Scalable visualization of Nsight Systems traces with Paraver

Marc Clascà Ramírez
marc.clasca@bsc.es

October 2025

POP 3 Webinars

Contents of the webinar

1. Our analysis methodology and strategy
2. How nsys2prv works
3. Analysis with Paraver
    a. Relevant metrics for accelerated workloads
    b. Efficiency model
    c. Following the lead of inefficiencies with applied examples
4. Useful links and resources

Tools overview

Extrae

- System level parallel performance analysis
- Timestamped events, configurable semantics
- CUDA support improving in progress
- Requires MPI for distributed memory applications

Paraver

- Configurable visualizations via DSL
- Suitable for large number of resources

NVIDIA Nsight Systems

- Comprehensive workload-level performance
- System level information: different runtimes and hardware metrics
- Typical behaviors to study: synchronization, parallelization, data movement
- Trace visualization integrated, usable up to ~8 processes

NVIDIA Nsight Compute

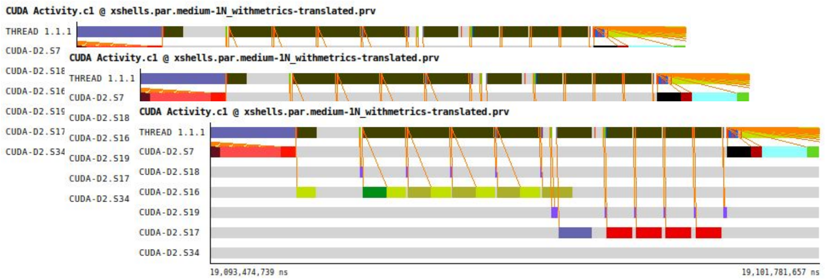
- Detailed CUDA kernel performance
- Isolated kernel execution information: requires replaying
- Typical behaviors to study: GPU utilization, kernel implementation, memory access

How we understand performance analysis

1. Navigating through **scales**

dynamic range allows to add up knowledge from different scales of time, resources, and data - in very large scale runs

2. Comparison and **quantification** of differences



across different traces (*how does a tuning mechanism affect my execution?*), or within the same trace (*how does the microstructure of my application change during time? or across processes?*)

Enabling "large-scale" GPU analysis

- Large scale also means **big "range" of scales**
- Large scale also means different **scale dimensions**

Time

- Macroscopic visualization and aggregation of metrics
- Microscopic runtime behavior
- All in the same timeline
- Very long runs or trace chops

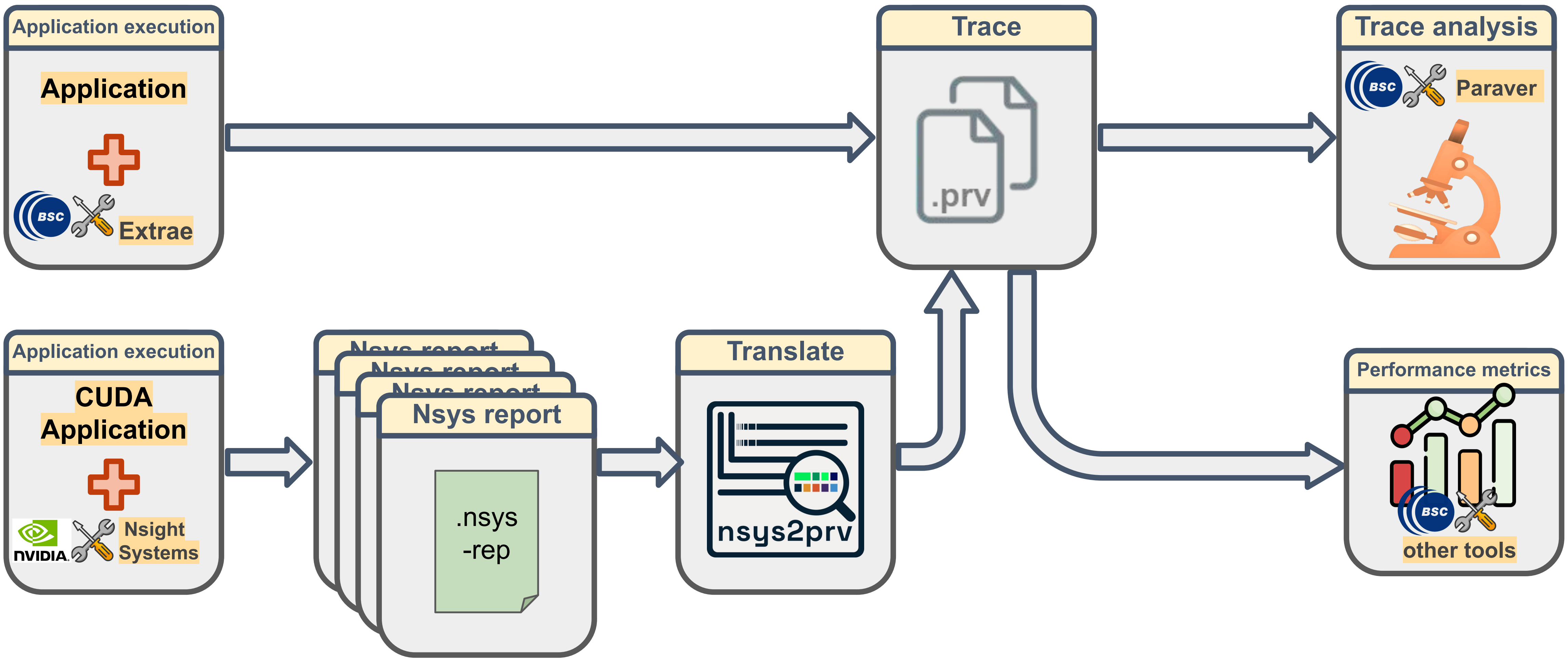
Resources

- Merging multiple reports from multi-node executions, only limited by final trace size.
- Filter and select which objects do you want to see during analysis.
- 1 GPU -> 100s

Data

- Performance information can be combined,
- aggregated,
- filtered,
- operated with…
  - different arithmetic and semantic functions

What we propose



What does nsys2prv currently support?

