Emerging Optical Interconnects for AI Systems

Manya Ghobadi
Massachusetts Institute of Technology
ghobadi@csail.mit.edu

Abstract: The ever-growing demand for accurate machine learning models resulted in an increase in dataset and model sizes of deep neural networks. This paper discusses reconfigurable optical networks as the key enabler for scaling AI systems.

1. Introduction

Large-scale AI systems are the foundation of modern online services. As the world is recovering from COVID-19, there is a vital reliance on online services powered by AI. However, today's networks are struggling to deliver high bandwidth, low end-to-end latency, and high availability requirements imposed by emerging AI workloads. For instance, the explosive growth of Machine Learning (ML) applications has created an enormous demand for distributed training. Hardware accelerators, such as GPUs and TPUs, have provided a significant amount of speed up in computing capabilities, but today's deep neural networks (DNNs) can still take days and even weeks to train.

Many factors impact the training time of large DNN jobs, including the parallelization strategy, model/data size, software libraries, and the interconnection network. As a result, a plethora of frameworks have been proposed to efficiently distribute and train DNN models in today's datacenters [1–4]. However, today's systems tend to only optimize *computation* and *communication* dimensions. Consequently, the impact of *co-optimizing the network topology* together with computation and communication dimensions for accelerating DNN training has been largely ignored. This paper argues for reconfiguring the network topology as an additional acceleration dimension to jointly optimize DNN training jobs across computation, communication, and topology dimensions.

Reconfiguring the network topology for datacenter traffic is a popular topic in networking and optics communities. Several academic papers demonstrated the benefits of optically reconfigurable circuit switch-based interconnects for datacenter workloads [5, 6]. However, prior work only considered using optical interconnects for general-purpose datacenter traffic, such as web search, storage, and cloud. Instead of focusing on general-purpose datacenter workloads, this paper turns its attention towards distributed ML workloads and argues that reconfigurable optical interconnects are an attractive solution to build the next generation of ML datacenters. To this end, there are three challenges that need to be addressed.

- **Challenge 1: Technology tradeoffs.** Are Fat-tree topologies the ideal interconnect for distributed DNN training clusters? Which optical technology is best suited for distributed DNN training workloads? What are the cost and practicality implications of using optical vs. electrical technologies? (§2)
- **Challenge 2: Algorithmic innovations.** Given a DNN model and its dataset, how can we co-optimize between the best parallelization strategy, routing, and network topology? (§3)
- **Challenge 3: Scale and heterogeneity considerations.** How to handle shared datacenters with heterogeneous jobs each with different model/data sizes and bandwidth requirements? Is a hybrid electrical/optical solution required to enable practical deployment with heterogeneous jobs? What are the techniques to keep the clusters robust to unforeseen circumstances, such as failures and other operational challenges? (§4)

2. Fundamental Tradeoffs between Technologies

Today's DNN training systems are built using traditional datacenter clusters with electrical packet switches arranged in a multi-tier Fat-tree topology. Fat-tree topologies, by design, work well for datacenters because the interconnect is *traffic oblivious*, allowing uniform bandwidth and latency between server pairs. However, traffic oblivious topologies are best suited for unpredictable workloads that consist mostly of short transfers–two inherent properties of legacy datacenter traffic. GPU-based training jobs have fundamental differences from legacy datacenter traffic. For instance, the communication pattern between DNN workers in GPU clusters remains unchanged across training iterations for the entire training duration, which can last hours, if not days. Hence, Fat-tree topologies are not the ideal interconnect for distributed DNN clusters.

Optical circuit switching is a powerful technology to achieve reconfigurability in datacenters. To be competitive with electrical packet-switching technologies, an ideal optical circuit switch (OCS) should have: (*i*) high port-count; (*ii*) low reconfiguration latency; (*iii*) low insertion loss; and (*iv*) low cost. Meeting all these requirements simultaneously is challenging. In particular, today's optical technologies bear a fundamental tradeoff between port-count and reconfiguration latency. Over the past decade, reconfigurable datacenter proposals have moved towards reducing the reconfiguration latency and, consequently, giving up on high port count. This shift makes sense for legacy datacenter workloads since they require a tight bound on the reconfiguration latency of circuits.

