Anemoi profiling - demo

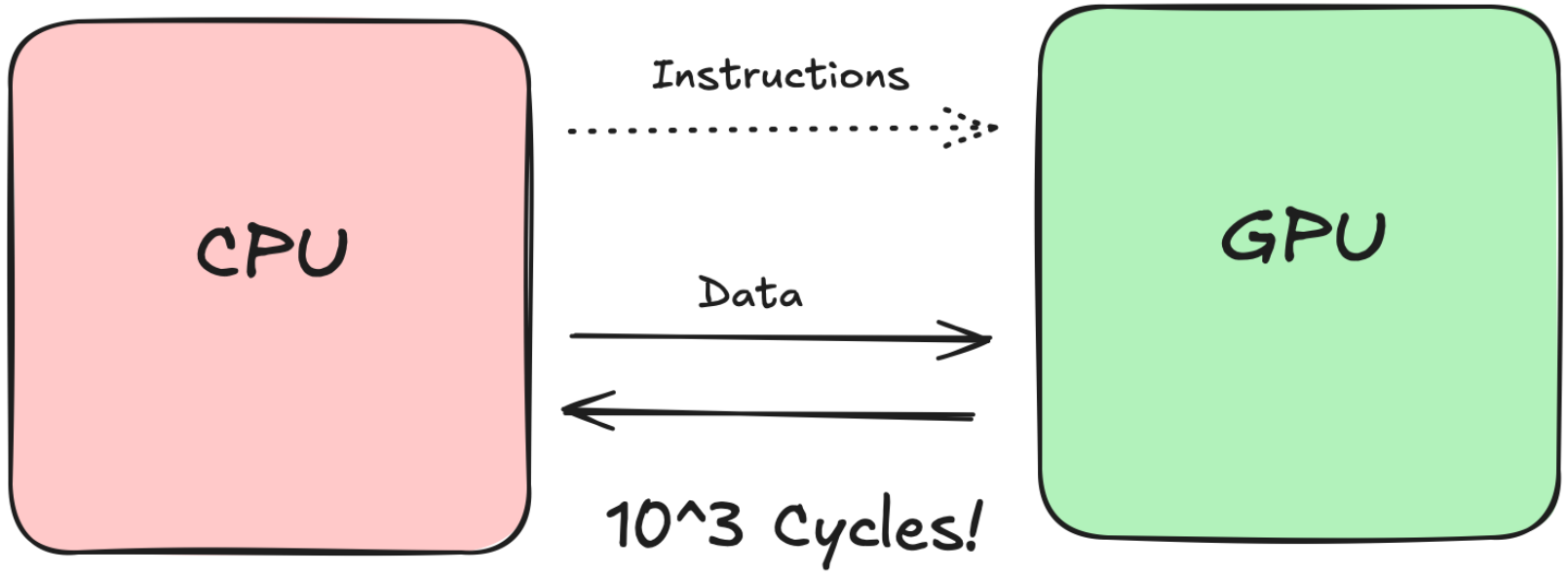
Jan Polster & Cathal O’Brien

ECMWF

Why profile?

- Enable you to run the largest models possible
- Ensure your program is running as fast as possible
- Understand bottlenecks
  - Memory usage
  - Compute time
- Identify and fix performance problems
  - Excessive data movement
  - Memory leaks
  - Slow dataloading

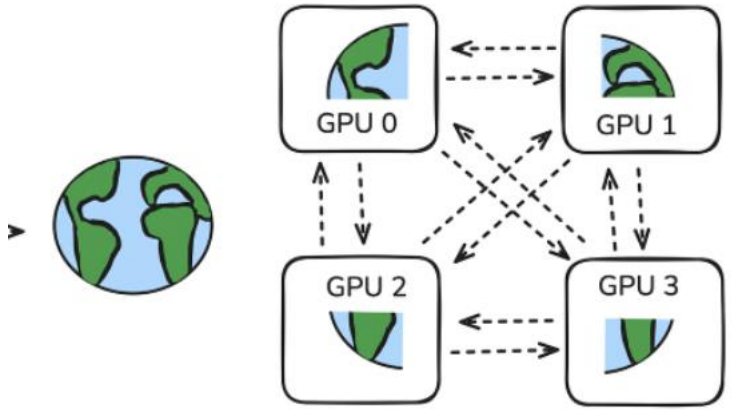
What is a CPU?



- Must be driven by a CPU
  - CPU launches all programs (called ‘kernels’) on GPU
  - CPU and GPU have separate memory spaces and must explicitly transfer data (slow)

Why do we parallelise?

