Deep Learning Recommendation Model Training Co-design with the Dynamic Opera Network

Connor Imes*, Andrew Rittenbach*, Peng Xie*, Dong In D. Kang*, John Paul Walters*, Stephen P. Crago*†
*Information Sciences Institute, University of Southern California, USA
†Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, USA
Email: {cimes, arittenb, pengxie, dkang, jwalters, crago}@isi.edu

***Abstract*—Training deep learning recommendation models (DLRMs) is increasingly dominated by all-to-all and many-to-many communication patterns. Current solutions often involve designing and implementing fully-connected—and costly—high-speed interconnects. The recently proposed dynamic Opera network optimizes bulk data flows using direct forwarding through time-varying circuits and has been shown to be particularly useful for all-to-all traffic patterns while remaining cost-equivalent with static network topologies. We propose co-designing DLRM models with the Opera network to improve training time while matching network infrastructure cost with a traditional fat-tree topology. We simulate strong-scaling DLRM training on Opera networks up to 1024 nodes, identify shifting bottlenecks, and suggest where co-designers should focus their efforts.**

***Index Terms*—machine learning, deep learning, recommendation models, networks, dynamic networks**

I. Introduction

Deep learning recommendation models (DLRMs) are commonly used by large organizations like Amazon [7], Facebook [9], Google [1], and Netflix [2] to personalize recommendations for users. For example, Facebook states that DLRMs have the largest infrastructure demand of their AI applications, and they propose a software/hardware co-design strategy using a fully-connected dedicated GPU network to better handle the growing all-to-all and many-to-many communication workloads in training these models. While highly performant, such networks may become cost-prohibitive as they scale.

The dynamic Opera network was recently proposed to improve bandwidth efficiency for bulk data transfers while maintaining cost equivalence with traditional static network topologies like fat-tree [8]. Opera reduces the bandwidth tax incurred by multiple network hops by introducing a circuit-switched topology that reconfigures a small number of links at a time so that every pair of network endpoints periodically have direct links. Multi-hop links are always available for traffic requiring low latency, i.e., traffic that cannot wait for a direct link to become available. The simplest use case is for the network to wait for direct links to deliver traffic above a size threshold (e.g., 15 MB) and to suffer the bandwidth tax for smaller messages by using the always-available multi-hop paths.

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0000851. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

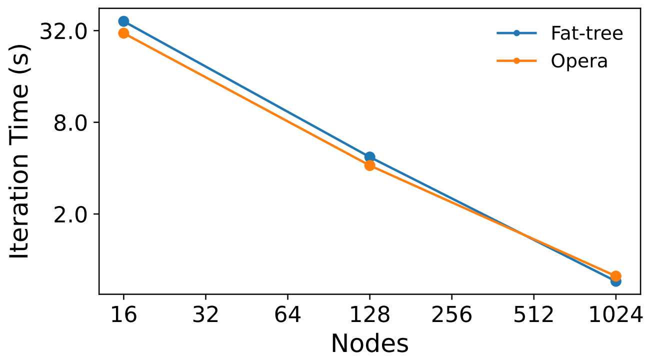

Fig. 1: Strong-scaling DLRM training from 16 to 1024 nodes on fat-tree and Opera networks.

We demonstrate the potential benefits of using Opera networks to train DLRMs by comparing with equivalent fat-tree networks. Our DLRM model is based on Facebook's `Model-A`, consisting of two 20-layer MLPs with 3375 weights per layer each, 1000 embedding tables, and a sparse feature size of 80, resulting in a model with roughly 800B total parameters [9]. Whereas they present results at 16 nodes, we simulate scaling from 16 to 1024 nodes. We discuss challenges in strong-scaling the training workload and how a good co-design might overcome these challenges to maintain performance benefits. Using performance measurements on NVIDIA A100 GPUs, we model DLRM component behaviors and data exchange patterns, then generate task placement graphs using FlexFlow [4]. We simulate training performance on fat-tree and Opera networks using the htsim packet simulator [3]. Our compute system model uses 8 GPUs per compute node, 200 Gbps inter-GPU bandwidth within a node, and 100 Gbps optical network between nodes.

We compare standard 1:1 fat-tree clusters [5] of 16, 128, and 1024 nodes with equal-size Opera networks. To this end, we simulate the following Opera configurations: a 16-node cluster with four top-of-rack (ToR) switches and four nodes per switch, a 128-node cluster with 32 ToR switches and four nodes per switch, and a 1024-node cluster with 128 ToR switches and eight nodes per switch.

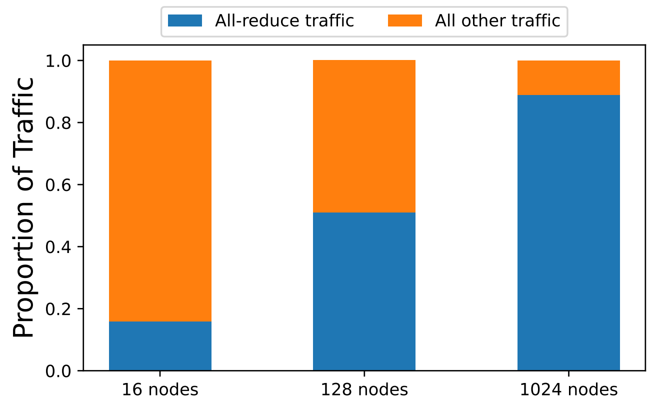

Fig. 2: Changing traffic patterns as we strong-scale the DLRM training workload. All-reduce dominates at larger scales.