But there is an opportunity that has not been leveraged before: to build the next-generation GPU clusters for DNN training, we can build optical interconnects with high reconfiguration latency enabling us to manufacture high-port count optical switches. In other words, because DNN training has a predictable traffic pattern, we can *plan* the circuit schedules such that each circuit is held for a long time (e.g., several hours).

One attractive design point is to only reconfigure the network topology once at the beginning of the training job instead of reconfiguring the network topology within each training iteration. As an example, consider a patch panel-based interconnect. Today, 1000-port patch panels with 0.5 dB insertion loss are already commercially available [7]. Because the reconfiguration latency of patch panels is in the order of minutes, we need to reconfigure the connectivity between servers participating in a DNN training only once, before training starts. This technique is called one-shot reconfiguration [8].

One-shot reconfigurability for DNN jobs is more practical than intra-iteration reconfigurability. In one-shot reconfigurable clusters, the datacenter operator finds the best topology at the same time as the parallelization strategy for each job. It then reconfigures the connectivity between servers associated with the job, installs routing and forwarding rules, and assigns DNN computations to GPUs. Each job's topology is kept intact during job training. Since the parallelization strategy remains the same, there is no need for reconfiguring the topology during training.

A reconfigurable ML cluster is a *shardable* interconnect where each job has its own dedicated partition. The size of each partition depends on the number of servers that the job requests. The following section describes how we can find the best parallelization strategy and topology for a given DNN training job and a set of servers.

3. Co-optimizing DNN Parallelization Strategy, Routing, and Network Topology

Unlike legacy datacenter workloads, a key feature of DNN workloads is that their communication matrix is *controllable* based on the parallelization strategy that places data and computation tasks on devices. This insight creates a new angle that has not been previously explored for DNN systems: "can we accelerate DNN training by making topology reconfigurability a *joint* decision as an optimization dimension together with routing and parallelization strategy?"

To find the best parallelization strategy, routing, and topology together, an extreme approach is to jointly optimize compute, communication, and topology dimensions using a cross-layer optimization function. But even for an off-line framework, the size of the search space is extremely large, making it practically impossible to solve a cross-dimension optimization formulation. The other extreme is to optimize the network topology, routing, and parallelization strategy, sequentially (one after the other). While this approach can find good routing paths and reconfigure the network topology to better match the traffic demand, the eventual combination of topology and parallelization strategy is sub-optimal in the global configuration space.

Many potential approaches sit between the above two extremes. For instance, SiP-ML [8] took the degree of each GPU (number of silicon photonics I/O ports on each GPU) as a parameter and fed it into the parallelization strategy algorithm. This way, the parallelization strategy was informed of the degree limitation and attempted to place DNN tasks while respecting the degree requirement of the GPUs. SiP-ML's approach largely overcame concerns such as limited communication degree and reconfigurability of optical circuit-switched networks for ML workloads and showed that its silicon photonics-based DNN cluster improves the training time of DNN jobs by up to $9\times$.

Another potential approach is to divide the search space into two planes: $Computation \times Communication$ and $Communication \times Topology$ and use an alternating optimization technique to iteratively search in one plane while keeping the result of the other plane constant. At each step in the process, the $Communication \times Topology$ plane receives a parallelization strategy from the other plane and finds the best topology and circuit schedules depending on the reconfiguration latency of the optical technology. This property adds an extra layer of complexity to finding a topology, thus making prior approaches for optically reconfigurable topologies infeasible. In particular, given a traffic demand matrix, prior papers [9, 10] find a *series* of matchings combined with a circuit hold-time to satisfy the demand using direct circuits. In contrast, depending on the reconfiguration latency, we need to find circuits that can satisfy two competing requirements: (*i*) prefer to have multiple parallel links between nodes with large transfer demands; and (*ii*) minimize the latency of indirect routing for nodes that do not have a direct link between them.

TopoOpt [11] meets both of the above goals by leveraging a unique property of distributed DNN training traffic, namely that the allreduce part of the traffic matrix is *mutable* and can be split across multiple permutations. Intuitively, this is because allreduce transfers contain network flows among nodes that handle the same part of the DNN model, providing flexibility on the order of nodes participating in the allreduce operation. Consequently, if a group of servers is connected in a certain order, simply permuting the labeling of the servers gives another order that would finish the allreduce operation with the same latency while potentially providing a smaller hop-count for other latency-sensitive transfers. TopoOpt's approach builds a series of *allreduce permutations* that not only carry large transfers efficiently, but also are well-positioned to carry Model Parallel transfers and, hence, improve the overall training performance.

