ET: Bridging the Gap on Energy Telemetry for Multi-GPU Communication Collectives

Brandon Tran*, Matthias Maiterth†, Woong Shin‡, Matthew D. Sinclair*, Shivaram Venkataraman*

*Computer Sciences Dept., University of Wisconsin-Madison, Email: bqtran2@wisc.edu, {sinclair, shivaram}@cs.wisc.edu

†NVIDIA, Email: mmaiterth@nvidia.com ‡Oak Ridge National Laboratory, Email: shinw@ornl.gov

***Abstract*—To meet growing high performance computing (HPC) demands, applications increasingly concurrently leverage many Graphics Processing Units (GPUs). As a result, HPC systems are now power-limited, requiring a more precise understanding of their energy consumption. One source of power consumption comes from communication over the interconnect used to coordinate data across multiple GPUs. The energy spent transferring data between GPUs has become increasingly important and is growing. However, to date, there is no way to accurately measure it. In the absence of counters to measure network energy consumption, users rely on models to estimate it. Unfortunately, precise models that accurately attribute energy to collective communication have yet to be developed.**

Accordingly, we explore using component-level measurements and network profiling to develop an energy-per-byte metric for estimating energy consumption for GPU collective communication. Our case studies explore the use of this metric across different GPU architectures and message sizes. Based on our observations, we propose a new approach, ET, that enables us to understand collective communication energy consumption and optimize application energy consumption, thereby improving system efficiency.

I. Introduction

As machine learning (ML) and high-performance computing (HPC) applications continue to evolve, the use of multiple GPUs has become the norm to meet computational demand. This is necessary due to the increasing sizes of models and datasets, which require more GPUs to effectively parallelize the work. However, parallelizing the work across multiple GPUs requires additional communication to coordinate data across the devices. To perform this coordination, developers use libraries such as NCCL that provide collective communication among GPUs [16].

While many tools have been developed to understand the power consumption of a single GPU [2], [4], [9], [10], [22], little attention has been paid to the power consumption of inter-GPU communication. However, understanding the energy cost of data movement is vital as high-speed, power-hungry communication links become prevalent [1], [7]. In Figure 1, we ran the `AllReduce` NCCL collective across three generations of NVIDIA GPUs and sampled the power while the collective was running. We find that running collectives on the newer GH200 GPU can draw up to 370W, corresponding to approximately 41% of the GPU's TDP. Moreover, the power consumed by the network is increasing with each generation, as prior work predicted [18].

Thus, understanding the significant power draw of collective communication is important. Accordingly, we devised a methodology to compute the energy per byte for each collective, and investigated how this metric varies across architectures and message sizes. We also examined whether bandwidth can serve as a simple proxy for energy consumption. We find that different collective communications can differ by up to $3.8\times$ in energy per byte, with significant differences across topology and message sizes that are not captured by simple bandwidth-driven proxies. Therefore, we propose ET, which models energy at a finer-grained level, enabling users to understand the energy consumption of their collective communication and supporting more energy-efficiency-aware optimizations.

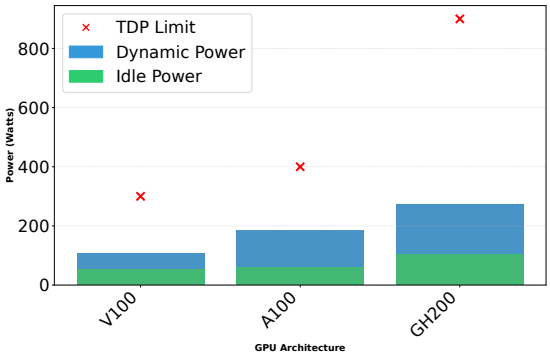

Fig. 1: Idle power and dynamic power when running the `AllReduce` NCCL collective across three generations of NVIDIA GPUs.

