ChipLight: Cross-Layer Optimization of Chiplet Design with Optical Interconnects for LLM Training

Kangbo Bai[1], Zhantong Zhu[1], Yifan Ding[1] and Tianyu Jia[1†]
[1]School of Integrated Circuits, Peking University, Beijing, China
[†]Corresponding Author: tianyuj@pku.edu.cn

***Abstract*—In large-scale distributed LLM training, communication between devices becomes the key performance bottleneck. Chiplet technology can integrate multiple dies into a package to scale-up node performance with higher bandwidth. Meanwhile, optical interconnect (OI) technology offers long-reach, high-bandwidth links, making it well suited for scale-out networks. The combination of these two technologies has the potential to overcome communication bottlenecks within and across packages. In this work, we present ChipLight, a cross-layer multi-objective design and optimization method for training clusters leveraging chiplet and OI. We first abstract an architecture model for such complex clusters, co-optimizing chiplet architecture, training parallel strategy, and OI network topology. Based on such models, we tailor the design space exploration flow by combining both black-box and white-box methodologies. Evaluated by our experimental results, ChipLight achieves significantly improved training efficiency and provides valuable design insights for the development of future training clusters.**

***Index Terms*—Chiplet, Optical Interconnect, LLM Training**

I. Introduction

Large language models (LLMs) have achieved widespread success across various applications. However, the growing model size and context length increase the cost and time of LLM training. As shown in Fig. 1(a), current LLM training typically relies on GPU clusters with scale-up nodes and scale-out networks. For example, GPU clusters can employ high-bandwidth NVLinks (900GB/s) for scale-up, and lower-bandwidth InfiniBand (IB, 60GB/s) links for scale-out [1], [2]. Prior research [3]–[5] reveals that communication is the major performance bottleneck in large-scale LLM training. Emerging technologies hold promise for overcoming the "communication wall" for both scale-up and scale-out, while also requiring novel architecture design and optimizations.

First, chiplet integration is a promising technology for node scale-up with higher bandwidth at larger scale. As shown in Fig. 1(b), chiplet is a technique using advanced packaging [6], [7] to integrate multiple dies into a single package and form a multi-chiplet module (MCM). Chiplet not only reduces the single die area to improve manufacturing yield, but also offers much higher intra-package die-to-die bandwidths, e.g., up to 658 GB/s/mm [8] for high-bandwidth domains (HBD). Chiplet has already been adopted by industrial products, e.g., AMD MI300X [9] integrates 8 chiplets. Attempts [10], [11] were also made to achieve larger integration scales.

Second, optical interconnect (OI) is a key technique for higher-bandwidth scale-out networks by replacing electrical interconnects with optical circuit switch (OCS) and links. As shown in Fig. 1(b), OI is critical, especially for MCM-based clusters, to avoid the benefits of on-package HBD being

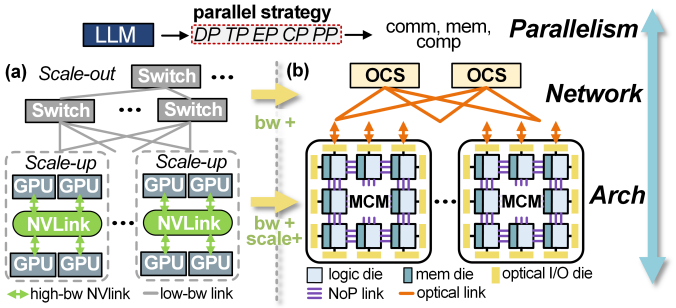


Fig. 1. LLM Training clusters and cross-layer optimization.

eliminated by low-bandwidth inter-MCM links. Co-packaged optics (CPO) can integrate optical modules within the package, achieving bandwidth density up to 128 GB/s/mm [12]. OI has already been adopted in Google TPUs [13] and will be employed in the upcoming Nvidia GPUs [14].

We believe that the combination of chiplet and OI is a promising approach for next-generation LLM training clusters. However, there is a lack of research on the design, optimization, and modeling of such systems. Prior studies have explored standalone MCM for inference tasks [11], [15]–[18] or solely on optical networks [19] for DNN tasks. Recently, RailX [20] explored a specific network design integrating MCMs and OI, but it lacks discussion on optimization of cluster architecture design and end-to-end evaluation of LLM training deployment.

In this work, we propose ChipLight, a cross-layer design and optimization framework for LLM training clusters leveraging chiplet and OI techniques. Through cross-layer innovations, ChipLight can effectively exploit MCMs and OI for large-scale LLM training. At the architecture level, the new design flexibility of MCMs is explored, including the optimization of single die area, integration scale, resource configuration of memory and OI ports. At the network level, the characteristics of training traffic is coordinated with the properties of MCM and OI for optimized scale-out networks. It is observed that the traffic distribution of LLM training exhibits both spatial and temporal traits, with the latter often neglected in prior optimizations [19], [21]. A network model is developed to abstract the complex and unexplored topology design space, with the optimization flow decoupling the physical and logical topologies. At the parallelism level, parallel strategies are also adapted to the novel MCM and OI interconnects. After cross-layer optimization, clusters with MCM and OI showcase 19.58× throughput gains over conventional GPUs. Our optimization achieves a 41% performance improvement over the latest network design [20] at a similar cost.