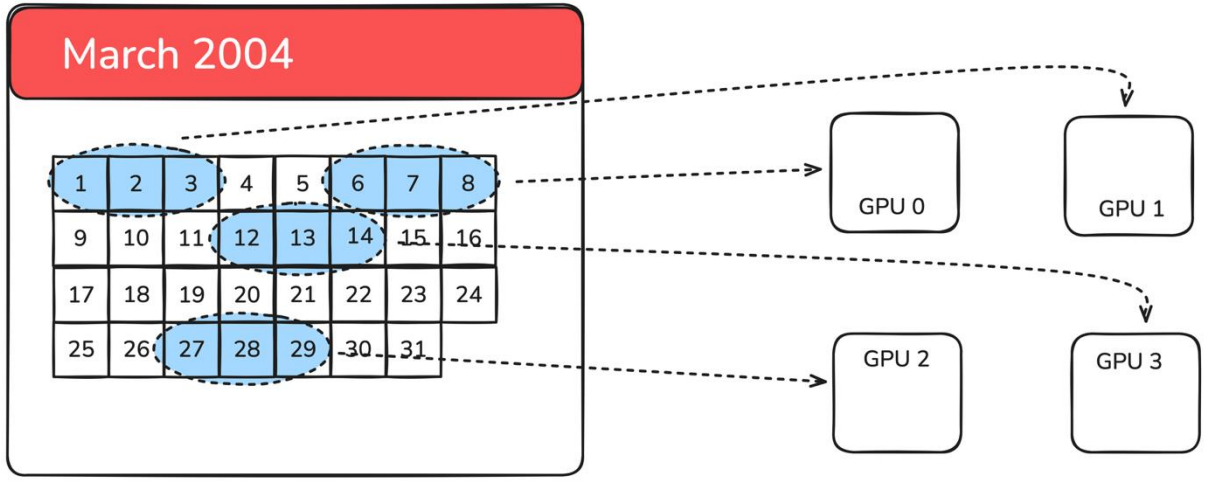
- Faster training!
  - Share the work across multiple GPUs
  - Each can compute a fraction in parallel
  - … and sychronise their results



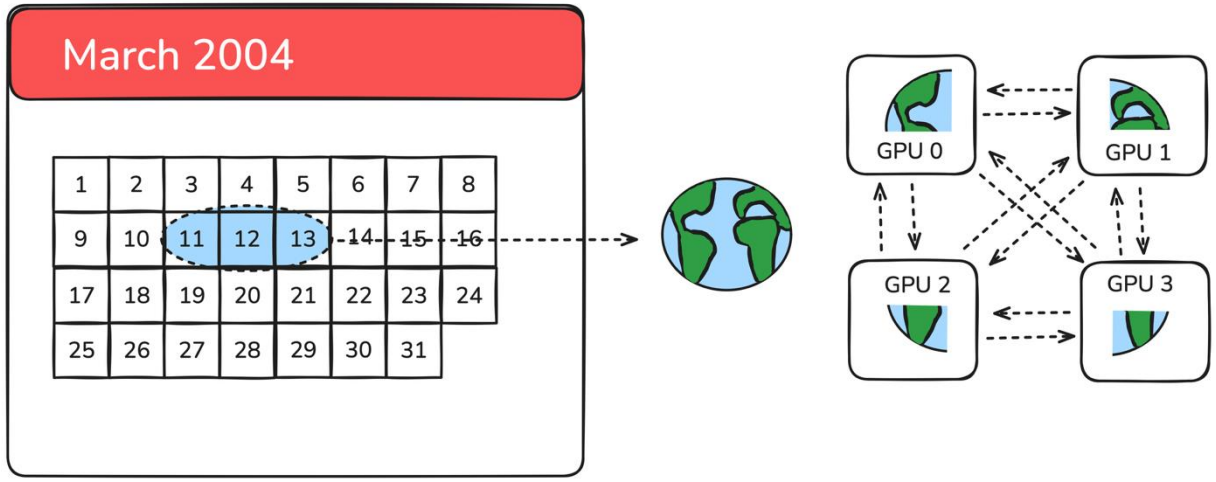
- Larger models!
  - Split the globe across multiple GPUs
  - Each GPU only needs to communicate the boundary conditions
  - Never materialise the full globe in a single GPUs memory

Parallelism in Anemoi

- Data Parallelism: DistributedDataParallel (PyTorch)
  - Distribute training batch across model replicas
  - Aggregate gradients via all-reduce, good scaling 🙂
  - Limited by batch size 😕



- Model parallelism: domain-specific sharding
  - Distribute input data and activations across GPUs
  - Collective communication to handle synchronisation
  - Limited by communication overheads



Memory profiling

- Problem: Maximise memory use without hitting the limit
  - More performant
  - More efficient

