How to Write Fast Numerical Code

Spring 2019

Lecture: Benchmarking

Instructor: Markus Püschel

TA: Tyler Smith, Gagandeep Singh, Alen Stojanov

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Overview

- **Measuring performance & benchmarking**
- **References:**
  - *Section 3.2 in: Chellappa, Franchetti, Püschel:* How To Write Fast Numerical Code: A Small Introduction, GTTSE 2008
  - *Hoefler and Belli:* Scientific Benchmarking of Parallel Computing Systems, Supercomputing 2015
  - *Whaley and Castaldo:* Achieving accurate and context-sensitive timing for code optimization, Software: Practice and Experience 2008

Benchmarking

- *First:* **Validate/test your code!**
- **Measure runtime (in [s] or [cycles]) for a set of relevant input sizes**
  - seconds: actual runtime
  - cycles: abstracts from CPU frequency
- **Usually: Compute and show performance (in [flop/s] or [flop/cycle])**
- *Careful:* **Better performance ≠ better runtime (why?)**
  - Op count could differ
  - Never show in one plot performance of two algorithms with substantially different op count

How to Measure Runtime?

- **C clock()**
  - process specific, low resolution, very portable
- **gettimeofday**
  - measures wall clock time, higher resolution, somewhat portable
- **Performance counter (e.g., TSC on Intel)**
  - measures cycles (i.e., also wall clock time), highest resolution, not portable
- **Careful:**
  - measure only what you want to measure
  - ensure proper machine state
    (e.g., cold or warm cache = input data is or is not in cache)
  - measure enough repetitions
  - check how reproducible; if not reproducible: fix it
- ***Getting proper measurements is not easy at all!***

Problems with Timing

- **Too few iterations: inaccurate non-reproducible timing**
- **Too many iterations: system events interfere**
- **Machine is under load: produces side effects**
- **Multiple timings performed on the same machine**
- **Bad data alignment of input/output vectors:**
  - align to multiples of cache line (on Core: address is divisible by 64)
  - sometimes aligning to page boundaries (address divisible by 4096) makes sense
- **Machine was not rebooted for a long time: state of operating system causes problems**
- **Computation is input data dependent: choose representative input data**
- **Computation is inplace and data grows until an exception is triggered (computation is done with NaNs)**
- **You work on a computer that has dynamic frequency scaling (e.g., turbo boost)**
- ***Always check whether timings make sense, are reproducible***

Benchmarks in Writing

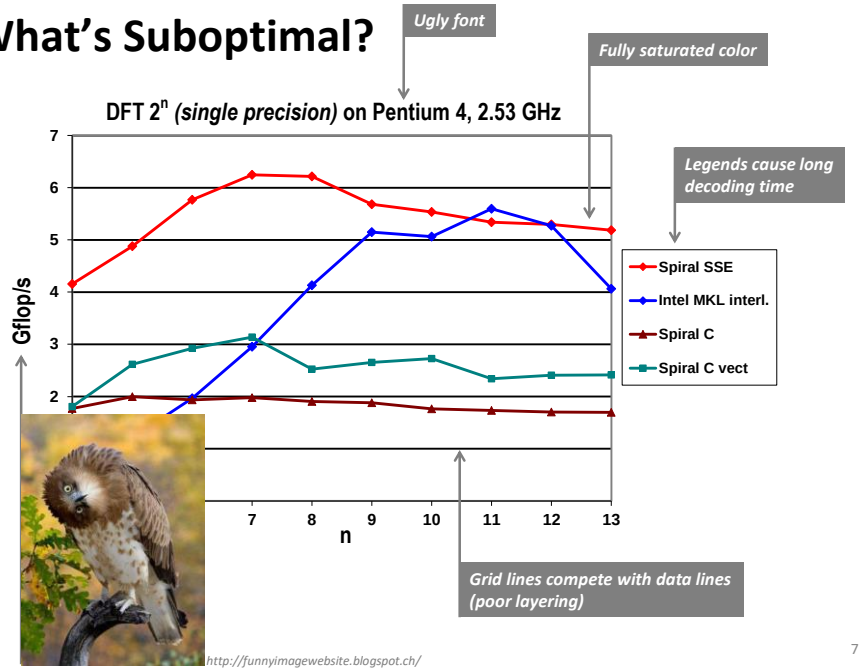
- **Specify experimental setup**
  - platform
  - compiler and version
  - compiler flags used
- **Plot: Very readable**
  - Title, x-label, y-label should be there
  - Fonts large enough
  - Enough contrast (e.g., no yellow on white please)
  - Proper number format

    No: *13.254687;* ***yes:*** *13.25*

    No: *2.0345e-05 s;* ***yes:*** *20.3 µs*

    No: *100000 B;* ***maybe:*** *100,000 B;* ***yes:*** *100 KB*

What's Suboptimal?



http://funnyimagewebsite.blogspot.ch/