Completion-Driven, Staging-Free AlltoAllv with Local-NIC Balancing

Emir Gencer*, Abhinav Bhatele*, Sajal Dash†
*Department of Computer Science, University of Maryland, College Park, USA
†Oak Ridge National Laboratory, Oak Ridge, USA
Email: egencer@umd.edu, dashs@ornl.gov, bhatele@cs.umd.edu

***Abstract*—Mixture-of-Experts (MoE) training workloads use imbalanced AlltoAllv collectives at bandwidth-bound message sizes. On Frontier [1], one-sided AlltoAll can beat RCCL and Cray-MPICH, but only with the right schedule and only as per-peer fetches shrink [2]. Other approaches [3] balance load across a node's NICs by explicitly copying data between GPUs, which costs GPU staging buffers and overlaps that copy with network traffic only coarsely. We introduce CURLEW, a completion-driven, GPU-staging-free RDMA AlltoAllv with dynamic local-NIC balancing. CURLEW achieves near-flat bandwidth across GPU counts and message sizes on balanced patterns, is $1.9\times$ faster than previous work [3] at max-to-mean skew above 8, and reduces end-to-end training time of a 201B-parameter model by 22% at a capacity factor of 1.25.**

I. Introduction and Motivation

Mixture of Experts (MoE) is a popular large language model architecture used by many recent models [4], [5], [6]. Training MoE models requires AlltoAll collective communication during training and inference. MoE training workloads use large-message AlltoAllv collectives with highly imbalanced communication patterns. Despite the popularity of MoE architectures, the AlltoAll routines in vendor libraries such as RCCL [7] and Cray-MPICH cannot fully utilize the available network bandwidth in the message-size and communicator-size regimes required by MoE training workloads.

Previous work shows that MPI_Get based AlltoAll can achieve high bandwidth utilization in large message size regimes [2]. However, it is schedule dependent and its performance increases as GPU count increases and per-peer message size gets smaller. Another approach at optimizing MoE workloads with high load imbalance is to use hierarchical implementations. Such implementations explicitly copy data over the intra-node fabric, NVLink [8] or InfinityFabric [9], balancing the node's load across GPUs, followed by an inter-node step [3], [10]. However such approaches require large GPU staging buffers, and can only overlap intra-node communication with inter-node communication coarsely.

We introduce CURLEW, a completion-driven, GPU-staging-free RDMA AlltoAllv with dynamic local-NIC balancing. CURLEW pipelines smaller RDMA operations and issues new operations as previous operations complete in steady state. By using multiple NICs on the imbalanced side, letting each NIC drive its own independent schedule, and exploiting xGMI's hardware routing, CURLEW can dynamically balance NIC loads and allow hardware to overlap xGMI copies with network messages in a fine-grained manner.

II. Background

Mixture of Experts. Mixture of Experts is a popular LLM architecture. MoE architectures replace the large feed-forward network after attention with a number of small expert networks. A router module routes each token's activations to one or more experts. After the expert computation, the activations are collected back.

Dispatch is the first AlltoAllv in the MoE forward pass. The dispatch step distributes activations to experts. Notably, because each GPU initially has the same number of tokens, during dispatch, sends are balanced. However, if some experts are more popular than others, the receiving side can be imbalanced.

Combine is the second AlltoAllv in MoE forward pass. It is the inverse of dispatch.

Skew is a metric used to express the imbalance in the communication matrix. It is defined as the maximum-to-mean ratio of sends and receives.

III. CURLEW Design

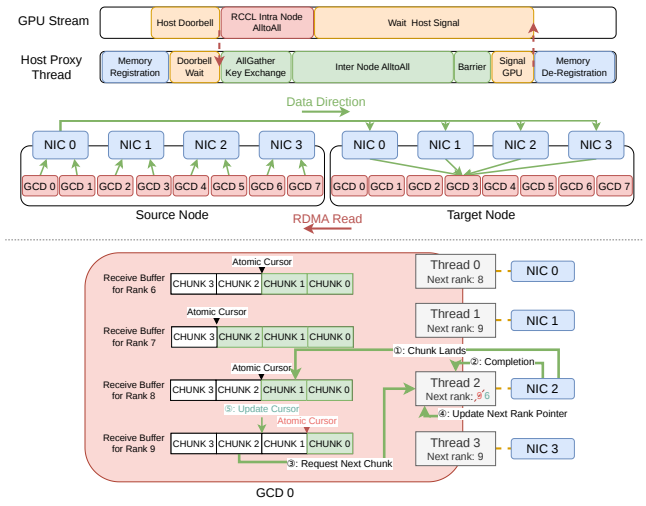


Fig. 1. CURLEW-MR dispatch design, showing how the proxy thread interacts with the GPU stream, and how completion-driven refill works.

Figure 1 shows CURLEW's proxy-thread architecture and completion-driven refill mechanism. CURLEW drives the

control plane in a proxy thread, which spawns one NIC worker thread per local NIC. Each worker drives its NIC independently.

