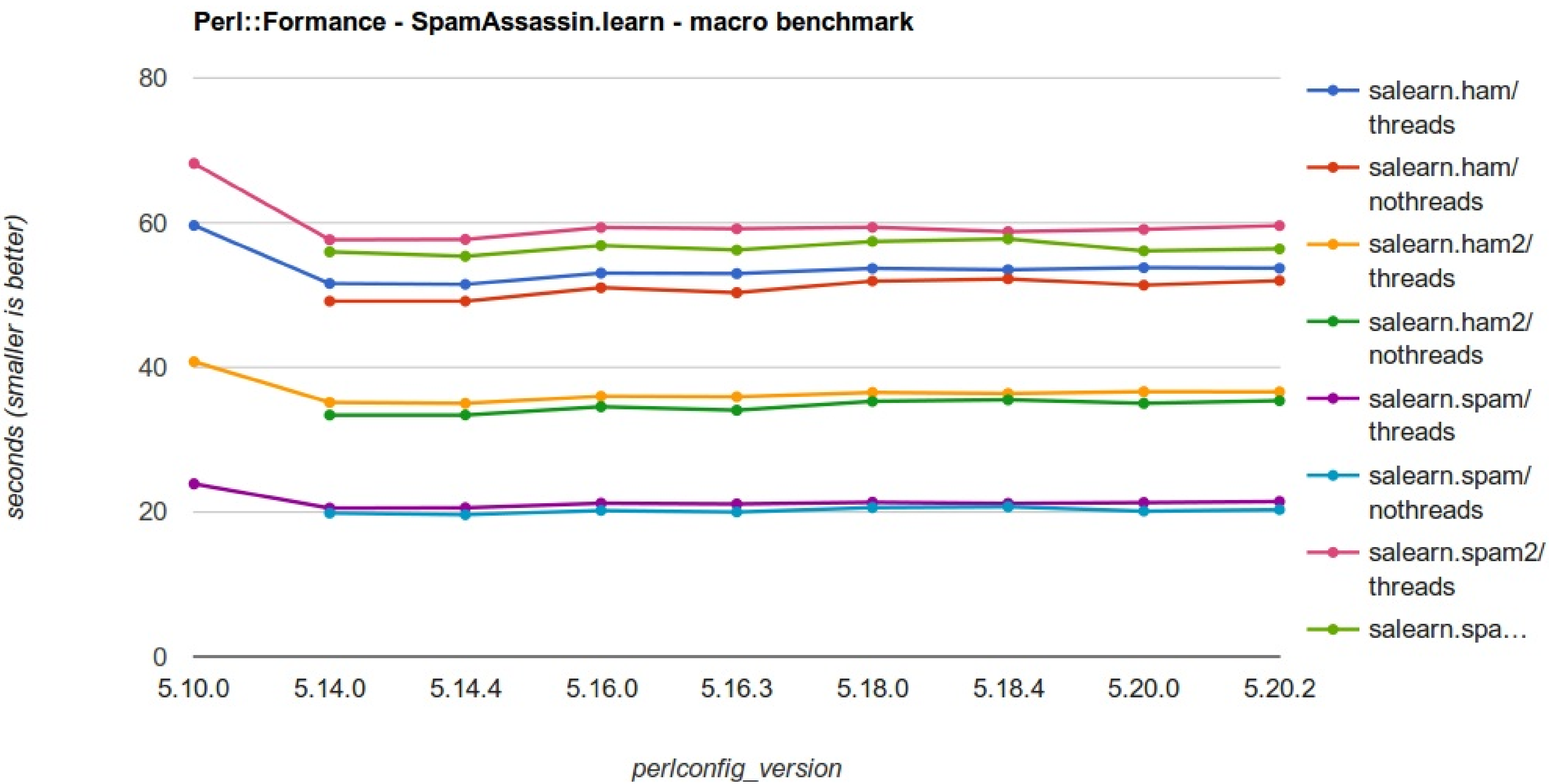
DPath - traverse data structures



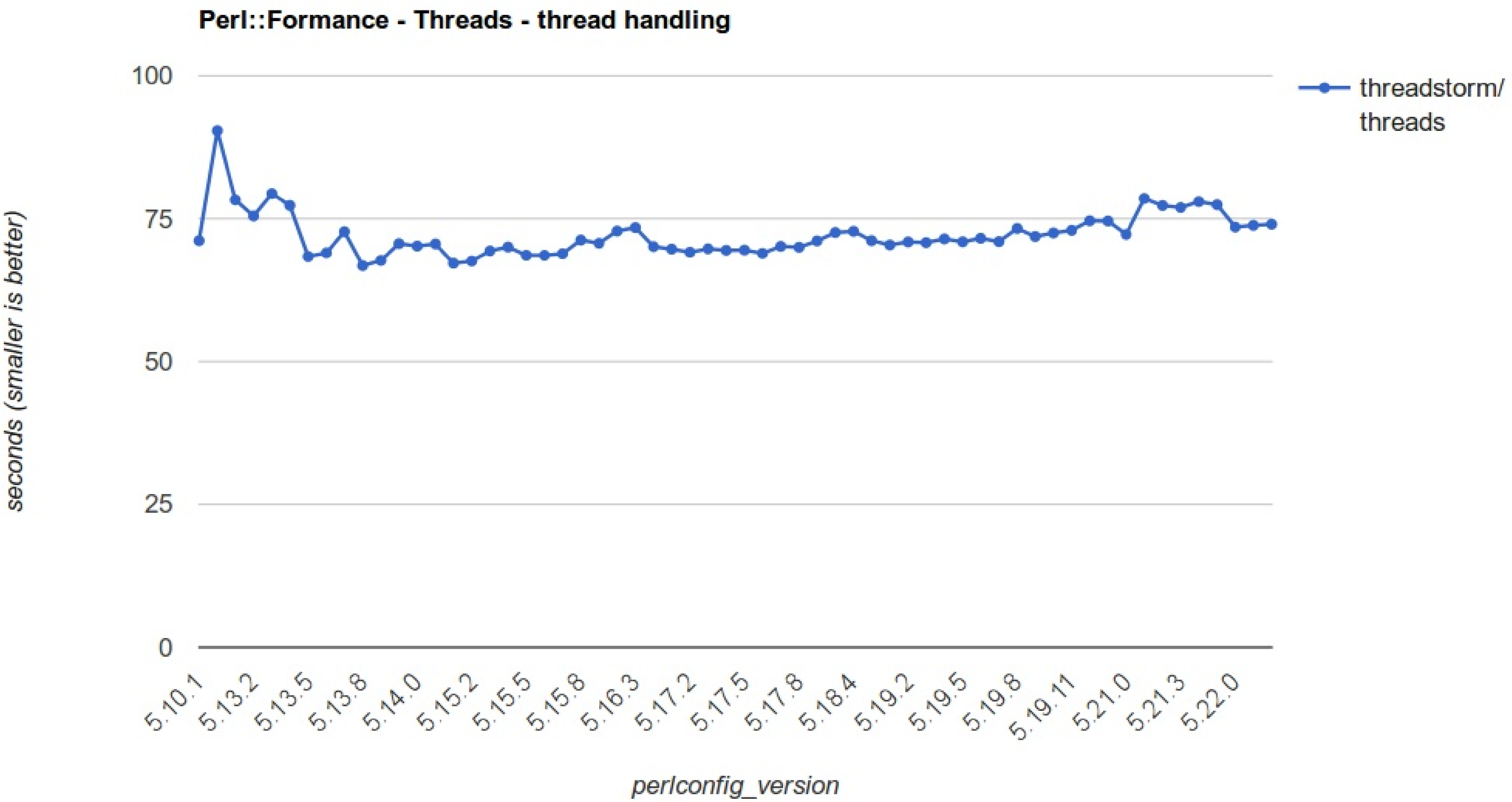