- Probably the most common performance issue when designing new experiments

- Avoid the dreaded CUDA OutOfMemory error

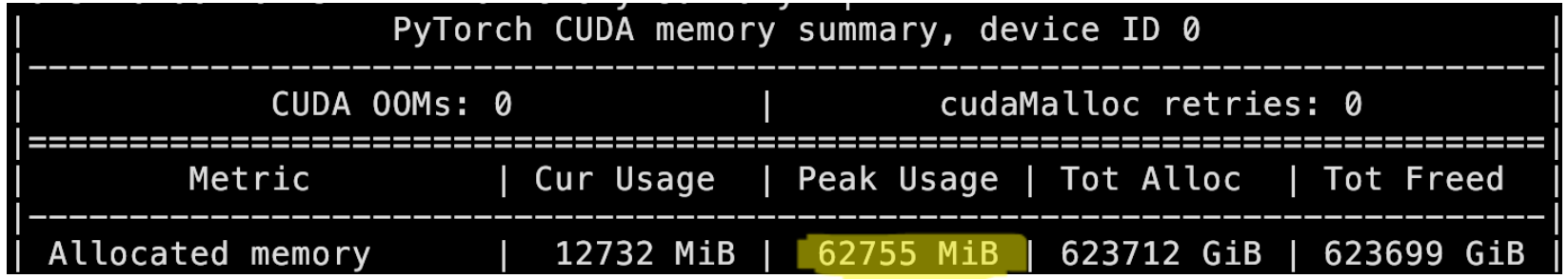
```
torch.OutOfMemoryError: CUDA out of memory. Tried to allocate 1.41 GiB. GPU 2 has a total capacity of 39.56 GiB of which 870.75 MiB is free.
```

  - Why did this happen?
  - Where is the GPU memory being used?

Highest level – peak memory usage

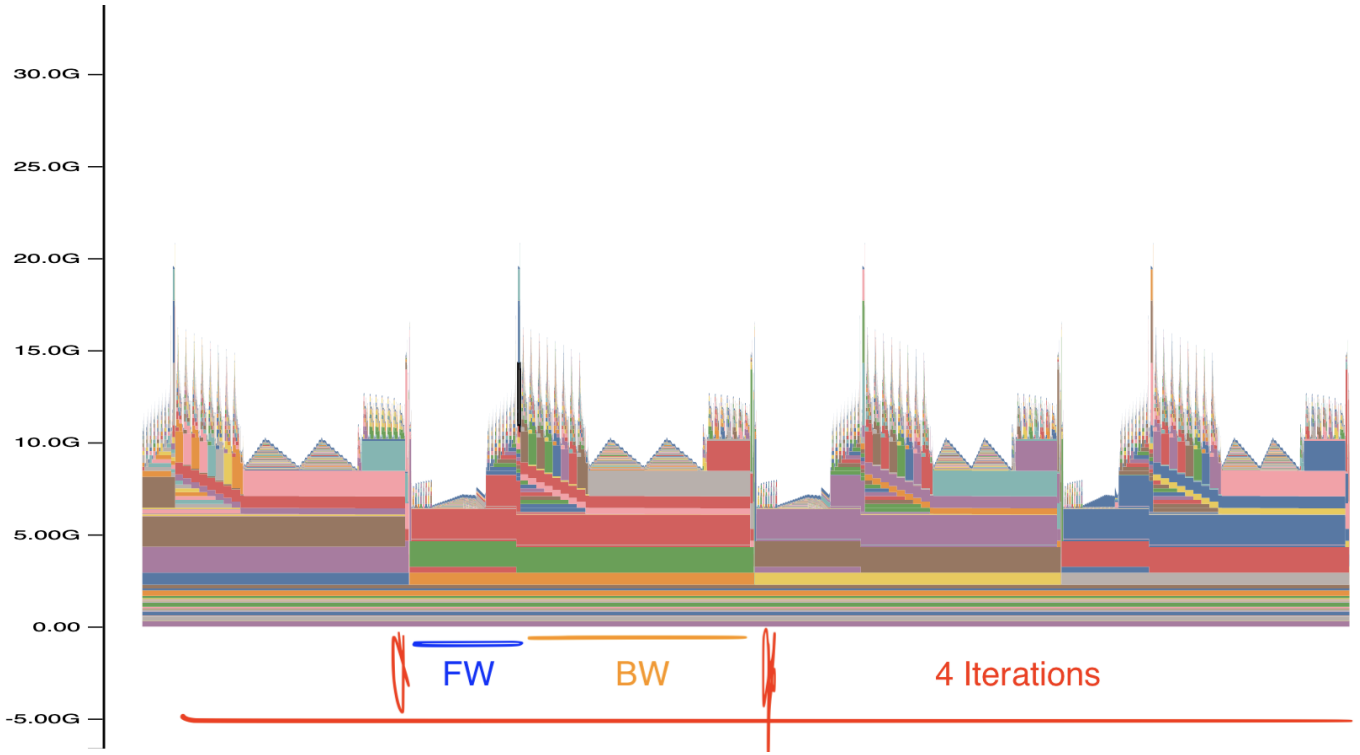
- Memory usage is impacted by a variety of factors
  - Model parameter count
  - Batch size
  - Input resolution

- First thing you should do is check the peak memory usage



- But that doesn't tell the whole story...

