

# MoEntwine: Unleashing the Potential of Wafer-scale Chips for Large-scale Expert Parallel Inference

Xinru Tang\*, Jingxiang Hou\*, Dingcheng Jiang\*, Taiquan Wei\*, Jiaxin Liu\*, Jinyi Deng\*,  
Huizheng Wang\*, Qize Yang\*, Haoran Shang\*, Chao Li†, Yang Hu\*, Shouyi Yin\*‡,

\*Tsinghua University, School of Integrated Circuits, BNRist, Beijing, China

†Shanghai Jiao Tong University, Shanghai, China

‡Shanghai Artificial Intelligence Laboratory, Shanghai, China

\*{tangxr23, houjx22, jdc24, weitq24, jiaxin-l24, wanghz22, yqz23, shanghr23}@mails.tsinghua.edu.cn,

\*dengjinyi@mail.tsinghua.edu.cn, hu\_yang@tsinghua.edu.cn, yinsy@tsinghua.edu.cn

†lichao@cs.sjtu.edu.cn

**Abstract**—As large language models (LLMs) continue to scale up, mixture-of-experts (MoE) has become a common technology in SOTA models. MoE models rely on expert parallelism (EP) to alleviate memory bottleneck, which introduces all-to-all communication to dispatch and combine tokens across devices. However, in widely-adopted GPU clusters, high-overhead cross-node communication makes all-to-all expensive, hindering the adoption of EP. Recently, wafer-scale chips (WSCs) have emerged as a platform integrating numerous devices on a wafer-sized interposer. WSCs provide a unified high-performance network connecting all devices, presenting a promising potential for hosting MoE models. Yet, their network is restricted to a mesh topology, causing imbalanced communication pressure and performance loss. Moreover, the lack of on-wafer disk leads to high-overhead expert migration on the critical path.

To fully unleash this potential, we first propose Entwined Ring Mapping (ER-Mapping), which co-designs the mapping of attention and MoE layers to balance communication pressure and achieve better performance. We find that under ER-Mapping, the distribution of cold and hot links in the attention and MoE layers is complementary. Therefore, to hide the migration overhead, we propose the Non-invasive Balancer (NI-Balancer), which splits a complete expert migration into multiple steps and alternately utilizes the cold links of both layers. Evaluation shows ER-Mapping achieves communication reduction up to 62%. NI-Balancer further delivers 54% and 22% improvements in MoE computation and communication, respectively. Compared with the SOTA NVL72 supernode, the WSC platform delivers an average 39% higher per-device MoE performance owing to its scalability to larger EP.

## I. INTRODUCTION

Traditional machine learning systems have reached a mature stage with well-established methodologies and deployments in various domains [51], [53], [54], but recent advances in large language models (LLMs) have rapidly surpassed these conventional approaches. LLMs have achieved state-of-the-art performance across a wide range of applications [15], [20], [52], [57], [58]. To further sustain this scaling trend, *Mixture-of-Experts* (MoE) architectures [48] have gained popularity for improving parameter efficiency. Recent models such as DeepSeek-R1-671B [38] and Qwen3-234B [61] adopt MoE designs with hundreds of lightweight experts, selectively activating a small subset per token (e.g., 8 out of 256 experts). This design reduces compute cost but imposes a significant

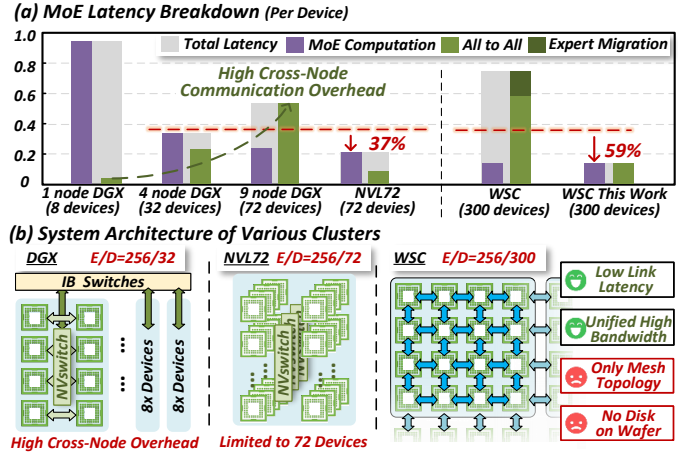


Fig. 1. (a) MoE Latency Breakdown of DeepSeek-V3 with EP equals to device count, total latency equals to the maximum of computation and communication time. (b) System Architecture of DGX, NVL72, and WSC.

memory footprint—especially when multiple experts are colocated on the same device during inference.

