Notes on

Lab 2: OpenMP + NUMA

CSE 6230: HPC Tools & Apps
Fall 2014 — September 9

Based in part on the **LLNL tutorial** @ https://computing.llnl.gov/tutorials/openMP/
See also the textbook, Chapters 6—8

Georgia Tech College of Computing
Computational Science and Engineering

hpcgarage

Part 0: **Code reviews + last shot at redemption!**
(Share ideas — if you already hit Lab 1, bonus points for you!)

Part 1: **Cilk Plus vs. OpenMP — fight!**
(spawn → omp task, parfor → omp for. Easy! Or is it?)

Part 2: **Science experiment: NUMA in action!**
(Next!)

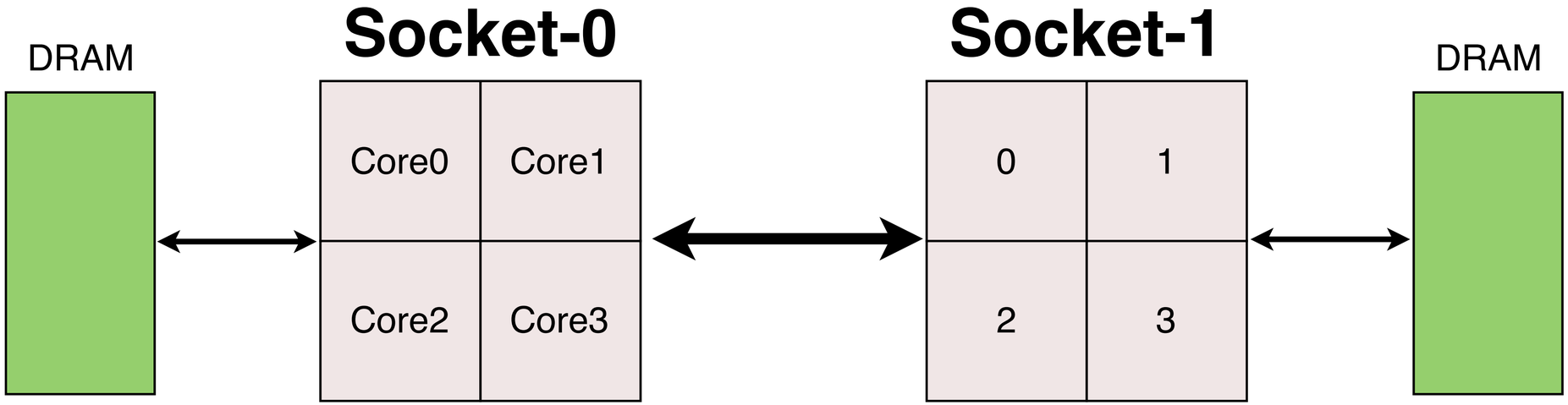
```
--------------------------------------------------------------
CPU type:       Intel Core Westmere processor
**************************************************************
Hardware Thread Topology
**************************************************************
Sockets:        2
Cores per socket:       6
Threads per core:       2
--------------------------------------------------------------
HWThread        Thread          Core            Socket
0               0               0               0
1               0               0               1
2               0               8               0
3               0               8               1
4               0               2               0
5               0               2               1
6               0               10              0
7               0               10              1
8               0               1               0
9               0               1               1
10              0               9               0
11              0               9               1
12              1               0               0
13              1               0               1
14              1               8               0
15              1               8               1
16              1               2               0
17              1               2               1
18              1               10              0
19              1               10              1
20              1               1               0
21              1               1               1
22              1               9               0