 Translate performance data acquired by Nsight Systems into Paraver **timestamped records**.

- CUDA API calls
- Kernels and memory copies (and related parameters)
- CUDA Graphs (node & graph level), instantiation and execution
- NCCL kernel execution and payload data (reduction operation, root rank, transfer size)
- GPU hardware counters
- NVTX regions
- OpenACC and MPI runtime calls…
- Operating System library calls
- POSIX pthread calls

 Merge multiple *.nsys-rep* reports, coming from a **multi-node execution**, into a single trace.

 And we provide all **predefined configuration files** for Paraver within the package to display all metrics described in the article and in this presentation

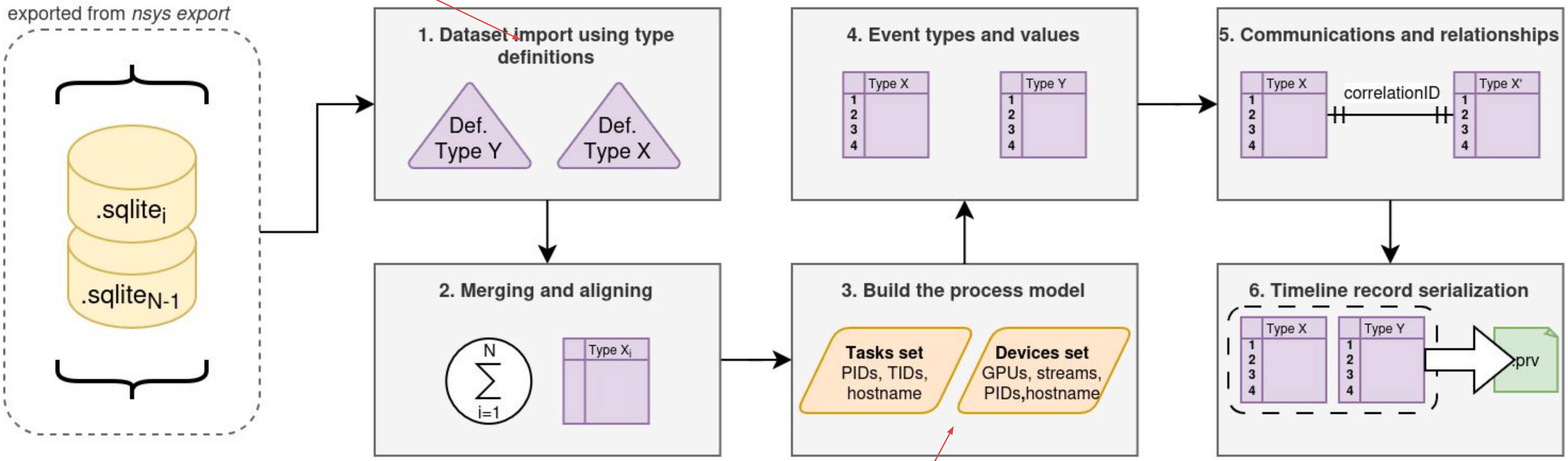
Installing the translator

```
$> python -m venv analysis-venv
$> source analysis-venv/bin/activate
$> python -m pip install nsys2prv
```

20 BSC YEARS

How does it work?

Expandable
to other info!



Leverages Paraver
trace format

How do we translate a trace?

```
$> source analysis-venv/bin/activate
$> module load nsight-systems/2025.3
$> nsys2prv -t nvtx_pushpop_trace,cuda_api_trace,gpu_metrics \
     -m ./llm_0.nsys-rep ./llm_1.nsys-rep ./llm_2.nsys-rep … llm_translated
```

