NHR@FAU HPC Café
October 14, 2025

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Assessing the efficiency of your jobs with ClusterCockpit

Jan Eitzinger, Georg Hager

Erlangen National High Performance Computing Center (NHR@FAU)

Here’s a hint

```
Dear User,

Many of your jobs are not properly utilizing the allocated GPU resources (e.g.
"https://monitoring.nhr.fau.de/monitoring/job/12345678", [...]

Please optimize your code to ensure efficient usage. You can check it yourself when logging
into our monitoring system ClusterCockpit at https://monitoring.nhr.fau.de/.

Please ensure that you only allocate nodes with GPUs if your code can actually make use of
them. Otherwise, these resources remain blocked for other users, and valuable project compute
time for you and your colleagues is wasted.

If you need help or have any questions, please do not hesitate to contact us.

Best regards,
NHR@FAU Support.
```

Identifying problems: Typical patterns

Reasons for bad performance

- Too many/too few nodes allocated
  - Resorces are under-/oversubscribed
- Non-usage of allocated GPU
  - Same issue, basically

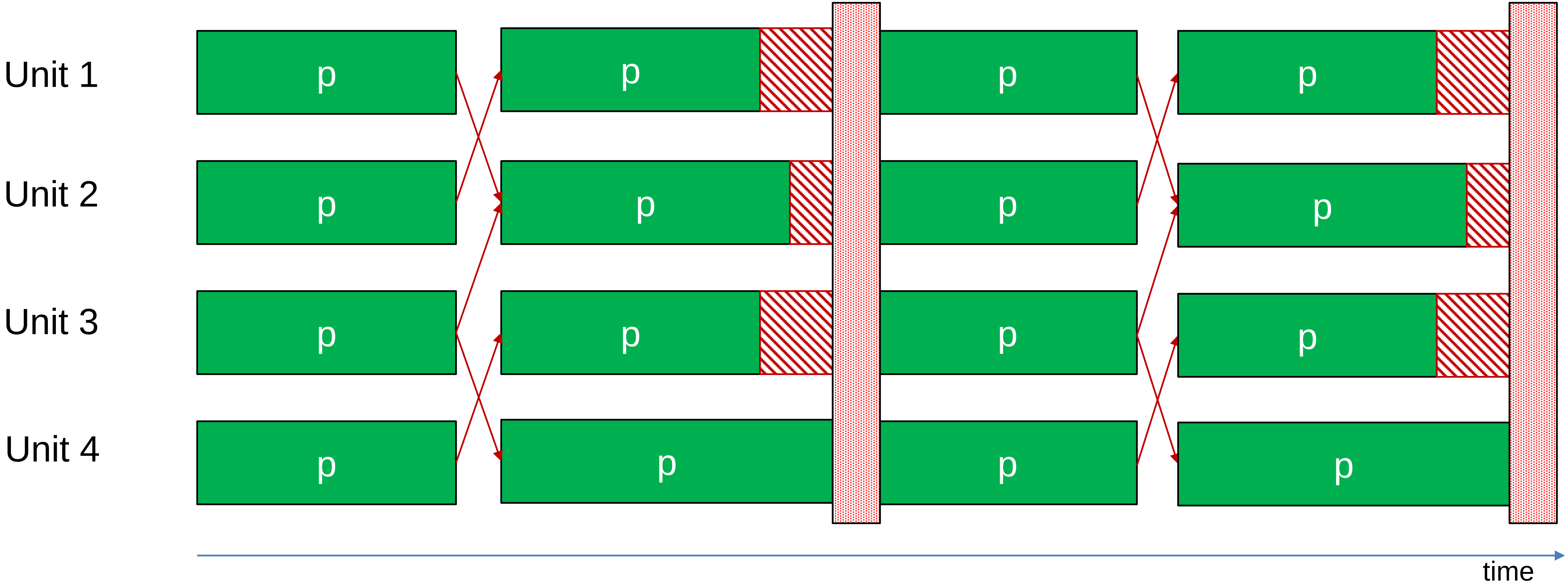
“Operational issues”
(easy to fix)

- Low GPU utilization
- Bad scaling (“death by communication”)
- Bad scaling (load imbalance)
- Dominating I/O time
- “Slow code”

“Workflow/code issues”
(may require work)

