Python Profiling Starter Kit

Christian Hudon

http://christianhudon.name/

Why?

A Quick Story...

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE STORE FORUMS SUBSCRIBE SIGN IN

LOAD IT UP —

Hacker reduces *GTA Online* load times by roughly 70 percent

Homebrewed DLL solves inefficient parsing of in-game shop files.

KYLE ORLAND - 3/1/2021, 1:32 PM

Source: https://arstechnica.com/gaming/2021/03/hacker-reduces-gta-online-load-times-by-over-70-percent/

Let's do the math!

120k players every day

Source: https://playercounter.com/grand-theft-auto-5/

game out for **7.5 years**

Source: https://en.wikipedia.org/wiki/Grand_Theft_Auto_Online

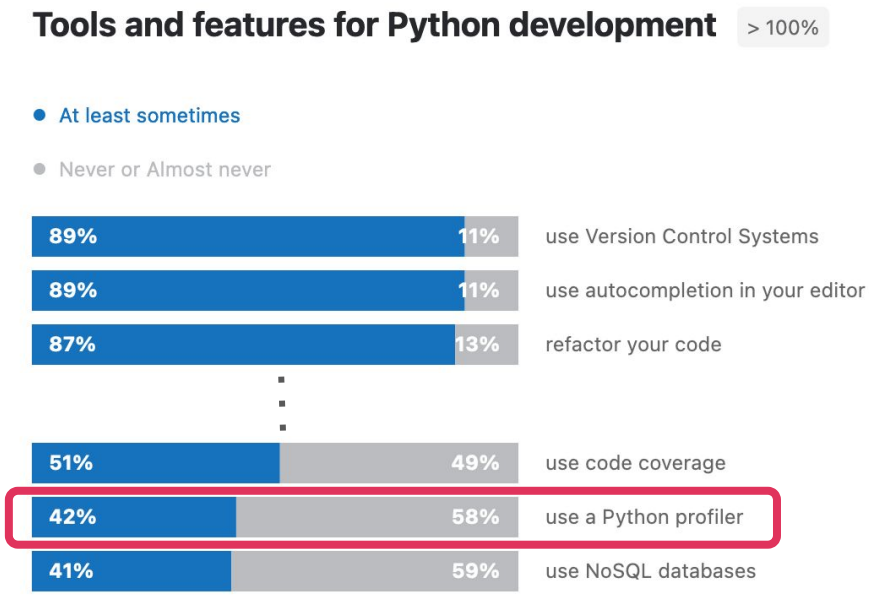
5 minutes average loading time

Source: https://www.reddit.com/r/gtaonline/comments/ht4i56/your_average_online_loading_time/

1 140 625

total days waiting for the game to load!

Profiling? What's That?



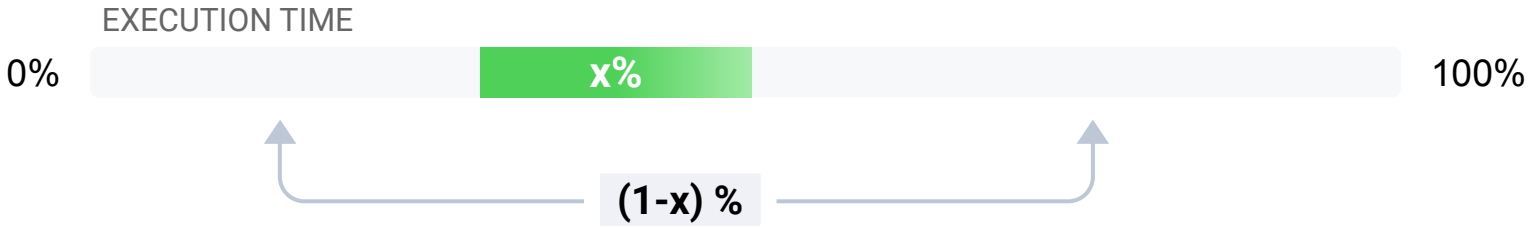
Source: https://www.jetbrains.com/lp/python-developers-survey-2020/

How?

How to Proceed: Build a Model

- “Minimum viable model”
- Pay attention to the bottlenecks!

