Threading and Multiprocessing

Advanced Python Programming

—

Valerio Velardo - The Sound of AI

Why bothering with threading / multiprocessing?

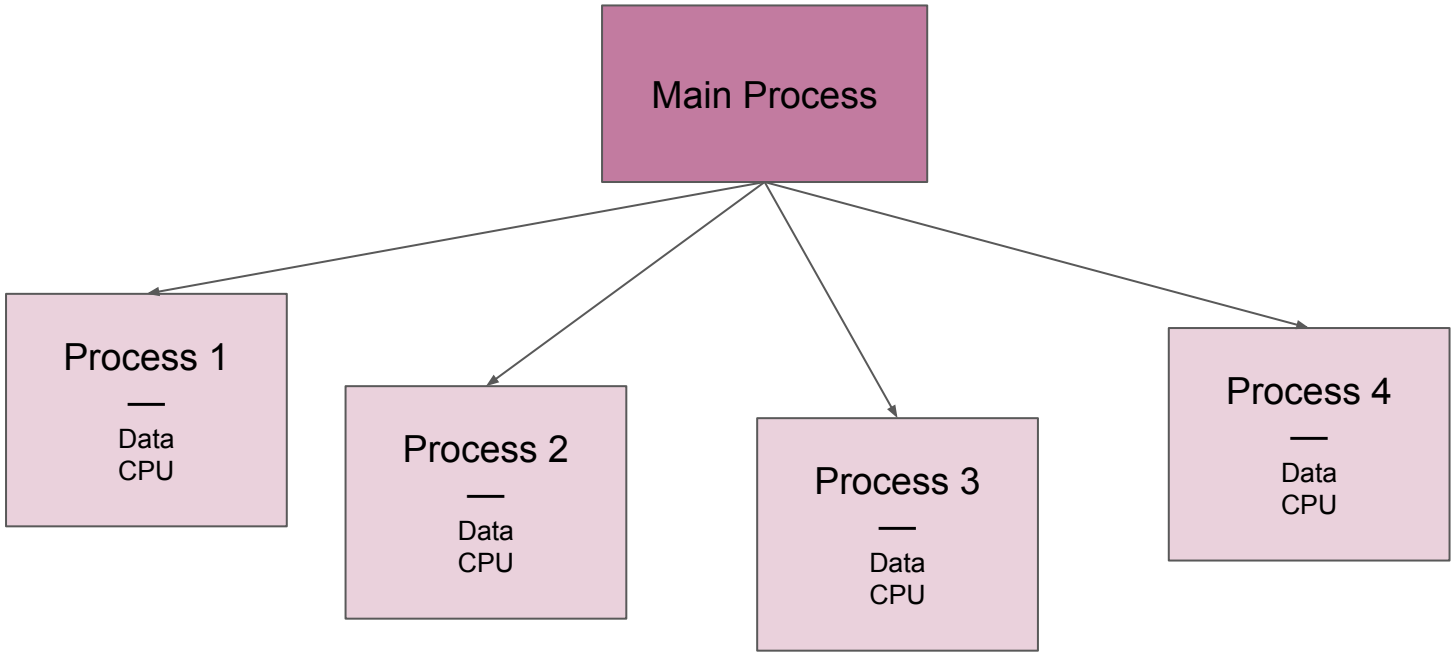
Why bothering with threading / multiprocessing?

To run our programmes faster

What’s a process?

- Execution environment for a programme
- A programme can be run in *parallel* in multiple processes
- Each process has different data and computer resources