After the GPU stream signals that its buffers are ready, the host exchanges memory region (MR) keys using Allgather. The NIC worker threads then drive the inter-node AlltoAll while the GPU stream concurrently performs the node-local RCCL AlltoAll over xGMI. Once inter-node communication completes, the host executes a Barrier, signals completion to the GPU stream, and deregisters the MRs.

CURLEW divides each remote rank's receive buffer into 16 KiB chunks managed by an atomic cursor. Worker threads start at strided remote ranks, independently claim chunks, and post operations in breadth-first order. As each operation completes, its worker posts another. This completion-driven policy prevents slower NICs from accumulating work and allows GCDs to claim otherwise unused NIC capacity.

For MoE combine, CURLEW uses the same algorithm with RDMA writes. This allows traffic to be striped only across the imbalanced side's NICs.

IV. Experimental Setup

We evaluate CURLEW on Frontier [1], where each node contains eight AMD MI250X GCDs and four Cassini NICs.

For balanced AlltoAll, we compare CURLEW with MPI_Alltoall, MPI_Alltoall_init, MPI_Get-based AlltoAll [2], and RCCL using 64 MiB to 1 GiB receive buffers on 16 to 1024 GCDs. We report 20 timed iterations after five warmups, excluding one-time persistent MPI and MPI_Get initialization but including CURLEW's MR registration and deregistration.

For end-to-end MoE training, we compare RCCL, CURLEW-SR (single rail), and CURLEW-MR (multi rail) on XMoE-Medium and Large [11], using the default scripts and a capacity factor of 1.25.

For highly skewed AlltoAllv, we compare CURLEW, RCCL, and SABRE [3] using matrices sampled every 25 iterations during the first 500 iterations of XMoE-Small training with token dropping disabled.

V. Results

Figure 2 shows that both single-rail and multi-rail CURLEW achieve 90% of peak bandwidth and exhibit near-flat scaling on balanced AlltoAll. Figure 3 shows that CURLEW performs on par with or better than RCCL across a range of message sizes and GCD counts, achieving up to a 2× speedup. Figure 4 shows that CURLEW reduces end-to-end training time by up to 22%. Figure 5 shows that CURLEW is faster than both RCCL (2.5×) and SABRE (1.9×) under highly skewed communication patterns.

VI. Conclusion and Future Work

We introduce CURLEW, a staging-buffer-free, completion-driven AlltoAllv designed primarily for MoE training. CURLEW achieves higher bandwidth utilization than vendor communication libraries and prior academic approaches across a range of use cases.

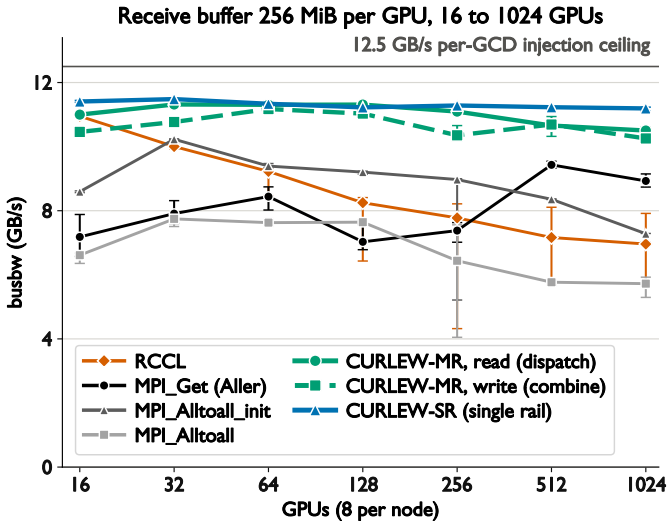


Fig. 2. Comparison of CURLEW with baselines for balanced AlltoAll using a fixed 256 MiB receive buffer.

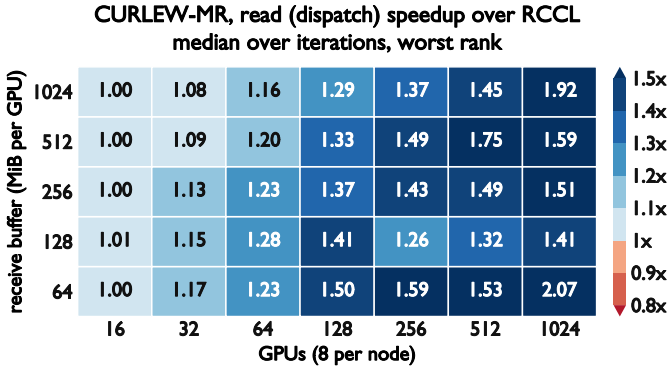


Fig. 3. Comparison of CURLEW with RCCL across message and communicator sizes.

Acknowledgment

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Advanced Scientific Computing Research programs in the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

OpenAI Codex [12] and Anthropic Claude Code [13] were used as coding and writing assistants for proofreading and editing the abstract and poster text. The authors have reviewed and take full responsibility for all content of this submission.

