enfabrica

Shared Memory Pool for AI Applications

August 2025

:: 3.2 tbps acf-s “millennium”

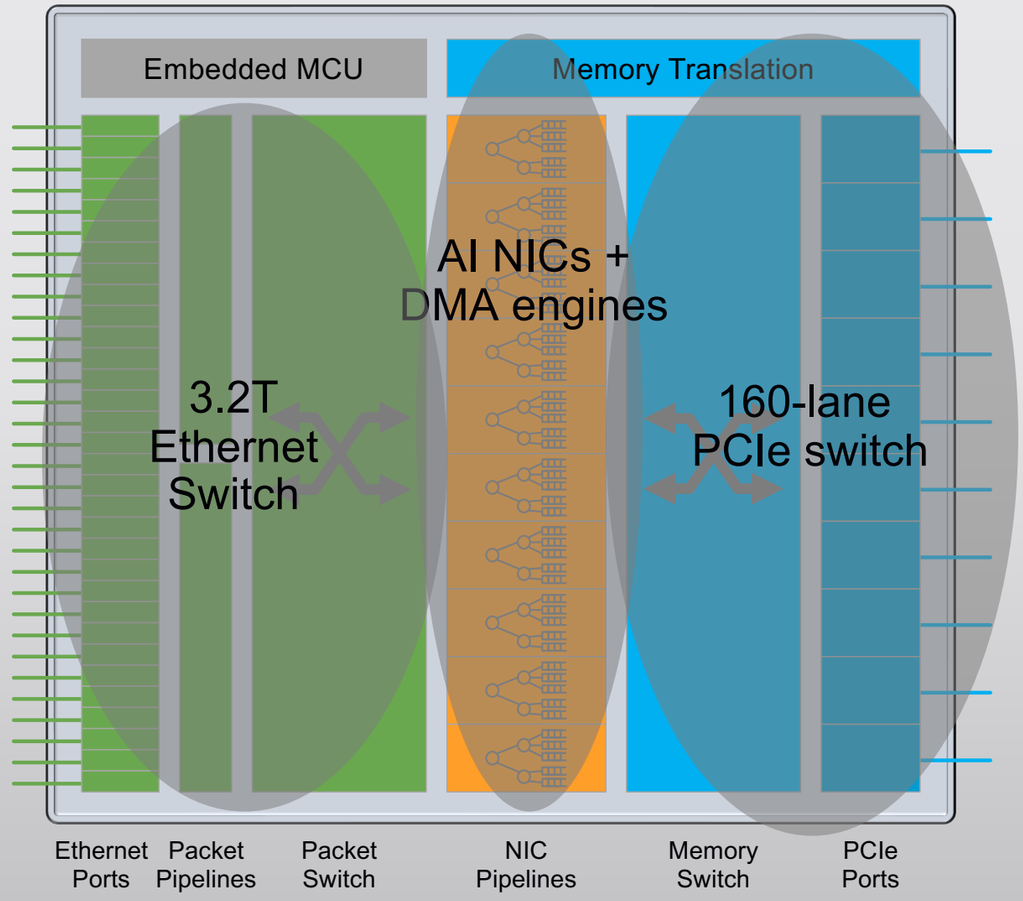
world’s highest throughput ai supernic chip

Resilient network
high port radix over fat (800G) or skinny (100G) links

Full Router:
consolidates NIC-TOR-PCIe fabrics with precise steering to/from queue pairs

User programmable transport
on scalable infra host cores at aggregate line rate PPS

ACF-S “Millennium” Chip



Delivers elastic, peak aggregate **3.2 Tbps bandwidth to accelerator**

Multi-planar internal switch fabric **absorbs GPU incast, optimizes data placement and reduces data transfer time**