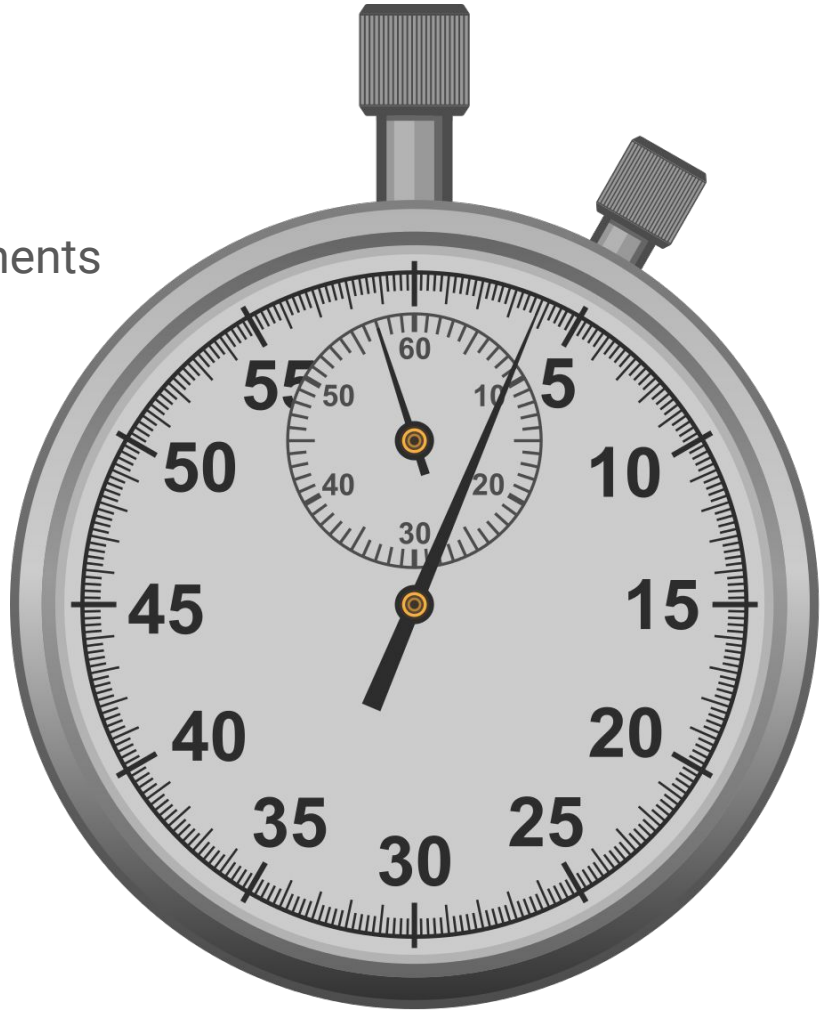
Where to Focus: Amdahl's Law*



* See Wikipedia article

How to Proceed: Be Systematic

1. Set yourself up for repeatable measurements
2. Shorter profiling runtime is better
3. Build a model!
4. Don't guess, **measure**!
   (aka Use the Tools, Luke!)
5. Change the code (1 change is easier)
6. Measure the impact
7. Repeat until satisfied / out of time



Use the Tools, Luke!

The Landscape

Classical vs. sampling vs. tracing profilers

Overview of 4 Profilers

Pyinstrument

What

- Sampling profiler
- Wall clock time & full stack traces
- Hooks to profile Django / Flask requests

PyPI package: pyinstrument

Homepage:
https://github.com/joerick/pyinstrument

How

```
$ pip install pyinstrument
```

```
$ pyinstrument slow_program.py
```

Or, for an HTML report, saved to a file

```
$ pyinstrument --html -outfile
out.html slow_program.py
```

Py-Spy

What

- Sampling profiler
- Can profile already running Python programs, in prod!
- Works entirely outside the process
- But… may require special permissions (sudo, etc.)

PyPI package: py-spy

Homepage: https://github.com/benfred/py-spy

How

```
$ pip install py-spy
```

```
$ py-spy record -o out.svg --
python3 slow_program.py
```

Or, for a SpeedScope-style flamegraph report, viewable via https://speedscope.app

```
$ py-spy record -o out.sscope -f
speedscope -- python3
slow_program.py
```

Also `--pid` option, `top` & `dump` commands.

Scalene

What

- Sampling profiler
- Profiles both CPU and memory usage
- Profiles at the line level

PyPI package: scalene

Homepage:
https://github.com/plasma-umass/scalene

How

```
$ pip install scalene
```

```
$ scalene slow_program.py
```

Or, for an HTML report, saved to a file

```
$ scalene --html -outfile out.html
slow_program.py
```

VizTracer

What

- Tracing profiler
- When you want to focus on rare events, not averages!
- But… more complex tool

PyPI package: viztracer

Homepage: https://github.com/gaogaotiantian/viztracer

How

```
$ pip install viztracer
```

```
$ viztracer -o out.html slow_program.py
```

Use Chrome to view HTML report.

Many options to control what is captured. See the doc at https://viztracer.readthedocs.io.

Making Code Faster

Some Ideas...

- Look at the big picture
- Focus on the bottlenecks
- Do less work / avoid repeated work
- Focus on overhead / batch work
- Understand the libraries you are using (e.g immutable vs. mutable)
- Vectorize / delegate the work to fast third-party code
- Parallelize
- Use more efficient data structures
- Use Cython / Numba

Two Crazy Ideas

1. Profile All Functionality at Least Once!

2. Profile in Your CI!

Thanks!

Summary

Focus where it pays / build a basic model / setup for repeatable, quick iterations / measure / iterate

Pyinstrument: sampling, full stack trace, web request hooks

Py-Spy: sampling, on already running code in prod

Scalene: sampling, line-level, memory usage

VizTracer: tracing (for rare events)

Bonus: profile in your CI!

Further Readings

Optimization in depth (memset for AMD64):
https://msrc-blog.microsoft.com/2021/01/11/building-faster-amd64-memset-routines/

Tracing the GIL with Perf & VizTracer:
https://www.maartenbreddels.com/perf/jupyter/python/tracing/gil/2021/01/14/Tracing-the-Python-GIL.html

Sampling vs. Tracing:
https://danluu.com/perf-tracing/

Data Center Tracing:
https://www.youtube.com/watch?v=QBu2Ae8-8LM

http://christianhudon.name/talks/#mp-python-profiling