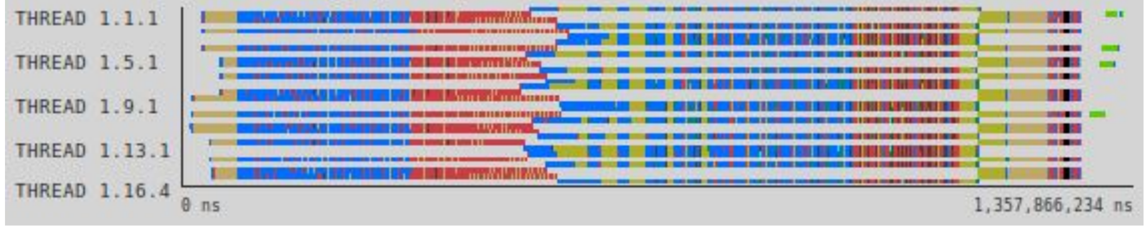
Multi-report flag

Source reports

Information to be translated

Basic visualizations

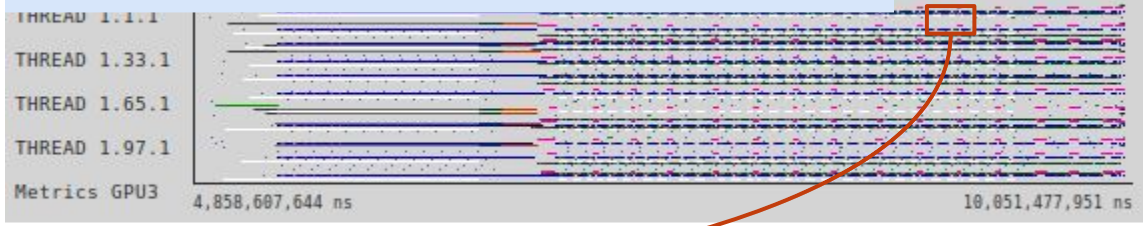
CUDA API calls



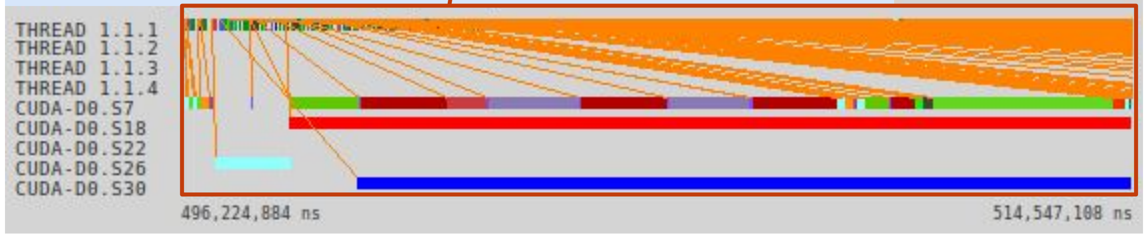
CUDA kernels

MPI calls

CUDA API calls + kernels @ 128 GPUs

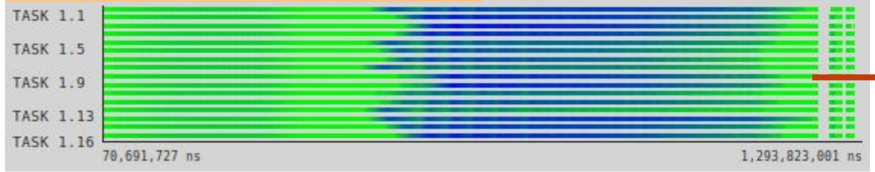


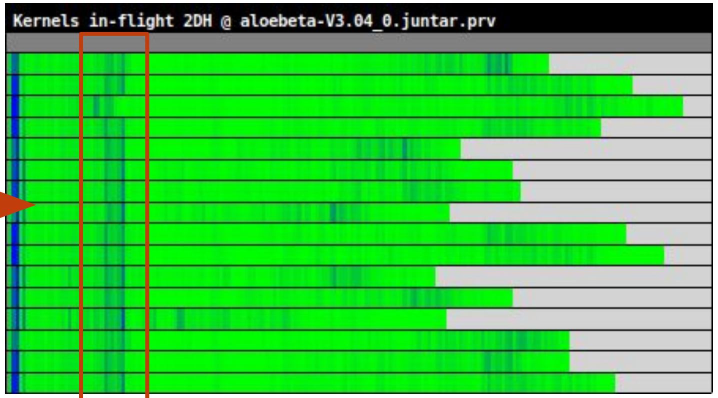
CUDA API calls + kernels, launch lines @ 1 process zoom

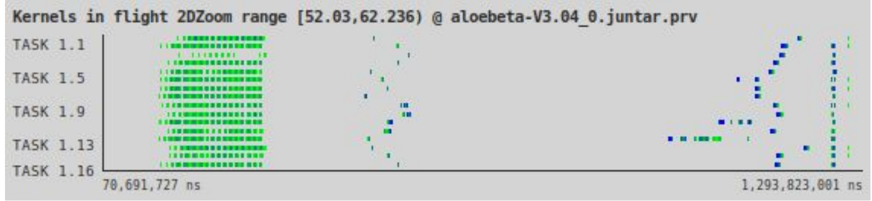


Relevant metrics

Kernels in-flight

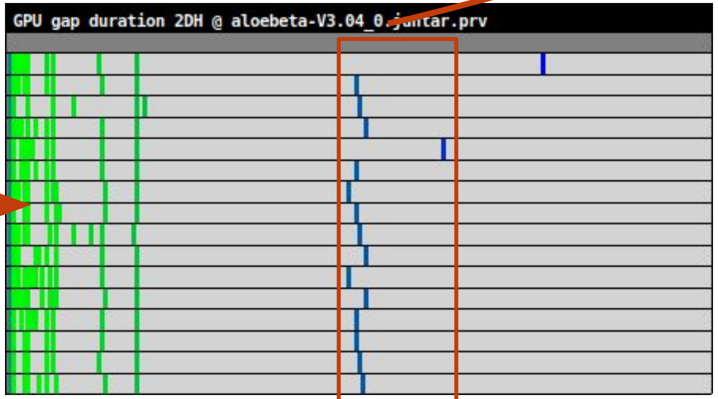


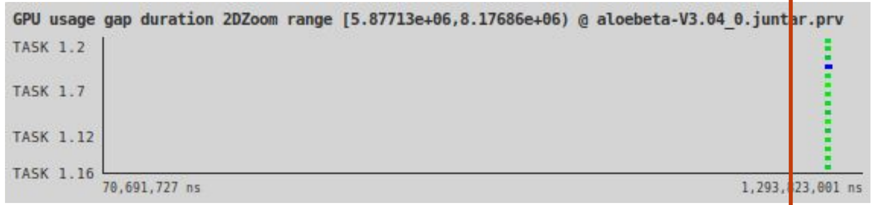




2D histogram: paint frequency (in time spent)of each value of metric

2D Zoom range: spot metric patterns in the timeline

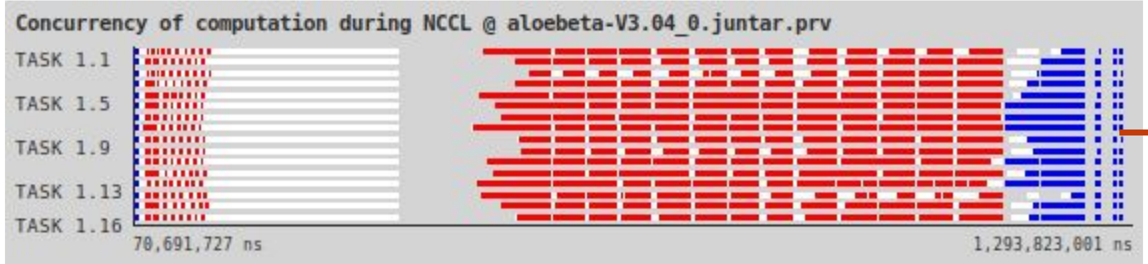




GPU usage gaps

Relevant metrics

Communication and compute overlap



0
1
2
3

Legend. Number of kernels, including the communication, running at the same time.
> 1 means overlap; 0 means no comm. going on; 1 means no overlap