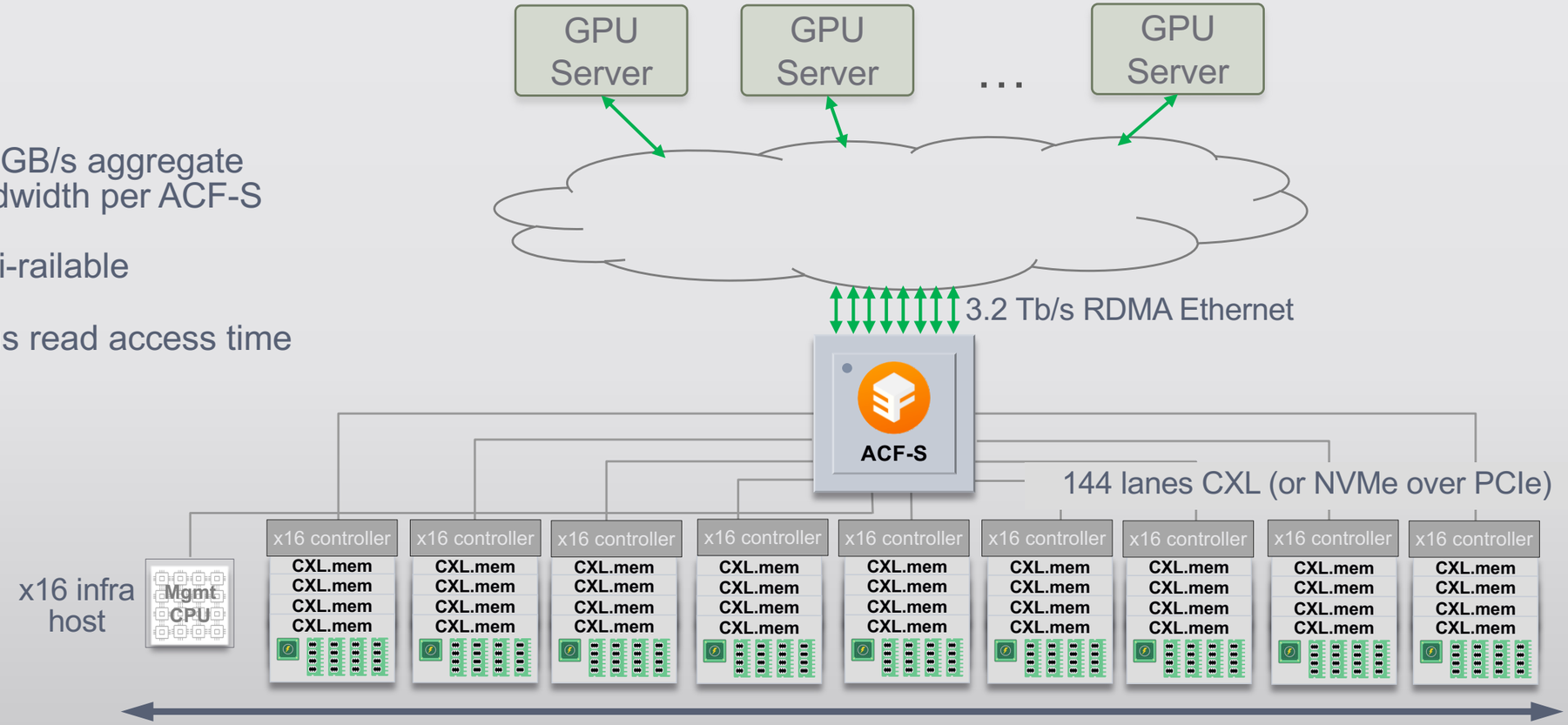
Composable DMA & collective ops offload GPU SMs:
40K copy engines / data movers

:: memory aggregation with acf-s

400 GB/s aggregate bandwidth per ACF-S

Multi-railable

~6 µs read access time



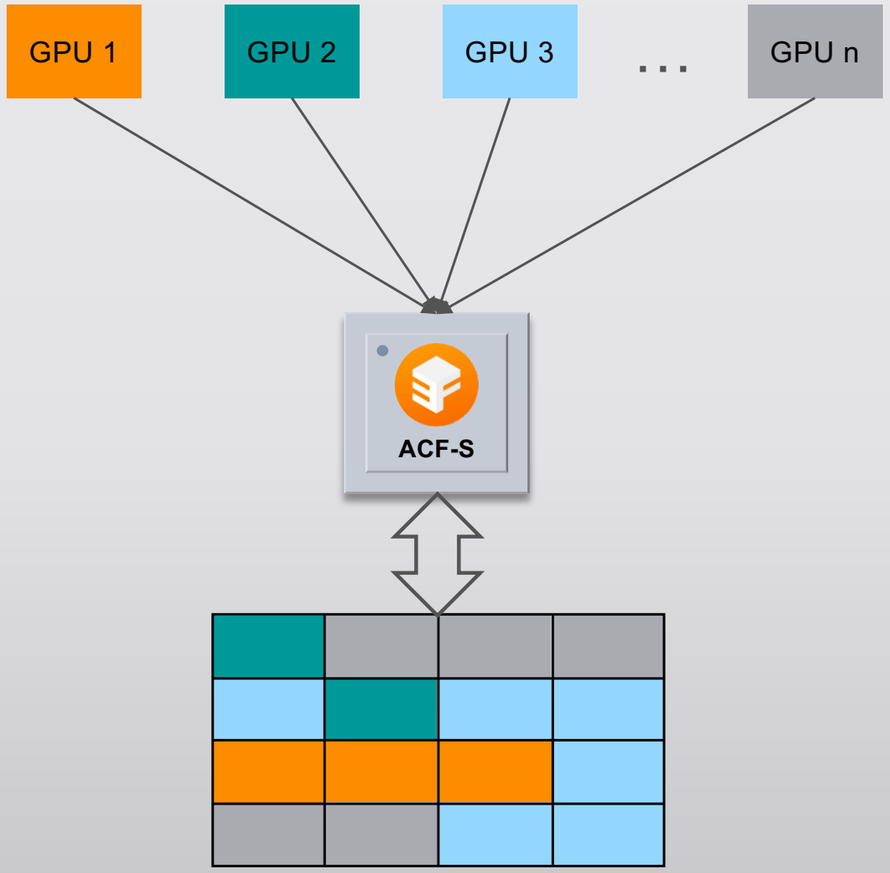
Can also aggregate NVMe Flash – up to 144 lanes

:: shared memory pool benefits

Very large memory space
- Can scale with additional systems

Fast network access
- Network access bandwidth matching memory
- One network hop away

Shared across multiple clients
- Efficient memory utilization vs dedicated memory per client
- Ability to share data between clients