Mem - mem allocation + copy



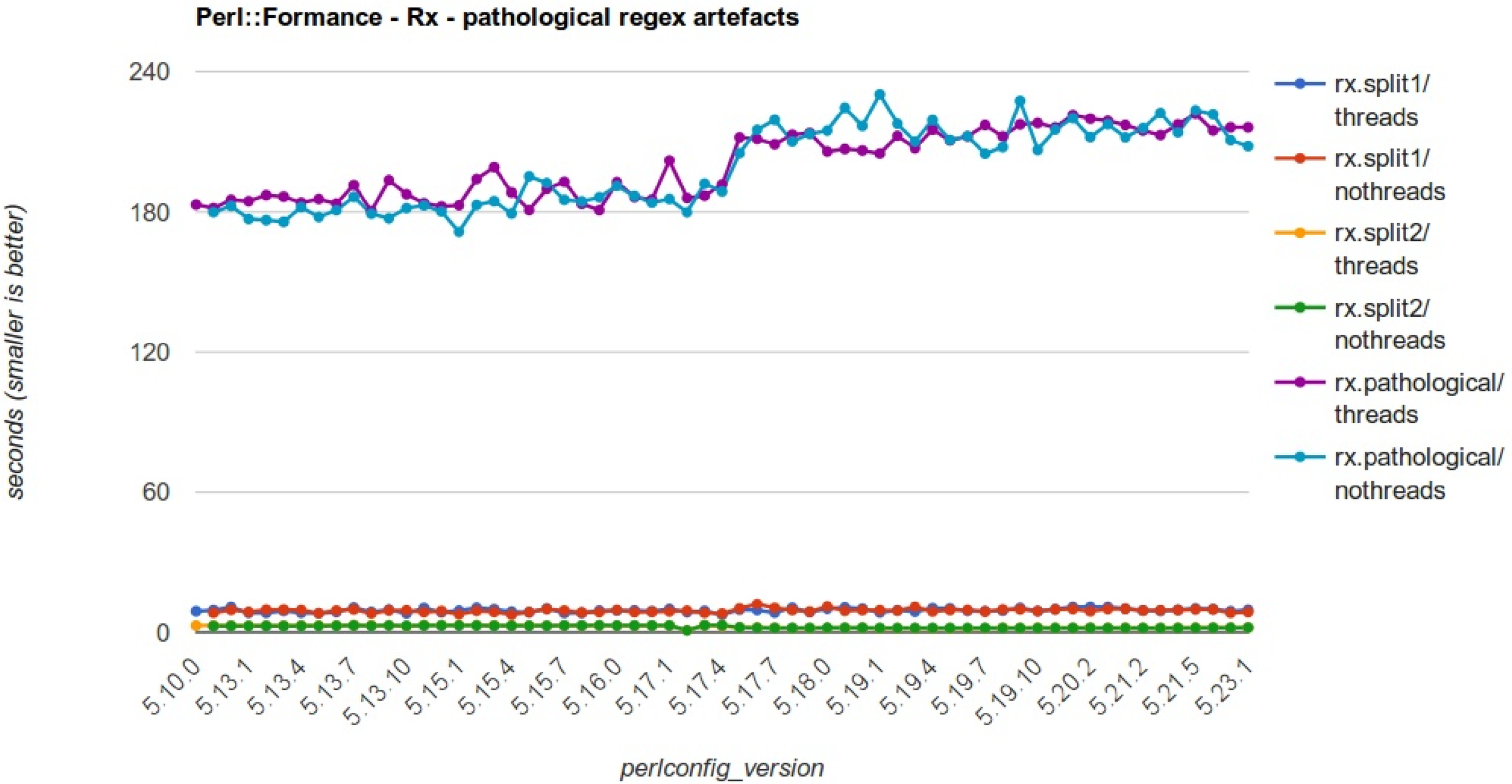
SpamAssassin - text processing



ThreadStorm - thread+join like hell



