HPC Café
A quick introduction to computer architecture and the proper use of resources

November 12, 2019
HPC Services, RRZE
The Stored Program Computer

- Improvements for **relevant** software
- **Technical** opportunities
- **Economical** and **marketing** concerns

```c
for (int j=0; j<size; j++) {
    sum = sum + V[j];
}
```

```
401d08:   f3 0f 58 04 82         addss xmm0,[rdx + rax * 4]
401d0d:   48 83 c0 01            add    rax,1
401d11:   39 c7                  cmp edi,eax
401d13:   77 f3                  ja     401d08
```

**Strategies**
- Increase clock speed
- Parallelism
- Specialization

**Execution and memory**
Major problem: How to get the instructions and data into the CPU

Memory and everything beyond it is slow

Solution: Caches!
Multicore nodes: Cores galore

Chip (up to 64 cores)

Node (2 chips)

~ 8 billion transistors on 500 mm²

© Intel
Topology of Erlangen systems: Emmy + Meggie

Node topology
- 2 sockets
- 10 cores / socket
- 2 HW threads / core

Memory Hierarchy
- 32kB L1 D-Cache
- 256kB L2 Cache
- 25MB shared L3
- 1 memory domain per socket

Core frequency: 2.2 GHz
Turbo max all cores emmy: 2.6 GHz
Turbo max all cores meggie: 2.4 GHz

Saturated Memory BW emmy
Load: 46 GB/s
Copy: 38 GB/s
Triad: 40 GB/s

Saturated Memory BW meggie
Load: 58 GB/s
Copy: 50 GB/s
Triad: 51 GB/s
A HPC System consists of many Cabinets!
Relevant resources: cores and caches

Core

- L2 Cache
- L1 Cache
- FPU
- FPU
- ALU
- LSU

Chip (up to 64 cores)

Questions

- Does every core do the same amount of work?
- Does every core take the same amount of time to do its work?
- Is the code limited by pure computation or by data transfers to the caches?
- Is the code actually doing floating-point work or something else?
Relevant resources: memory interface

Questions

• How well is the memory interface utilized?

• Is the data access “friendly” to the architecture? Does the code “jump around” in memory a lot?

• Is the data traffic balanced across the memory domains?
Scaling types on the chip

saturating type: probably memory bound

scalable type: bound by execution or cache access
Questions

- Is communication among nodes a bottleneck for my code?
- What is the pattern of communication? Is there a “hot spot”?
- Does my code do a lot of I/O to the disks?
Good practices

- **Be considerate.** Clusters are valuable shared resources that have been paid by the taxpayer.
- **Check your jobs regularly**
  - Are the results OK?
  - Does the job actually use the allocated nodes in the intended way? Does it run with the expected performance?
  - Memory consumption? Disk quota exceeded?

**Why should I care?**
- Minimum total cost (hardware, power, infrastructure, people) for one node-hour on modern HPC cluster: 0.5 €
- 1 node for 1 year: 4000 €
- 300 nodes for 1 day: 3600 €

This is money. Money that you burn cannot be used by others.
Good practices

- Use the **appropriate** amount of parallelism
  - Most workloads are not highly scalable
  - Best to run scaling experiments to figure out the “sweet spot”
  - Parallel efficiency < 50% is generally a waste of resources!

\[
\varepsilon(n) = \frac{P(n)}{n \cdot P(1)}
\]
Questions!? 

HPC@RRZE 

https://hpc.fau.de