Profile of time, percentage of time wrt comms volume

Overlap time percentage

	1	2	3
TASK 1.1	7.97 %	54.76 %	37.26 %
TASK 1.2	13.11 %	49.16 %	37.72 %
TASK 1.3	14.71 %	56.06 %	29.23 %
TASK 1.4	14.97 %	46.08 %	38.95 %
TASK 1.5	14.18 %	41.23 %	44.59 %
TASK 1.6	17.10 %	41.33 %	41.57 %
TASK 1.7	17.58 %	40.53 %	41.89 %
TASK 1.8	15.80 %	40.74 %	43.46 %
TASK 1.9	15.11 %	45.30 %	39.59 %
TASK 1.10	14.67 %	49.37 %	35.96 %
TASK 1.11	16.99 %	41.99 %	41.01 %
TASK 1.12	14.35 %	43.48 %	42.17 %
TASK 1.13	18.47 %	42.91 %	38.62 %
TASK 1.14	6.64 %	58.95 %	34.41 %
TASK 1.15	13.11 %	45.89 %	41.00 %
TASK 1.16	14.73 %	45.85 %	39.42 %
Total	229.50 %	743.64 %	626.86 %
Average	14.34 %	46.48 %	39.18 %
Maximum	18.47 %	58.95 %	44.59 %
Minimum	6.64 %	40.53 %	29.23 %
StDev	3.05 %	5.58 %	3.66 %
Avg/Max	0.78	0.79	0.88

Profile of time, percentage of time wrt trace time

New Histogram #1 @ aloebeta-V3.04_0.juntar.prv

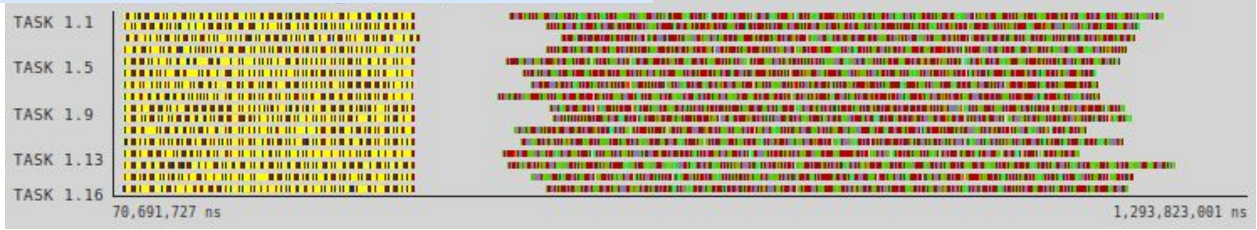
	1	2	3
TASK 1.1	6.13 %	42.06 %	28.62 %
TASK 1.2	10.17 %	38.12 %	29.25 %
TASK 1.3	10.94 %	41.69 %	21.74 %
TASK 1.4	11.66 %	35.88 %	30.33 %
TASK 1.5	11.62 %	33.79 %	36.54 %
TASK 1.6	13.82 %	33.39 %	33.58 %
TASK 1.7	14.13 %	32.56 %	33.65 %
TASK 1.8	13.14 %	33.87 %	36.13 %
TASK 1.9	11.77 %	35.29 %	30.83 %
TASK 1.10	11.12 %	37.43 %	27.27 %
TASK 1.11	13.88 %	34.31 %	33.51 %
TASK 1.12	11.56 %	35.04 %	33.99 %
TASK 1.13	15.23 %	35.38 %	31.84 %
TASK 1.14	4.94 %	43.92 %	25.64 %
TASK 1.15	10.33 %	36.17 %	32.32 %
TASK 1.16	11.44 %	35.62 %	30.63 %
Total	181.88 %	584.52 %	495.87 %
Average	11.37 %	36.53 %	30.99 %
Maximum	15.23 %	43.92 %	36.54 %
Minimum	4.94 %	32.56 %	21.74 %
StDev	2.62 %	3.23 %	3.76 %
Avg/Max	0.75	0.83	0.85

Relevant metrics

Tensor core usage per GEMM kernel

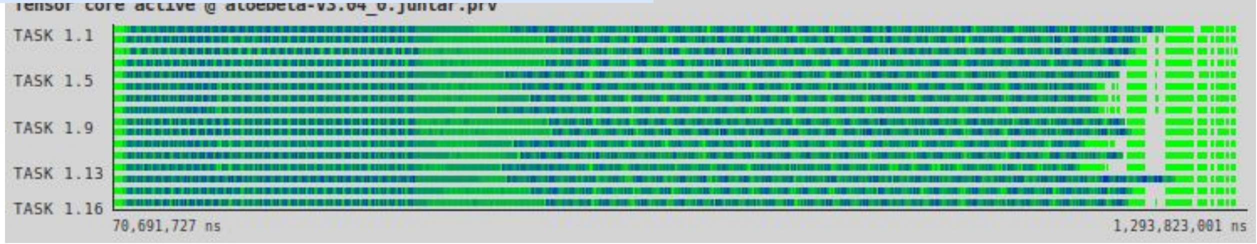
- GEMM nt 128x128x64
- GEMM tn 128x128x64
- GEMM nt 128x256x64
- flash bwd dot do o
- GEMM nn 128x128x64
- GEMM nn 256x128x64
- flash bwd convert dq
- flash_bwd
- flash_fwd
- GEMM tn 256x128x64