978-3-9826741-1-7/DATE26/© 2026 EDAA

The contributions of this work are summarized as follows:

- We perform an in-depth analysis of LLM training traffic, leveraging its characteristics to design networks based on MCM and OI to enhance communication efficiency.
- A design model of MCM- and OI-based network is developed to abstract the complex optimization space and to highlight key design parameters from MCM architecture, parallelism, and network topology.
- A cross-layer multi-objective optimization method is proposed to perform effective explorations of the design for training clusters based on chiplet and OI. MCM architecture, network topology, and parallel strategy are co-optimized.
- After thorough optimization, we evaluate how chiplet and OI technologies enhance LLM training clusters, providing key insights for designing such systems.

II. Background

A. Chiplet and Optical Interconnect

Chiplet has emerged as a crucial technology to scale-up performance and reduce costs. By leveraging advanced packaging technology, diverse dies are integrated on a substrate [22] or interposer [6] and connected by the network-on-package (NoP) to form an MCM. The NoP offers high bandwidth, low power, and low latency transfer compared to off-package interconnects. For example, novel die-to-die (D2D) interface achieves 658 GB/s/mm bandwidth at 0.29 pJ/b [8]. Chiplet also reduces costs by decreasing the single-die area and improving the manufacturing yield. The chiplet technology introduces greater flexibility in hardware design, and new design considerations have also been raised: For LLM training, what scale is required for a single chiplet and an MCM package? Should hardware configuration be adapted differently from GPUs? We aim to answer these questions.

To scale-out the cluster, optical interconnects are supplanting electrical links with higher bandwidth. With the recent CPO [23]–[26] technology, optical I/O dies are integrated at the package edge for signal conversion between logic dies and external optical links. The optical links are connected with optical circuit switches (OCS). Our work focuses on micro-electrical mechanical switches (MEMS) OCS, which is currently the most promising pathway due to its high port count and technical maturity, and has been deployed in Google TPUs [13]. However, because of millisecond-scale switching latency, MEMS OCS needs one-time configuration of connections before training starts, or deterministic reconfiguration during non-communication phases. How can we build efficient OI networks, and what gains can they provide? We aim to answer these questions.

B. Network Topology

As shown in Fig. 2(a), the network topology has two aspects: *physical topology* and *logical topology*. *Physical topology* describes the actual wiring between devices and switches, and the *logical topology* is the network structure upon which communication is implemented. Logical topology is derived

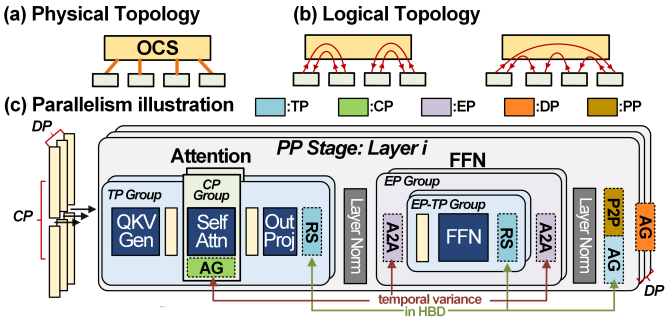


Fig. 2. Parallelism and corresponding communications in LLM training.

from the physical topology by configuring switch connections and pruning idle links. Owing to the switch reconfiguration, a physical topology can support multiple logical topologies. As the example in Fig. 2(a), four devices are physically connected to an OCS, which can logically configure the devices as either two rings of diameter 2 or a single ring of diameter 4, as shown in Fig. 2(b). For LLM training, TPUv4 [13] employs a 3D-torus topology with OCS. RailX [20] combines Chiplet and OI, which employs a HammingMesh-like physical topology and can be configured as a high-dimensional logical topology.

C. Parallelism in LLM Training

Large-scale LLM training employs diverse parallelisms to distribute the workloads across numerous devices, e.g., GPUs or chiplets. Fig. 2(b) illustrates an example of computation and communication for a Transformer layer under hybrid parallelisms, which is commonly supported by mainstream training frameworks [27], [28].

Tensor parallelism (TP) splits weight matrices and distributes shards across devices. Reduce-scatter (RS) and all-gather (AG) are required to aggregate the results.

Data parallelism (DP) splits data samples across devices, each of which keeps a model replica and processes data independently. All-gather (AG) communication is introduced during backward propagation to synchronize gradients.

Pipeline parallelism (PP) divides the model's layers into stages across devices in a pipelined manner, introducing peer-to-peer (P2P) send between adjacent stages.

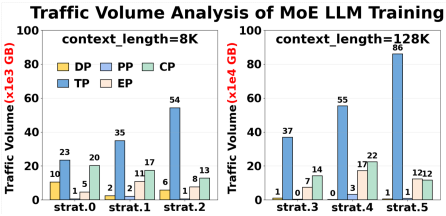
Context Parallelism (CP) splits sequences across devices, completing attention computation through ring-attention [29] to enable long context training, and all-gathering (AG) communication (green block in Fig. 2) is required in the attention.

Expert Parallelism (EP) is specialized for prevalent MoE LLMs [2], [30], distributes FFN experts across devices. Before and after the FFN, all-to-all (A2A) communications are required to dispatch / recombine tokens to / from the devices hosting the corresponding experts.

III. Profiling and Motivations

A. Profiling Analysis

We profile the training traffic volume for Qwen3-235B-A22B [30] on a total of 1024 devices (e.g., GPUs or chiplets), using ASTRA-sim [31] simulator to derive generalizable observations for subsequent optimizations. The collective communication is based on the ring algorithm. Fig. 3 shows the parallelism-wise



	DP	TP	PP	EP	CP
strat.0	4	4	4	2	8
strat.1	2	4	4	8	4
strat.2	4	8	2	4	4
strat.3	4	4	4	2	8
strat.4	2	4	4	8	4
strat.5	4	8	2	4	4

Fig. 3. Traffic volume of LLM training in different cases.

traffic volumes under different parallel strategies (listed in the right table) and context length.

