AI&HPC Datacenter Infrastructures

Convergence, Diverge...

Definitions

[i]HPC uses clusters of powerful processors that work in parallel to process multidimensional data sets and solve complex problems at extremely high speeds.

[ii]AI: Computer systems that can perform tasks autonomously, ingesting and analyzing enormous volumes of data, then recognizing patterns in that data.

Characteristics

HPC: Parallelism [Task Division] / Tightly coupled workloads / Ultra-low-latency across the cluster / FP64: double precision / MPI (Message passing interface) / Extremely high read/write storage.

AI: Massive datasets / Parallelism [Gradient sync per step] / High-throughput within scale-up and scale-out / FP16-4: mid/low-precision / RAG during the inference.

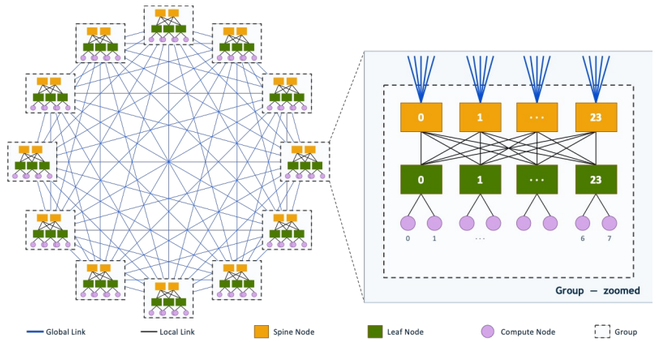
Frontend Networks & Remote Storage Access

AI setups utilize a separate frontend network from backend with shared bandwidth to remote storage. Since data is cached locally on node SSDs, cost-effective design is preferred.

HPC systems require highly uniform, ultra-high-bandwidth network fabrics that allow any node to access the centralized shared storage cluster at peak speeds without bottlenecks. Performance optimized dedicated storage fabrics are preferred.

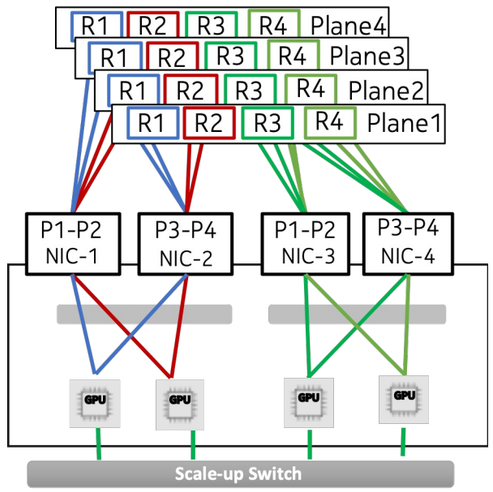
Backend Networks

Existing HPC exascale cluster deployments favor Dragonfly/+ topologies. Given that fat-tree topologies are emerging due to hybrid-purpose aspects. Dragonfly eliminates the network scaling problem by design. Instead of building massive switch trees to handle intra/inter cluster communication, Dragonfly uses a smart, physical layout that maximizes local connections within the groups, and minimizes long-reach optics across the groups. It uses smart adaptive routing software to handle the heavy lifting.



Fat-tree 2-tier architectures already win the race for AI backend network designs. Rail-optimized principles are fundamental for either single or multi plane deployments.

Flattened topologies minimize the hop count for GPU-to-GPU traffic while providing full bisection bandwidth across the fabric.



Oversubscription ratio is a critical design aspect that often differs between AI and HPC backend networks. AI backend fabrics demand a 1:1 (non-blocking) ratio between GPU-facing ports and leaf-to-spine inter-switch links. On the other hand HPC backend network can typically tolerate 2:1 or even 4:1 oversubscription ratios. Hybrid clusters require more balanced approach to provide lose-less and low-latency fabrics.

Storage Strategy & Data Localization

AI (Node-Local SSDs): In AI training, systems spend much more time processing data on GPUs than reading data from storage. Because of this, AI clusters often use fast local NVMe SSDs inside each server instead of relying only on a very fast centralized storage system. Data is copied to these local drives ahead of time, so training can run efficiently without constantly accessing shared network storage.

HPC (Shared Parallel File Systems): Traditional HPC workloads typically sync massive simulation checkpoints and/or read global datasets continuously. They rely heavily on massive, high-bandwidth remote shared file systems to move data quickly across the entire fabric.