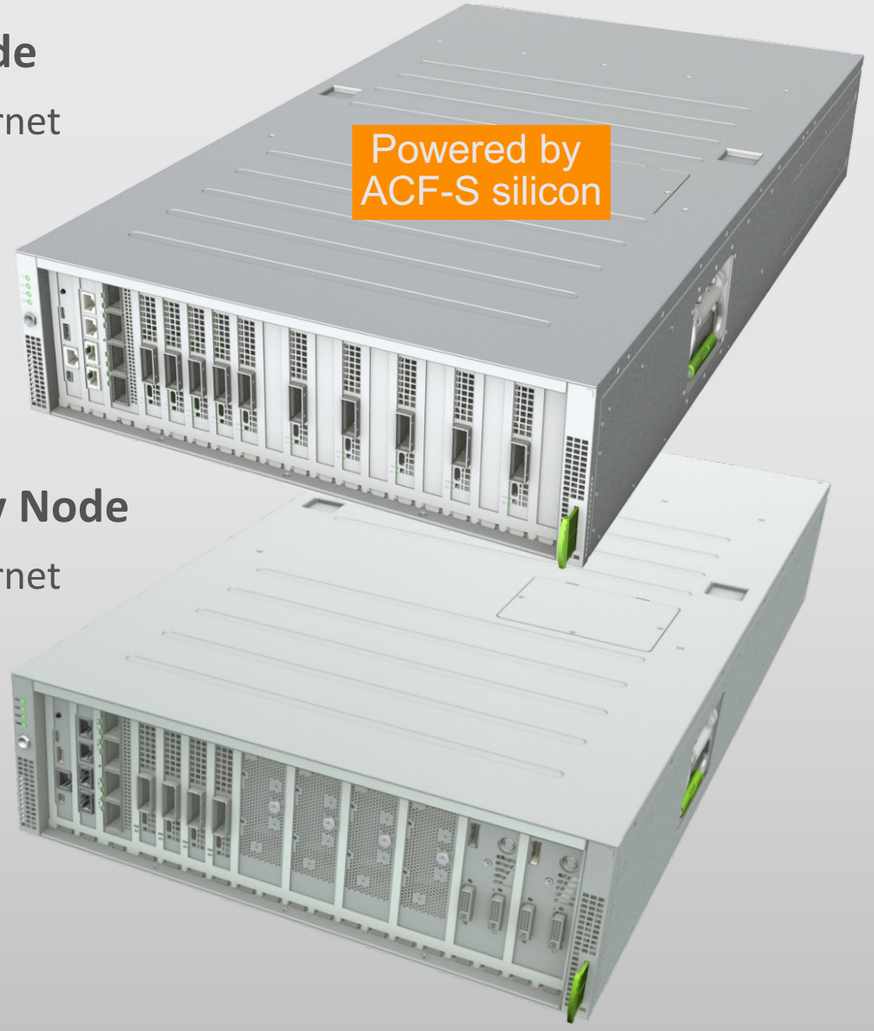
:: thames system

GPU networking node

- **4 x 800G OSFP** Ethernet
- **10 x16 PCIe** cabled

In-Network Memory Node

- **4 x 800G OSFP** Ethernet
- **4.5TB CXL DDR5**



8 Tbps AI Networking Node

Connect any combination of GPUs, CPUs, CXL memory, SSD to network