Memory snapshots

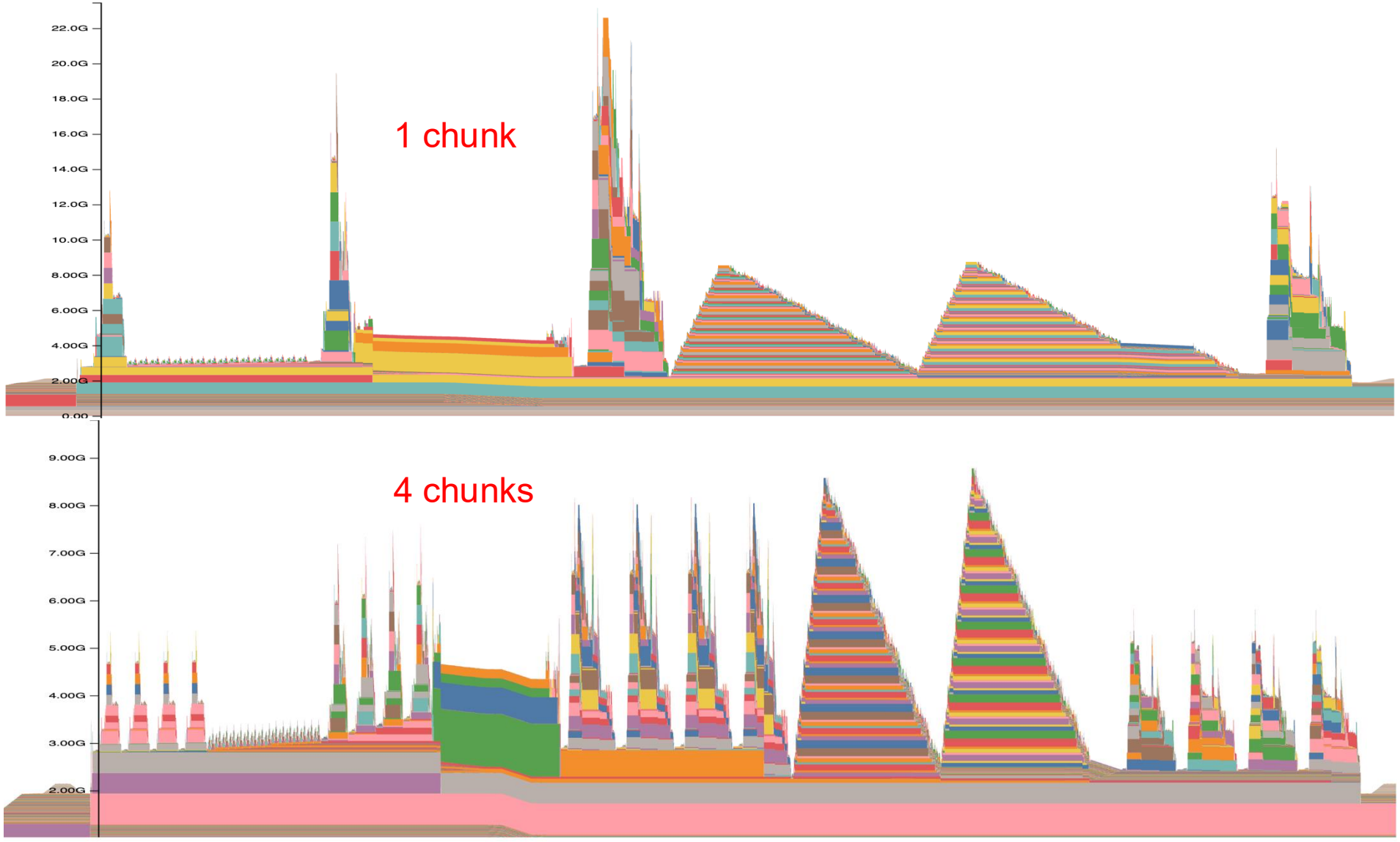


- Memory usage varies over time
  - Peak memory usage of 20GB vs average of ~12GB

