Hybrid Simulator Infrastructure to Explore Scalable, Large-Scale Distributed Training Architectures

Gyeonggeun Jung
School of Electrical Engineering
KAIST
Daejeon, South Korea
gyeonggeun@kaist.ac.kr

Hans Kasan
School of Electrical Engineering
KAIST
Daejeon, South Korea
hanskasan@kaist.ac.kr

John Kim
School of Electrical Engineering
KAIST
Daejeon, South Korea
jjk12@kaist.edu

***Abstract*—The growing scale of distributed training highlights the need for accurate simulation of interconnects and collective communication. Existing simulators provide useful abstractions but often rely on analytical models for the interconnection networks, limiting their ability to capture cycle-level details such as congestion and contention that can occur within the network. In this paper, we discuss simulation framework that are commonly used in prior work and their limitations. We then discuss a hybrid simulator infrastructure that combines a cycle-accurate interconnection network simulator with a system level (MPI-driving) runtime of SST/Macro framework. This integration enables realistic evaluation of not only large-scale physical topologies but also logical topologies that describe the communication algorithm and preserve collective communication semantics required for training workloads. Although compute and memory operations are modeled statically, the framework is specialized for network-centric studies, offering valuable insights into contention, topology scaling, and collective performance in large-scale distributed training systems. In addition, the framework can be expanded to include more detailed compute model of the endpoints.**

***Index Terms*—Interconnection Network, Distributed Training, Simulation, Collective Communication.**

I. Introduction

The rapid growth of large-scale distributed training has intensified the need for accurate and scalable simulation frameworks that can capture the communication bottlenecks of modern interconnection networks. Existing tools [1]–[5] offer valuable insights into parallelization strategies and communication overheads. However, they rely largely on analytical models or high-level abstractions, which fail to capture cycle-level congestion dynamics and contention behavior in realistic interconnects. Conversely, cycle-accurate network simulators such as BookSim 2.0 [6] provide fine-grained fidelity in modeling router pipelines, buffer occupancy, and flow control, yet they lack the ability to enforce application-level semantics or execute MPI-based collectives required in distributed training workloads.

To bridge this gap, we propose **BookSim+SST**, a hybrid simulator that integrates the cycle-accurate fidelity of BookSim 2.0 [6] with the MPI-driven runtime coordination of SST/Macro [7]. This integration allows collective communication to be simulated with both application-level correctness and network-level detail. Specifically, SST/Macro [7] executes MPI traces or codes, enforces synchronization barriers, and orchestrates collectives, while BookSim [6] handles packet-level injection into arbitrary topologies with realistic contention modeling. Together, the framework enables cycle-accurate simulation of distributed training networks, providing insights into congestion, topology choice, and collective scaling behavior that are otherwise inaccessible in purely analytical tools.

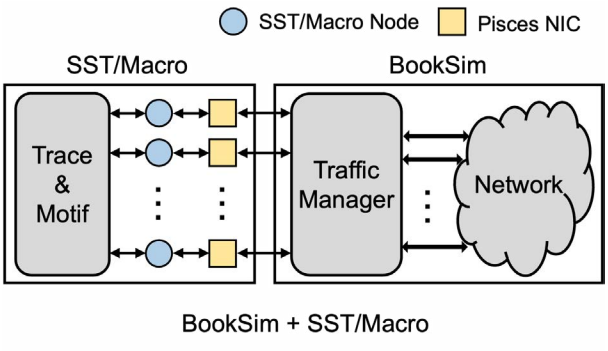

Fig. 1. High-level overview of integrating BookSim and SST/Macro

II. Simulator Comparison

In this section, we compare five recent simulators focusing specifically on how they model communication and network behavior in distributed training. Since the accuracy and scalability of these tools largely depend on their underlying network abstraction, we highlight the strengths and limitations of each approach. Table I summarizes the comparison across dimensions such as simulator scope, network modeling fidelity, workload input granularity, interconnect topology, supported parallelism schemes, and scalability.