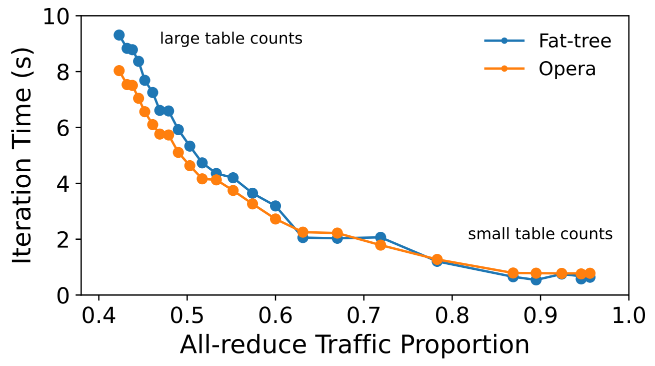

Fig. 3: Varying the embedding table counts at 128 nodes affects training iteration time and all-reduce traffic patterns.

II. Results

Figure 1 presents the simulated training iteration time when strong-scaling the DLRM training process. Note the log axes. Opera achieves $1.20\times$ speedup over fat-tree at 16 nodes. As we scale out, the benefit reduces to $1.14\times$ at 128 nodes, then drops below fat-tree to $0.93\times$ at 1024 nodes.

Figure 2 shows that as we strong-scale the workload, the proportion of all-reduce traffic increases. This is because the DLRM model's MLP components are replicated on each node, but the number of embedding tables remains fixed—each is located on a single node using table-wise parallelism [9]. The strong-scaling challenge is thus that all-to-all traffic no longer dominates. With fat-tree, other collective operations can efficiently overlap communication and computation and thus would not benefit from Opera's intermittently available direct links. We use a ring all-reduce algorithm in the backward pass with fat-tree, which is bandwidth-optimal for tree networks [10], and a distributed parameter server with Opera [6].

Figure 3 explores the impact of embedding table counts on training iteration times for fat-tree and Opera with 128 nodes. Viewed right to left, the four rightmost points are at 10, 25, 50, and 75 tables, then we scale at 100-table increments from 100 to 2000. At small table counts, network traffic is dominated by all-reduce during the backward pass as other model components update their weights, which constitute the bulk of the model. Since Opera is not beneficial for all-reduce traffic, fat-tree tends to have slightly shorter training times. At larger embedding table counts, embedding tables constitute a larger portion of the total model size, and thus more traffic (and more time) is spent in all-to-all exchanges during the training forward pass. Therefore, Opera improves the training time over fat-tree by waiting for direct links and avoiding the multi-hop bandwidth tax.

III. Conclusion

Training DLRMs with a dynamic Opera network can provide performance benefits over static fat-tree topologies. However, network traffic becomes increasingly dominated by all-reduce as we strong-scale a training workload, eventually negating the benefits of using Opera. These results suggest that careful co-design of a DLRM model and the Opera network is necessary to achieve the best results. Particular attention must be paid to all-to-all traffic, which Opera performs well on but is strongly affected by the number of embedding tables when using table-wise parallelism. Our ongoing work is exploring other parameters that impact these traffic patterns, such as feature sizes and training batch sizes.

Acknowledgments

The authors acknowledge the Center for Advanced Research Computing (CARC) at the University of Southern California for providing computing resources that have contributed to the research results reported within this publication. URL: https://carc.usc.edu.

References

[1] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16, 2016, p. 191–198.
[2] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 4, dec 2016.
[3] Ht-sim, "The htsim simulator," https://github.com/nets-cs-pub-ro/NDP/wiki/NDP-Simulator.
[4] Z. Jia, M. Zaharia, and A. Aiken, "Beyond data and model parallelism for deep neural networks," in *Proceedings of Machine Learning and Systems*, vol. 1, 2019, pp. 1–13.
[5] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Transactions on Computers*, vol. C-34, no. 10, pp. 892–901, 1985.
[6] M. Li *et al.*, "Scaling distributed machine learning with the parameter server," in *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'14, 2014, p. 583–598.
[7] R. Lopez, I. S. Dhillon, and M. Jordan, "Learning from extreme bandit feedback," in *AAAI 2021*, 2021.
[8] W. M. Mellette, R. Das, Y. Guo, R. McGuinness, A. C. Snoeren, and G. Porter, "Expanding across time to deliver bandwidth efficiency and low latency," in *Proceedings of the 17th Usenix Conference on Networked Systems Design and Implementation*, ser. NSDI'20, 2020, p. 1–18.
[9] D. Mudigere *et al.*, "Software-hardware co-design for fast and scalable training of deep learning recommendation models," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, ser. ISCA '22, 2022, p. 993–1011.
[10] P. Patarasuk and X. Yuan, "Bandwidth optimal all-reduce algorithms for clusters of workstations," *J. Parallel Distrib. Comput.*, vol. 69, no. 2, p. 117–124, feb 2009.