# Understanding and Improving Multi-Rack Symmetric-Memory AllGather in NCCL

Prajwal Singhania*†, Vijay Korthikanti*, Jared Casper*, Mohammad Shoeybi*, Abhinav Bhatele†
*NVIDIA, Inc., Santa Clara, USA †Department of Computer Science, University of Maryland, College Park, USA
E-mail: prajwal@umd.edu

***Abstract*—Rack-scale GPU systems such as the GB200 NVL72 extend the NVLink scale-up domain from four or eight GPUs per server to 72 GPUs per rack. Large language model (LLM) training still scales across many racks, with *all-gathers* spanning heterogeneous NVLink and InfiniBand fabrics. These all-gathers operate in a message-size regime where both latency and bandwidth matter. In this work, we characterize NCCL's symmetric-memory and legacy all-gather performance for this regime. We derive a closed-form topology-aware bandwidth bound and extend it with a latency term for comparison with empirical measurements. Based on observed trends and model insights, we design two new GPU-initiated symmetric-memory all-gather implementations that outperform NCCL's symmetric-memory all-gathers by up to $2.6\times$ for 32–512 MB messages.**

***Index Terms*—collective communication, device-initiated communication, performance modeling**

## I. Introduction and Motivation

Rack-scale systems are an emerging class of GPU clusters that expand the NVLink-connected scale-up domain from four or eight GPUs per server to 72 GPUs per rack. Training large language models (LLMs), however, still scales across many racks. ZeRO-3 [1], a popular parameter-sharding strategy, materializes model parameters on demand via *all-gather* communication that can span multiple racks. Higher scale-up and scale-out bandwidths place these messages in a regime where both latency and bandwidth matter, motivating a study of multi-rack all-gather performance.

Device-initiated communication APIs [2–4] enable GPU kernels to issue one-sided RDMA operations through symmetric-memory registration. NCCL [2] provides *symmetric* all-gather implementations that use these features, whereas its *legacy* host-initiated implementations do not. In this work, we study the scaling performance of both across GB200 NVL72 racks. We derive a closed-form, topology-aware bandwidth bound for all-gather and evaluate the scaling performance of NCCL's all-gather implementations against it. The resulting insights guide new implementations that outperform NCCL's symmetric implementations in the medium-message regime.

## II. Speed-of-Light Performance Model

On homogeneous networks, the measured bus bandwidth of a collective can be compared directly with the link bandwidth. Heterogeneous networks, however, offer no single reference rate. To address this, we build on ForestColl [5], which bounds all-gather time via the bottleneck cut of the network graph $G = (V, E)$, where $V = V_s \cup V_g$ comprises switches and GPUs:

$$T_{comm} \geq \frac{M}{N} \max_{S \subset V,\, V_g \not\subseteq S} \frac{|S \cap V_g|}{B^+(S)}, \tag{1}$$

where $M$ is the aggregate message size, $N$ is the number of GPUs, and $B^+(S)$ is the total bandwidth leaving cut $S$. ForestColl evaluates (1) for arbitrary topologies using binary search and max-flow in $O(N|V|^2|E|(\log|V| + \log b_{\max}))$ time, where $b_{\max} = \max_{e \in E} b_e$ and $b_e$ is the bandwidth of edge $e$.

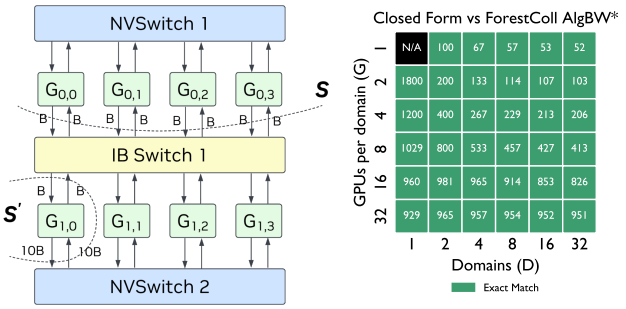



Fig. 1. (Left) The two candidate bottleneck cuts: $S$ (single-domain) and $S'$ (single-GPU), for symmetric two-level topologies. (Right) Validation that the closed-form $AlgBW^*$ bound (2) exactly matches ForestColl's computation.

