XSEDE

Extreme Science and Engineering Discovery Environment

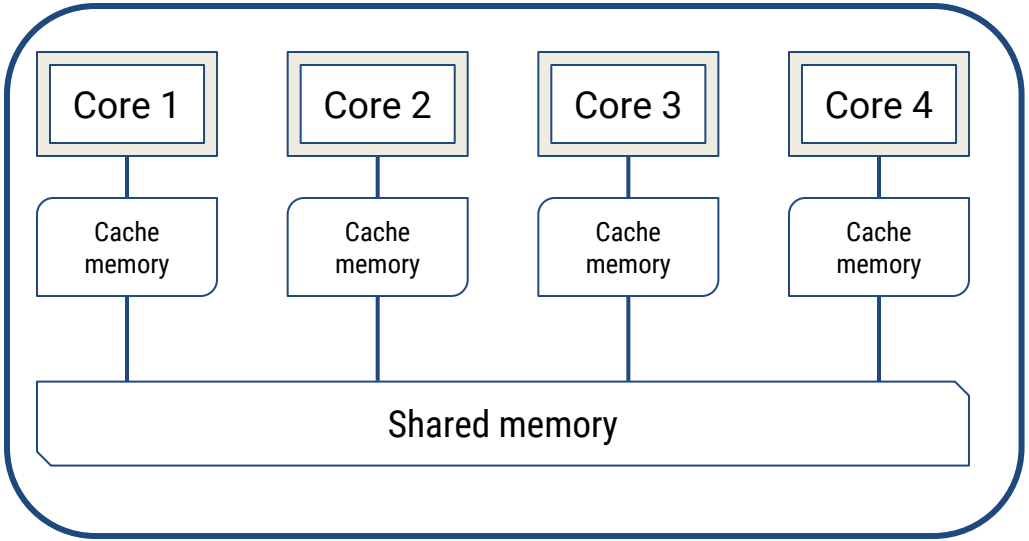
High performance computing in Python

Presented by:

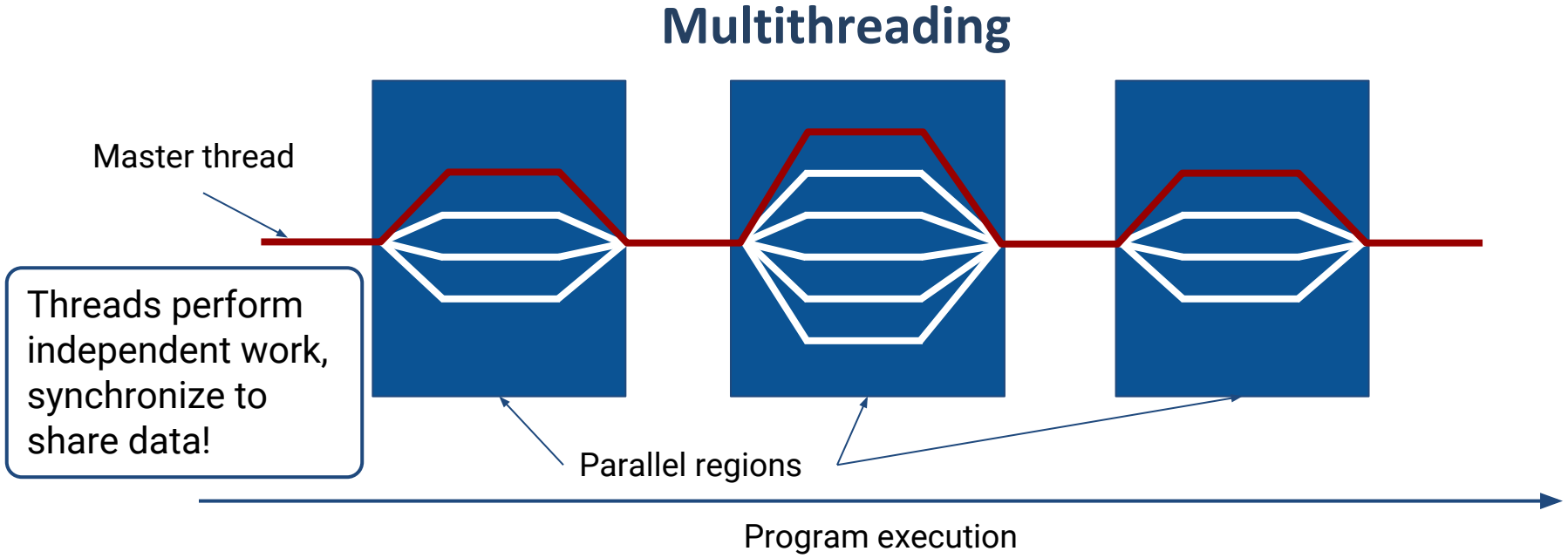
- Steve Lantz - steve.lantz@cornell.edu
- Roberto Camacho - rcamachobarranco@utep.edu