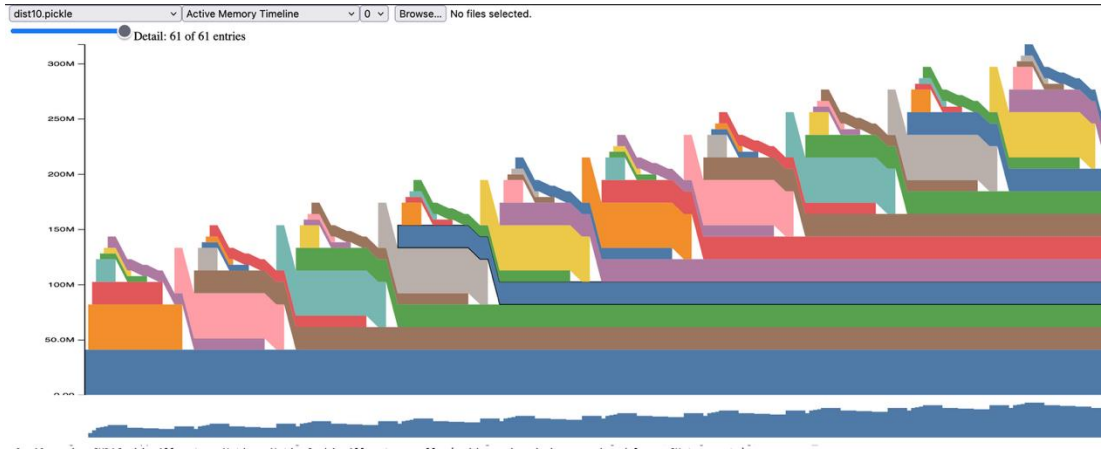
Mapper chunking

- Running inside 1 GPU now
  - but we are breaking our subgraph into smaller subgraphs to reduce peak mem

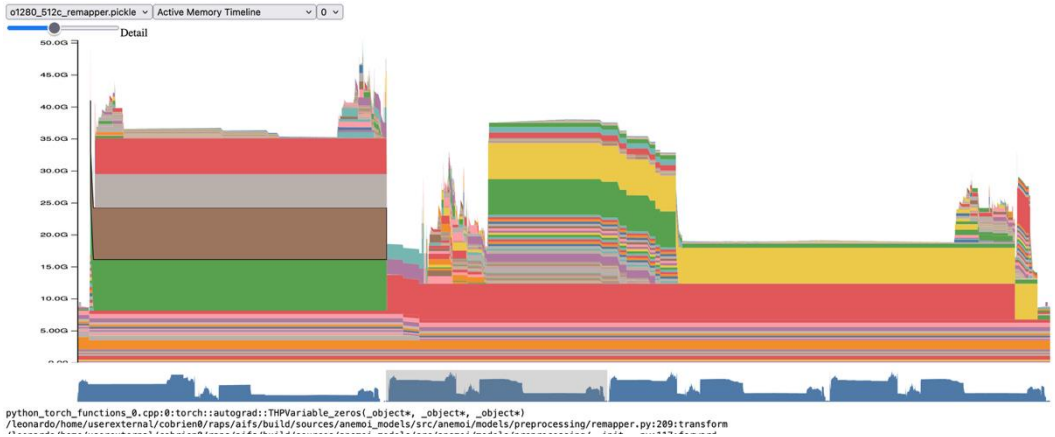
Memory usage



- Memory leak



- Large blocks of memory



How to view a memory snapshot

- Add an entry to your config ->

- Find the output file

```
ls $HOME/anemoi/outputs/profiler/*
```

```
profiler/d62851ba-ab68-465c-b093-e6bc9de2fefd:
memory_snapshot.pickle
```

- Upload to https://docs.pytorch.org/memory_viz

```yaml
defaults:
- data: zarr
- dataloader: native_grid
- diagnostics: evaluation
- datamodule: single
- hardware: example
- graph: multi_scale
- model: gnn
- training: default
- _self_

config_validation: False

# Add the following lines to enable memory snapshots
diagnostics:
  benchmark_profiler:
    snapshot:
      enabled: True
```

Runtime profiling

- You might wonder
  - Where is the GPU spending its time while running?
  - Is it going as fast as it possibly could?