Programmable Network Transport: RoCE, RDMA over TCP

Replaces NICs, PCIe switches, Ethernet TOR

800G server I/O

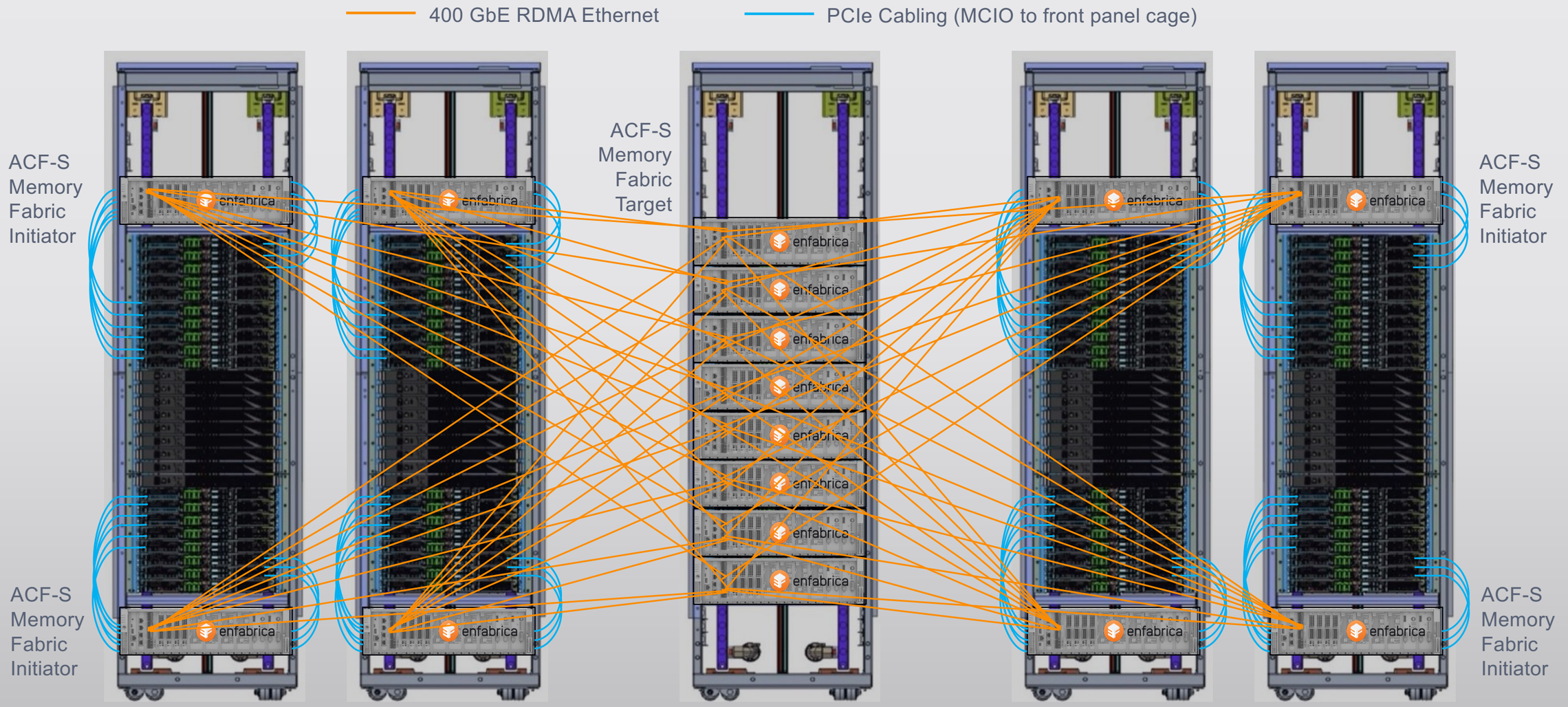
Composable, modular, production-grade

:: enfabrica EMFASYS

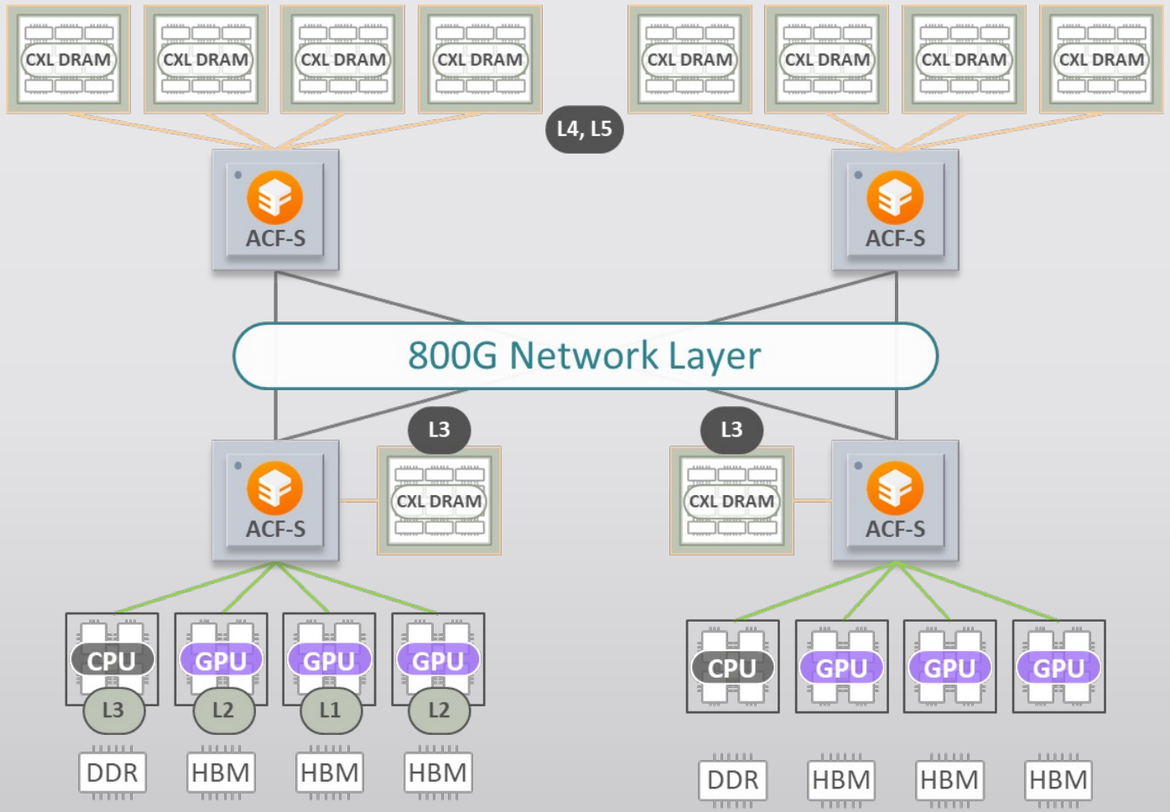
EMFASYS : Enfabrica Elastic AI Memory Fabric