Rx - known regex pathologicaloids

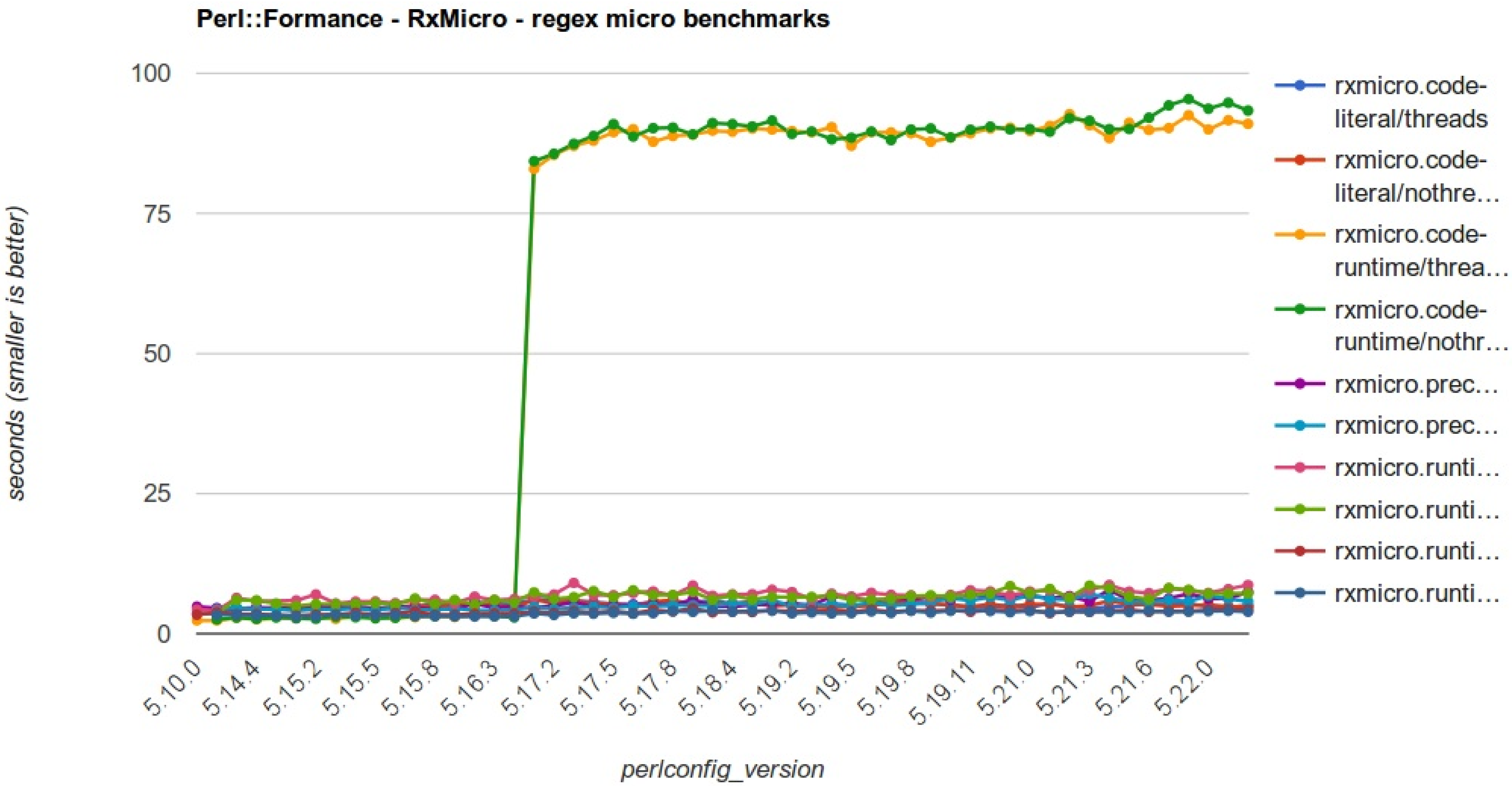


RxMicro - regex micro benchmarks

suggested snippets from Dave Mitchell

- way back in 2012
- "I'm sorry Dave, I'm afraid I couldn't do that" ...faster.
- and now they even find regressions