Based on Fig. 3, we have concluded the **Observation 1: the parallel strategy significantly influences traffic volume, while it generally follows *TP>(CP,EP)>(DP,PP)***. The relative order of the two parallelisms in parentheses also varies with the context length. Therefore, suitable parallel strategies are crucial, and networks with parallelism group mapping should be tailored to traffic volume. Naturally, a TP group should be mapped within one GPU or MCM to confine the majority of traffic to on-package HBD. For chiplet node, depending on the MCM scale and parallel strategy, a parallelism other than TP may also be mapped within MCMs, while traffic of other parallelisms must traverse lower-bandwidth inter-MCM links. The current InfiniBand only offers 60GB/s bandwidth [2], presenting a tens-of-times gap compared to TB/s-scale NoP bandwidth. However, as shown in Fig. 3, the CP and EP traffic is not much lower than TP. We have the **Observation 2: electronic links negate the benefits of on-package HBD due to traffic–bandwidth mismatch**. In contrast, OI can increase inter-MCM bandwidth to 400GB/s or 128GB/s/mm [12], [32], significantly reducing the gap with NoP.

B. Traffic Distribution Analysis

In this section, we analyze the spatial distribution of training traffic for a ring topology, in which traffic is mainly concentrated on a small subset of links between neighbor devices. Fig. 4 shows a traffic heatmap with indices of source and destination devices on x- and y-axes, and each grid's color indicates the corresponding training traffic volume. It is evident that the training traffic is highly sparse and regular. Traffic is confined within parallel groups, and due to the ring communication algorithm, each device typically communicates only with a few devices that are adjacent in the ring. This sparsity leaves many links idle in conventional all-to-all networks in data centers. Additionally, the traffic volumes vary greatly, reflecting uneven distribution. Based on the above characteristics, we have the **Observation 3: training traffic is spatially sparse and link resource allocation should be guided by traffic distribution.**

We also analyze the temporal distribution of training traffic, in which communication of various parallelisms does not occur continuously during the training. As shown in Fig. 2, CP and EP traffic occur in the attention and FFN operators respectively, exhibiting temporal variance due to intervening output_proj and layernorm computations. We have the **observation 4: the temporal traffic variance allows dynamic reuse of bandwidth resources**, which has been neglected in prior optimizations [19], [21]. If two independent parts of links are allocated

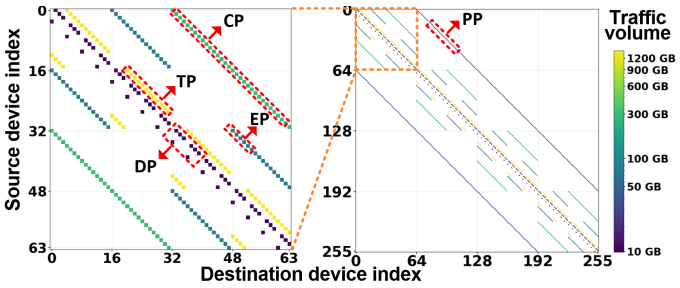


Fig. 4. Traffic distribution heatmap of LLM training.

to the CP and EP traffic, one part will remain idle while the other is active. Conversely, reconfiguring the network via OCS between the interval of CP and EP traffic allows links to be reused for both traffic, improving network efficiency. Opportunities for link reuse also exist among CP, DP, and PP. But as mentioned in Sec. III-A, CP and EP traffic typically dominate the inter-MCM network. Therefore, reuse between these two parallelisms is most beneficial. To support dynamic link reuse, the OCS switching latency should be smaller than the traffic interval, which is satisfied in practice.

IV. Cross-Layer Optimization for Training Cluster with Chiplet and OI

A. Model of Chiplet and OI Network

Characterizing the design points and defining the optimization space of MCM- and OI-based training clusters is non-trivial due to their complex structure. To explore a wide range of promising and crucial design possibilities under controllable optimization complexity, we abstract a cluster model for subsequent optimizations.

Fig. 5(a) shows the model of an MCM. To explore the design flexibility of MCM, we set the total compute performance C of the cluster as the input constant rather than the chiplet count. C is split into N MCMs. An MCM is further split into $x \times y$ logic dies, each coupled with m memory dies. Optical I/O dies are integrated at the edge of the MCM package, each providing o optical links. An MCM has a total of $L = 2 \times (x + y) \times o$ links for external connectivity. The maximum of o is proportional to the edge length of the logic die. A larger o requires more OCS, significantly increasing cost. The design variables N, x, y, m, o define the MCM architecture. We assume that NoP employs mesh topology with bandwidth B_p, which scales linearly with the length of D2D. The edges of logic die are shared by the D2D link interfaces and memory dies, leading to a trade-off between m and B_p.

Fig. 5(b) shows the OI network model. It is challenging to describe complex and diverse physical topologies using quantitative variables. Inspired by previous research [13], [20], we abstract diverse physical topologies into a rail-based model, which consists of multiple *rail dimensions*. Fig. 5(b) illustrates an example with two rail dimensions, i.e., D_i and D_j. A rail is a connection comprising multiple MCMs, where each MCM connects to an OCS via a single link. A rail dimension D_i represents a set of connections, where S_i OCSs are used to connect N_i MCMs, occupying R_i links on each MCM, and is denoted as $D_i = (N_i, R_i, S_i)$. Each MCM connects k_i

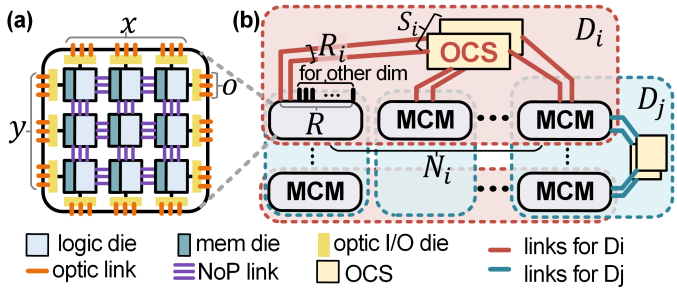


Fig. 5. Our cluster model for training cluster with MCM and OI.