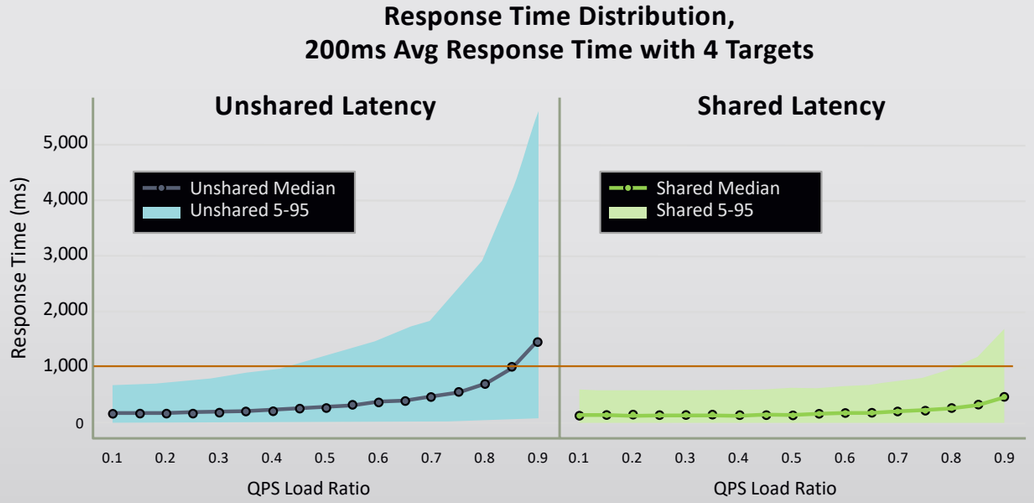
- **3.2Tbps RDMA Target** with 4.5 TB -18 TB CXL DDR5 Store per node (**50-100X HBM**)
- **Low Jitter 6µs RDMA read** access time between GPU HBM and ACF-S Target Memory (**50-200X latency reduction** vs GPU Direct Storage)
- **Unconstrained write/erase cycles** vs Flash Storage for sustained high-throughput KV cache / token / activation data movement
- **<$15/GB** fully loaded fabric-attached ACF-S memory vs **~$100/GB for HBM3e** stored KV cache, token, or activation buffers → **5X cost reduction**
- Memory Target to **any initiator GPU server** (H100/200, B200/300, MI3xx) via PCIe to **drive down FLOPs/HBM consumption** and **cost per query up to 50%**
- **No CPU memory controller contention or CPU memory locality constraints**, 100% pooled, fabric-attached and headless

:: cluster scale EMFASYS architecture



:: impact of memory tiering on inference at scale





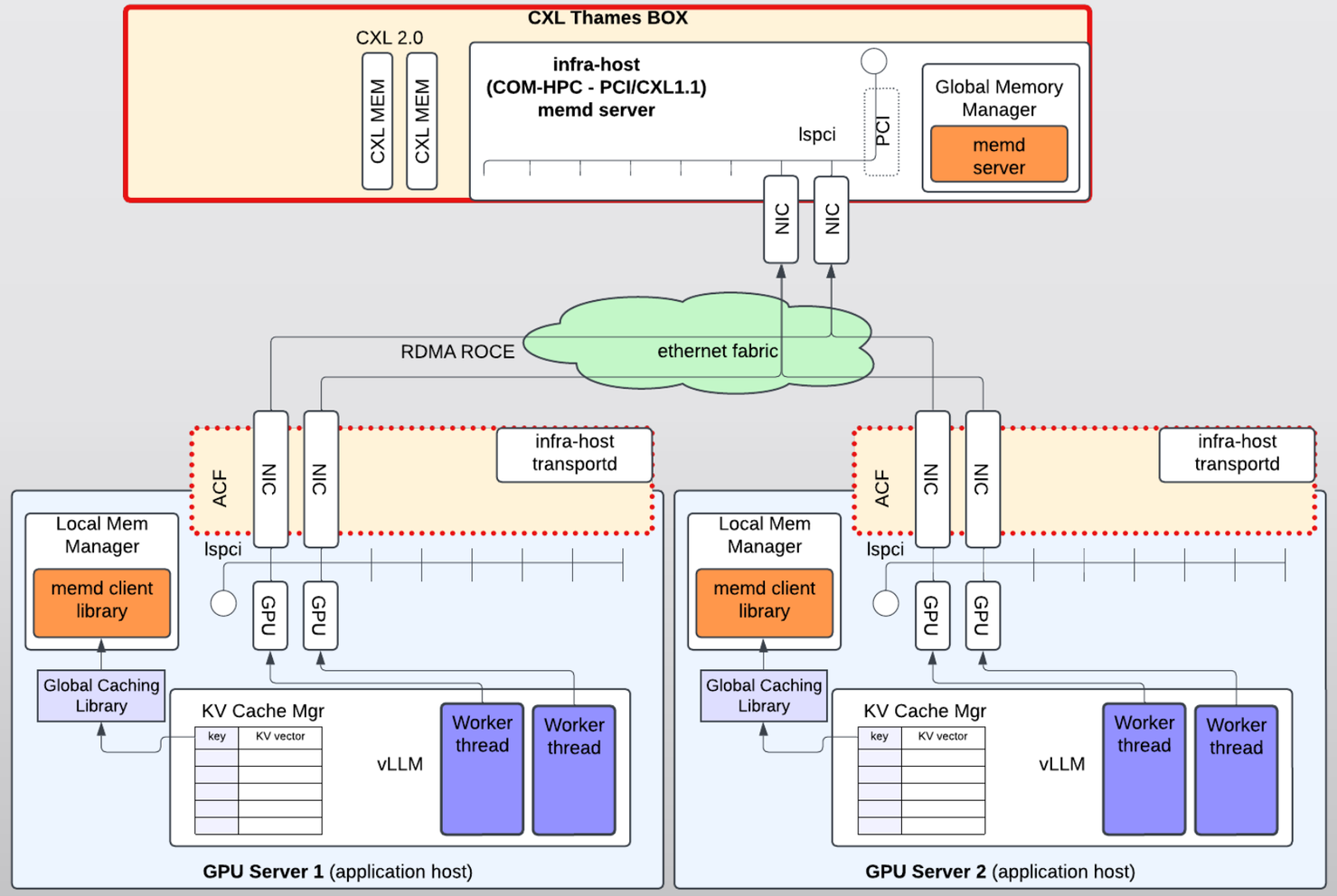
GPU capacity subject to high overprovisioning
to meet prefill-to-decode latency requirements

ACF-S can drive up to 50% fewer FLOPS required for large-sequence-length inference workloads at scale

:: inference acceleration with shared memory pool

Testing environment
- Enfabrica Memory Box with up to 9 CXL memory cards
- 1 or more GPU servers
- 400Gb/s Ethernet network