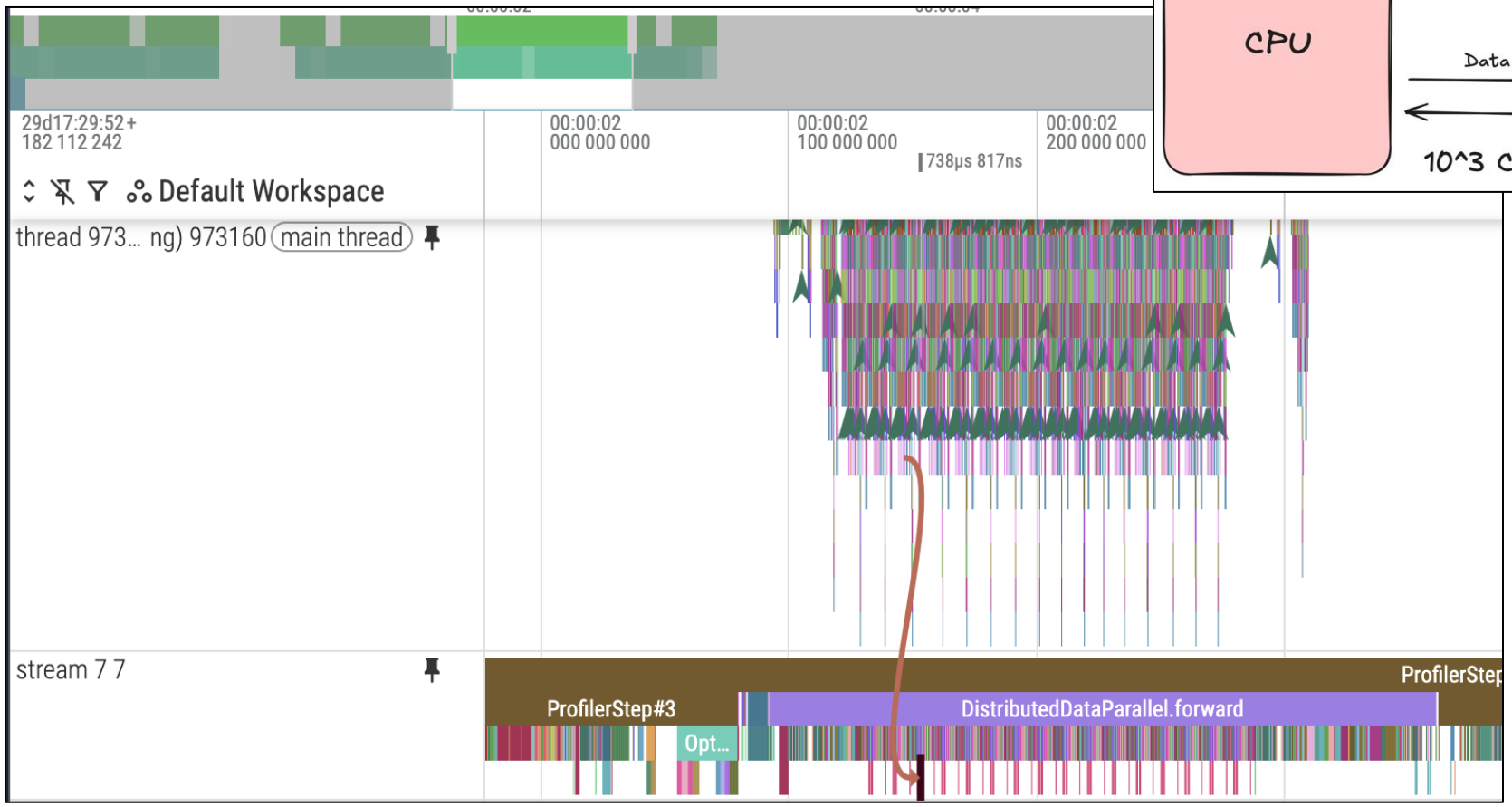
- Examining traces can help understand a GPUs runtime

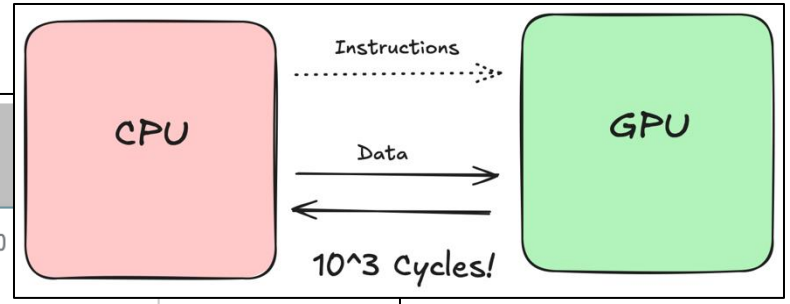
Enabling runtime traces

```yaml
defaults:
  - hardware: slurm
  - data: zarr
  - dataloader: native_grid
  - datamodule: single
  - model: transformer
  - graph: encoder_decoder_only
  - training: default
  - diagnostics: evaluation
  - override hydra/job_logging: none
  - override hydra/hydra_logging: none
  - _self_

# Add the following lines to enable runtime traces
diagnostics:
    benchmark_profiler:
        memory:
            enbaled: True
            steps: 10
```