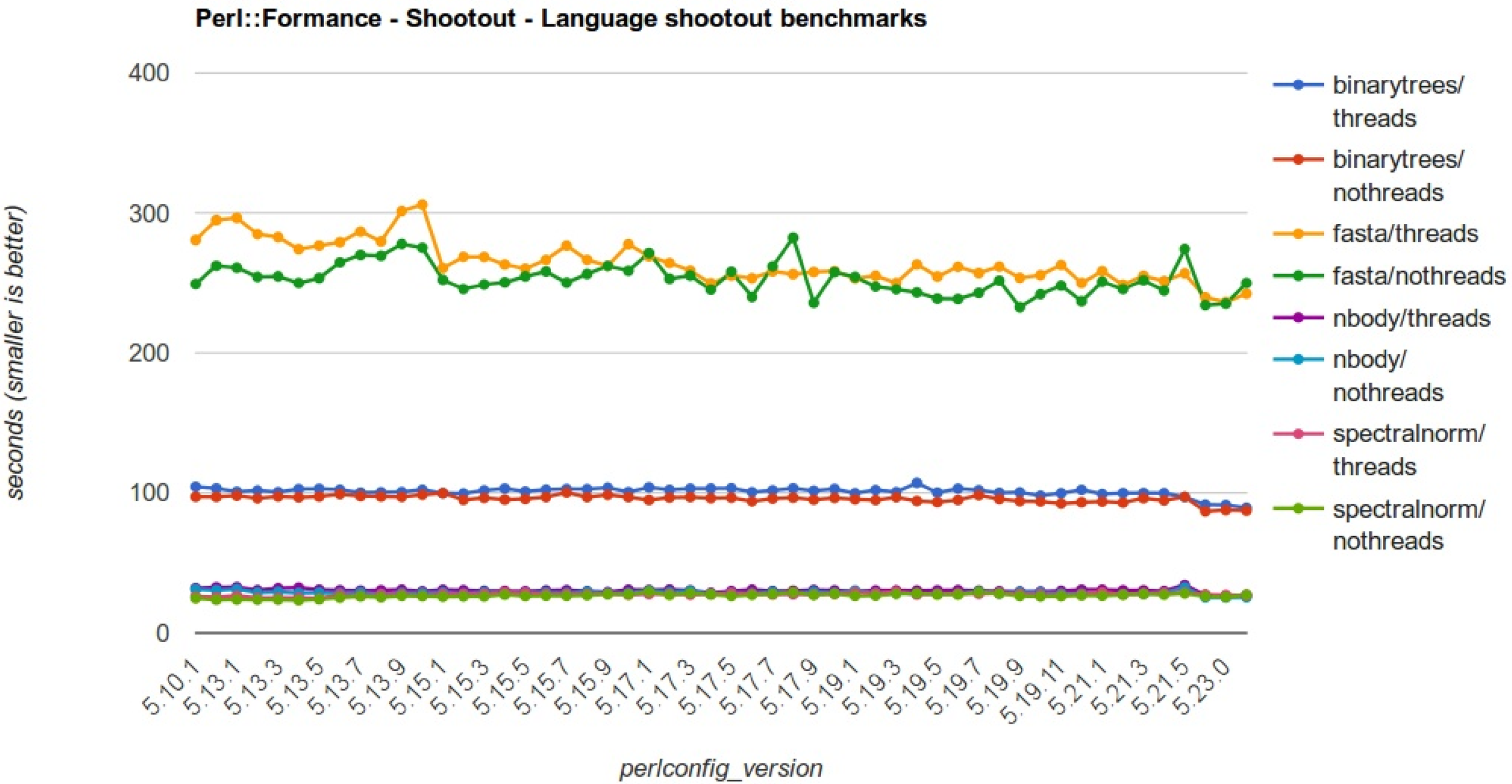
RxMicro - regex micro benchmarks



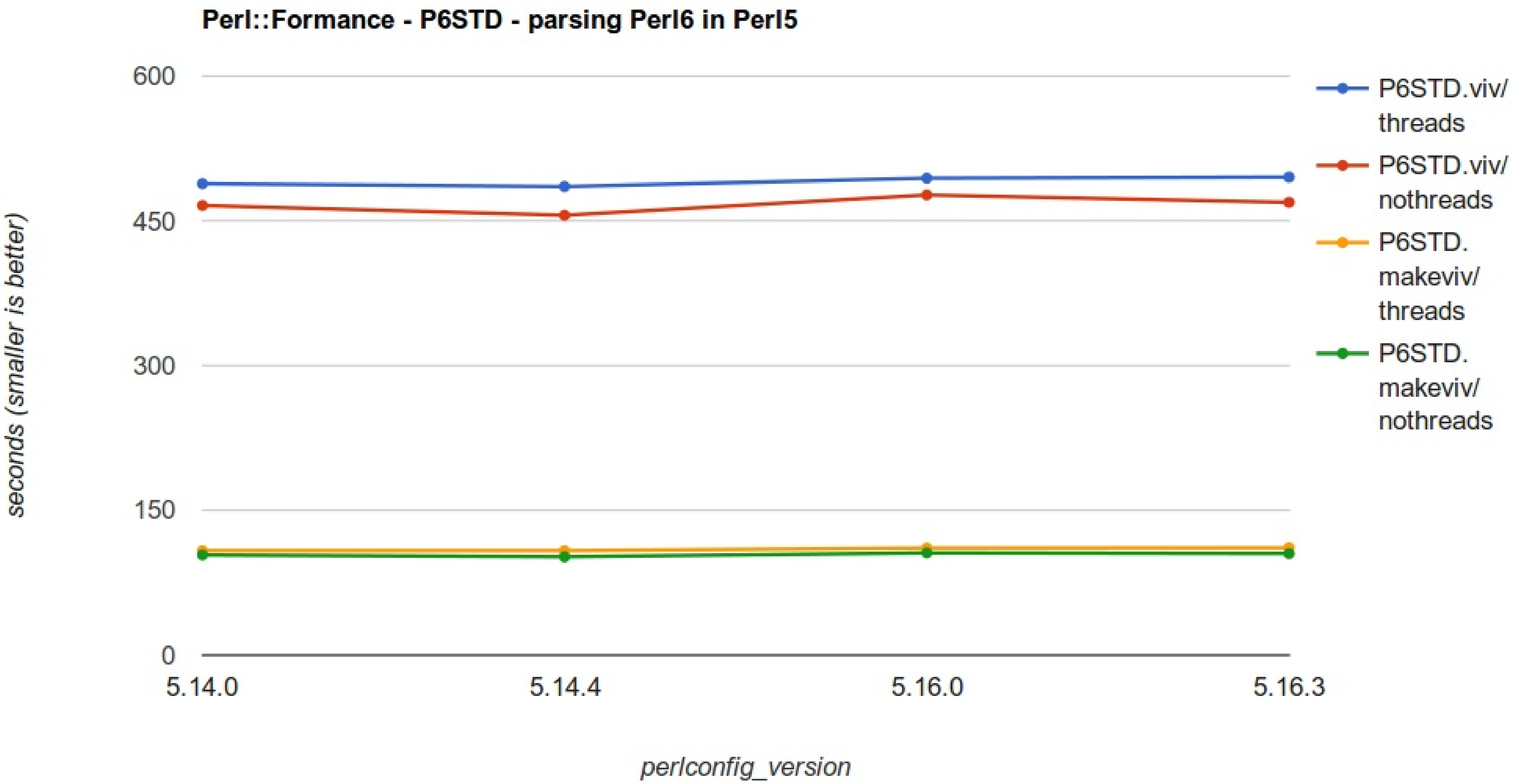
Shootout - algorithmic and parallel

- Perl code taken from
  The Computer Language Benchmarks game
  `http://shootout.alioth.debian.org`
- fork / threads