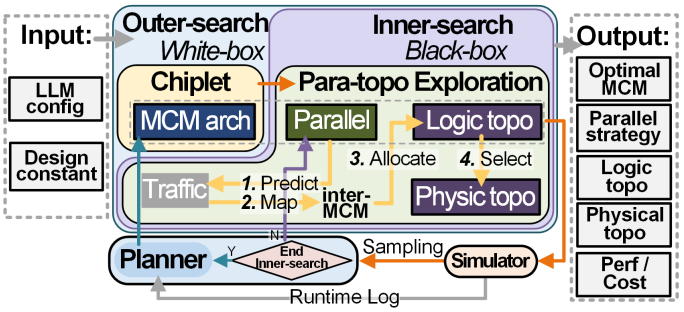


Fig. 6. Workflow of ChipLight cross-layer optimizations.

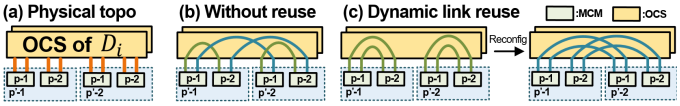


Fig. 7. Illustration of the dynamic link reuse.

links to the same OCS. So $S_i = \lfloor R_i/k_i \rfloor$, Fig. 5(b) shows the $k_i = 1$ case. Each OCS has P ports, and $k_i \times N_i \leq P$ needs to be satisfied. Each MCM is involved in multiple rail dimensions, and their interweaving constitutes the entire network. The MCM count satisfies $\prod N_i = N$, and OCS count $S = \sum_i (\prod_{j \neq i} N_j \times S_i)$. Due to symmetry, link allocation across D_i is uniform for all MCMs and satisfies $\sum R_i = L$.

Besides OI network, the parallelism values are also incorporated as variables in the model. Parallelism groups are mapped to intra-MCM HBD or inter-MCM rail dimensions. There is no one-to-one correspondence between rail dimension and parallelism, as multiple parallelisms can map to a single D_i. Since up to four inter-package parallelisms in LLM, $i \leq 4$.

The advantage of our model lies in comprehensive coverage of optimization space, allowing cross-stack exploration of design possibilities that have been ignored before. For example, RailX [20] and TPUv4 [13] can be viewed as specific cases that have two or three rail dimensions with uniformly distributed links. Prior optimization considered a single-dimensional network [19], or simplified practical physical topology [21]. Multiple key design aspects of training clusters have been considered, including the scale of MCM and chiplet, and the allocation of compute, memory, and NoP.

B. ChipLight Optimization Methodology

1) Optimization Workflow: ChipLight employs a cross-layer multi-objective method as explained in Fig. 6. The LLM configuration and other design constants are set as input. Since the MCM architecture defines the design space of parallel strategy and OI networks, we adopt a nested optimization flow: The outer-search explores the MCM architecture, while for each MCM sample, the inner-search optimizes the parallel strategy and OI network topology, i.e. the para-topo explorations. Each sampled design point is evaluated by simulator. For the next iteration, the planner decides whether to continue the inner search or switch to the outer-search. In the latter case, the new MCM sample is also determined. Through iterations, a performance-cost Pareto frontier is generated, representing the optimized MCM architecture, topology, and parallelization strategy. We combine black-box and white-box optimization methods for above flow, detailed as follows.

2) Optimization for Para-Topo: In inner-search for parallelism and topology co-explorations, a straightforward flow first set the OI physical topology, then tailors the parallel strategy and logical topology accordingly. However, such methods may converge to a local optimum due to the pre-fixed physical topology and have low search efficiency. In contrast, ChipLight follows a **parallel-centric** search flow.

As shown in Fig. 6, the parallelism values are set as optimization variables in the inner-search and sampled by a black-box algorithm, e.g., PRF [33]. First, the traffic volume is projected based on the determinacy of training traffic and is independent of the underlying network, similar as the results in Fig. 4. Second, parallelism groups are mapped to intra-MCM or inter-MCM. The TP group and possibly one other group are mapped to intra-MCM, while the traffic of other parallelisms traverses the OI network. Third, links of an MCM are allocated to the traffic of inter-MCM mapped parallelism. For parallelism p with traffic volume v_p, each MCM assigns l_p links to parallelism p based on its traffic proportion, i.e., $l_p = \lfloor L * v_i / (\sum v(i)) \rfloor$. An ideal logical topology can be generated in which all MCMs involved in an inter-MCM parallelism group form a ring. The ring size equals the value of this parallelism, and bandwidth scales with l_p. For EP, fully-connected (FC) topology can also be considered for A2A communications. The overall logical topology is a high-dimensional composition of local ring/FC of each parallelism, with all links being utilized and aligned to the spatial-temporal distribution of traffic. Finally, the physical topology is derived from the ideal logical topology. For each parallelism p, the ring of p must be mapped to a rail dimension D_i, with $l_p = D_i$. If two parallelisms are mapped to the same D_i, the product of their values equals N_i. By enumerating all parallelism-rail mappings, feasible physical topologies can be identified, and the one with the fewest OCSs is selected.