To address this, *Expert Parallelism* (EP) [29], [66] distributes experts across devices, ideally one expert per device, to alleviate memory pressure. However, EP requires all-to-all communication to route tokens to and from the activated experts, with overhead scaling rapidly as the device count increases. Thus, a critical factor influencing EP performance is the ratio of experts to available devices, defined as the **E/D ratio**. A lower E/D ratio indicates fewer experts per device, reducing memory contention and improving inference throughput. The optimal EP performance is theoretically achieved when E/D equals one, provided all devices are interconnected by a unified high-bandwidth, low-latency network.

However, as shown in Fig. 1(b), in widely-deployed DGX systems [10], high-performance networking is confined to each 8-GPU nodes, with high-overhead inter-node links (e.g., IB Link [42]) degrading cross-node communication [35], [40]. As Fig. 1(a) shows, when the cluster scale exceeds 4 nodes

(32 GPUs), the all-to-all overhead exceeds computation by 2.3 $\times$ , forcing suboptimal  $E/D=8$  (256/32) and significant performance loss. This highlights the need for system-wide high-speed interconnects to minimize  $E/D$  and unlock EP scalability. To address this, NVIDIA introduced the NVL72 supernode [11], connecting 72 GB200 dies via a custom scale-up network. It improves  $E/D$  to 3.6 and boosts performance by 37% over 4-node DGX. However, its reliance on numerous switches and cables leads to high energy and infrastructure costs, limiting scalability and preventing  $E/D = 1$ .

Recently, **wafer-scale chips (WSCs)** [37], [39], [56], [64] have emerged as a promising approach to overcome these scaling bottlenecks. By directly interconnecting compute dies via wafer-scale interposers, WSCs offer unprecedented bandwidth and latency characteristics. Tesla’s Dojo platform [56], for instance, achieves 4 TB/s intra-wafer bandwidth between dies and 9 TB/s inter-wafer bandwidth, significantly outperforming NVLink’s 1.8 TB/s (by 4.4 $\times$ ). Such designs facilitate a unified, high-performance network spanning 300 dies in a single cabinet, enabling even  $E/D$  ratios below one, which further boosts EP performance up to 59%.

Despite their theoretical advantages, directly porting existing GPU-cluster optimizations onto WSCs fails to fully exploit their capabilities due to two unique challenges. First, signal integrity (SI) constraints compel practical WSC implementations [37], [56] to adopt mesh topologies instead of ideal all-to-all networks. As a result, all-to-all communication traffic must traverse multiple hops, causing significant congestion and performance degradation in the wafer’s central regions. Second, the lack of on-wafer storage exacerbates congestion: expert migration, widely utilized for load balancing, must frequently transfer large expert weights via the already congested wafer interconnects, further degrading performance.

Motivated by these unique challenges, we introduce **MoEntwine**, a specialized MoE scaling solution for WSCs, featuring two novel designs: **Entwined Ring Mapping (ER-Mapping)** and **Non-invasive Balancer (NI-Balancer)**.

We first observe that MoE inference workloads involve two primary types of collective communication: the all-to-all operations for token dispatching in MoE layers, and the all-reduce operations within attention layers. Notably, these two types of communication exhibit different latency characteristics as system scale increases; all-to-all latency escalates quickly with increased device count, while all-reduce latency remains relatively stable. Critically, the parallelism mapping strategy of the attention layers significantly affects the initial token distribution, thus indirectly influencing communication overhead in subsequent MoE layers. To capture this interaction systematically, we propose the **Full Token Domain (FTD)** framework, analyzing the trade-off between all-to-all and all-reduce overheads. Guided by this analysis, **ER-Mapping** co-designs the parallelism mapping strategies for attention and MoE layers, balancing communication pressure and dramatically reducing latency.

Secondly, we surprisingly find that ER-Mapping provides an opportunity to hide the overhead of expert migration. By

analyzing the link traffic, it’s observed that the distribution of “hot links” and “cold links” in the attention and MoE layers is complementary. Therefore, we can split a complete expert migration into multiple steps and use the cold links in these two layers alternately without overhead. Based on this, we propose **NI-Balancer**, a multi-step expert migration scheme that strategically exploits idle (“cold”) communication links in both layers to perform expert weight transfers without additional overhead. Specifically, NI-Balancer first identifies temporal locality patterns of expert selection during inference, then orchestrates expert migration across layers, effectively hiding migration overhead and ensuring agile load balancing.