References

[1] O. R. N. L. (ORNL), "Frontier (hpe cray ex) exascale supercomputer at the oak ridge leadership computing facility," 05 2022. [Online]. Available: https://www.osti.gov/biblio/3023606

[2] J. B. White, "Large-message all-to-all communication at frontier scale," in *Proceedings of the SC '25 Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC Workshops '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 461–467. [Online]. Available: https://doi.org/10.1145/3731599.3767389

[3] C. Wei and A. Bhatele, "Skew-aware adaptive all-to-allv algorithms for dynamic deep learning workloads," in *Proceedings of the 40th ACM International Conference on Supercomputing*, ser. ICS '26. New York,

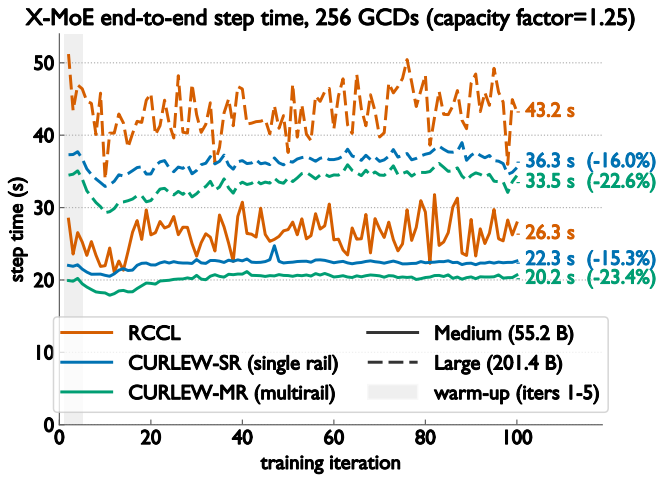


Fig. 4. Comparison of CURLEW with RCCL on end-to-end MoE training.

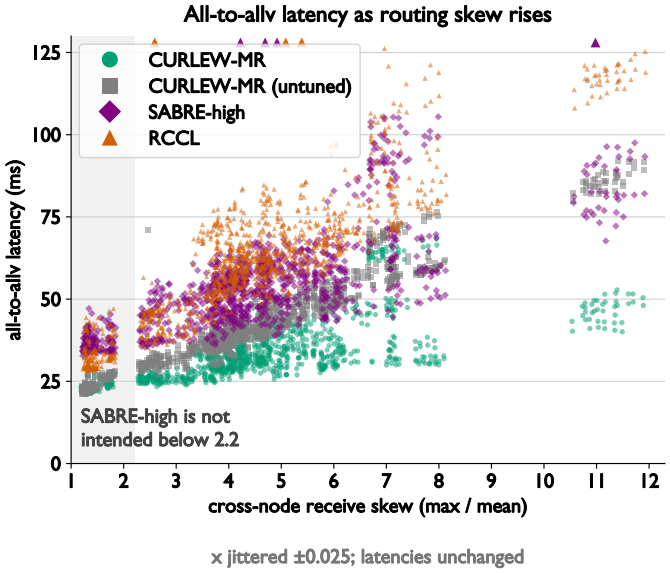


Fig. 5. Comparison of CURLEW with RCCL and SABRE on highly skewed AlltoAllv.

NY, USA: Association for Computing Machinery, 2026, p. 450–462. [Online]. Available: https://doi.org/10.1145/3797905.3800541

[4] J. Parmar, S. Prabhumoye, J. Jennings, M. Patwary, S. Subramanian, D. Su, C. Zhu, D. Narayanan, A. Jhunjhunwala, A. Dattagupta, V. Jawa, J. Liu, A. Mahabaleshwarkar, O. Nitski, A. Brundyn, J. Maki, M. Martinez, J. You, J. Kamalu, P. LeGresley, D. Fridman, J. Casper, A. Aithal, O. Kuchaiev, M. Shoeybi, J. Cohen, and B. Catanzaro, "Nemotron-4 15b technical report," 2024.

[5] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[6] K. Team, Y. Bai, Y. Bao, G. Chen, J. Chen, N. Chen, R. Chen, Y. Chen, Y. Chen, Y. Chen *et al.*, "Kimi k2: Open agentic intelligence," *arXiv preprint arXiv:2507.20534*, 2025.

[7] "Rccl," 2025. [Online]. Available: https://github.com/ROCm/rccl

[8] NVIDIA, "Nvlink and nvswitch: The building blocks of advanced multi-gpu communication," https://www.nvidia.com/en-us/data-center/nvlink/.

[9] G. Schieffer, R. Shi, S. Markidis, A. Herten, J. Faj, and I. Peng, "Understanding data movement in amd multi-gpu systems with infinity fabric," 2024. [Online]. Available: https://arxiv.org/abs/2410.00801

[10] C.-C. Chen, J. Yao, L. Xu, H. Subramoni, and D. K. Panda, "Unified designs of multi-rail-aware mpi allreduce and alltoall operations across diverse gpu and interconnect systems," in *2025 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2025, pp. 938–949.

[11] Y. Yuan, A. Gupta, J. Li, S. Dash, F. Wang, and M. Zhang, "X-moe: Enabling scalable training for emerging mixture-of-experts architectures on hpc platforms," 2025. [Online]. Available: https://arxiv.org/abs/2508.13337

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

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