Multiprocessing

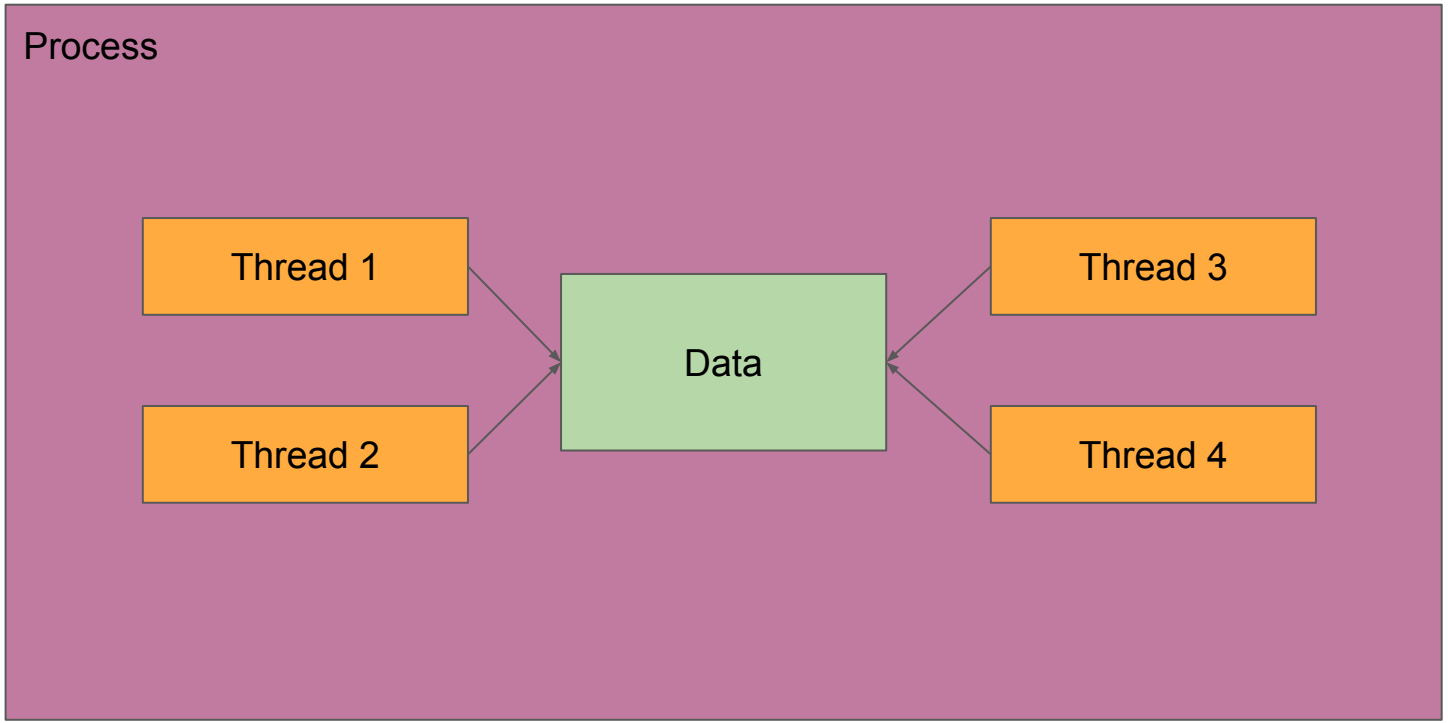


What’s a thread?

- Unit of execution in a process
- A process can have multiple threads (*concurrency*)
- Threads in a process share data and CPU

Multithreading

Multithreading

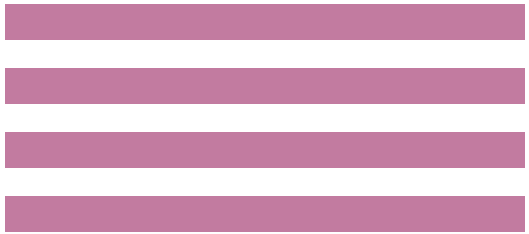


Global Interpreter Lock (GIL)

- Process lock
- GIL prevents multiple threads from executing simultaneously
- Multiple threads can run concurrently
- GIL enforces threads are executed serially

Multiprocessing vs threads

Multiprocessing



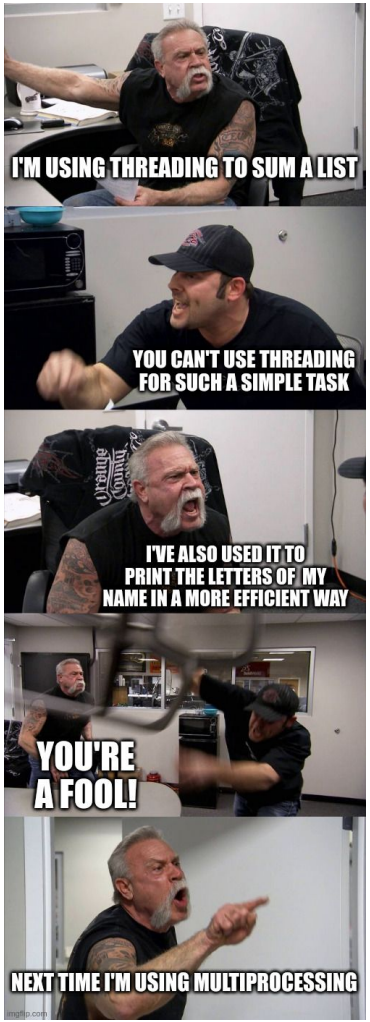
Threading

Multiprocessing pros and cons

- +
  - Significant performance boost
  - Make the most out of your multi-core CPU
- -
  - Significant overhead to manage processes
  - Entire memory copied in each subprocess

Threads pros and cons

- +
  - Run tasks concurrently
  - Don’t be blocked by slow tasks (e.g., I/O)
- -
  - Increase complexity of the programme
  - Overhead to manage threads


I'M USING THREADING TO SUM A LIST
YOU CAN'T USE THREADING FOR SUCH A SIMPLE TASK
I'VE ALSO USED IT TO PRINT THE LETTERS OF MY NAME IN A MORE EFFICIENT WAY
YOU'RE A FOOL!
NEXT TIME I'M USING MULTIPROCESSING
imgflip.com

Use cases

Multiprocessing

- CPU bound tasks
  - Crunching data
  - Data preprocessing
  - …

Threading

- I/O
- Network access
- GUI
- Audio programming