ASTRA-sim 2.0 [1] extends the original ASTRA-sim by introducing a graph-based execution engine and hierarchical topology support. From the networking perspective, it provides analytical models for collective communication and can represent complex topologies such as multi-dimensional meshes and wafer-scale fabrics. However, it relies on simplified bandwidth-latency parameters rather than cycle-accurate simulation, which means it cannot fully capture congestion dynamics or contention effects in large clusters.

TABLE I
COMPARISON FOR VARIOUS DISTRIBUTED TRAINING SIMULATORS

Simulator	Simulation Scope	Network Modeling	Input	Interconnect Topology	Parallelism Comm. Support	Scalability
Astra-sim2.0 [1]	xPU + Memory	Latency/Bandwidth model	Operator-level	Hierarchical, Configurable	DP, TP, PP (extensible)	~1000
AMPed [2]	xPU-only	Latency/Bandwidth model	System & parallelism configs	Fat-Tree Topology	DP, TP, PP, MoE	~1000
Calculon [3]	xPU + Memory	Latency/Bandwidth model	Layer-level	Fat-Tree Topology	DP, TP, PP	~1000
vTrain [4]	xPU-only	Profiling-based Latency/Bandwidth model	Operator-level	Fat-Tree Topology	DP, TP, PP, 3D	~100
Echo [5]	xPU-only	Profiling-based	Operator-level	Fat-Tree Topology	DP, TP, PP, 3D	~100
BookSim+SST (Ours)	xPU + Memory	Cycle-accurate	Operator-level	Hierarchical, Configurable	DP, TP, PP, 3D, MoE	~100

AMPeD [2] adopts an analytical approach with tunable parameters for data, tensor, pipeline, and expert parallelism. Its network model is primarily based on high-level communication cost formulas that approximate collective overheads. While this allows rapid exploration of different parallelization strategies, it treats the interconnect as a black box and lacks detailed modeling of routing, contention, or topology-aware scheduling, limiting its usefulness for network micro-architecture studies.

Calculon [3] is designed for high-level co-design of LLM systems and employs parameterized analytical models to capture training efficiency. In terms of networking, it abstracts communication through simplified estimates of data movement cost between processors. This enables large-scale design space exploration but does not incorporate topology-specific effects or fine-grained collective behavior. As a result, it is better suited for identifying global efficiency cliffs rather than analyzing detailed network bottlenecks.

vTrain [4] focuses on cost-effective and compute-optimal training planning, with a profiling-driven framework that produces deterministic DAGs for training workloads. Its network modeling is relatively lightweight, relying on simplified abstractions of collective operations rather than detailed packet-level or topology-aware simulations. This makes vTrain efficient for exploring trade-offs in training time and monetary cost but less capable of capturing network-level contention or the effects of specific interconnect designs.

Echo [5] takes a hybrid approach that explicitly addresses network fidelity. Instead of relying solely on analytical models, Echo implements white-box modeling of collective communication kernels, including intra- and inter-server transmission phases, and augments it with profiling data from NCCL. Furthermore, it introduces an ML-based predictor to capture slowdowns when computation and communication overlap, which is often overlooked in other simulators. This makes Echo significantly more accurate in modeling real-world communication bottlenecks, though at the cost of increased complexity and dependence on empirical profiling.

III. HYBRID SCALABLE INTERCONNECT SIMULATOR

BookSim 2.0 [6] is a cycle-accurate interconnection-network simulator that models router pipelines, virtual channels, buffer occupancy, and credit-based flow control while allowing researchers to instantiate virtually any topology (e.g., mesh/torus, fat-tree/Clos, dragonfly, custom graphs). Its fidelity makes it ideal for studying contention hot spots, path diversity, and micro-architectural router choices. However, BookSim operates purely at the network layer—it neither executes MPI codes nor provides global coordination semantics. In particular, collective communication requires knowing when every participant has completed its local compute and is ready to enter (and exit) each phase; BookSim by itself cannot enforce these application-level barriers. **SST/Macro** [7] complements this by providing a scalable discrete-event HPC runtime that executes MPI programs, orchestrates collectives and synchronization, and emits communication events with application-level timing. Its built-in network models, however, are coarser than BookSim's and cannot expose cycle-level congestion behavior.

