POSTER: Taking the Pulse: Diagnosing GPU Collectives from Endpoint Timestamps Alone

Bekmukhamed Tursunbayev
Illinois Institute of Technology
btursunbayev@hawk.illinoistech.edu

Abstract

Diagnosing network degradation in distributed GPU training typically requires switch telemetry or application instrumentation, neither available in multi-tenant clouds. We show that per-packet NIC timestamps, already present at every endpoint, encode algorithm-specific contention signatures. During GPT-2 training on a 4-node NCCL ring spanning two sites, contention on one hop caused a 17% throughput drop indistinguishable from normal variance, while our inter-arrival time entropy analysis detected a 101% spike at the exact affected node. Ring and Tree AllReduce responded with structurally different entropy patterns: Ring's sequential pipeline concentrated the impact at one receiver, while Tree's fan-out distributed it across the root and its children. These results establish that endpoint NIC timestamps are a sufficient and zero-instrumentation diagnostic for network behavior in GPU collectives.

CCS Concepts

• **Networks** → **Network monitoring**; **Network measurement**; • **Computing methodologies** → *Distributed computing methodologies*; *Machine learning*.

Keywords

Collective Communication, NCCL, Inter-Arrival Time, Network Diagnosis, Distributed Training, GPU Clusters, Machine Learning Systems

ACM Reference Format:
Bekmukhamed Tursunbayev. 2026. POSTER: Taking the Pulse: Diagnosing GPU Collectives from Endpoint Timestamps Alone. In *ACM SIGCOMM 2026 Conference (SIGCOMM Posters and Demos '26), August 17–21, 2026, Denver, CO, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3789240.3830290

1 Introduction

Large-scale AI training synchronizes gradients across GPUs every training step. When a network link degrades, training slows, but the operator cannot determine *which* link is responsible from aggregate throughput alone. SkeletonHunter [7] reports 4,816 network failures in one production GPU cluster over six months, each requiring per-hop attribution that training metrics do not provide.

Existing diagnostic tools leave a gap. Application profilers such as NCCL Inspector [8] and Nsight Systems [9] report per-collective bandwidth but not per-hop attribution. Network tools (INT [10], FlowPulse [6]) can attribute degradation to specific links but require network-level visibility that multi-tenant clouds withhold. Recent endpoint systems [2–5, 7, 11] reduce switch dependence but each requires active cooperation: modifying NCCL, instrumenting proxy threads, attaching kernel probes, injecting active probes, or decomposing algorithm internals.

Our approach is entirely out-of-band. Ring and Tree AllReduce use different communication topologies (sequential pipeline vs. fan-out), and this difference produces distinct patterns in per-packet inter-arrival times under contention. We quantify these with Shannon entropy over 5 s windows of binned IATs:

$$H = -\sum_i p_i \log_2 p_i$$

Entropy captures distributional shape changes that mean and percentile statistics miss; the 5 s window balances sample density against temporal resolution. In an ablation, entropy under a CUSUM change-detector fires only at the contended rank with zero false alarms; mean IAT and P99 fire on clean ranks too.

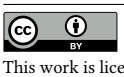
Aspan, a GPU-accelerated passive receiver (27 Mpps, 25 Gbps, zero loss), extracts these patterns from NIC timestamps alone. We deploy it on a 4-node NCCL ring during GPT-2 distributed training and show that identical contention produces algorithm-specific entropy signatures: Ring concentrates a 101% spike at one node, Tree distributes it across two, while training throughput drops stay within normal variance. We trace this difference to the algorithms' communication topologies and validate it across multiple contention levels, transport protocols, and collective operations.

2 Experimental Setup

Aspan captures packets at line rate using strided receive queues and raw completion queue polling, harvesting HCA nanosecond timestamps and computing IAT entropy on GPU without reducing capture throughput. The capture path uses a dedicated receive queue separate from the training data path, adding 2.15% to training step time.

We deploy four nodes on FABRIC [1]: ranks 0–1 at STAR and ranks 2–3 at FIU. Each node has a Tesla T4 GPU and ConnectX-5 25 GbE SmartNIC. The ring traverses two intra-site LAN hops and two inter-site WAN hops (10–30 ms latency). FABRIC exposes no switch telemetry; the HCA timestamp at each node is the sole observable.

The workload is GPT-2 small (124M parameters) trained with PyTorch DDP across all four nodes using the NCCL TCP backend. Each step performs a 498 MB gradient AllReduce. We evaluate both Ring and Tree algorithms with 20 warmup and 100 measured steps, repeated across 3 independent runs. While production clusters typically use RDMA, our supporting experiments show that RoCE and TCP produce distinct IAT signatures (JSD = 0.49) on the same



This work is licensed under a Creative Commons Attribution 4.0 International License.
SIGCOMM Posters and Demos '26, Denver, CO, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2467-1/26/08
https://doi.org/10.1145/3789240.3830290

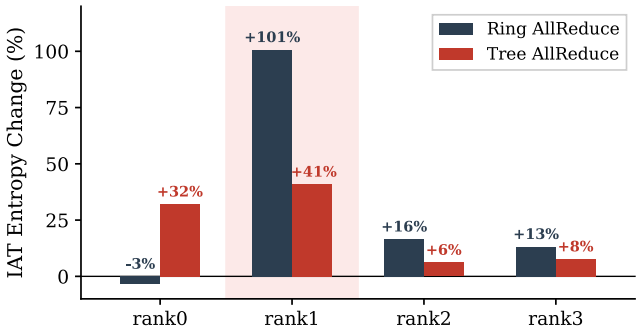


Figure 1: Per-node entropy change under contention (shaded: affected hop).

physical link, confirming that per-packet timing patterns carry transport-level information under both stacks.