Another peek at how microprocessors work

Multi-core processors

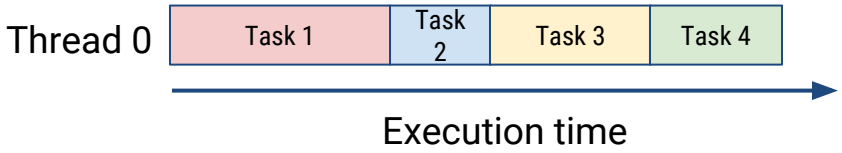


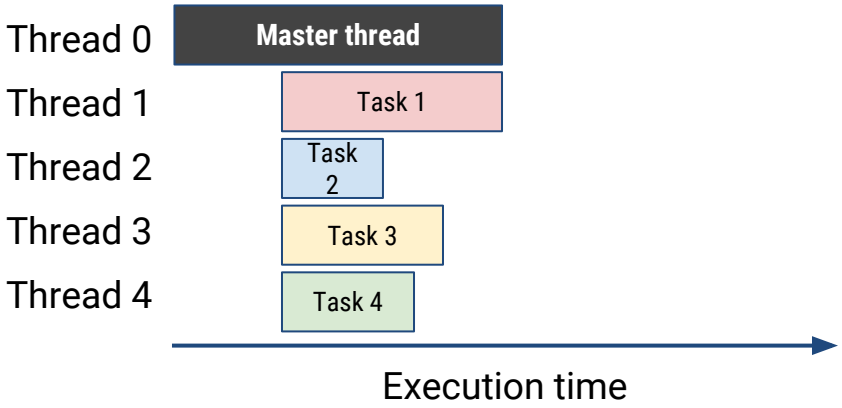
Another peek at how microprocessors work



Another peek at how microprocessors work

Single thread



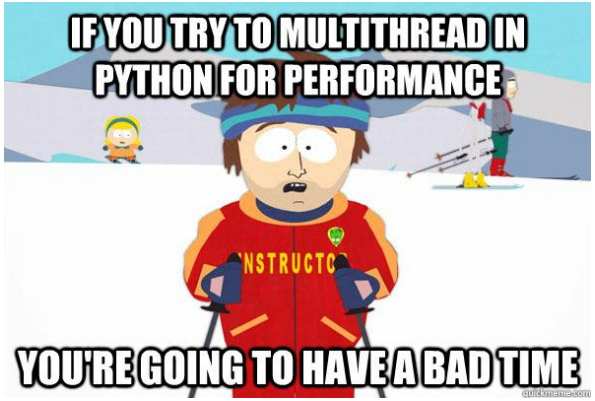


Multiple threads

Great! How do we exploit this in Python?

Global Interpreter Lock in Python → no multi-threading

- Alternatives:
  - Python multiprocessing module
  - Multi-processing via the OS
  - External application that calls your Python code (e.g., Spark or Hadoop)
  - Code that your Python code calls (e.g. you could have your Python code call a C function that does the expensive multi-threaded stuff).



Exploiting parallelism in Monte Carlo applications

- Remember, we are trying to model an unknown distribution!
- Works best with a HUGE number of random samples
- Luckily, it's "embarrassingly" parallel
  - Every random sample is independent
- Do the work faster with N parallel tasks
  - Each does 1/N of the total samples
  - Each must use a different seed
  - Merge results at the end

Exploiting parallelism in Monte Carlo applications