High-level view of parallel code execution

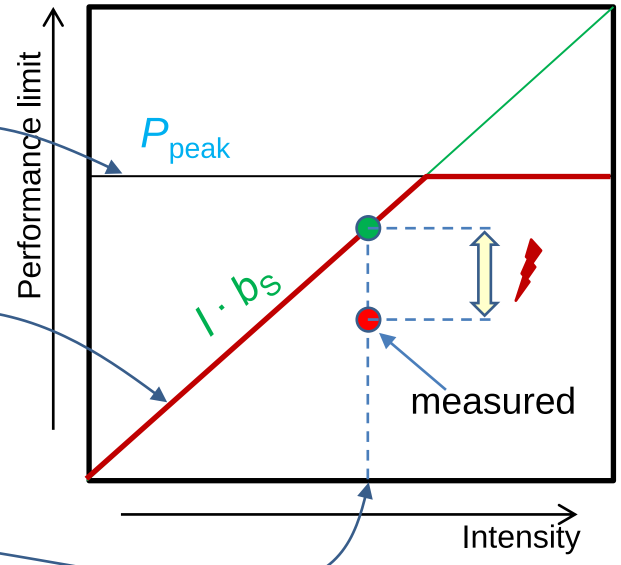
Communication, synchronization, work imbalance, I/O



Diagnosing “code issues” with rooflines

- Roofline diagram in quantifies machine (node) bottlenecks and code property
- Machine bottlenecks:
  - Peak computational performance P_{peak} e.g., 5.5 Tflop/s
  - Peak memory bandwidth b_S e.g., 320 Gbyte/s
- Code property: computational intensity I How much work (flops) does the code do for every byte transferred from memory?
  - E.g., Sparse solvers $I \approx 0.16 \text{ flop/byte}$, dense matrix-matrix mult $I \gg 20 \text{ flop/byte}$

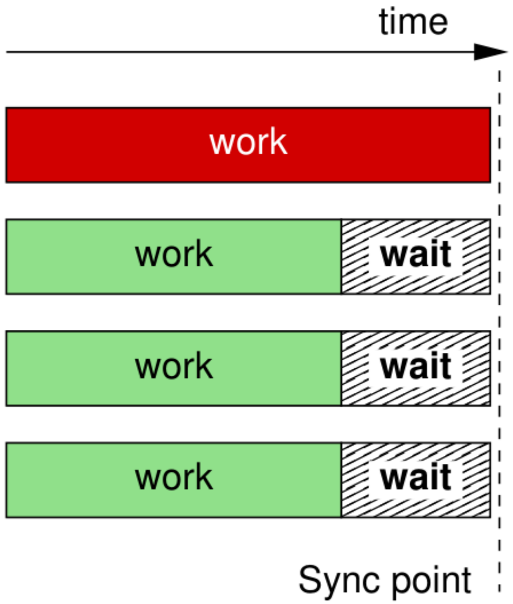
$$P_{limit} = \min\left(P_{peak}, I \times b_S\right)$$

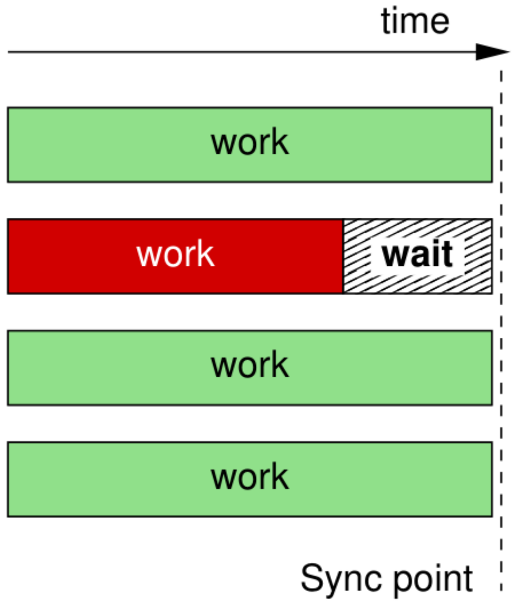


Reasons for “away from roofline limit”

- Low vectorization ratio
  Low ratio of vectorized (SIMD) vs. scalar instructions; not necessarily bad
  - Some codes just cannot be vectorized
  - If hardware utilization is still good, you might not care
  - If SIMD pays off, a factor of up to 8x (DP) might be achievable

- Load imbalance (actually, execution time imbalance)
  Should usually be fixed; however, memory-bound code is more forgiving towards load imbalance (why?)
  Caveat: Two extreme cases!





Reasons for “away from roofline limit”

- High IB package rate
  - IB latency is in the low-µs range; hundreds of millions of IB packages per second are thus near the limit
  - Remedy: Communicate less ☺, aggregation
  - Probably you are just using too many nodes/processes

- High NFS rate

  Some codes write to NFS-mounted volumes frequently; a “fat” server can take up to 500 MB/sec

- Fine-grained, high-frequency I/O

  Rapid-fire I/O requests can overload the metadata servers and severely slow down the shared file system for all users