Dynamic link reuse can be supported by the above flow with minor modifications. If dynamic link reuse is applied between p and p', they must be mapped to the same D_i. In other words, the MCMs involved in p and p' must physically be connected to the OCS of D_i as shown in Fig. 7(a). Fig. 7(b) illustrates the case without link reuse, where each MCM uses a green link for p traffic and a blue link for p' traffic. Fig. 7(c) shows the link reuse implementation, where each MCM employs two links for p; and through OCS reconfiguration, the two links are fully used for p'. Therefore, in link allocation, p and p' share

the links with count:

$$l_{reuse} = \left\lfloor \frac{L * max(v_p, v_{p'})}{\sum v_{others} + max(v_p, v_{p'})} \right\rfloor \quad (1)$$

During the physical topology selection, we impose the constraint that p and p' are mapped to the same rail dimension.

3) Optimization for Chiplets: By leveraging the simulation runtime logs, we conducted an efficient outer-search for chiplet architecture. Outer-search sampling is costly since each sample introduces a full inner-search. Current optimizations [16] often neglect the development of sampling strategies in outer-search. However, for highly flexible MCM architectures, exploring the vast search space with limited outer-sampling iterations necessitates efficient sampling methods. Conventional black-box algorithms perform poorly with limited samples.

To improve the search efficiency, additional information is required beyond the standalone final performance and cost metrics. Simulation logs contain abundant information that reveals hardware bottlenecks and utilization. Resampling of MCM architecture occurs after an inner-search, when parallelism and network are near-optimal, and the bottleneck lies in the MCM architecture. We implement a heuristic planner to identify the bottleneck by collecting key metrics in logs, such as compute resource utilization, memory consumption, and communication bottlenecks, and modify the MCM architecture to break bottlenecks or reduce redundancy.

V. Evaluations and Insights

A. Evaluation Setup

In our experiment, the design parameters of logic dies are based on H100 GPU [34], while the memory die employs HBM3 [35]. The parameters for the chiplet are derived from [8], [18], while the CPO implementation is based on [12], [32]. Target model is Qwen3-235B-A22B [30] with context length of 10k. We modified the open-source ASTRA-sim [31] to serve as our evaluation simulator. The cost estimation is based on [20], [36].

B. Breaking Communication Bound by Chiplet and OI

We first evaluate the throughput trends of the training cluster under different hardware choices. Fig. 8 shows the end-to-end training throughput (y-axis) changes as the total compute performance C (x-axis) increases. We apply our optimization to each experimental group to the extent permitted, exploiting their potential. H100 [34] GPU clusters with clos network formed by NVLink and IB are used as the comparison baseline. Throughput of such clusters increases proportionally with C, or GPU counts, at small scales. However, the growth rate significantly slows down when C exceeds 4×10^6TOPS, which we define as the scaling point. While the per-device computation load decreases proportionally with the number of devices, the communication traffic volume and allocated bandwidth resources remain relatively constant. Therefore, the training throughput encounters a communication wall at large scale.

Next, we evaluate the gains of chiplet and OI technology. To ensure a fair comparison, each logic die and its coupled HBM are consistent with the H100. Chiplet+IB employs NoP

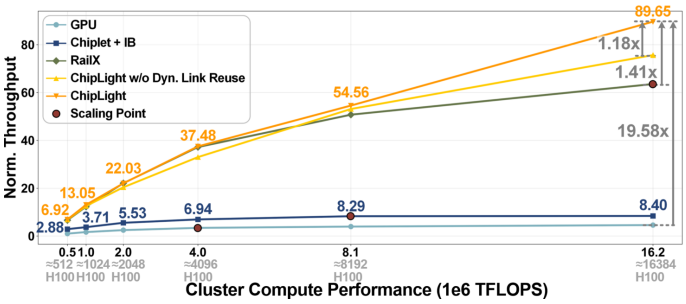


Fig. 8. Training throughput scaling across different clusters.

instead of NVLink for scale-up, while retaining electrical IB in scale-out network. As shown in Fig. 8, Chiplet+IB demonstrates modest gains over GPU baseline and pushes the scaling point to about 8×10^6TOPS. However, it still encounters a clear boundary caused by IB. Both RailX and ChipLight combine chiplets with OI, the former builds on existing design [20], and the latter is refined through our network optimization. Both show a clear throughput boost over GPU clusters, especially at large scales. The scaling point is pushed far out, indicating enhanced system scalability. Therefore, we obtain **insight 1: chiplet technology alone cannot fully overcome the communication bound, while its combination with OI greatly alleviates the communication bottleneck.**

In Fig. 8, RailX and ChipLight are configured with the same per-edge OI ports o, resulting in similar total cost. We also apply dynamic link reuse in RailX. At small scales, throughput of RailX and ChipLight is comparable, as the OI network is not the bottleneck, reducing the necessity of optimization. However, at $C = 16 \times 10^6$TFLOPS, ChipLight achieves 41% improvement over RailX due to optimized network topology. We also evaluate the case without dynamic link reuse, which has a throughput drop of 30%. This leads to **insight 2: through optimizations on network topology and dynamic link reuse, ChipLight achieves better network efficiency with similar cost**.