II. Design

To derive an energy-per-byte metric for different collective communications, we sample GPUs to obtain power traces (discussed further in Section III). We integrate these power traces over time to obtain the total GPU energy consumed during collective communication.Next, to determine the dynamic component of the collective communication, we break the total energy into an idle[1] and a dynamic component.

$$E_{total} = E_{idle} + E_{dynamic} \quad (1)$$

Idle energy is the product of the idle power and the application's execution time. This allows us to subtract the idle energy from the total energy, leaving us with the dynamic energy when the GPU is executing a collective.

$$E_{dynamic} = E_{total} - (P_{idle} \times t_{exec}) \quad (2)$$

[1]We assume static power is included in idle power.

TABLE I: Summary of studied GPUs and HPC clusters.

Cluster	GPU	# GPUs per Node	NVLinks
Cloudlab [5]	V100	4	2
Leconte [17]	V100	6	2
Delta [12]	A100	4	4
Delta AI [13]	GH200	4	7

From there, we can derive the dynamic energy per byte by dividing by the total number of bytes that were sent during the experiment.

$$\text{Energy per Byte} = \frac{E_{total} - (P_{idle} \times t_{exec})}{total\ bytes} \tag{3}$$

The total number of bytes can be computed as the product of the number of iterations and the collective message size.

$$\text{Energy per Byte} = \frac{E_{total} - (P_{idle} \times t_{exec})}{N_{iterations} \times \text{Size}_{bytes}} \tag{4}$$

Overall, this approach allows us to isolate the energy spent communicating between GPUs.

III. Methodology

Evaluated Systems: Table I provides details on the four GPU clusters we evaluate this design across, each containing V100, A100, and GH200 GPUs. V100, we include two topologies: one with 4 GPUs, similar to the A100 and GH200 studied, and another with 6 GPUs. In particular, we focus on measuring GPU power within a single node, as monitoring energy consumption across nodes is not readily available.

Evaluated Applications: For our experiments, we use NVIDIA's NCCL collectives' test suite [15], which includes `AllGather`, `AllReduce`, `AlltoAll`, `Reduce`, and `ReduceScatter`. For our initial experiment, we fix the message size to 1GB. We choose 1GB because it is near the peak usage in ML training [23]. To obtain the energy of the entire collective, we sum the energy contributions from all participating GPUs. In our experiments, the variance [3], [6], [19], [20] among GPUs participating in collective communication is approximately `1%`, except for `Reduce`, where rank 0 consumes `8–13%` more energy than the other GPUs.

To mitigate noise, we run experiments with different numbers of iterations. This allows us to construct a least-squares line of best fit for energy as a function of the number of iterations, yielding the energy per iteration. Since all of our experiments have a fixed message size, we can amortize the energy per iteration to determine energy per byte.

Power Measurements: To profile power, we use NVIDIA Management Library (NVML), which runs concurrently in the background as GPUs execute each collective [14]. As NVML measures the power of the GPU and associated circuitry, it also captures all dynamic power used in intra-node collectives. This limits our current methodology to the intra-node level until additional profiling tools are incorporated. Moreover, since NVML samples the GPU every 1-2 ms granularity [24], for very short-running kernels, the energy value we compute may not reflect the kernel's actual power consumption. Thus, we tune the number of iterations so that each run is at least 30 seconds, ensuring sufficient steady-state power samples for integration.

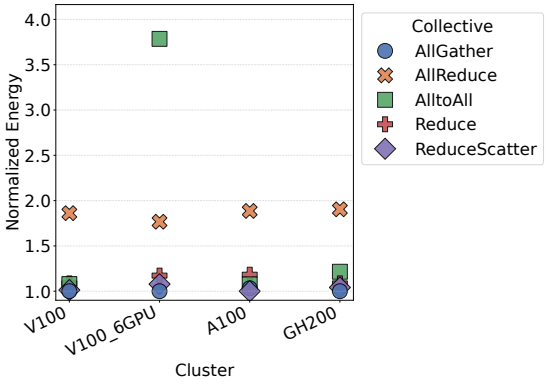

Fig. 2: Energy per byte of five collectives across different architectures normalized to the energy of the most energy efficient collective of that cluster.

IV. Motivating Case Studies

To understand the impact of our energy-per-byte metric, we examine its variation across architectures, topologies, and message sizes. We find that the collective communication energy can vary across these factors. Thus, linearly scaling energy per byte is unlikely to accurately model the network's energy consumption. Furthermore, we show that other metrics, such as bandwidth, are poor proxies for energy-per-byte. Overall, these results demonstrate that a more complex approach (Section V) is needed to effectively measure network power consumption in multi-GPU systems.