Runtime traces



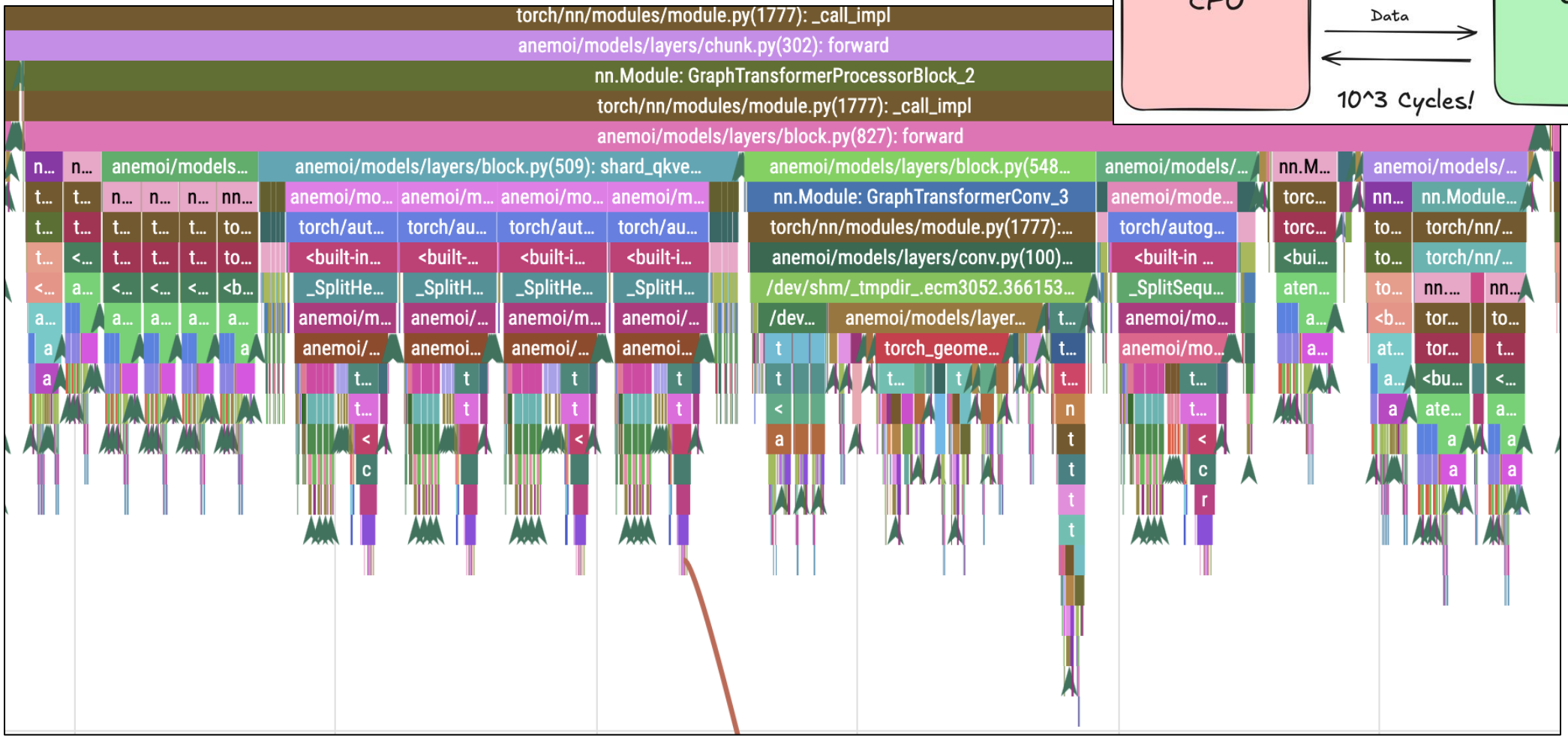


CPU

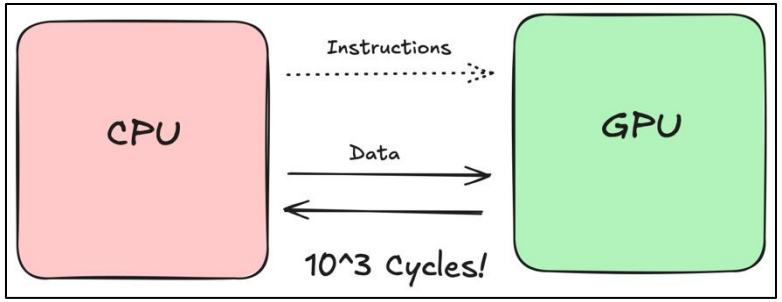
GPU

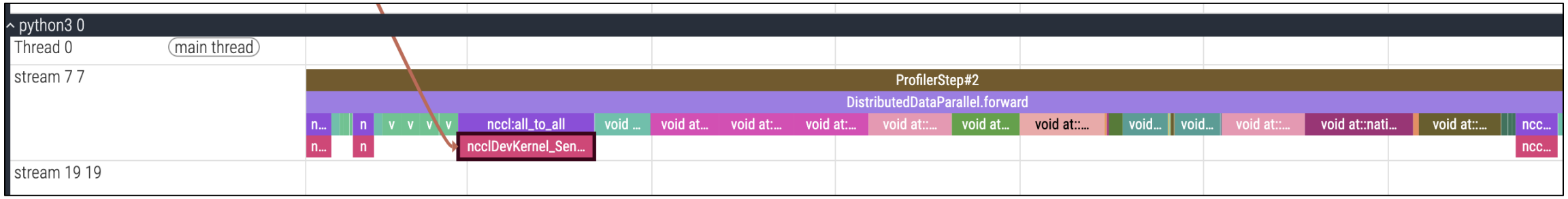
Time ->

CPU view


torch/nn/modules/module.py(1777): _call_impl
anemoi/models/layers/chunk.py(302): forward
nn.Module: GraphTransformerProcessorBlock_2
torch/nn/modules/module.py(1777): _call_impl
anemoi/models/layers/block.py(827): forward
anemoi/models/layers/block.py(509): shard_qkve...
anemoi/models/layers/block.py(548...
nn.Module: GraphTransformerConv_3
torch/nn/modules/module.py(1777):...
anemoi/models/layers/conv.py(100)...
/dev/shm/_tmpdir_.ecm3052.366153...
anemoi/models/layer...
torch_geome...
anemoi/models/...
anemoi/mode...
torch/autog...
<built-in ...
_SplitSequ...
anemoi/mo...
nn.M...
torc...
<bui...
aten...
anemoi/models/...
nn.Module...
torch/nn/...
CPU
Instructions
Data
GPU
10^3 Cycles!

