hyperfine Command-line benchmarking

CONCEPTS

> Commands run directly without a shell. Quote each command, and add -S when you need pipes, && or redirects.

setup once
↓ then
prepare before each run
↓ then
command timed
↓ then
conclude after each run
↓ finally
cleanup once
One benchmark, per command

> The first command is the reference; the others are reported relative to it. Default is at least 10 runs, aiming for about 3 seconds.

> Very fast commands (a few ms) are noisy: prefer no shell, and compare many runs.

RUNNING

hyperfine '*<cmd>*'
Benchmark a command

hyperfine '*<cmd1>*' '*<cmd2>*'
Compare commands side by side

hyperfine -r *<n>* '*<cmd>*'
Run exactly n times

hyperfine -m *<n>* '*<cmd>*'
Run at least n times

hyperfine -M *<n>* '*<cmd>*'
Run at most n times

hyperfine -w *<n>* '*<cmd>*'
Do n warmup runs first

hyperfine -n *<name>* '*<cmd>*'
Give a command a display name

hyperfine -i '*<cmd>*'
Ignore a non-zero exit code

hyperfine -i=1,2 '*<cmd>*'
Ignore only listed exit codes

hyperfine --show-output '*<cmd>*'
Show the command's output

hyperfine --output *<where>* '*<cmd>*'
Send output to null, pipe, inherit or file

RUNNING (cont.)

hyperfine --input *<file>* '*<cmd>*'
Feed a file to stdin

PREPARING RUNS

-p '*<cmd>*' / --prepare
Run before every timing run

-C '*<cmd>*' / --conclude
Run after every timing run

-s '*<cmd>*' / --setup
Run once before a benchmark

-c '*<cmd>*' / --cleanup
Run once after a benchmark

--prepare 'make clean' 'make'
Rebuild from scratch each run

--prepare 'sync; echo 3 | sudo tee /proc/sys/vm/drop_caches'
Drop the Linux page cache

--env '*<K=V ...>*'
Set environment variables

> Setup, prepare, conclude and cleanup always run through a shell, even without -S.

PARAMETERS

-P *<var>* *<min>* *<max>* '*<cmd>* {var}'
Scan a numeric range

-P *<var>* *<min>* *<max>* -D *<step>* ...
Set the scan step size

-L *<var>* a,b,c '*<cmd>* {var}'
Try a list of values

-L *<v1>* a,b -L *<v2>* x,y '*<cmd>* {v1} {v2}'
Benchmark all combinations

-P threads 1 8 --env 'OMP_NUM_THREADS={threads}' ...
Use a parameter in the environment

SHELL

-N / --shell=none
Run without a shell (default)

-S
Run through the default shell

--shell *<name>*
Use a specific shell, e.g. bash or zsh

SHELL (cont.)

> With a shell, hyperfine measures an empty shell's startup and subtracts it. Skip the shell for commands under about 5 ms.

OUTPUT AND EXPORT

--export-json *<file>*
Save stats and every sample as JSON

--export-markdown *<file>*
Save a Markdown table

--export-csv *<file>*
Save one CSV row per command

--style *<type>*
auto, basic, full, nocolor, color or none

--metrics *<list>*
Choose metrics, e.g. time_wall_clock:ms

--metrics memory_peak_resident:MiB
Report peak memory in MiB

-h / --help
Show help

-V / --version
Show version

RECIPES

Cold versus warm cache

```
# warm: ignore the first slow runs
hyperfine -S --warmup 3 'grep -R TODO .'
# cold: drop the cache before every run
sudo -v
hyperfine -S --prepare 'sync; echo 3 | sudo tee /proc/sys/vm/drop_caches' 'grep -R TODO .'
```

Compare two tools and keep a report

```
hyperfine -w 3 -n grep -n rg 'grep -r TODO .' 'rg TODO' \
  --export-markdown bench.md --export-json bench.json
```

Compare git branches

```
hyperfine -L branch main,faster \
  --setup 'git switch {branch}' \
  'python main.py'
```

Thread scaling build

```
hyperfine --prepare 'make clean' \
  -P jobs 1 8 'make -j {jobs}'
```