Performance

This is only the introduction

Required reading: Chapter 5:

Introduction and 5.1, plus the first paragraphs of sections 5.2 – 5.9. Look over 5.15

C/assembly

How Fast?

Godot TBM owned by The Boring Company

Gary, The Boring Company’s snail mascot

Usain Bolt

Smart For2

Tesla Model S P100DL

Jetliners

Falcon 9 / Long March to orbit

Earth Escape velocity

Photons

How Fast?

Godot TBM owned by The Boring Company (300’/week?)

Gary, The Boring Company’s snail mascot (14x faster)

Usain Bolt (27 MPH / 44 km/hr)

Smart For2 (96 MPH)

Tesla Model S P100DL (155 MPH)

Jetliners [.87 to .93 Mach] (Mach 1 is 760 MPH / 1200 km/hr)

Falcon 9 / Long March to orbit (Mach 20)

Earth Escape velocity (Mach 33)

Photons at c

How Quick? (0–60mph / 0–100)

- Godot?
- Gary the snail?
- Usain Bolt?
- Smart For2?
- Tesla Model S P100DL?
- Boeing 777?
- Falcon 9?
- Photon?

How Quick? (0-60mph / 0-100)

- Godot – N/A
- Gary – N/A
- Usain Bolt – (N/A)
- Smart For2 – 11sec
- Tesla Model S P100DL – 2.5 sec
- Boeing 777 (unladen) 6 sec, most jetliners can make it to 150 MPH in 30 seconds or less.
- Falcon 9 in ~ 10-11 seconds
- Photon to c in 0 seconds

What do those measures mean?

- Categories of performance
  - “too slow”
  - “OK”
  - “ridiculously fast”
- Sufficient Performance
  - Meets need
  - Doesn’t meet the need

Measures

- We can’t talk about performance without talking about measures
- **If you don’t measure, you don’t know**
- **If you don’t profile, you can’t tell where**

Knuth

- Programmers waste enormous amounts of time thinking about, or worrying about, the speed of noncritical parts of their programs, and these attempts at efficiency actually have a strong negative impact when debugging and maintenance are considered. We should forget about small efficiencies, say about 97% of the time: **premature optimization is the root of all evil**. Yet we should not pass up our opportunities in that critical 3%.

How fast is determined by timing your code

- Different languages have access to different clock functions
  - 1 second resolution
  - Sub-second resolution
- Use stats when you clock is too coarse-grained.
- Test code should include timing data as an output.

Bottlenecks are found by profiling

- Profile optimized code, not debug code
  - Do not check performance on debug code compiled with -g
  - Only check performance on optimized code compiled with -O
  - To enable profiling compile with -pg then use gprof
- gprof is the finest technology from 1980's – cell phones back then were called "brick phones"
- Newer tools have newer profiling capabilities that are platform dependent.

Systems need performance checking, too!

- top command under Unix
- task manager and perfmon under windows

Which is faster? Why? Does it matter?

- Sum = 0;
- For (i=0; i<limit; i++)Sum = sum + a[i];

- Sum = 0;
- Ptr = a;
- For(i=0; i<limit;i++)Sum = sum + *ptr++;

Which is faster? Why? Does it matter?

```
Sum = 0;
for (i=0; i<rows; i++)
{
    for (k=0; k<cols; k++)
    {
        Sum += a[i][k];
    }
}
```

```
Sum = 0;
for (i=0; i<rows; i++)
{
    for (k=0; k<cols; k++)
    {
        Sum += a[k][i];
    }
}
```

C is row-major order
Fortran is column-major order

Java uses Iliffe vectors instead

Performance At Scale

- AWS Whitepaper from 2014
  - https://www.enterpriseai.news/2014/11/14/rare-peek-massive-scale-aws/?fbclid=IwAR2swFdPs5PlfAg8rS3uce95y5VRSkr81TXE4kfJjOtTaIHApSPUwuZ3SL0
- Availability zones in a region are always under 2ms apart, usually under 1 ms apart (186 miles) – the write time of an SSD device.
- Data centers in an availability zone are under ¼ ms apart (46 miles).
- A data center is over 50,000 servers in 2,000 racks using 25-30 MW of power
- Custom network stack on custom hardware

What Do Those Numbers Mean?

- New York to LA is 74ms latency on a real network
- Ground to geosynchronous orbit is 120ms one way at light speed
- Starlink is 340 miles up (1.83ms one way, 41ms actual average ping)

- High performance computer systems and networks are about latency as well as throughput, so we measure distances in terms of light-speed delays to get a minimum delay knowing that real networks take longer

- Inside a data center, software latency is far worse than light-time lag:
  - Milliseconds to get from the application to the network card
  - Microseconds to get through the card
  - Nanoseconds to transit the fiber

But does it matter? Python->C speedup

The program does matrix multiplication

- Switching to C: 47x speedup
- Parallel loops on many cores: 7x speedup
- Optimal memory layout for cache: 20x speedup
- Using SIMD floating point extensions: 7x speedup

Total speedup: 62,000 faster!

Un-asked questions:

- Did anyone care?
- Was it worth the effort?