We simplify the bound to a closed-form for two-level symmetric topologies: $D$ domains, each with $G$ GPUs, with per-GPU NVLink and IB injection bandwidths $B_{NVL} \gg B_{IB}$. The bottleneck cut can be either the single-GPU cut $S'$ or the single-domain cut $S$, as depicted in Figure 1 (left):

$$\text{AlgBW}^* = \min\left(N\,\frac{B_{NVL} + B_{IB}}{N-1},\ \frac{D\,(G \cdot B_{IB})}{D-1}\right). \tag{2}$$

Intuitively, a GPU cannot receive the other $N-1$ shards faster than its $B_{NVL}+B_{IB}$ ingress bandwidth, nor can a domain receive the other $D-1$ domains' shards faster than its $G \cdot B_{IB}$ IB ingress bandwidth. Figure 1 (right) empirically validates the closed-form bound against ForestColl's computation across $(D, G)$ configurations. We omit the full proof for brevity.

### A. A Latency-Aware SOL Model

ForestColl models only steady-state messaging in the bandwidth-bound regime. For any algorithm realized as a sequence of $K$ communication steps, we use the Hockney model [6]. The model adds a latency term to the SOL effective bandwidth:

$$T_{comm} \approx K\alpha + M/(Nx^*), \tag{3}$$

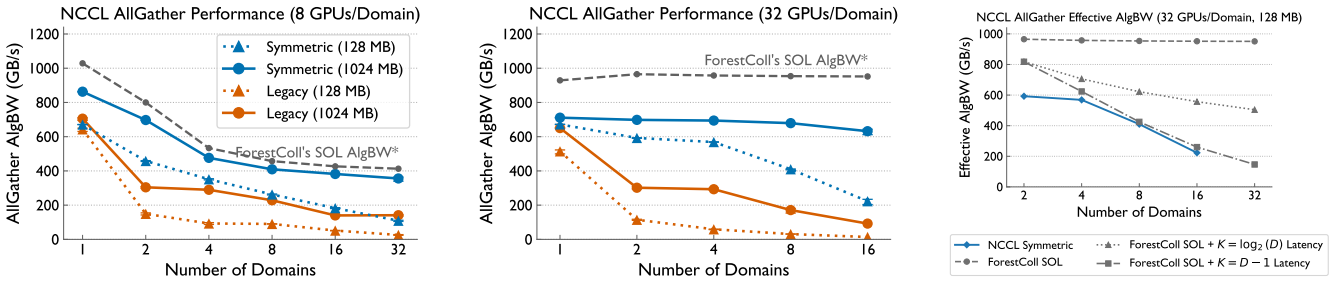



Fig. 2. (Left and Middle) NCCL AllGather bandwidth against the bandwidth-only SOL bound for 128 MB and 1024 MB messages on GB200. (Right) 128 MB NCCL symmetric AllGather bandwidth against the latency-aware model (3): $K{=}D{-}1$ models ring exchange, while $K{=}\log_2 D$ models tree exchange.

where $x^* = \mathrm{AlgBW}^*/N$ is the per-GPU effective bandwidth from (2), $K$ is the number of serialized communication steps on the critical path, and $\alpha$ is the per-step latency. To apply the model to our empirical study, we assume that IB communication latency dominates NVLink latency and set $\alpha = \alpha_{IB}$, the empirically measured one-hop IB latency. Additionally, IB multicast is disabled on our hardware, so $K \neq 1$ and is determined by the algorithm's serialized inter-domain steps.

## III. Benchmarking NCCL Performance

Figure 2 benchmarks NCCL v2.30.6 symmetric-memory and legacy all-gather implementations against our speed-of-light bounds on a GB200 system with CX-7 InfiniBand NICs (50 GB/s per direction). The left and middle panels fix eight and 32 GPUs per domain, respectively, while scaling domains. The former is domain-cut (IB) limited, while the latter is single-GPU-cut (GPU-ingress) limited. At both message sizes, symmetric all-gather significantly outperforms legacy all-gather. At 1024 MB, symmetric all-gather closely tracks the optimal $\mathrm{AlgBW}^*$ bound. At 128 MB, however, it drops sharply beyond four domains at 32 GPUs per domain, a behavior unexplained by the bandwidth-only model (2).

Figure 2 (right) compares 128 MB NCCL symmetric all-gather bandwidth with the latency-aware model (3). We use $K = D - 1$ to model ring exchange and $K = \log_2 D$ for tree exchange. The ring model explains the performance drop-off, consistent with NCCL's rail-ring implementation, which pipelines NVLS multicast. The tree latency model suggests that a lower-latency algorithm could mitigate the drop.

## IV. Improving All-Gather Performance

Based on insights from our analysis in Section III, we design two new high-performance symmetric-memory all-gather kernels.