We therefore propose **BookSim+SST**, a combined simulator that integrates SST/Macro [7] with BookSim 2.0 [6] to run MPI-driven, cycle-accurate network simulations. SST/Macro executes the MPI code (or its trace), advances virtual time through compute phases, and determines when ranks enter each collective or point-to-point phase; an adapter then translates those messages into BookSim packet streams injected into the chosen topology. BookSim returns cycle-accurate latencies and throughput under realistic contention, and SST/Macro uses these completions to drive the application forward, thus preserving collective semantics while maintaining network-level fidelity. This design inherits two clear advantages: (i) the ability to model arbitrary topologies and router micro-architecture in detail, and (ii) correct handling of collectives and global barriers that real training jobs rely on. Its limitations are the flip side of its specialization: accurate studies require MPI traces (or runnable codes) that describe the training job's communication, and compute/memory are repre-

sented with static or analytical timing rather than fine-grained device models. As such, the framework is purpose-built for network-centric analysis of distributed training—revealing congestion and topology effects that analytical simulators miss—while deferring detailed compute/memory modeling to future extensions.

IV. Summary

In this work, we presented BookSim+SST, a hybrid simulator purpose-built for network-centric evaluation of distributed training. By combining SST/Macro's MPI runtime and collective orchestration with BookSim's cycle-accurate interconnection network modeling, our framework offers both application-level correctness and micro-architectural fidelity. This dual advantage makes it uniquely suited to investigate congestion hotspots, topology-aware performance trade-offs, and collective scaling in distributed training workloads.

References

[1] W. Won, T. Heo, S. Rashidi, S. Sridharan, S. Srinivasan and T. Krishna, "ASTRA-sim2.0: Modeling Hierarchical Networks and Disaggregated Systems for Large-model Training at Scale," 2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Raleigh, NC, USA, 2023, pp. 283-294, doi: 10.1109/ISPASS57527.2023.00035.

[2] D. Moolchandani, J. Kundu, F. Ruelens, P. Vrancx, T. Evenblij and M. Perumkunnil, "AMPeD: An Analytical Model for Performance in Distributed Training of Transformers," 2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Raleigh, NC, USA, 2023, pp. 306-315, doi: 10.1109/ISPASS57527.2023.00037.

[3] Mikhail Isaev, Nic Mcdonald, Larry Dennison, and Richard Vuduc. 2023. Calculon: a methodology and tool for high-level co-design of systems and large language models. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'23). Association for Computing Machinery, New York, NY, USA, Article 71, 1–14. https://doi.org/10.1145/3581784.3607102

[4] J. Bang, Y. Choi, M. Kim, Y. Kim and M. Rhu, "vTrain: A Simulation Framework for Evaluating Cost-Effective and Compute-Optimal Large Language Model Training," 2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO), Austin, TX, USA, 2024, pp. 153-167, doi: 10.1109/MICRO61859.2024.00021

[5] Y. Feng, Y. Chen, K. Chen, J. Li, T. Wu, P. Cheng, C. Wu, W. Wang, T.-Y. Ho and H. Xu, "Echo: Simulating Distributed Training At Scale," arXiv preprint arXiv:2412.12487, 2024. Available: https://arxiv.org/abs/2412.12487

[6] Nan Jiang et al., "A detailed and flexible cycle-accurate Network-on-Chip simulator," 2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Austin, TX, USA, 2013, pp. 86-96, doi: 10.1109/ISPASS.2013.6557149.

[7] G. Hendry, J. J. Wilke, B. A. Allan, K. Sargsyan, M. Drohmann, and J. P. Kenny, SST/macro: A coarse-grained simulation workflow. Livermore, CA: Sandia National Laboratories, Tech. Rep. SAND2013-5422C, 2013.

[8] S. Rashidi, S. Sridharan, S. Srinivasan and T. Krishna, "ASTRA-SIM: Enabling SW/HW Co-Design Exploration for Distributed DL Training Platforms," 2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Boston, MA, USA, 2020, pp. 81-92, doi: 10.1109/ISPASS48437.2020.00018.