A. Energy per Byte Across Architectures and Topology

Figure 1 showed that dynamic power increases with newer GPU generations. Thus, we examine how the energy-per-byte metric changes across generations. Figure 2 shows the energy per byte between the different collectives normalized against the most energy efficient collective for that architecture. We find that collectives can vary in energy consumption across architectures. For example, on the V100 and GH200, the `Reduce` collective only consumes ≈8% more energy than the most energy efficient collective. However, on the A100, `Reduce` consumes 18% more energy than the most energy-efficient collective, indicating that the relative cost of communication collectives can vary with architecture.

The number of GPUs participating in the collective can also play a major role in the energy per byte. For example, `AlltoAll` is roughly 3.8× more energy-intensive than the most energy-efficient collective for the V100_6GPU node, but on other nodes, it is at most only 1.2× more energy-intensive than the most energy-efficient collective. Thus, a collective that is expensive on one hardware system may become cheaper on another, thereby presenting additional opportunities to reduce energy consumption.

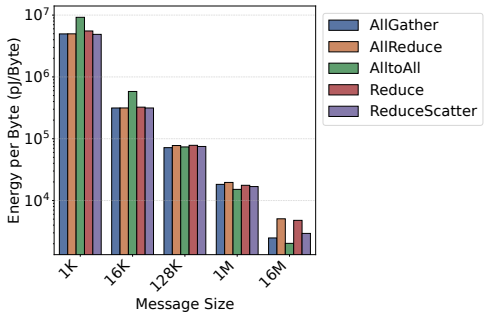

Fig. 3: Calculated energy per byte for various NCCL collectives at different message sizes from 1KB to 16MB on the GH200 node. The y-axis is on a log scale.

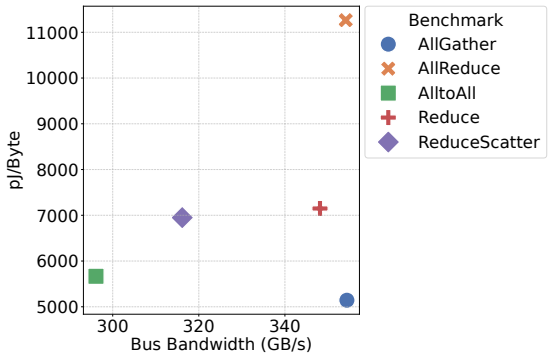

Fig. 4: Calculated energy per byte compared with measured in-place bus bandwidth on the GH200 node.

B. Energy per Byte with Varying Message Sizes

Next, we explore whether changing the message size also changes the energy per byte. To do this, we vary the message size from 1KB to 16MB. Figure 3 shows the results of this experiment on the GH200 node, with the values presented in log scale. Overall, varying message sizes significantly affect energy per byte. More importantly, this difference is not consistent across collectives. For example, the `AlltoAll` collective is the most expensive at 1KB, but as we increase the message size to 16MB, it becomes one of the most energy-efficient. This suggests that optimizing energy for a specific message size may not be the most efficient choice for other message sizes. It also highlights how determining the energy consumption of collectives requires more information than simply linearly scaling energy per byte.

C. Bandwidth

Alternatively, to measure GPU energy during collective communication, users may consider whether bandwidth can serve as a simple proxy for energy. However, Figure 4 shows there is no clear correlation between bandwidth and energy per collective. Although `Reduce` and `ReduceScatter` consume similar energy, `ReduceScatter`'s calculated bandwidth is 9% lower than `Reduce`'s. Although higher bandwidth may finish the collective faster, it comes at the cost of higher dynamic power, which may partially counteract the energy savings from faster completion. Thus, bandwidth is not a useful proxy for energy.

V. Proposal

Overall, Section IV demonstrates significant promise, as our energy-per-byte metric shows that collective communication can vary across architectures and message sizes in ways that require supplemental information to explain. Moreover, using bandwidth-driven metrics as a proxy for energy is also insufficient. Thus, a new modeling framework is required that is robust to hardware and collective nuances. To this end, we propose ET for measuring energy at finer granularity, supported by additional network parameters, which is applicable across a wider range of use cases.

