Small Guide to Software Benchmarking

(update planned)

Markus Püschel
Carnegie Mellon University
www.ece.cmu.edu/~pueschel

Guide to Benchmarking: How?

- **First: Verify your code!**

- **Measure runtime, compare against the best available code**
  - compile other code correctly (as good as possible)
  - use same timing method
  - be fair
  - always sanity check: compare to published results etc.

- **Measure performance: flops (number floating point ops/second), compare to peak performance**
  - needs peak performance, which can be difficult
  - get instruction count statically (cost analysis) or dynamically (tool that counts, or replace ops by counters through macros)
  - Careful: Different algorithms may have different op count, i.e., best flops is not always best runtime

Guide to benchmarking: 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 Pentiums)**
  - measures cycles (i.e., also wall clock time), highest resolution, not portable
- **Careful:**
  - measure only what you want to measure (maybe subtract overhead)
  - proper machine state (e.g., cold/warm cache)
  - measure enough repetitions
  - check how reproducible; if not reproducible: fix it

Guide to Benchmarking: How to present results (in writing)?

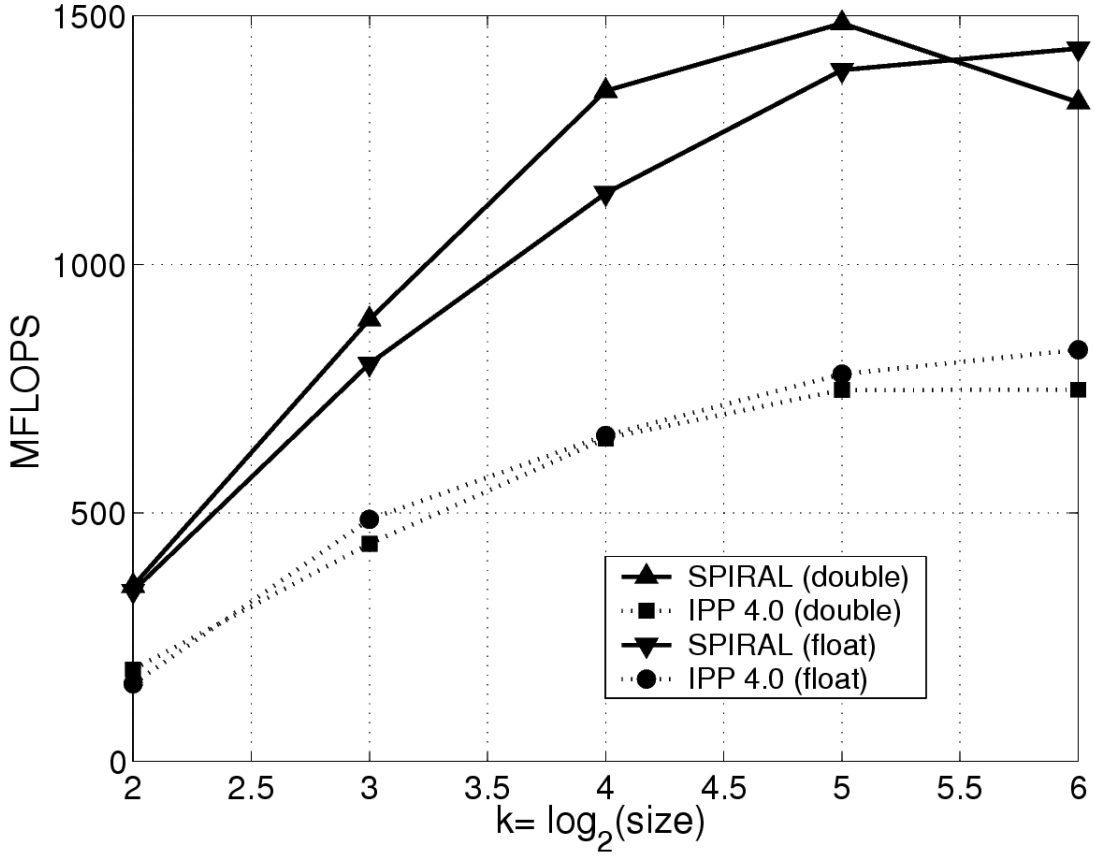
- **Specify machine**
  - processor type, frequency
  - relevant caches and their sizes
  - operating system
- **Specify compilation**
  - compiler incl. version
  - flags
- **Explain timing method**
- **Plot**
  - Has to be very readable (colors if possible, thick lines, fonts, etc.)
  - Choose proper type of plot: message as visible as possible

Guide to Benchmarking: How to present results (talking)?

- Briefly explain the experiment
- Explain x- and y-axis
- Say, e.g., “higher is better” if appropriate
- If many lines, maybe explain one as example
- Extract a message in the end

Example

Performance of code for the discrete cosine transform (DCT):



Platform:
P4 (HT), 3GHz,
8KB L1, 512KB L2,
WinXP

Compiler:
icc 8.0

Compiler flags:
/QxKW /G7 /O3

- **Spiral-generated code is a factor of 2 faster**
- **reaches up to 50% of the peak performance**