Timeline of kernels, only GEMMs and flash



$\bar{X}$

Tensor core usage in %, sampling

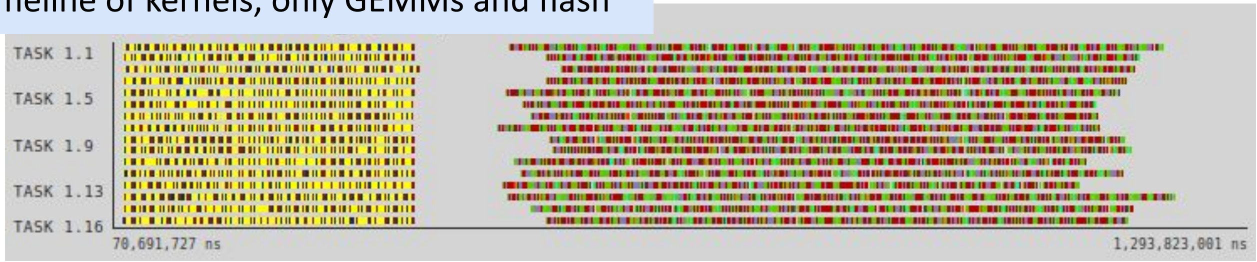


Relevant metrics

Tensor core usage per GEMM kernel

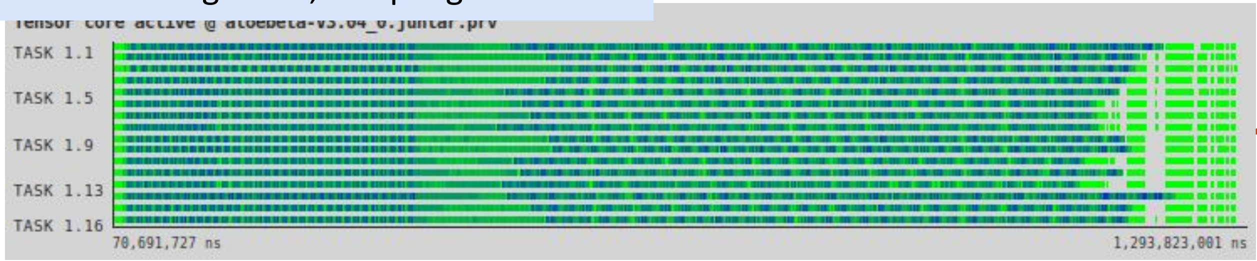
- GEMM nt 128x128x64
- GEMM tn 128x128x64
- GEMM nt 128x256x64
- flash bwd dot do o
- GEMM nn 128x128x64
- GEMM nn 256x128x64
- flash bwd convert dq
- flash_bwd
- flash_fwd
- GEMM tn 256x128x64

Timeline of kernels, only GEMMs and flash

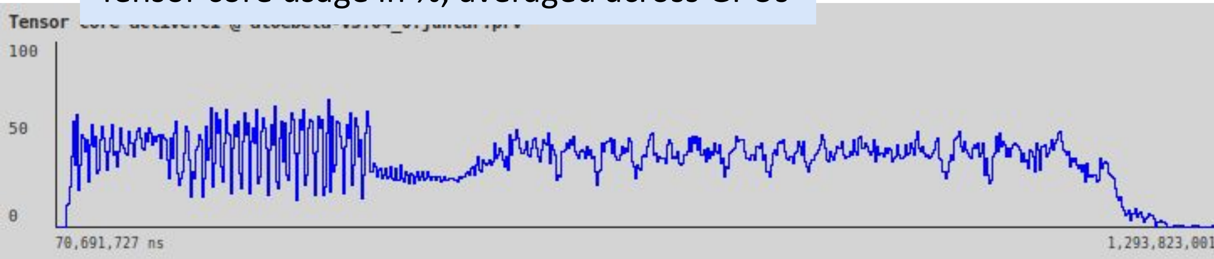


$\bar{x}$

Tensor core usage in %, sampling



Tensor core usage in %, averaged across GPUs

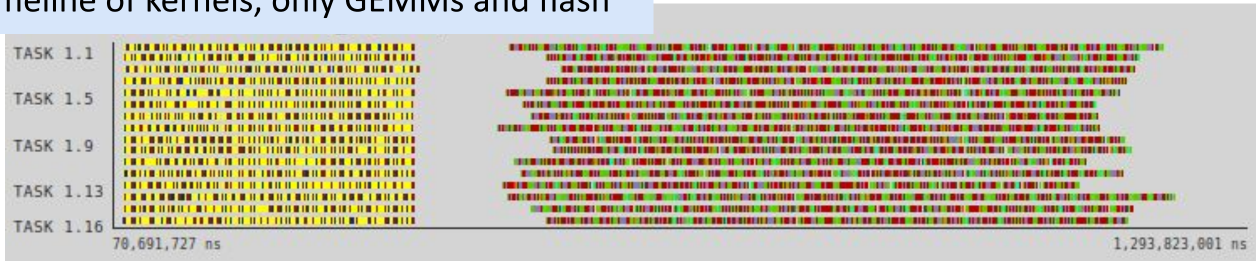


Relevant metrics

Tensor core usage per GEMM kernel

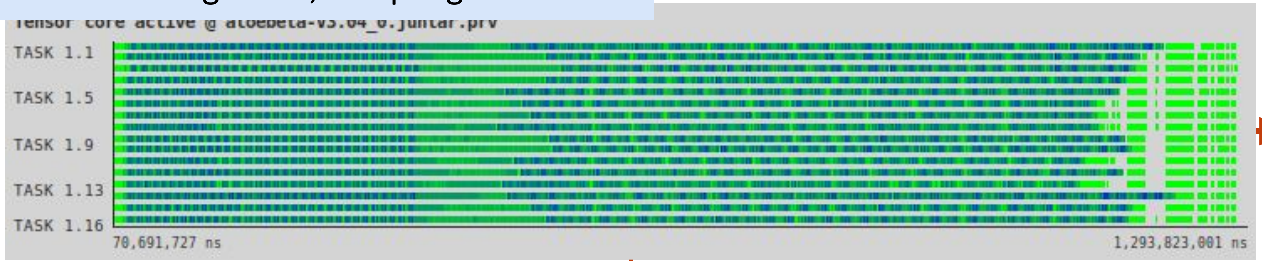
- GEMM nt 128x128x64
- GEMM tn 128x128x64
- GEMM nt 128x256x64
- flash bwd dot do o
- GEMM nn 128x128x64
- GEMM nn 256x128x64
- flash bwd convert dq
- flash_bwd
- flash_fwd
- GEMM tn 256x128x64