*a) Rail hybrid recursive-doubling + ring:* As we scale to more domains under strong scaling, the per-GPU message size decreases. We therefore design a hybrid recursive-doubling [7] plus ring exchange kernel, pipelined with intra-domain NVLS multicast. For small messages, recursive-doubling is latency-optimal. However, it transfers progressively larger volumes of data in later steps, making it inefficient for large, bandwidth-bound messages, where ring exchange performs better. In the hybrid approach, we partition the $D$ rail peers into subgroups. Each subgroup first performs several recursive-doubling steps to expand the data per rank, followed by a ring exchange across the subgroups to complete the operation. The number of recursive-doubling steps is a configurable parameter that interpolates $K$ between $\lceil \log_2 D \rceil$ and $D - 1$.

*b) Rail round-robin broadcast:* We also design a one-shot rail-based broadcast kernel in which each rank sends each data chunk directly to all rail peers in round-robin order and pipelines these transfers with intra-domain NVLS multicast. Although IB does not support multicast in our setup, this approach uses NIC-level pipelining to achieve an effective fan-out of $1 \leq K \leq D - 1$.

We implement both approaches as fused kernels using the NCCL Device API.

### A. Performance Results and Conclusion

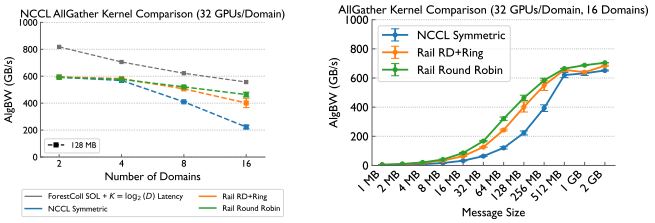



Fig. 3. Benchmarking our new all-gather kernels against NCCL symmetric AllGather at 32 GPUs per domain. (Left) Scaling 128 MB messages across domains. (Right) Message-size sweep at 16 domains.

Figure 3 compares our two kernels with NCCL's symmetric-memory all-gather at 32 GPUs per domain. At 128 MB (left), all three implementations perform similarly up to four domains. Beyond four domains, our kernels track the tree-based latency-aware model more closely, whereas NCCL's ring-based implementation degrades sharply. At 16 domains, rail round-robin achieves 2× NCCL's bandwidth. In the 16-domain message-size sweep (right), both kernels reach high bandwidth at small to medium message sizes and outperform NCCL by up to 2.6× for 32–512 MB messages.

Together, these results demonstrate that jointly modeling bandwidth and communication depth explains NCCL's medium-message drop-off and guides faster multi-rack all-gather kernels. As future work, we plan to extend our model and analysis to reduce-scatter operations.

## ACKNOWLEDGMENT

The authors thank the NCCL team for valuable insights during the development of this work. OpenAI Codex [8] and Anthropic Claude Code [9] were used as coding and writing assistants in kernel development and in proofreading and editing the abstract and poster text. The authors reviewed and take full responsibility for all content of this submission.

## REFERENCES

[1] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, "Zero: Memory optimizations toward training trillion parameter models," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '20. IEEE Press, 2020.

[2] "Nccl," 2020. [Online]. Available: https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/overview.html

[3] K. Hamidouche, J. Bachan, P. Markthub, P.-J. Gootzen, E. Agostini, S. Jeaugey, A. Shafi, G. Theodorakis, and M. G. Venkata, "Gpu-initiated networking for nccl," *arXiv preprint arXiv:2511.15076*, 2025.

[4] NVIDIA, "Nvshmem," https://developer.nvidia.com/nvshmem, 2020.

[5] L. Zhao, S. Maleki, Y. Wang, Z. Wang, Z. Yang, H. Pourreza, and A. Krishnamurthy, "ForestColl: Throughput-Optimal collective communications on heterogeneous network fabrics," in *23rd USENIX Symposium on Networked Systems Design and Implementation (NSDI 26)*. Renton, WA: USENIX Association, May 2026, pp. 2067–2093. [Online]. Available: https://www.usenix.org/conference/nsdi26/presentation/zhao-liangyu

[6] R. W. Hockney, "The communication challenge for mpp: Intel paragon and meiko cs-2," *Parallel Comput.*, vol. 20, no. 3, p. 389–398, Mar. 1994. [Online]. Available: https://doi.org/10.1016/S0167-8191(06)80021-9

[7] R. Thakur, R. Rabenseifner, and W. Gropp, "Optimization of collective communication operations in mpich," *The International Journal of High Performance Computing Applications*, vol. 19, no. 1, pp. 49–66, 2005. [Online]. Available: https://doi.org/10.1177/1094342005051521

[8] OpenAI, "Codex," 2026, accessed 2026-04-07. [Online]. Available: https://openai.com/codex/

[9] Anthropic, "Claude code," 2026, accessed 2026-04-07. [Online]. Available: https://www.anthropic.com/product/claude-code