4. Handling Heterogeneity, Scale, and Failures

While reconfigurable optical interconnects have many attractive properties, building a DNN interconnect based on them is not without its challenges. An important practical challenge is how to handle large-scale interconnects with a heterogeneous set of jobs, failures, and maintenance events, while keeping the operations simple and practical?

A popular technique in the research community to handle heterogeneity is to keep the electrical interconnect as the basis of communication and augment it with additional optical connections. This approach has been proposed in several prior papers [10, 12] for general-purpose datacenter traffic. However, recent research [13, 14] has shown several cost-saving benefits when the interconnect is all-optical. Which of these approaches is more efficient for DNN training clusters is currently an open problem.

To start a job in a cluster serving multiple jobs simultaneously, the datacenter operator needs to reconfigure the interconnection without disturbing other jobs. Depending on the reconfiguration latency, we need to make sure GPU servers do not stay idle when the links are being reconfigured. For instance, patch panels take several minutes to reconfigure. To avoid keeping the GPUs idle while part of the interconnect is being reconfigured for a newly arrived job, one potential design point is to use a look-ahead approach to pre-provision the next topology while current jobs are running [11].

Finally, a straightforward approach to scale the size of the cluster is to create hierarchical interconnects by placing the servers under Top-of-Rack switches and connecting the ToR switches to the optical layer. Another option is to build a Clos topology using a hierarchy of patch panels. This design point is possible because patch panels have only 0.5 dB insertion loss [7]. Note that even a single layer design using 1000-port patch panels is attractive: each server has 8 GPUs, hence the cluster can accommodate 8000 GPUs.

5. Conclusion

The design of today's AI infrastructure still follows the telephony model where the datacenter operators treat the physical layer of networks as a static black box with no reconfigurability. As a result, the network is provisioned to carry the worst-case traffic demand, making it excessively inefficient and prohibitively expensive. Yet, ML training workloads have unique characteristics that can benefit from a dynamically reconfigurable physical layer to enable high throughput, low latency, and seamless recovery from failures. This paper discusses the benefits of enabling physical-layer reconfigurability in large-scale AI systems and highlights the foundations for future network architectures, algorithms, and protocols to increase efficiency and reduce the cost of large-scale AI networks.

References

1. N. Gebara *et al.*, "PANAMA: In-network Aggregation for Shared Machine Learning Clusters," *MLSys*, 2021.
2. Y. Peng *et al.*, "A Generic Communication Scheduler for Distributed DNN Training Acceleration," *USENIX SOSP*, 2019.
3. D. Narayanan *et al.*, "PipeDream: Generalized Pipeline Parallelism for DNN Training," *USENIX SOSP*, 2019.
4. Z. Zhia *et al.*, "Beyond Data and Model Parallelism for Deep Neural Networks," *MLSys*, 2019.
5. S. Lange *et al.*, "Sub-nanosecond Optical Switching Using Chip-Based Soliton Microcombs," *OFC*, 2020.
6. Y. S. Fainman *et al.*, "LEED: A Lightwave Energy-Efficient Datacenter," *OFC*, 2019.
7. Telescent G4 Network Topology Manager, `https://www.telescent.com/products`.
8. M. Khani *et al.*, "SiP-ML: High-Bandwidth Optical Network Interconnects for Machine Learning Training," *ACM SIGCOMM*, 2021.
9. H. Liu *et al.*, "Scheduling Techniques for Hybrid Circuit/Packet Networks," *ACM CoNext*, 2015.
10. N. Farrington *et al.*, "Helios: A Hybrid Electrical/Optical Switch Architecture for Modular Data Centers," *ACM SIGCOMM*, 2010.
11. W. Wang *et al.*, "TopoOpt: Optimizing the Network Topology for Distributed DNN Training," 2021.
12. G. Porter *et al.*, "Integrating Microsecond Circuit Switching into the Data Center," *ACM SIGCOMM*, 2013.
13. H. Ballani *et al.*, "Sirius: A Flat Datacenter Network with Nanosecond Optical Switching," *ACM SIGCOMM*, 2020.
14. W. M. Mellette *et al.*, "Expanding across time to deliver bandwidth efficiency and low latency," *USENIX NSDI*, 2020.