Overview

Bookmark this page

Introduction

In this module we cover strategies, tools and pitfalls around understanding the performance of a python code base. We mention or demonstrate analysis tools built into specific operating systems or available via python itself. We then we look at techniques for improving performance, including coding techniques, alternate Python interpreters, and interaction with C.

Keep in mind that the material presented herein is not the last word on this topic, rather, as is often the case with the material in the program, it is merely the first. There are related topics covered elsewhere in the curriculum, such as concurrency and asynchronous programming; both topics are about improving the performance or responsiveness of systems. Also keep in mind that related topics are covered more deeply in other areas of the curriculum, so that for instance it might be worthwhile referencing the material on map/filter, comprehensions, lambdas and other functional programming constructs from their respective lessons while or before continuing with this lesson.

Learning Objectives

Upon successful completion of this lesson, you will be able to:

- use profiling strategies to identify bottlenecks in Python code.
- use timing strategies to assess code constructs.
- use PyPy to run simple Python scripts to improve their performance.
- refactor Python code to use Cython for performance improvement.

New Words, Concepts, and Tools

- Profiling
- cProfile
- time (GNU Time)
- timeit
- Great Circle
- memoization
- PyPy
- Cython
- cdef

Prerequisites

Beyond the content covered on this and the prior class, there are no specific prerequisites for this lesson.

Before you Start

Required Reading

- Profiling & Timing

  https://docs.python.org/3.6/library/debug.html
  https://docs.python.org/3.6/library/profile.html
  https://docs.python.org/3.6/library/timeit.html

- PyPy

  http://pypy.org/

- Cython

  http://cython.org

Optional Reading

- Profiling
  https://pypi.python.org/pypi/memory_profiler
  https://www.jetbrains.com/help/pycharm/profiler.html
- https://wiki.python.org/moin/PythonSpeed/PerformanceTips

- https://www.codementor.io/satwikkansal/python-practices-for-efficient-code-performance-memory-and-usability-aze6oiq65
- https://nyu-cds.github.io/python-performance-tuning/

- https://pypy.org/performance.html

Suggested Workflow

- Explore the "Before you Start" readings and video
- Work through the lesson content pages
- Watch the required videos
- Do the practice activity
- Submit your assignment

At the End of the Lesson

What material in this lesson do you still feel unclear about? Please use the discussion forum for this lesson to describe any concepts or ideas you've struggled with this week. Your instructor will address concepts that seem problematic for several students. You may also want to address issues raised by other students or ask for their help.