Timeline of kernels, only GEMMs and flash

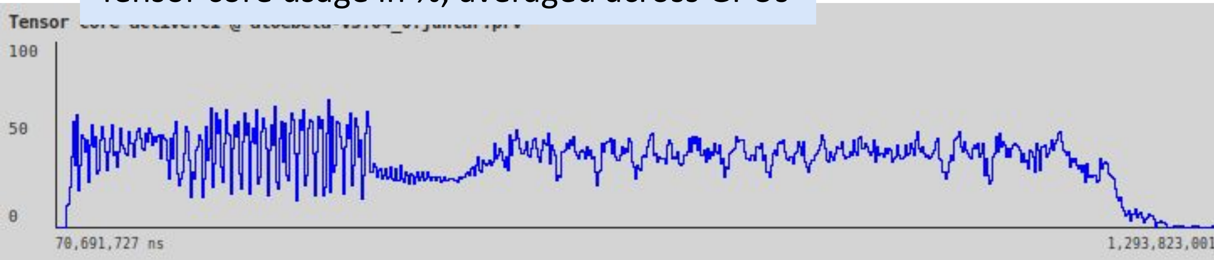


$\bar{x}$

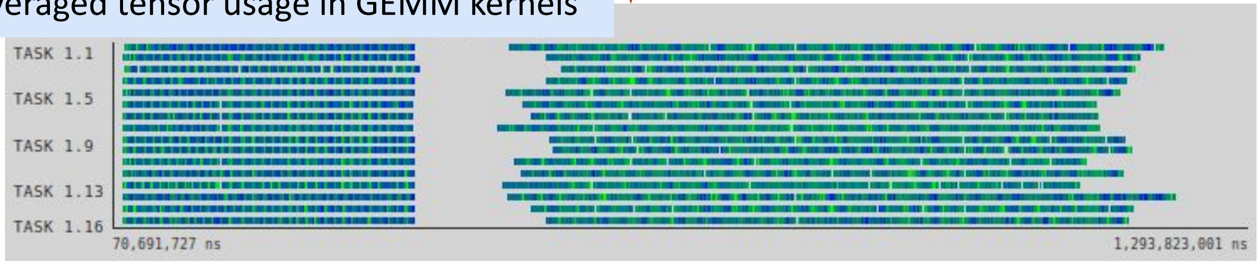
Tensor core usage in %, sampling



Tensor core usage in %, averaged across GPUs



Averaged tensor usage in GEMM kernels



Profile in next slide…

Relevant metrics

Tensor core usage per GEMM kernel

Profile of average tensor core utilization wrt peak, in %

	GEMM nt 128x128x64	GEMM tn 128x128x64	GEMM nt 128x256x64	flash bwd dot do o	GEMM nn 128x128x64	GEMM nn 256x128x64	flash bwd convert dq	flash bwd	flash fwd	GEMM tn 256x128x64
TASK 1.1	50,14	59,91	30,49	21,39	52,47	70,45	0,87	41,14	43,12	68,27
TASK 1.2	47,53	60,13	30,66	16,42	51,3	71,12	0,81	32,07	28,55	66,41
TASK 1.3	48,29	59,39	30,61	12,96	51,6	70,19	0,91	23,91	19,85	[illegible]
TASK 1.4	47,48	59,81	30,43	12,71	49,28	68,63	0,85	25,37	20,87	[illegible]
TASK 1.5	47,08	59	29,87	18,9	48,37	68,69	0,86	36,95	36,9	[illegible]
TASK 1.6	47,02	57,45	29,2	17,83	48,57	69,27	0,85	28,37	23,06	[illegible]
TASK 1.7	45,18	57,71	30,34	18,56	49,32	67,3	0,92	24,3	17,54	[illegible]
TASK 1.8	45,74	56,42	29,85	22,77	48,83	63,56	1,03	37,4	37,03	[illegible]
TASK 1.9	47,34	58,93	30,6	14,69	50,22	67,25	0,86	24,68	19,58	65,46
TASK 1.10	49,51	60,07	29,49	15,17	51,36	71,36	0,91	25,91	20,96	66,13
TASK 1.11	46,4	56,95	30,62	22,56	48,55	64,82	0,97	29,61	24,92	61,54
TASK 1.12	47,16	58,46	29,62	20,64	48,26	68,45	0,91	34,93	33,67	64,24
TASK 1.13	45,67	57,65	28	29,73	50,38	57,67	1	13,5	6,31	56,64
TASK 1.14	49,89	59,44	30,94	16,95	51,96	72,83	0,9	41,82	45,11	66,75
TASK 1.15	47,59	58,95	29,18	16,41	49,23	70,77	0,94	32,47	30,63	66,17
TASK 1.16	46,96	58,64	30,97	14,92	49,18	68,59	0,91	28,96	24,03	64,11
Average	47,44	58,68	30,05	18,29	49,93	68,18	0,91	30,09	27,01	64,09
Maximum	50,14	60,13	30,97	29,73	52,47	72,83	1,03	41,82	45,11	68,27
Minimum	45,18	56,42	28	12,71	48,26	57,67	0,81	13,5	6,31	56,64
StDev	1,39	1,11	0,78	4,24	1,36	3,56	0,06	7,13	9,92	2,75
Avg/Max	0,95	0,98	0,97	0,62	0,95	0,94	0,88	0,72	0,6	0,94

3. Very different behavior of specific GPU in some kernels (usually lower, sometimes higher)

1. High variability between GPUs in flash_attn kernels

2. Different usage depending on GEMM type…

Efficiency model for GPU traces

Device Global Efficiency

Device Parallel Efficiency

Load Balance

Quantifies how much time the devices are idle due to one device spending more time in useful work than others.

Communication efficiency

Quantifies how much time the devices are busy due to data movements.

Orchestration efficiency

Quantifies how much time the devices are idle because there is no pending work to do.