C. Explorations on Optimal Packaging and Single Die Scale

With our ChipLight method, we can explore the optimal chiplet design scale for training clusters. In our experiment, the total compute performance of the cluster is fixed at $C = 8 \times 10^6$TFLOPS, and the single logic die has the same specifications as H100 GPU. We swept the number of logic dies integrated on MCM (i.e., MCM_scale) to explore the impact of package scale on cluster throughput and cost.

Fig. 9(a) presents the throughput-cost distribution of the cluster as the MCM_scale ranges from 4 to 64, plotted in different colors. The Pareto fronts are marked with dashed lines. After our thorough optimization, clusters based on small-scale MCMs achieve comparable optimal throughput to those using large-scale MCMs, and even marginally surpass the latter in some cases. This is because OI narrows the bandwidth gap between HBD and the scale-out network, reducing the benefit of enlarging HBD. And in larger MCMs, there are more inner logic dies, reducing the per-die OI bandwidth. Furthermore, the mesh topology becomes less efficient in large-scale NoP. Under

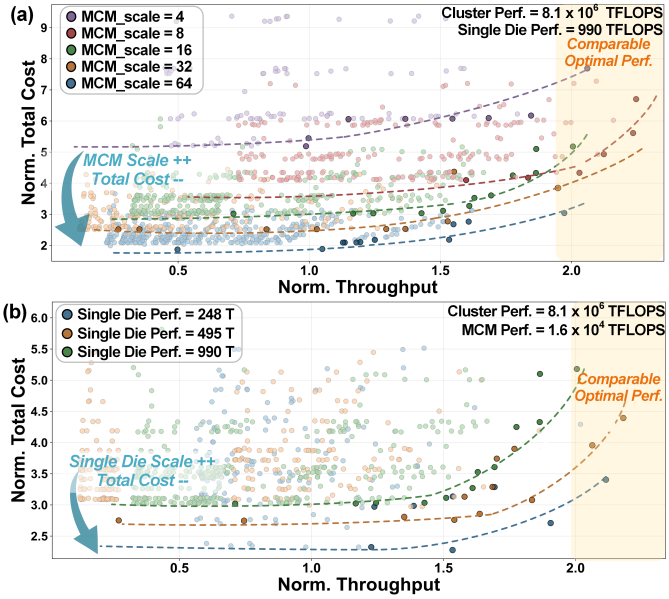


Fig. 9. Cost-throughput landscape for different MCM and single die scale.

fixed total compute performance, it is observed that larger-scale chiplet integration shifts more interconnects from OI to NoP, requiring fewer OCS and consequently greatly lowering cost. Hence, we have **insight 3: small-scale MCMs with OI can match the performance of large-scale MCMs, while large-scale integration greatly reduces cluster cost.**

Chiplet technology has better manufacturing yield and lower cost. However, smaller and more logic dies increase traffic volume, potentially leading to performance degradation. Fig. 9(b) shows the cost-throughput distribution of clusters with varying logic die scale. We maintain a constant compute performance of the entire cluster and an MCM for comparison. Fig. 9(b) indicates that reducing the performance of a logic die by half or even a quarter, i.e., cutting its area, does not lead to significant performance degradation. It is manifested as a downward shift of the Pareto frontier. The reason is that the high-bandwidth NoP can accommodate the traffic load growth caused by the increased count of logic dies. The total cluster cost decreases by approximately 23% when the single die scale is reduced to one quarter. We have **insight 4: by co-optimization such as ChipLight, it is feasible to reduce the die area and cost while maintaining the cluster performance.**

D. Explorations on Memory Resource and OI Links

Beyond the logic die, memory and optical I/O dies also influence the cost and performance of training clusters. For such novel clusters with MCM and OI, the configuration of these resources also needs to be re-evaluated.

Fig. 10(a) plots the trend of cluster cost (in line) and throughput (in bar) versus the number of memory dies (m) that are coupled with a logic die, while other hardware configurations are fixed. In practice, H100 is equipped with six HBM dies, providing about 3.3 TB/s of memory bandwidth. For a logic die with compute performance equivalent to H100 GPU, increasing the number of attached HBM improves cluster throughput effectively, until m is near 14, which indicates more HBMs

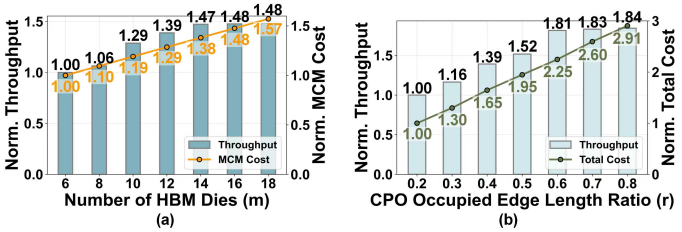


Fig. 10. Trade-off for memory and OI resource.

should be placed with GPUs. The reason is straightforward. Collective communications, such as reduce-scatter, occur along with aggregation computation. After data is received, memory dies must perform at least one read and one write. Since the GPU memory bandwidth is multiple times that of NVLink, the latency of memory access can be overlapped in data transmission. For MCMs, the bandwidth of the NoP and the memory are comparable, the latter can become the bottleneck. However, a large m also increases the cost, as shown in Fig. 10(a). In our case, 14 HBMs is the optimal configuration for abundant bandwidth, but it is difficult for practical physical implementations. So we have **insight 5: compared with GPUs, logic dies in MCMs require more HBMs and call for future higher bandwidth memory solutions**.