23              1               9               1
--------------------------------------------------------------
Socket 0: ( 0 12 8 20 4 16 2 14 10 22 6 18 )
Socket 1: ( 1 13 9 21 5 17 3 15 11 23 7 19 )
--------------------------------------------------------------
```

```
**************************************************************
NUMA domains: 2
--------------------------------------------------------------
Domain 0:
Processors:  0 2 4 6 8 10 12 14 16 18 20 22
Memory: 10988.6 MB free of total 12277.8 MB
--------------------------------------------------------------
Domain 1:
Processors:  1 3 5 7 9 11 13 15 17 19 21 23
Memory: 10986.1 MB free of total 12288 MB
--------------------------------------------------------------

**************************************************************
Graphical:
**************************************************************
Socket 0:
+-------------------------------------------------------------------------+
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| |  0  12 | |  8  20 | |  4  16 | |  2  14 | | 10  22 | |  6  18 | |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| |  32kB  | |  32kB  | |  32kB  | |  32kB  | |  32kB  | |  32kB  | |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| |  256kB | |  256kB | |  256kB | |  256kB | |  256kB | |  256kB | |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| +---------------------------------------------------------------+ |
| |                              12MB                             | |
| +---------------------------------------------------------------+ |
+-------------------------------------------------------------------------+
Socket 1:
+-------------------------------------------------------------------------+
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| |  1  13 | |  9  21 | |  5  17 | |  3  15 | | 11  23 | |  7  19 | |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
| |  32kB  | |  32kB  | |  32kB  | |  32kB  | |  32kB  | |  32kB  | |
| +--------+ +--------+ +--------+ +--------+ +--------+ +--------+ |
```

```
jinx9 $  /nethome/rvuduc3/local/jinx/likwid/2.2.1-ICC/bin/likwid-topology -g
```