Shootout - algorithmic and parallel



P6STD - Perl 6 parsing in Perl 5



Raw data

```
Perl::Formance - chart rendering: Thu Sep  3 17:16:51 2015

* rx.pathological/threads - perlformance.perl5.Rx.regexes.pathological
* rx.pathological/nothreads - perlformance.perl5.Rx.regexes.pathological
  rx.pathological/threads . 5.19.3  . (ci95l..avg..ci95u) = (203.21 .. 207.30 .. 211.39) +- stdv  2.09 (  2 points)
  rx.pathological/threads . 5.15.7  . (ci95l..avg..ci95u) = (192.20 .. 192.88 .. 193.55) +- stdv  0.34 (  2 points)
  rx.pathological/threads . 5.14.0  . (ci95l..avg..ci95u) = (183.38 .. 183.76 .. 184.14) +- stdv  0.84 ( 20 points)
  rx.pathological/threads . 5.15.6  . (ci95l..avg..ci95u) = (181.63 .. 189.87 .. 198.11) +- stdv  4.20 (  2 points)
  rx.pathological/threads . 5.17.3  . (ci95l..avg..ci95u) = (184.42 .. 186.92 .. 189.42) +- stdv  1.27 (  2 points)
  rx.pathological/threads . 5.13.8  . (ci95l..avg..ci95u) = (161.29 .. 180.42 .. 199.54) +- stdv  9.76 (  2 points)
  rx.pathological/threads . 5.18.0  . (ci95l..avg..ci95u) = (203.13 .. 205.87 .. 208.61) +- stdv  6.10 ( 20 points)
  rx.pathological/threads . 5.17.5  . (ci95l..avg..ci95u) = (211.70 .. 211.87 .. 212.04) +- stdv  0.08 (  2 points)
  rx.pathological/threads . 5.19.4  . (ci95l..avg..ci95u) = (214.43 .. 215.22 .. 216.01) +- stdv  0.40 (  2 points)
  rx.pathological/threads . 5.15.9  . (ci95l..avg..ci95u) = (180.66 .. 180.75 .. 180.85) +- stdv  0.05 (  2 points)
  rx.pathological/threads . 5.13.4  . (ci95l..avg..ci95u) = (181.56 .. 183.98 .. 186.40) +- stdv  1.23 (  2 points)
  rx.pathological/threads . 5.13.3  . (ci95l..avg..ci95u) = (184.94 .. 186.53 .. 188.12) +- stdv  0.81 (  2 points)
  rx.pathological/threads . 5.19.0  . (ci95l..avg..ci95u) = (194.65 .. 206.32 .. 217.99) +- stdv  5.95 (  2 points)
  rx.pathological/threads . 5.23.1  . (ci95l..avg..ci95u) = (216.20 .. 216.22 .. 216.25) +- stdv  0.01 (  2 points)
  rx.pathological/threads . 5.19.1  . (ci95l..avg..ci95u) = (202.30 .. 204.98 .. 207.66) +- stdv  1.37 (  2 points)
  rx.pathological/threads . 5.21.0  . (ci95l..avg..ci95u) = (218.81 .. 219.01 .. 219.21) +- stdv  0.10 (  2 points)
  rx.pathological/threads . 5.17.7  . (ci95l..avg..ci95u) = (208.05 .. 208.94 .. 209.83) +- stdv  0.45 (  2 points)
  rx.pathological/threads . 5.19.5  . (ci95l..avg..ci95u) = (210.27 .. 210.60 .. 210.94) +- stdv  0.17 (  2 points)
  rx.pathological/threads . 5.17.0  . (ci95l..avg..ci95u) = (185.22 .. 185.27 .. 185.32) +- stdv  0.03 (  2 points)
  rx.pathological/threads . 5.17.4  . (ci95l..avg..ci95u) = (190.75 .. 191.75 .. 192.75) +- stdv  0.51 (  2 points)
  rx.pathological/threads . 5.14.4  . (ci95l..avg..ci95u) = (181.98 .. 182.37 .. 182.77) +- stdv  0.88 ( 20 points)
  rx.pathological/threads . 5.13.0  . (ci95l..avg..ci95u) = (184.05 .. 185.22 .. 186.39) +- stdv  0.60 (  2 points)
  rx.pathological/threads . 5.21.3  . (ci95l..avg..ci95u) = (211.13 .. 212.89 .. 214.66) +- stdv  0.90 (  2 points)
  rx.pathological/threads . 5.10.0  . (ci95l..avg..ci95u) = (182.99 .. 183.14 .. 183.30) +- stdv  0.24 ( 10 points)
  rx.pathological/threads . 5.20.0  . (ci95l..avg..ci95u) = (219.62 .. 221.50 .. 223.38) +- stdv  4.18 ( 20 points)
  rx.pathological/threads . 5.16.0  . (ci95l..avg..ci95u) = (190.13 .. 192.80 .. 195.48) +- stdv  5.95 ( 20 points)
  rx.pathological/threads . 5.17.1  . (ci95l..avg..ci95u) = (199.25 .. 202.02 .. 204.79) +- stdv  1.42 (  2 points)
  rx.pathological/threads . 5.13.2  . (ci95l..avg..ci95u) = (181.92 .. 187.16 .. 192.40) +- stdv  2.67 (  2 points)
  rx.pathological/threads . 5.19.2  . (ci95l..avg..ci95u) = (210.19 .. 212.56 .. 214.92) +- stdv  1.21 (  2 points)
  rx.pathological/threads . 5.13.9  . (ci95l..avg..ci95u) = (192.50 .. 193.64 .. 194.78) +- stdv  0.58 (  2 points)
  rx.pathological/threads . 5.15.3  . (ci95l..avg..ci95u) = (196.55 .. 199.13 .. 201.70) +- stdv  1.31 (  2 points)
```

<YOUR BENCHMARK SHOULD BE HERE>

<YOUR BENCHMARK SHOULD BE HERE>

fork
github.com/renormalist/
Benchmark-Perl-Formance

copy
lib/Benchmark/Perl/Formance/Plugin/**Skeleton.pm**

<YOUR BENCHMARK SHOULD BE HERE>

Or ask me

ss5@renormalist.net

Observations

- Micro benchmarks
  - some up, some down
- Macro benchmarks remain stable
- Some medium benchmarks improve

Observations

- What's that regex micro benchmark explosion thing?