Fig. 10(b) shows the impact of cluster throughput and cost as more OI ports are integrated on a logic die. Current CPO achieves a bandwidth density of 128 GB/s/mm [32]. However, during explorations, we observe that it is not optimal for logic die to dedicate the entire edge for CPO to maximize optical ports. We use r, the ratio of edge length that is occupied for CPO as the x-axis in Fig. 10(b), with the corresponding cluster cost and optimal throughput as the y-axis. Increasing r indicates more optical links are utilized to boost bandwidth, and more OCS are required to support the connections. It is observed that the high cost of OCSs imposes a significant increase on the total cluster cost. As shown in the Fig. 10(b), the throughput growth becomes disproportionate to the additional cost when r is beyond 0.6. This indicates that a balanced cost-performance configuration only utilizes over half of the maximum bandwidth of CPOs. Hence, we have **insight 6: the current bandwidth density of CPO is sufficient to meet interconnect demands, while the costly OCS limits the available OI bandwidth in practice**.

VI. Conclusion

In this paper, we explore the design space of training clusters with chiplet technology and optical interconnect. We present ChipLight, a cross-layer optimization method with cluster models for MCM and OI. Compared to conventional GPU clusters, the optimized clusters with MCM and OI achieve a 19.58× improvement in training throughput under our explorations. Under the same technology and comparable cost, ChipLight achieves a 41% throughput improvement compared to prior network design [20]. Valuable design insights for cluster architectures have also been provided.

Acknowledgment

This work was supported in part by NSFC No. 92464202.

REFERENCES

[1] M. Naumov, J. Kim, D. Mudigere, S. Sridharan, X. Wang, W. Zhao, S. Yilmaz, C. Kim, H. Yuen, M. Ozdal, *et al.*, "Deep learning training in facebook data centers: Design of scale-up and scale-out systems," *arXiv preprint arXiv:2003.09518*, 2020.

[2] 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.

[3] J. Duan, S. Zhang, Z. Wang, L. Jiang, W. Qu, Q. Hu, G. Wang, Q. Weng, H. Yan, X. Zhang, *et al.*, "Efficient training of large language models on distributed infrastructures: a survey," *arXiv preprint arXiv:2407.20018*, 2024.

[4] H. Liao, B. Liu, X. Chen, Z. Guo, C. Cheng, J. Wang, X. Chen, P. Dong, R. Meng, W. Liu, Z. Zhou, Z. Zhang, Y. Gai, C. Qian, Y. Xiong, Z. Cheng, J. Xia, Y. Ma, X. Chen, W. Du, S. Xiao, C. Li, Y. Qin, L. Xiong, Z. Yu, L. Chen, L. Chen, B. Wang, P. Wu, J. Gao, X. Li, J. He, S. Yan, and B. McColl, "Ub-mesh: a hierarchically localized nd-fullmesh datacenter network architecture," 2025.

[5] X. Liao, Y. Sun, H. Tian, X. Wan, Y. Jin, Z. Wang, Z. Ren, X. Huang, W. Li, K. F. Tse, *et al.*, "mfabric: An efficient and scalable fabric for mixture-of-experts training," *arXiv preprint arXiv:2501.03905*, 2025.

[6] P. K. Huang, C. Y. Lu, W. H. Wei, C. Chiu, K. C. Ting, C. Hu, C. Tsai, S. Y. Hou, W. C. Chiou, C. T. Wang, and D. Yu, "Wafer level system integration of the fifth generation cowos®-s with high performance si interposer at 2500 mm2," in *2021 IEEE 71st Electronic Components and Technology Conference (ECTC)*, pp. 101–104, 2021.

[7] R. Mahajan, R. Sankman, N. Patel, D.-W. Kim, K. Aygun, Z. Qian, Y. Mekonnen, I. Salama, S. Sharan, D. Iyengar, and D. Mallik, "Embedded multi-die interconnect bridge (emib) – a high density, high bandwidth packaging interconnect," in *2016 IEEE 66th Electronic Components and Technology Conference (ECTC)*, pp. 557–565, 2016.

[8] D. T. Melek, R. Navinkumar, J. Vandersand, P. Sarkar, B. Prakash, A. Leuciuc, K. Geary, S. Ma, C. M. Mehta, S. Jain, B. Bothra, P. Sabharwal, R. Vaish, K. Bhanushali, Y. Ding, C. Frost, J. Annunziata, K. Sadhu, D. Kyritsis, J. Bostak, M. Li, S. Williams, and K. Chang, "A 0.29pj/b 5.27tb/s/mm ucie advanced package link in 3nm finfet with 2.5d cowos packaging," in *2025 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 68, pp. 590–592, 2025.

[9] A. Smith, E. Chapman, C. Patel, R. Swaminathan, J. Wuu, T. Huang, W. Jung, A. Kaganov, H. McIntyre, and R. Mangaser, "11.1 amd instinctтm mi300 series modular chiplet package – hpc and ai accelerator for exa-class systems," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, pp. 490–492, 2024.

[10] E. Talpes, D. D. Sarma, D. Williams, S. Arora, T. Kunjan, B. Floering, A. Jalote, C. Hsiong, C. Poorna, V. Samant, *et al.*, "The microarchitecture of dojo, tesla's exa-scale computer," *IEEE Micro*, vol. 43, no. 3, pp. 31–39, 2023.

[11] Q. Yang, T. Wei, S. Guan, C. Li, H. Shang, J. Deng, H. Wang, C. Li, L. Wang, Y. Zhang, S. Yin, and Y. Hu, "Pd constraint-aware physical/logical topology co-design for network on wafer," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, ISCA '25, (New York, NY, USA), p. 49–64, Association for Computing Machinery, 2025.

[12] S. Fathololoumi, "4 tb/s optical compute interconnect chiplet for xpu-to-xpu connectivity," in *2024 IEEE Hot Chips 36 Symposium (HCS)*, pp. 1–18, IEEE Computer Society, 2024.