Computational scalability

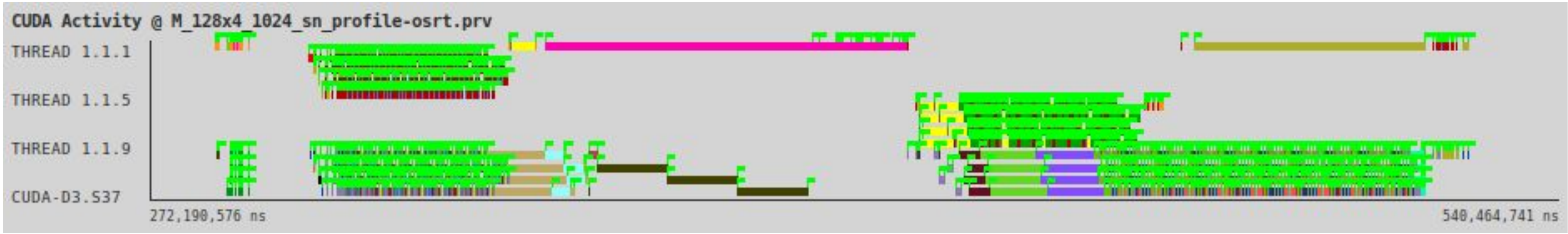
WIP

- Tensor Core usage?
- Occupancy scalability?
- Active warp scheduling?
- Executed instructions?
- SM issue rate?
- …

Complementary metrics

- Computation / communication overlap (stream level)
- Inflight kernels
- CUDA Graphs ready?
- Hardware metric aggregation
- Tensor usage in GEMMs
- Data exchange

Efficiency model for GPU traces



Configuration files

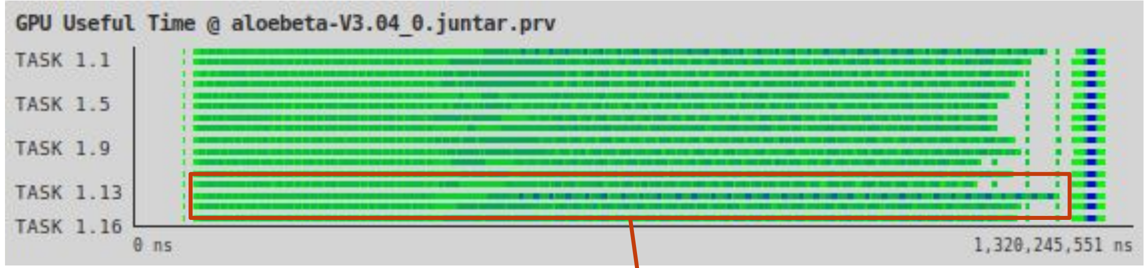
LB_eff.cfg

Case	M_128x4_1024_sn
GPUs	4
Device Parallel Efficiency	44%
-- Device Load Balance	99%
-- Device Comm. Efficiency	89%
-- Device Orchestration Efficiency	50%

Applied examples

Useful time and load balance

Load imbalance in backward phase



GPU 13 shows higher kernel duration times steadily across the training step

GPU Load Balance

	1
Total	16,634,897,517 ns
Average	1,039,681,094.81 ns
Maximum	1,126,654,021 ns
Minimum	941,880,722 ns
StDev	40,713,047.98 ns
Avg/Max	0.92

Addition of all “useful” time (compute kernels). Avg/Max is a metric for Load Balance

Putting the pieces all together

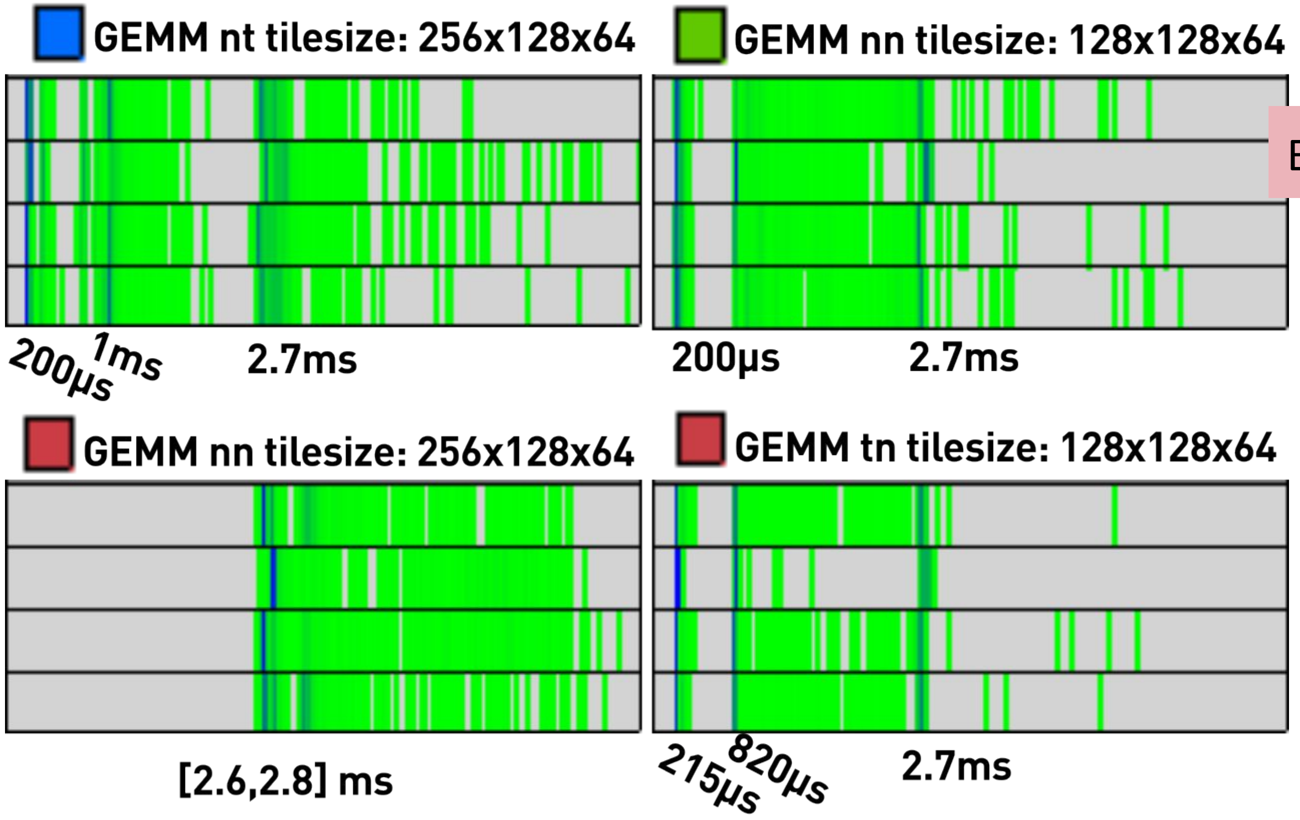
- **Microscopic** behavior
  - Tensor core utilization differences
  - Specific GPU shows worse performance in for some GEMMs and for the flash attention kernels
- **Macroscopic** effects
  - 92% of Load balance efficiency. Not bad but considerable in only 16 GPUs run, could go worse when scaling up
  - Impacts communication phase at the end of the step (other GPUs have to wait)
  - We see a higher execution time for the same specific GPU observed earlier