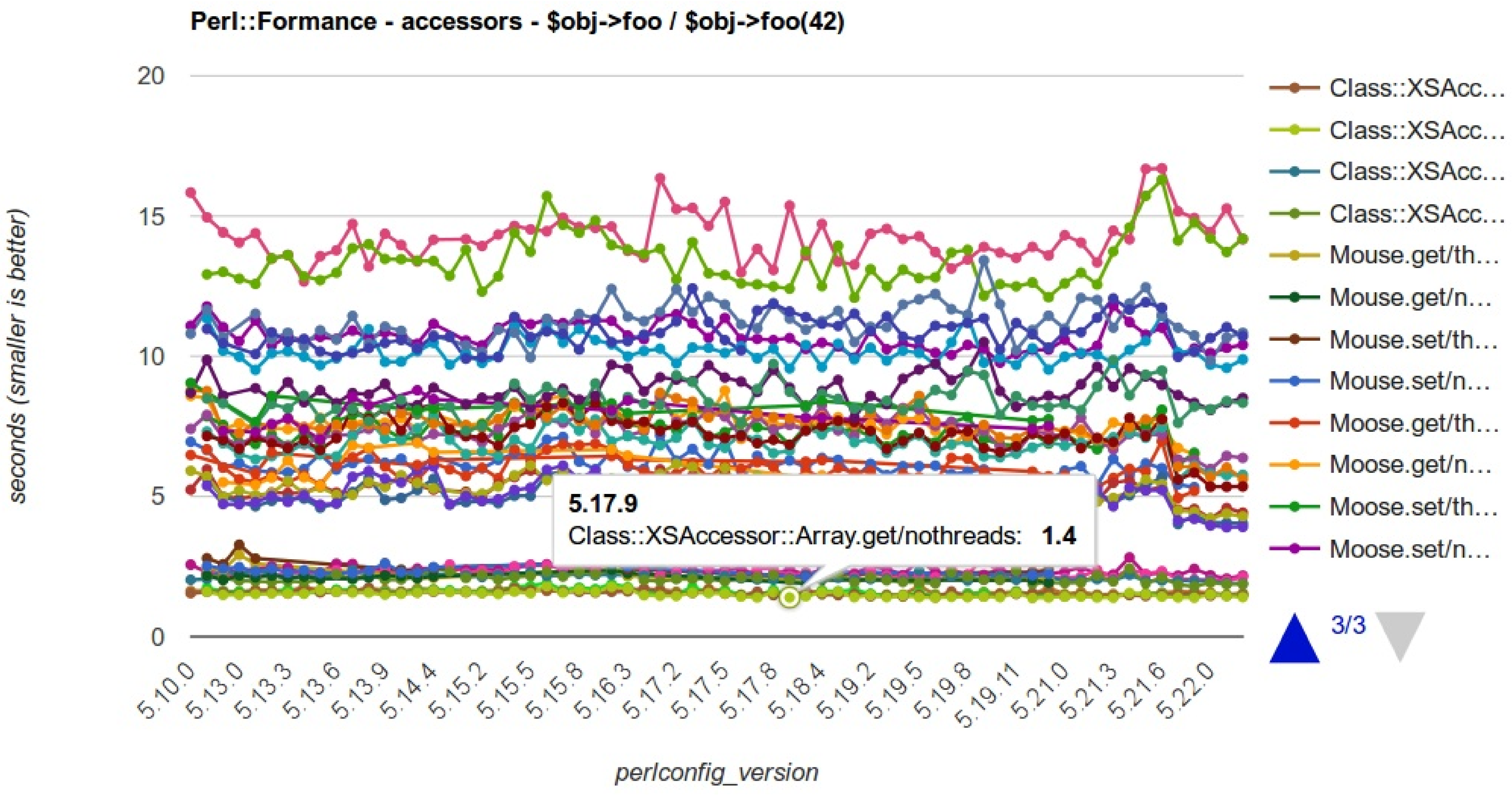
Observations

- Threaded Perl isn't that slow!

Observations

- Who won the accessor speed combat?

Observations



Thanks!

Questions?


DRESDEN PERL MONGERS
use style;