A. Collective Primitives

Sections IV-A and IV-B showed that energy per byte is sensitive to different collective and message sizes. Thus, we propose finer-grained energy attribution. Prior work has shown that collectives can be broken down into primitives such as `send` and `recv` [8]. However, these primitives omit differences that may arise from the implementation of these collectives. For example, NCCL provides three protocols: Simple, LL, and LL128, that perform the same collective algorithmically, but make several changes to trade off latency and bandwidth that are not inherently captured at the primitive level. To address this, we consider further decomposing the primitives (**ET**). In particular, we propose to extend our Wattchmen GPU energy model [21] to provide instruction-level collective communication energy attribution. In turn, this allows us to ensure ET is more robust to changes in algorithm, implementation, and message sizes. Furthermore, by modeling at the instruction level, we can account for additional, future collective communication patterns (e.g., from new collectives).

B. Supporting Network Parameters

As noted in Section IV-A, differing topologies can play a major role in the energy of collective communication. Thus, ET must also consider a range of parameters, such as message size and system topology, to accurately capture the nuances we identified. For collective communication that includes computations, such as `Reduce`, we will also consider which mathematical operation(s) are performed. Furthermore, while bandwidth itself may not serve as a proxy for energy, prior work [11] has shown that bandwidth can vary across interconnect links. Thus, this information could improve the accuracy of ET. However, only by cross-validating the impact of specific parameters for a given technology can we assess the need to include them.

C. Use Cases

We propose using our robust framework to explore interesting use cases, including providing profiling support, comparing energy efficiency across different communication libraries, and guiding portability across different hardware platforms.

1) Application Profiling: By profiling and detecting collective communication, our framework can also ascribe energy consumption in more complex applications. For example, LLM training often overlaps with computation and communication. Profilers, such as NVML, sample power across the entire GPU and its associated circuitry. This means that they cannot separately attribute power to concurrently running compute and communication kernels. However, since ET models fine-grained per-instruction energy, we can attribute the energy to compute and communication. Thus, this approach yields a richer energy profile that helps users optimize workloads.

2) Collective Libraries: We can also use ET to evaluate the energy consumption of various communication libraries. While our experiments focused on the most commonly used NCCL library, there are other communication libraries [25]. These libraries often significantly reduce communication latency and/or increase throughput compared to NCCL, prompting consideration of energy-performance trade-offs when selecting a library implementation.

3) Portability: As new systems become available, code must often be ported to them. However, in Section IV-A we showed that collective communication varies across architectures. ET can easily identify the most energy-efficient collectives for a given system. This can help explain changes in energy consumption, which often occur when running applications on different systems. Overall, our proposed approach provides a guided path to tune an application for energy efficiency when porting across systems.

VI. Conclusion

Overall, there is an increasing need for a detailed understanding of data movement energy consumption. Our experiments with an energy-per-byte metric perform a preliminary study across collectives, topologies, message sizes, and architectures. Thus, we propose developing a framework, ET, to model energy at finer-grained primitives, thereby properly accounting for these differences. While we currently focus on single-node collective communication, ET can serve as a foundation for understanding multi-node interconnect energy consumption. Moreover, ET can also help developers understand and optimize their workload's energy consumption.

VII. Acknowledgments

This research used resources of the Oak Ridge Leadership Computing Facility and the Experimental Computing Laboratory (ExCL) at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research used both the Delta and DeltaAI advanced computing and data resource, which is supported by the National Science Foundation (award OAC 2005572 and award OAC 2320345) and the State of Illinois through allocation CIS251202 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. This work is supported by NSF grant CNS-2312688 and NSF CIRC-2450448.

References

[1] D. Abts *et al.*, "Energy Proportional Datacenter Networks," in *Proceedings of the International Symposium on Computer Architecture*, ser. ISCA, 2010, pp. 338–347.

[2] Y. Arafa *et al.*, "Verified Instruction-level Energy Consumption Measurement for NVIDIA GPUs," in *Proceedings of the 17th ACM International Conference on Computing Frontiers*, ser. CF, 2020, p. 60–70.