Performance tuning tip:
Exploit non-uniform memory access (NUMA)



Example: Two quad-core CPUs with logically shared but physically distributed memory

Exploiting NUMA: Linux “first-touch” policy

```
a = /* … allocate buffer … */;

for (i = 0; i < n; ++i) {
  a[i] = /* … initial value … */ ;
}
```

```
#pragma omp parallel for …
for (i = 0; i < n; ++i) {
  a[i] += foo (i);
}
```

Exploiting NUMA: Linux “first-touch” policy

```
a = /* … allocate buffer … */;
#pragma omp parallel for …
for (i = 0; i < n; ++i) {
  a[i] = /* … initial value … */ ;
}
```

```
#pragma omp parallel for …
for (i = 0; i < n; ++i) {
  a[i] += foo (i);
}
```

Exploiting NUMA: Linux “first-touch” policy

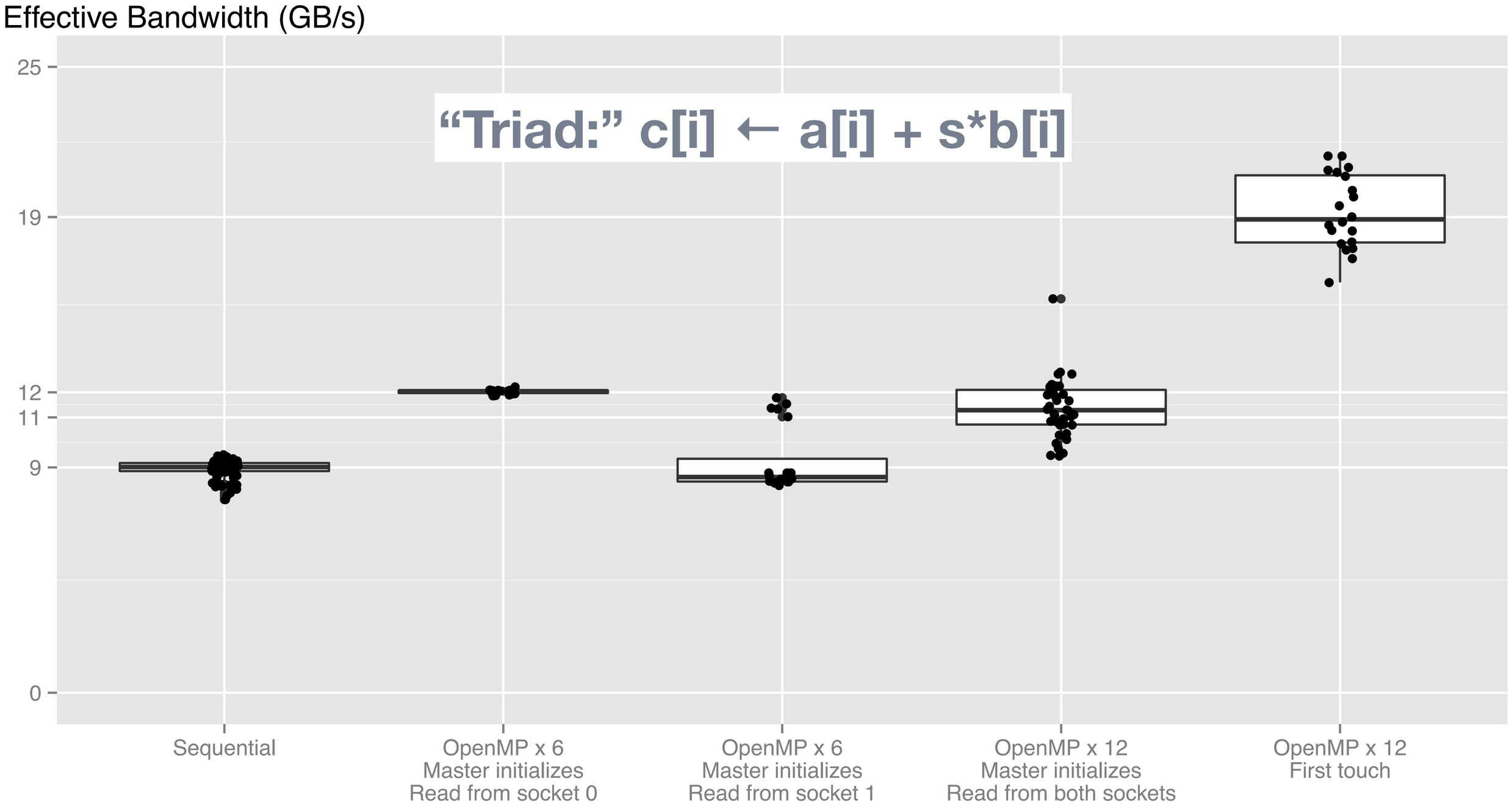
```
a = /* … allocate buffer … */;
#pragma omp parallel for … schedule(static)
for (i = 0; i < n; ++i) {
  a[i] = /* … initial value … */ ;
}
```

```
#pragma omp parallel for … schedule(static)
for (i = 0; i < n; ++i) {
  a[i] += foo (i);
}
```