- 🚧 *Research currently in progress with HPAI group @ BSC*

Applied examples

Are all GEMMs born equal?

Histogram of kernel duration for different GEMM kernels



Bimodal?

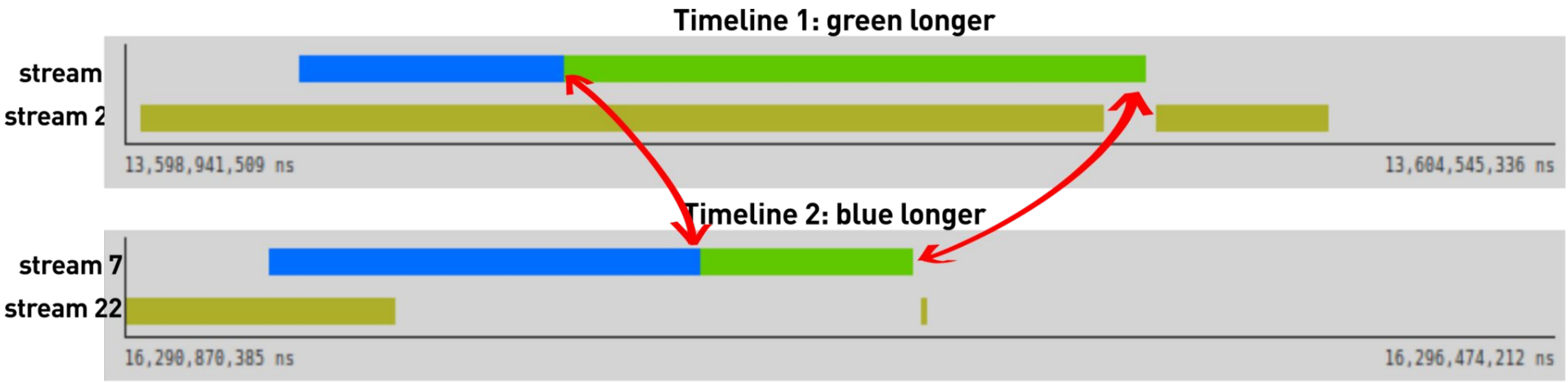
Bimodal?

One mode

Multimodal?

Applied examples

Are all GEMMs born equal?



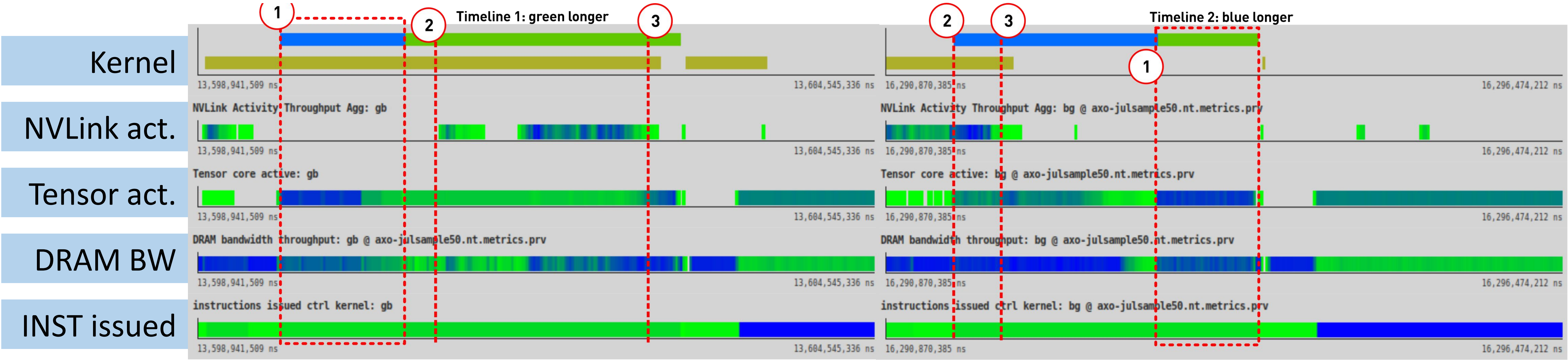
- Idle
- NCCL AllReduce
- GEMM nn tilesize 128x128x64
- GEMM nt tilesize 256x128x64

20 BSC YEARS

Applied examples

- Idle
- NCCL AllReduce
- GEMM nn tilesize 128x128x64
- GEMM nt tilesize 256x128x64

Are all GEMMs born equal?



Green longer

Blue longer

Putting the pieces all together

- **Comparing** the microscopic behavior at different moments on the trace
  - HW metrics show internal kernel behavior, at us level
  - Gives insight about the effects of overlapping communication with compute

- 🚧 *Research currently in progress with HPAI group @ BSC*

Useful links

- Package repository: https://gitlab.pm.bsc.es/beppp/nsys2prv
- Documentation: https://gitlab.pm.bsc.es/beppp/nsys2prv/-/wikis/Home
  - Basic usage
  - Feature status
  - Troubleshooting
- CFGs for the presented metrics included in the repo!
- And don’t miss the opportunity: if you have a use case, apply for a POP assessment! :)

20 YEARS BSC