Software environment
- vLLM inference framework
- LMCache serving engine
- Enfabrica rmem layer



:: initiator software stack

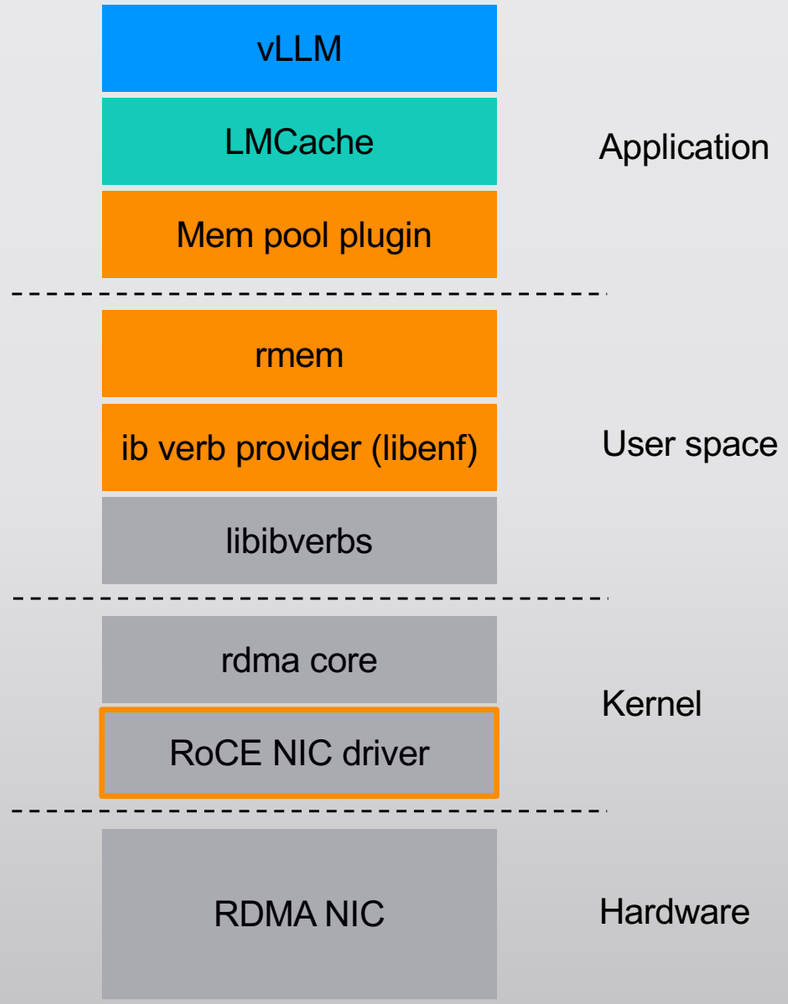
Standard RDMA stack + Enfabrica libraries

- RDMA device driver (Enfabrica driver for ACF-S)
- IB verb provider (libenf)
- rmem client
- LMCache plugin

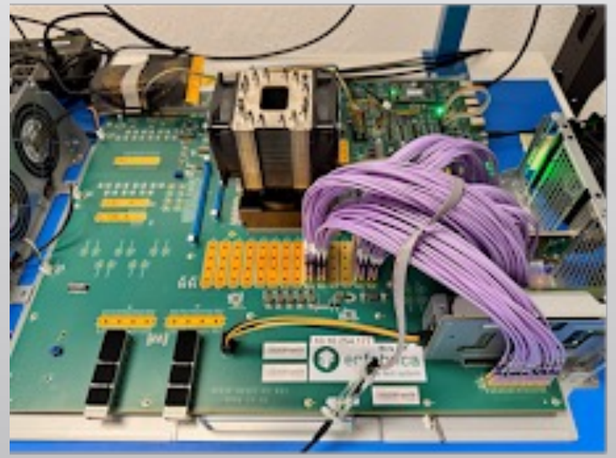
Simple key-value store API

```
rmem_err_t rmem_client_init(ibv_ctx, pd, config, **ctx);
rmem_err_t rmem_client_finalize(ctx);

rmem_err_t rmem_client_pool_open(ctx, size, config, **pool)
rmem_err_t rmem_client_pool_close(ctx, pool)

rmem_err_t rmem_client_put_block(pool,
    const char* key, const enf_sge *src_sge, size_t src_cnt);
rmem_err_t rmem_client_get_block(pool,
    const char* key, const enf_sge *dst_sge, size_t dst_cnt);

rmem_err_t rmem_client_wait(pool);
```

