H1 Understanding performant python

What are the elements of a computer architecture?

In short: (1) memory unit (2) the computing unit and (3) connections in between.

Computing unit

CPU or GPU. Bits go in, bits come out.
Measure of speed: how many instructions can it do in 1 cycle (IPC), and how many cycles can it do in 1 second (clock speed).
IPC and clock speed don't improve much more, because physical limitations are reached. But other tricks can help such as multi-cores, multi-threading, and asynchronous compute.

GIL: global interpreter lock. Python uses this, and it means that a python process can only perform 1 instruction at the time, no matter how many cores there are available.

So lets say you want to multi-thread your code, e.g. summing up numbers 1 to 20 with 4 thread, such that each thread sums up 5 numbers. E.g thread 1 sums up 1 to 5, thread 2 sums up 6 to 10 etc.

However, the python interpreter only wants to communicate with 1 core at the time.

This is sad, BUT libraries like
`numpy` or `multiprocessing` can overcome this.

Multiprocessing: means spinning up multiple copies of the python interpreter,
Numpy: is using C and not python, so does not depend on the GIL

Memory unit

RAM and hard drive. stores bits.
Measure of speed: bits it can read and write per second. But also how long it takes to find the data (latency).

Sub units:
- spinning hard drive: long term storage, stays when computer is shut down. Slow read/write speed. Large capacity (±10 terabyte)
- solid state hard drive: same as spinning, but a little bit faster, less capacity (±1 terabyte)
- RAM: temporal storage, very fast. Capacity in range of ±64 gigabyte.
- L1/L2 cache: very very fast, capacity in range of megabytes.

Communication layer

Often called a “bus”. For example, the frontside bus is between RAM and L1/L2 cache.
Bus width: how much data per transfer
Bus frequency: how many transfers per second

NOTE: GPU’s have less efficient busses, which means moving data from and to GPU’s is slower then CPU.

NAS: network attached storage, which is attached to the memory unit that uses the network.

Vectorization

To send multiple values to the CPU at once, as it can process things in parallel.

What are some hurdles to making python code performant?

1. python has garbage collection: this causes memory allocation to be suboptimal
2. dynamic typing/code is not compiled: so pre-optimization is hard.
3. as mentioned before, the GIL.

How to become highly performant?

The main point of this chapter is : test your code.