[13] N. Jouppi, G. Kurian, S. Li, P. Ma, R. Nagarajan, L. Nai, N. Patil, S. Subramanian, A. Swing, B. Towles, *et al.*, "Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings," in *Proceedings of the 50th annual international symposium on computer architecture*, pp. 1–14, 2023.

[14] Online in *https://www.nvidia.com/en-sg/networking/products/silicon-photonics/*.

[15] Y. S. Shao, J. Clemons, R. Venkatesan, B. Zimmer, M. Fojtik, N. Jiang, B. Keller, A. Klinefelter, N. Pinckney, P. Raina, *et al.*, "Simba: Scaling deep-learning inference with multi-chip-module-based architecture," in *Proceedings of the 52nd annual IEEE/ACM international symposium on microarchitecture*, pp. 14–27, 2019.

[16] J. Cai, Z. Wu, S. Peng, Y. Wei, Z. Tan, G. Shi, M. Gao, and K. Ma, "Gemini: Mapping and architecture co-exploration for large-scale dnn chiplet accelerators," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 156–171, IEEE, 2024.

[17] R. Prabhakar, R. Sivaramakrishnan, D. Gandhi, Y. Du, M. Wang, X. Song, K. Zhang, T. Gao, A. Wang, X. Li, *et al.*, "Sambanova sn40l: Scaling the ai memory wall with dataflow and composition of experts," in *2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 1353–1366, IEEE, 2024.

[18] Z. Xu, D. Kong, J. Liu, J. Li, J. Hou, X. Dai, C. Li, S. Wei, Y. Hu, and S. Yin, "Wsc-llm: Efficient llm service and architecture co-exploration for wafer-scale chips," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, ISCA '25, (New York, NY, USA), p. 1–17, Association for Computing Machinery, 2025.

[19] W. Wang, M. Khazraee, Z. Zhong, M. Ghobadi, Z. Jia, D. Mudigere, Y. Zhang, and A. Kewitsch, "{TopoOpt}: Co-optimizing network topology and parallelization strategy for distributed training jobs," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pp. 739–767, 2023.

[20] Y. Feng, T. Chen, Y. Wei, S. Shen, S. Wang, W. Li, K. Ma, and T. Hoefler, "Railx: A flexible, scalable, and low-cost network architecture for hyper-scale llm training systems," *arXiv preprint arXiv:2507.18889*, 2025.

[21] A. Raju, J. Ni, W. Won, C. Man, S. Krishnan, S. Sridharan, A. Yazdanbakhsh, T. Krishna, and V. J. Reddi, "Cosmic: Enabling full-stack co-design and optimization of distributed machine learning systems," *arXiv preprint arXiv:2505.15020*, 2025.

[22] S. Naffziger, N. Beck, T. Burd, K. Lepak, G. H. Loh, M. Subramony, and S. White, "Pioneering chiplet technology and design for the amd epyc™ and ryzen™ processor families : Industrial product," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pp. 57–70, 2021.

[23] D. Kuchta, J. Proesel, F. Doany, W. Lee, T. Dickson, H. Ainspan, M. Meghelli, P. Pepeljugoski, X. Gu, M. Beakes, *et al.*, "Multi-wavelength optical transceivers integrated on node (motion)," in *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, pp. 1–3, IEEE, 2019.

[24] P. Maniotis, L. Schares, B. G. Lee, M. A. Taubenblatt, and D. M. Kuchta, "Scaling hpc networks with co-packaged optics," in *Optical Fiber Communication Conference*, pp. T3K–7, Optica Publishing Group, 2020.

[25] J. H. Lau, "Co-packaged optics—heterogeneous integration of photonic integrated circuits and electronic integrated circuits," *Journal of Electronic Packaging*, vol. 147, no. 1, 2025.

[26] Y.-T. Yang and C.-M. Hung, "Heterogeneous integration in co-packaged optics," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2025.

[27] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-lm: Training multi-billion parameter language models using model parallelism," *arXiv preprint arXiv:1909.08053*, 2019.

[28] S. Singh, O. Ruwase, A. A. Awan, S. Rajbhandari, Y. He, and A. Bhatele, "A hybrid tensor-expert-data parallelism approach to optimize mixture-of-experts training," in *Proceedings of the 37th ACM International Conference on Supercomputing*, ICS '23, (New York, NY, USA), p. 203–214, Association for Computing Machinery, 2023.

[29] H. Liu, M. Zaharia, and P. Abbeel, "Ring attention with blockwise transformers for near-infinite context," *arXiv preprint arXiv:2310.01889*, 2023.

[30] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[31] 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," in *2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pp. 283–294, IEEE, 2023.

[32] M. Mehta, "An ai compute asic with optical attach to enable next generation scale-up architectures," in *2024 IEEE Hot Chips 36 Symposium (HCS)*, pp. 1–30, IEEE, 2024.

[33] I. Reis, D. Baron, and S. Shahaf, "Probabilistic random forest: A machine learning algorithm for noisy data sets," *The Astronomical Journal*, vol. 157, no. 1, p. 16, 2018.

[34] Online in *https://resources.nvidia.com/en-us-hopper-architecture/nvidia-h100-tensor-c?ncid=no-ncid*.

[35] Online in *https://product.skhynix.com/products/dram/hbm/hbm3.go*.

[36] Y. Feng and K. Ma, "Chiplet actuary: A quantitative cost model and multi-chiplet architecture exploration," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pp. 121–126, 2022.