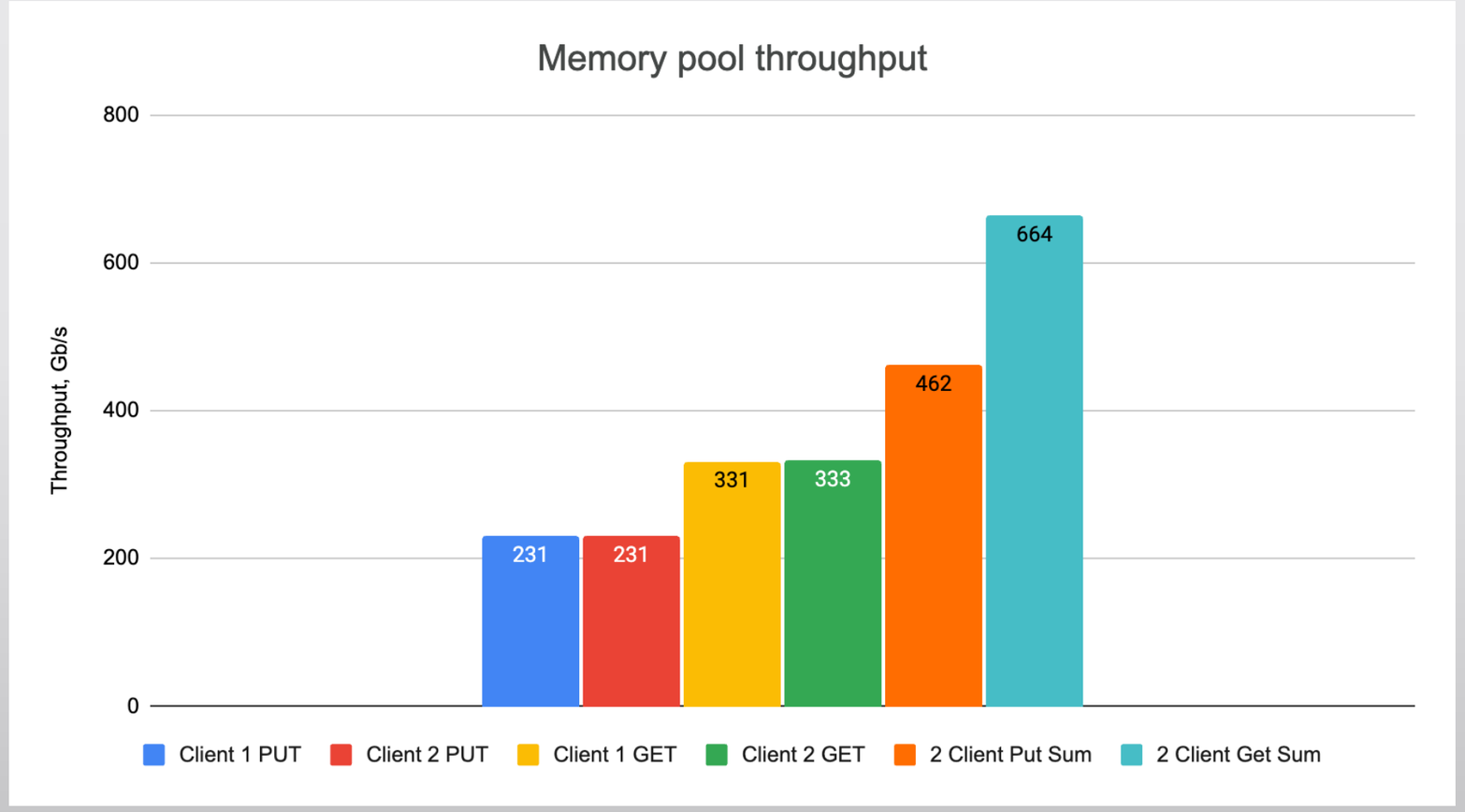
… a few more to query server info, status, etc …



:: let the fun begin!



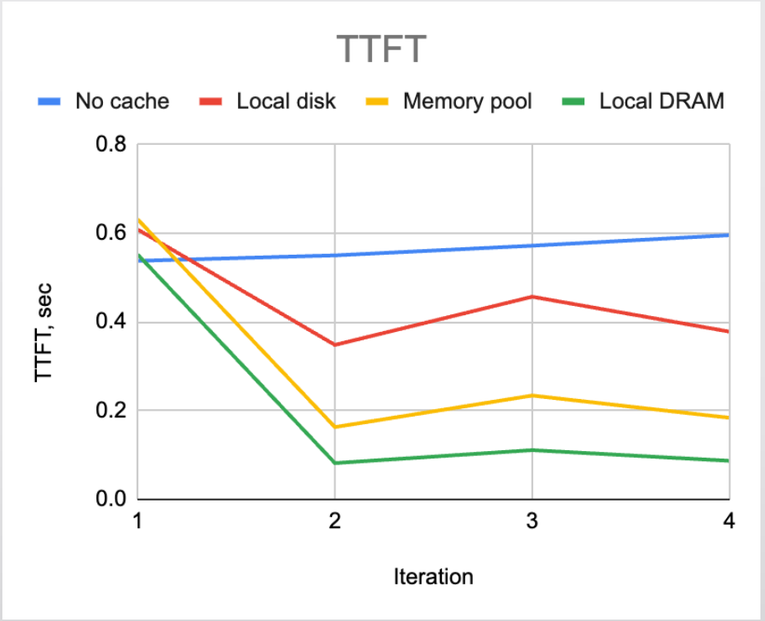
:: memory pool performance
(optimizations ongoing)