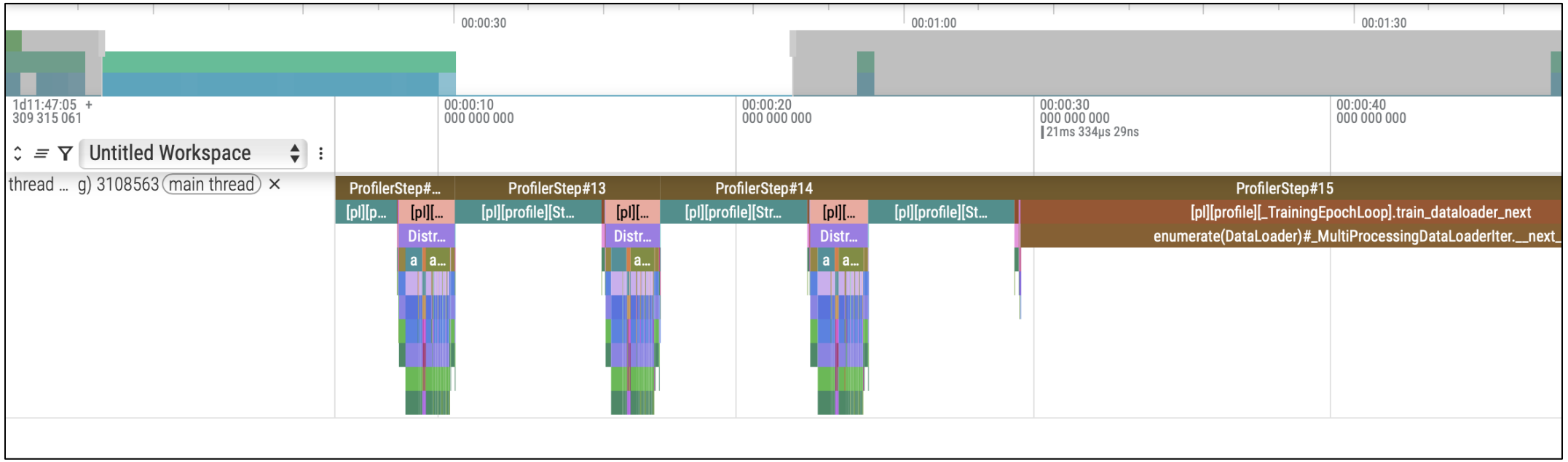
GPU view





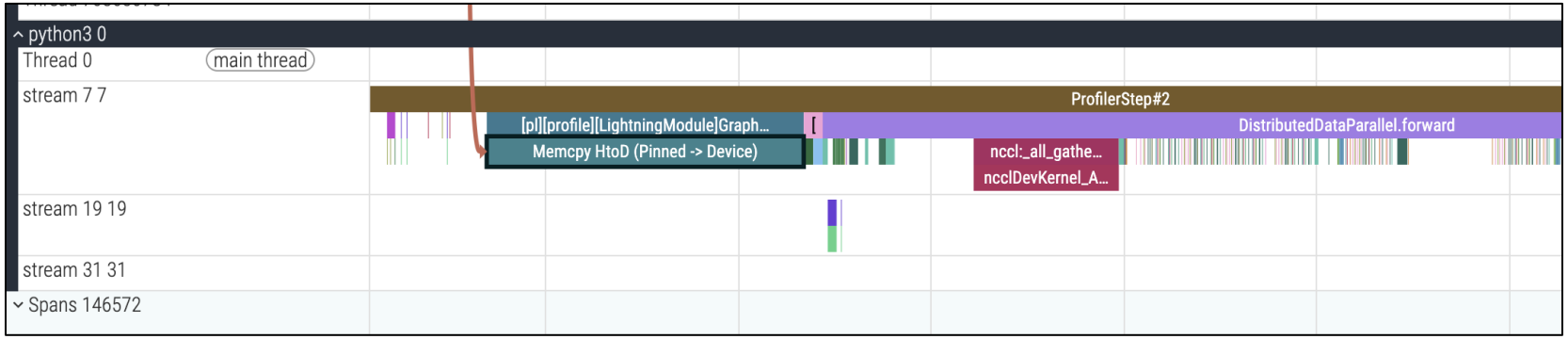
- Look out for gaps here

Dataloader bound



- Filesystem can't provide data fast enough for GPU
  - First, try increasing your number of dataloader workers
    - Until CPU memory is full
  - If still dataloader bound, optimise your data layout to match access pattern

Data transfers



- Sending the input batch to GPU
  - If very large, try using ‘pinned memory’