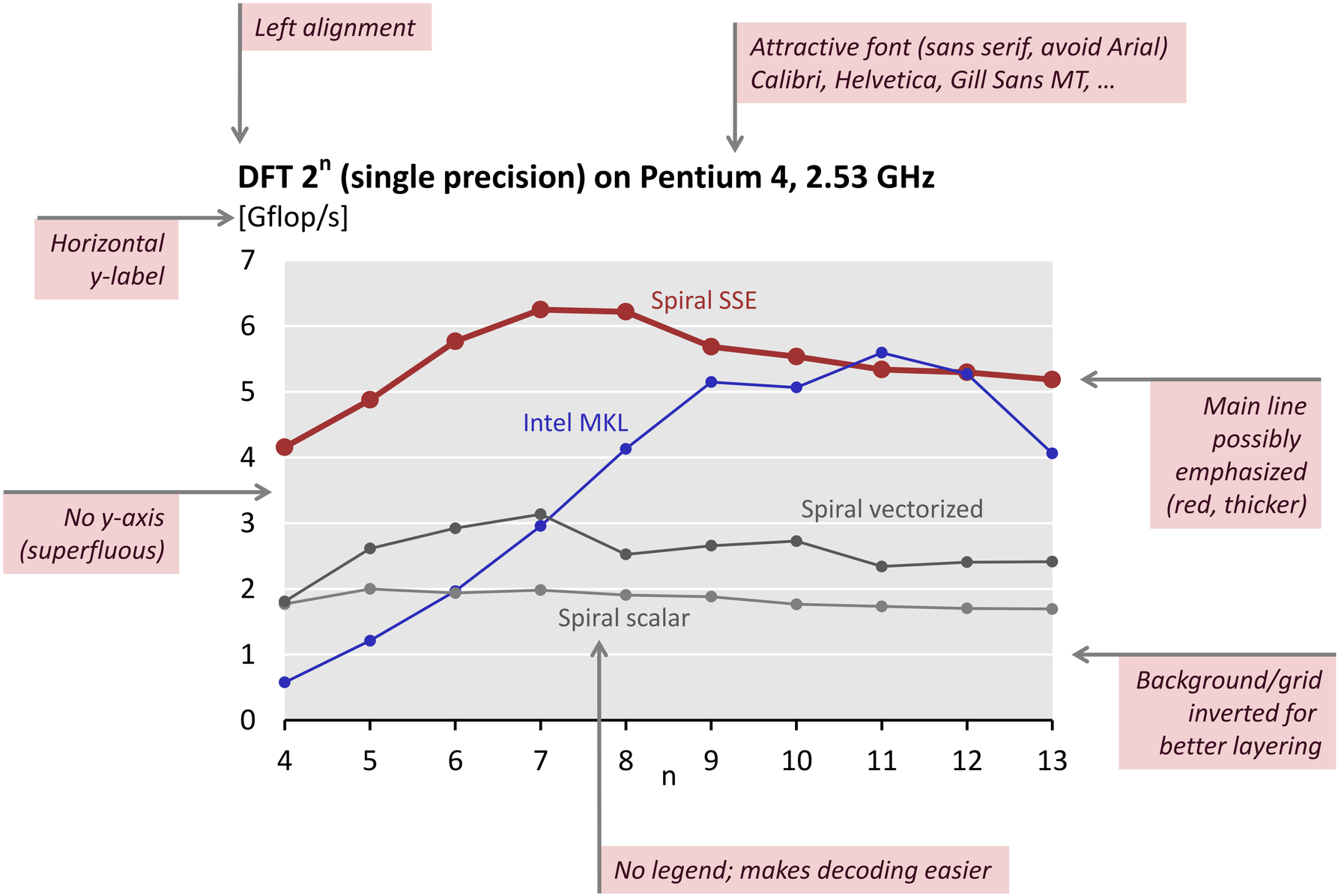
Thread binding

- Key environment variables
  - **OMP_NUM_THREADS**: Number of OpenMP threads
  - **GOMP_CPU_AFFINITY**: Specify thread-to-core binding

- Consider: 2-socket x 6-core system, main thread initializes data and ‘6’ OpenMP threads operate
  - env OMP_NUM_THREADS=6 GOMP_CPU_AFFINITY=“0 2 4 6 … 22” ./run-program …
    (shorthand: GOMP_CPU_AFFINITY=“0-22:2”)
  - env OMP_NUM_THREADS=6 GOMP_CPU_AFFINITY=“1 3 5 7 … 23” ./run-program …
    (shorthand: GOMP_CPU_AFFINITY=“1-23:2”)


Effective Bandwidth (GB/s)
25
19
12
11
9
0
“Triad:” c[i] ← a[i] + s*b[i]
Sequential
OpenMP x 6
Master initializes
Read from socket 0
OpenMP x 6
Master initializes
Read from socket 1
OpenMP x 12
Master initializes
Read from both sockets
OpenMP x 12
First touch

Plotting tips



http://www.inf.ethz.ch/personal/markusp/teaching/263-2300-ETH-spring14/slides/05-bench-compiler-limitations.pdf