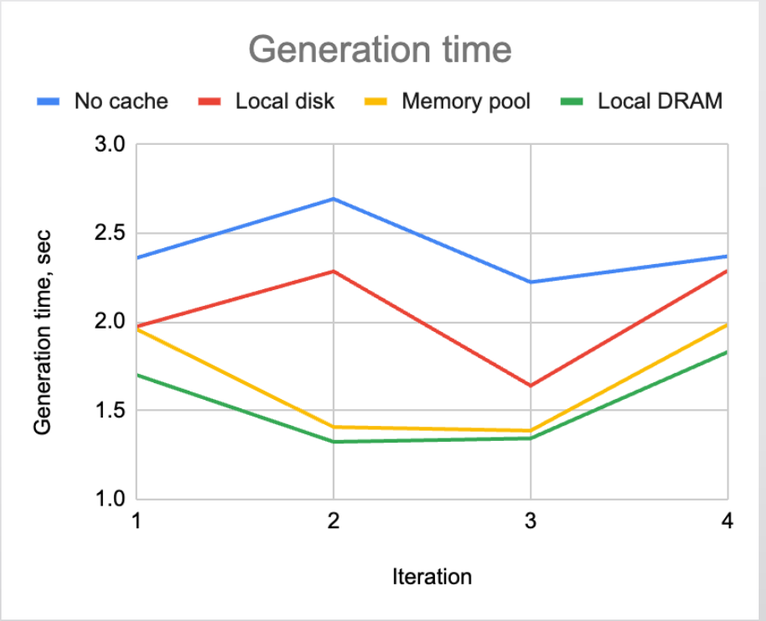


:: inference test results: burst workload

Test conditions

- 1x GPU (H100)
- GPU HBM size: 43GB
- Local DRAM size:
  - 50GB (no L3 tier available)
  - 8GB (disk/memory pool present)
- Memory pool size: 50GB
- Concurrent users: 20
- Input tokens per query: ~5K-8K
- Output tokens per query: 100
- Iterations per user: 3
- Workload: all users issue their queries in parallel



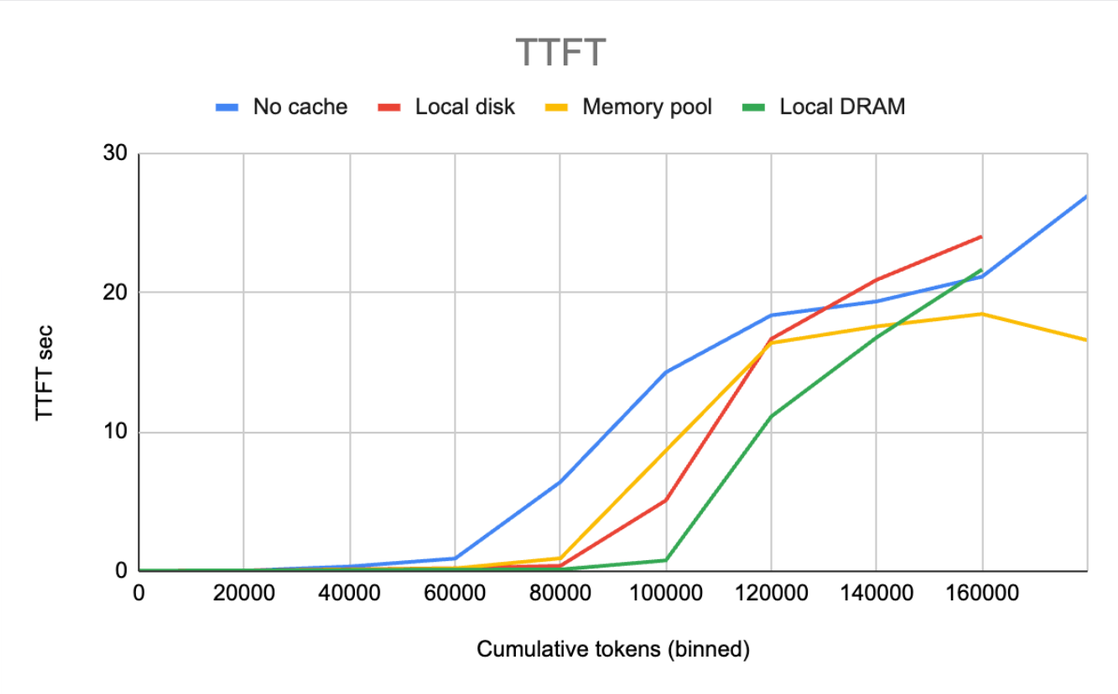


Results

- Without additional memory GPU HBM is too small resulting in KV cache recalculation for every new query since
- Large local DRAM shows best performance (as long as it can contain all generated data)
- Remote memory pool shows much better results than local disk due to its speed

:: inference test results: iterative workload

Test conditions
- 1x GPU (H100)
- GPU HBM size: 43GB
- Local DRAM size:
  - 50GB (no L3 tier available)
  - 8GB (disk/memory pool present)
- Memory pool size: 50GB
- Concurrent users: 10
- Input tokens per query:
  - First prompt ~200
  - Last prompt: ~14K
- Output tokens per query: 600
- Iterations per user: 26
- Workload:
  - All users start at the same time
  - Gradually increase the prompt over 26 iterations



Results
- GPU starts running out of HBM capacity around 60K tokens served
- Remote memory pool outperforms local disk and GPU w/o additional memory after 120K tokens served
- Remote memory pool outperforms local DRAM after 150K tokens served

:: conclusions

Large RDMA accessible memory pool

- Accelerates inference response times
- Decouples KV cache storage from the GPU server
- Allows dynamic GPU allocation anywhere in the cluster (decoupled from server's local DRAM)
- Can be utilized to store other components (model data, encodings, intermediate storage for reinforced learning and more)

Key EMFASYS characteristics

- Network throughput matched with CXL memory bandwidth allows fully non-blocking get/put access
- Low access latency results in much faster response times compared to flash storage
- Simple get/put API over standard RoCEv2 protocol makes it easy to use

:: call for action

Interested in experimenting with Enfabrica Memory Pool solution?

- Want to explore new use cases?
- Compare performance to alternative solutions?
- Collaborate with Enfabrica on application software development/integration?

Contact Boris Shpolyansky

boris@enfabrica.net

:: Thank You