Job scheduling

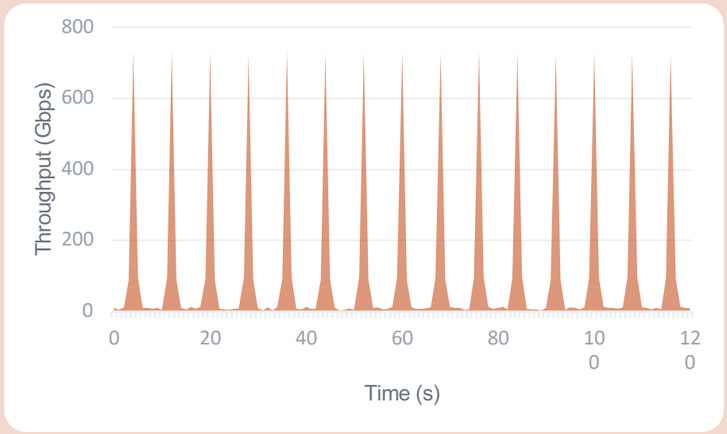
Both AI training and HPC systems often uses slurm due to the bulk-synchronous, batch-style nature of large-scale LLM training in AI and queue management of large-scale parallel simulations in HPC clusters.

Behavioral characteristics and fabric-level effects — AI/ML training & inference, HPC, and hybrid clusters

Network traffic patterns across compute paradigms

1 AI/ML Training & Inference

Microbursts on a backend fabric

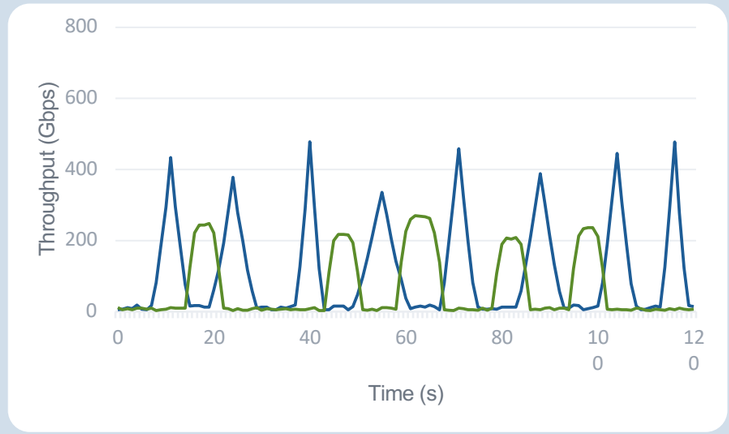


■ CCL AllReduce (NCCL / RCCL)

- Bulk-synchronous on/off: compute-idle fabric alternates with full-rate collective burst.
- Gradient synchronization: AllReduce / AllGather over RDMA (NCCL / RCCL)
- Elephant flows; very high peak
- Network impact: synchronized incast, microburst congestion, tail-latency bound (slowest flow gates the step)

2 HPC clusters (sim.)

Aggregated multi-job traffic; CPU and/or GPU ranks

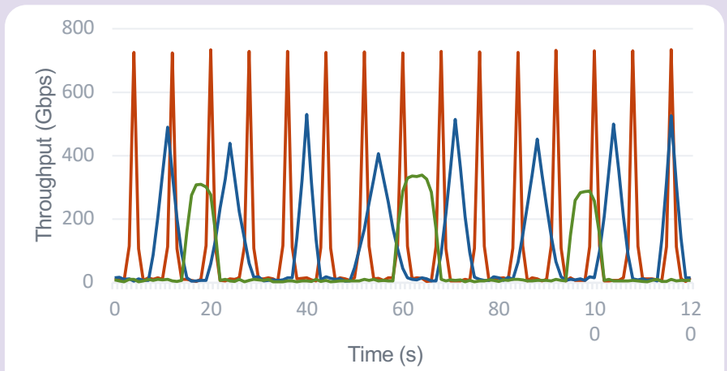


■ MPI collective ■ I/O checkpoint

- Irregular periodic bursts; many concurrent independent jobs on a shared fabric
- MPI collectives at app cadence — CUDA-aware MPI (Inter-GPU) + parallel file system (PFS) I/O
- Many medium flows; small control and bulk halo messages coexist
- Network impact: spine incast, head-of-line blocking, MPI & I/O link sharing.

3 Hybrid AI/ML + HPC

Superimposed CCL + MPI + I/O on one fabric



■ CCL ■ MPI ■ I/O

- Multiple uncorrelated periodicities superimposed; fabric rarely idle.
- CCL training + MPI simulation + I/O sharing one fabric
- Bimodal flow sizes: elephant (CCL, checkpoints) + medium (MPI halos); volatile frame-size mix
- Network impact: cross-job phase collisions → compounded incast and inter-tenant interference

References

Studies

- Understanding Data Movement Patterns in HPC: A NERSC Case Study
- A Study of Network Congestion in Two Supercomputing High-Speed Interconnects
- Characterizing the Impact of Congestion in Modern HPC Interconnects
- Modeling and Analysis of Application Interference on Dragonfly+

Definitions

- https://www.ibm.com/think/topics/hpc
- https://www.nvidia.com/en-eu/glossary/artificial-intelligence/

Publications

- Resilient AI Supercomputer Networking using MRC and SRv6
- Insights into DeepSeek-V3: Scaling Challenges and Reflections on Hardware for AI Architectures
- Introducing Virgo Network, Google's scale-out AI data center fabric

Articles, Studies, White Papers...