[3] N. DeBardeleben *et al.*, "GPU Behavior on a Large HPC Cluster," in *Euro-Par 2013: Parallel Processing Workshops. Revised Selected Papers*, ser. Lecture Notes in Computer Science, D. an Mey *et al.*, Eds., vol. 8374, 2013, pp. 680–689.

[4] P. Delestrac *et al.*, "Analyzing GPU Energy Consumption in Data Movement and Storage," in *IEEE 35th International Conference on Application-specific Systems, Architectures and Processors*, ser. ASAP. Los Alamitos, CA, USA: IEEE Computer Society, 7 2024, pp. 143–151.

[5] D. Duplyakin *et al.*, "The Design and Operation of Cloudlab," in *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference*, ser. USENIX ATC, 2019, p. 1–14.

[6] F. Fraternali *et al.*, "Quantifying the Impact of Variability and Heterogeneity on the Energy Efficiency for a Next-Generation Ultra-Green Supercomputer," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 7, pp. 1575–1588, 2018.

[7] M. Horowitz, "Computing's Energy Problem and What We Can Do About It," Plenary at ISSCC, 2014.

[8] Z. Hu *et al.*, "Demystifying nccl: An in-depth analysis of gpu communication protocols and algorithms," 08 2025, pp. 48–59.

[9] V. Kandiah *et al.*, "AccelWattch: A Power Modeling Framework for Modern GPUs," in *Proceedings of the 54th IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO. New York, NY, USA: Association for Computing Machinery, 10 2021, p. 738–753.

[10] J. Leng *et al.*, "GPUWattch: enabling energy optimizations in GPGPUs," in *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ser. ISCA, 2013, p. 487–498.

[11] A. Li *et al.*, "Evaluating Modern GPU Interconnect: PCIe, NVLink, NV-SLI, NVSwitch and GPUDirect," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 1, pp. 94–110, 2020.

[12] National Center for Supercomputing Applications, "NCSA Delta," 2026.

[13] National Center for Supercomputing Applications, "NCSA DeltaAi," 2026.

[14] NVIDIA Corporation, "NVIDIA Management Library (NVML)."

[15] NVIDIA Corporation, "NCCL Tests," 2025.

[16] NVIDIA Corporation, "NVIDIA Collective Communications Library (NCCL)," 2025.

[17] Oak Ridge National Laboratory, "ExCL LeConte," 2026.

[18] S. Pati *et al.*, "Tale of Two Cs: Computation vs Communication Scaling for Future Transformers on Future Hardware," in *IEEE International Symposium on Workload Characterization*, ser. IISWC, 10 2023.

[19] P. Sinha *et al.*, "Not All GPUs Are Created Equal: Characterizing Variability in Large-Scale, Accelerator-Rich Systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC. IEEE Press, 2022.

[20] J. Tan and K. Yan, "Efficiently Managing the Impact of Hardware Variability on GPUs' Streaming Processors," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 24, no. 1, dec 2018.

[21] B. Tran *et al.*, "Wattchmen: Watching the Wattchers – High Fidelity, Flexible GPU Energy Modeling," in *Proceedings of the ACM International Conference on Supercomputing*, ser. ICS, 2026.

[22] G. Wu *et al.*, "GPGPU Performance and Power Estimation Using Machine Learning," in *IEEE 21st International Symposium on High Performance Computer Architecture*, ser. HPCA, 2015, pp. 564–576.

[23] G. Xu *et al.*, "AutoCCL: Automated Collective Communication Tuning for Accelerating Distributed and Parallel DNN Training," in *Proceedings of the 22nd USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI. USA: USENIX Association, 2025.

[24] Z. Yang, K. Adamek, and W. Armour, "Accurate and convenient energy measurements for gpus: A detailed study of nvidia gpu's built-in power sensor," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*, ser. SC '24. IEEE Press, 2024.

[25] Y. Zhou *et al.*, "An Extensible Software Transport Layer for GPU Networking," *arXiv preprint arXiv:2504.17307*, 2025.