To model contention, we inject 50 ms ± 10 ms of delay on both outgoing interfaces of rank0 using `tc netem`. This models the latency increase caused by WAN path degradation or upstream congestion events. Because the delay applies to existing NCCL packets rather than injecting new traffic, the Aspan capture remains uncontaminated.

3 Results

Mean training throughput drops 17% for Ring (2010 to 1675 tokens/sec) and 11% for Tree (1783 to 1591 tokens/sec) under contention. However, individual training steps range from 1427 to 2470 tokens/sec regardless of whether contention is present. An operator monitoring step-level throughput cannot reliably detect the degradation, nor attribute it to a specific hop.

IAT entropy tells a different story. Figure 1 shows per-node entropy changes under the same contention for both algorithms.

For Ring, rank1 entropy doubles (+101%, from 2.15 to 4.31 bits, a 7σ shift). Rank0 stays flat (−3%). Downstream nodes show +13–17% cascade from NCCL pipeline propagation.

For Tree, the pattern differs: rank0 (root) shows +32% and rank1 (child) +41%, with mild cascade at leaves (+6–8%). The difference is topological. Ring moves data sequentially ($0 \to 1 \to 2 \to 3 \to 0$), concentrating contention at one receiver. Tree coordinates through fan-out, distributing the impact. This topological difference creates algorithm-specific fingerprints readable from timestamps alone. Independent replications confirmed these patterns.

These DDP results are part of a broader pattern. On the same testbed, Aspan localizes contention across multiple severity levels (200 Mbps to 2 Gbps) with consistent 43–49% entropy spikes and automatic CUSUM detection in one 5 s window, while NCCL Inspector reports only bandwidth fluctuation (0.085–0.226 GB/s) with no hop attribution. The IAT signal encodes more than location. Transport protocols produce distinct signatures (JSD = 0.49 between RoCE and TCP), and on a two-node RoCE testbed a competing RDMA flow raises endpoint IAT +75% while throughput shifts only +19%, localizing RDMA-path contention rather than TCP alone. The timing rhythm even reveals *which* collective is running: AllReduce, AllGather, and ReduceScatter leave distinct autocorrelation fingerprints that sharpen rather than blur under noise (r: 0.17→0.60). The same capture also reads LLM inference: the per-layer AllReduce of tensor-parallel serving (Llama-3.2-3B) and the all-to-all of MoE expert parallelism (OLMoE-1B-7B), so the method is not specific to training AllReduce. Together, endpoint timestamps answer *where* the bottleneck is, *what* transport is active, *which* collective is running, and *how* contention propagates.

Each diagnostic operates independently at the receiving node, requiring no cross-node coordination. FABRIC's real WAN hops provide a realistic multi-site environment. We are investigating PFC/DCQCN environments and closed-loop algorithm selection where entropy spikes trigger automatic Ring-to-Tree switching.

References

[1] Ilya Baldin, Anita Nikolich, James Griffioen, et al. 2019. FABRIC: A National-Scale Programmable Experimental Network Infrastructure. *IEEE Internet Computing* 23, 6 (2019). doi:10.1109/MIC.2019.2958545

[2] Yuxuan Chen, Menghao Zhang, Xiheng Li, et al. 2025. Vedrfolnir: RDMA Network Performance Anomalies Diagnosis in Collective Communications. In *Proceedings of the ACM SIGCOMM Posters and Demos*. doi:10.1145/3744969.3748396

[3] Ziteng Chen, Xiaohe Hu, Menghao Zhang, et al. 2025. An Efficient, Reliable and Observable Collective Communication Library in Large-scale GPU Training Clusters. *arXiv preprint arXiv:2510.00991* (2025).

[4] Erfan Darzi, Aldo Pareja, and Shreeanant Bharadwaj. 2025. Host-Side Telemetry for Performance Diagnosis in Cloud and HPC GPU Infrastructure. *arXiv preprint arXiv:2510.16946* (2025).

[5] Yangtao Deng, Lei Zhang, Qinlong Wang, et al. 2025. Mycroft: Tracing Dependencies in Collective Communication Towards Reliable LLM Training. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*. doi:10.1145/3731569.3764848

[6] Jakob Krebs, Dimitry Gavrilenko, Daniel Amir, et al. 2025. FlowPulse: Catching Network Failures in ML Clusters. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*. doi:10.1145/3772356.3772384

[7] Wei Liu, Kun Qian, Zhenhua Li, et al. 2025. SkeletonHunter: Diagnosing and Localizing Network Failures in Containerized Large Model Training. In *Proceedings of the ACM SIGCOMM Conference*. doi:10.1145/3718958.3750513

[8] NVIDIA. 2026. NCCL Inspector: Profiler Plugin for NCCL. https://github.com/NVIDIA/nccl/tree/master/plugins/profiler/inspector. Accessed May 2026.

[9] NVIDIA. 2026. Nsight Systems. https://developer.nvidia.com/nsight-systems. Accessed May 2026.

[10] P4.org Applications Working Group. 2020. In-band Network Telemetry (INT) Dataplane Specification. https://p4.org/p4-spec/docs/INT_v2_1.pdf. Version 2.1.

[11] Zhiyi Yao, Pengbo Hu, Congcong Miao, et al. 2025. Holmes: Localizing Irregularities in LLM Training with Mega-scale GPU Clusters. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. doi:10.5555/3767955.3767983