

https://docs.pytorch.org/memory_viz (or just search “memory viz”), best with pytorch >=2.12

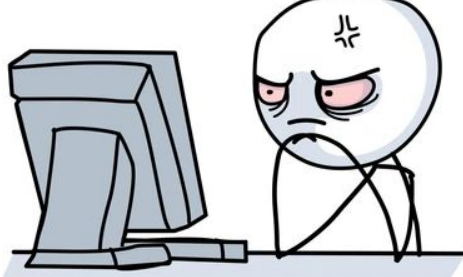
Motivation

Nvidia CUDA Graph mitigates CPU launch overhead, especially on Blackwell GPUs. However, CUDA Graph usually comes with high memory overhead, which hinders its adoption in large-scale memory-tight workloads such as GenAI training. The lack of existing tools makes it difficult to understand and mitigate the memory overhead from CUDA Graph.

We built a Private MemPool Visualizer to present CUDA Graph memory consumption. Private MemPool Visualizer demonstrates alloc and free events in private mempool and the high watermark of memory consumption. It also visualizes the mempool fragmentation due to different mempools or CUDA streams.

Existing Memory Visualizer (Doesn’t show memory held in private pool)

Why does it OOM? I free’d my tensors!!



```
graph = torch.cuda.CUDAGraph()
with torch.cuda.graph(graph):
    data = torch.empty(
        _180GB, device="cuda", dtype=torch.uint8
    )
del data

# not oom
x0 = torch.empty(_60GB, device="cuda", dtype=torch.uint8)
# oom
x1 = torch.empty(_60GB, device="cuda", dtype=torch.uint8)
```

memory_snapshot.pickle ▾ Active Memory Timeline ▾ 0 ▾

☐ Require click to show trace (applies on file load)

Choose Files No file chosen

Detail: 2 of 2 entries | Search address (hex)... | Search stack frame...



Empty but oom???

Private MemPool Visualizer (Shows memory held in private mempool)

A **private mempool** is GPU memory reserved for a specific owner (e.g., CUDA graph) and isolated from eager mempool. Private mempool can cause fragmentation because free memory cannot be directly reused across:

- Eager memory pool and private pool.
- Different private pools.
- Different CUDA streams within the same private pool.

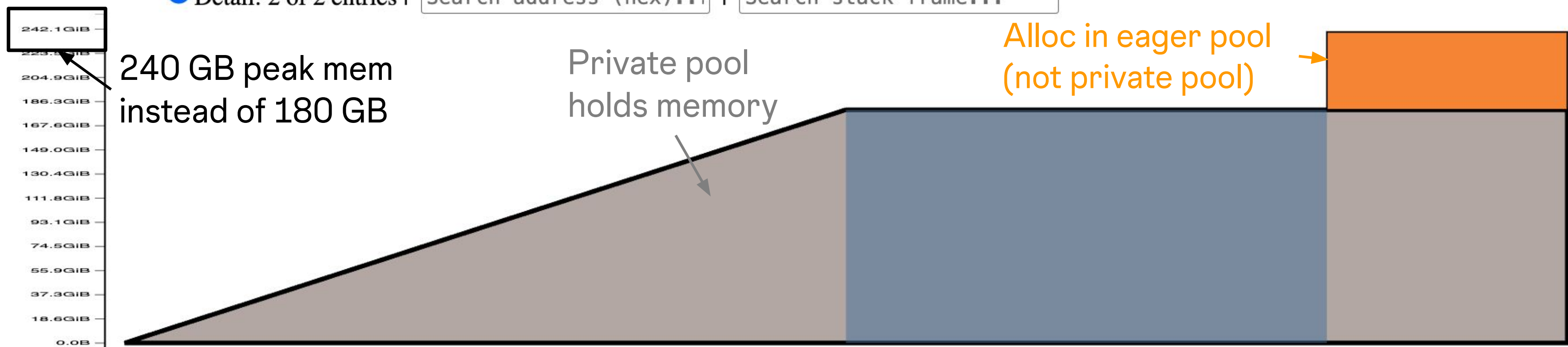
memory_snapshot.pickle ▾ Allocated Memory (incl. Private Pools) ▾ 0 ▾

☐ Require click to show trace (applies on file load)

Choose Files No file chosen

Note: Private pool memory (the gray bar) is shown as allocated until the pool's segment is freed. This view requires that MemPools are not deleted before torch.cuda.memory._snapshot() is called.

Detail: 2 of 2 entries | Search address (hex)... | Search stack frame...



240 GB peak mem instead of 180 GB

Private pool holds memory

Alloc in eager pool (not private pool)

(Naive) model example: training with CUDA Graph

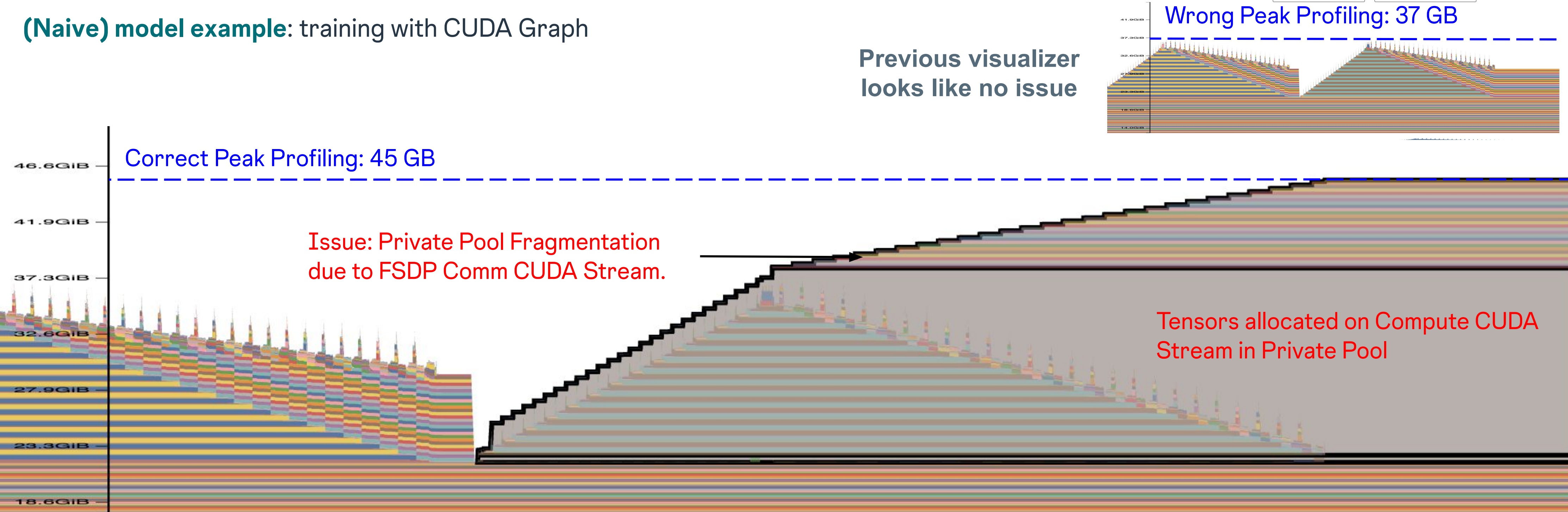
Previous visualizer looks like no issue

Wrong Peak Profiling: 37 GB

Correct Peak Profiling: 45 GB

Issue: Private Pool Fragmentation due to FSDP Comm CUDA Stream.

Tensors allocated on Compute CUDA Stream in Private Pool



New visualizer shows the issue