Our evaluations show that WSC inherently reduces communication latency by 56% compared to DGX, benefiting from its unified wafer-scale interconnect. Further optimizations via ER-Mapping achieve up to 62% additional latency reduction. NI-Balancer completely eliminates expert migration overhead while significantly improving load balance, reducing MoE computation and communication latency by up to 54% and 22%, respectively. These innovations effectively address the fundamental communication and migration bottlenecks inherent to wafer-scale EP implementations. Compared to the state-of-the-art NVL72 supernode, the WSC enhanced by MoEntwine achieves 39% higher average per-device MoE performance, unlocking the full scaling potential of wafer-scale MoE inference.

## II. BACKGROUND

In this section, we first introduce the structure of LLMs with MoE. We then describe the architecture of wafer-scale chips.

### A. Large Language Models with Mixture-of-Experts

The LLMs with MoE [48] technology comprises a stack of dense and sparse blocks. Within sparse blocks, an MoE layer replaces the traditional MLP layer. Since MoE substantially reduces both computation and memory access, it has become a pivotal technology in SOTA models [38], [61] for scaling model sizes. As illustrated in Fig. 2(a), the MoE layer comprises a gating network and multiple expert networks, each specializing in distinct domains. The gating network selects the *top-k* experts per token according to affinity scores. Tokens are then routed to their respective experts. After computation, expert outputs are weighted by the affinity scores and combined into a final output.

Fig. 2(b) illustrates a common deployment strategy using expert parallelism (EP) [29], [43], [66] for the MoE layer, which distributes experts across devices while maintaining each expert’s integrity. Although EP enhances performance, it introduces two key problems. First, with experts distributed across devices, tokens must be dispatched to devices hosting their assigned experts and subsequently recombined on their original devices after computation. These two all-to-all communications may incur latency up to 2.4 $\times$  that of computation (Fig. 1(a)), forming a major bottleneck.

Second, within each layer, certain experts stochastically attract more tokens, causing devices hosting these “hot”



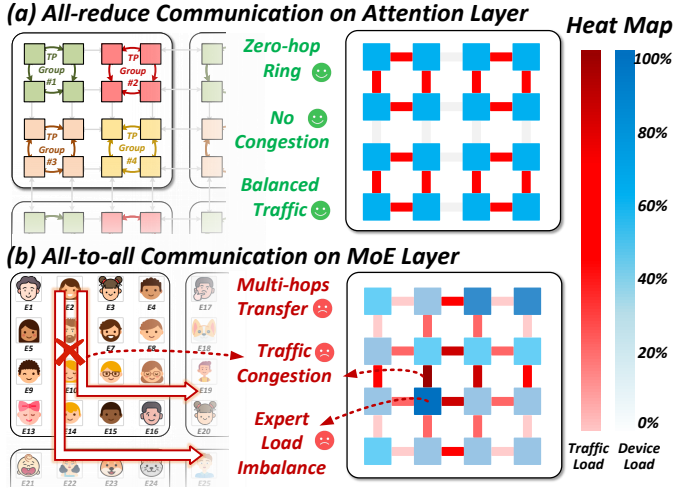


Fig. 5. (a) All-reduce of attention layer with  $DP=4$ ,  $TP=4$ . (b) All-to-all of MoE layer with  $EP=16$ . Each face denotes an expert.

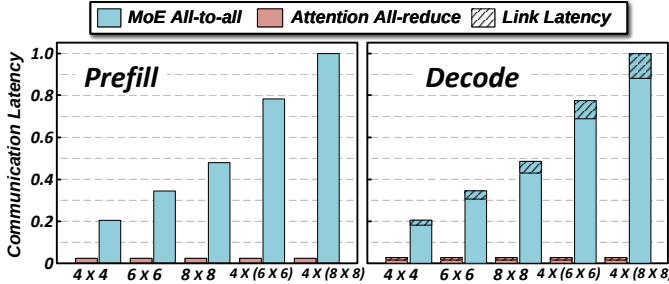


Fig. 6. Communication latency comparison between all-to-all and all-reduce; link latency during prefill stage is negligible and hence omitted.

The token generation phase in inference presents a severe memory bottleneck, necessitating a reduction in the number of experts per device to alleviate weight access pressure. Consequently, MoE relies on large-scale  $EP$  to minimize experts per device—potentially down to one. As illustrated in Fig. 4, increasing  $EP$  progressively reduces memory access ratio while improving per-device performance. However, due to high-overhead all-to-all communications, the optimal  $EP$  configuration for a cluster should match the number of devices covered by its high-performance network. For traditional DGX systems [10], this corresponds to  $EP=8\sim32$ . The NVL72 supernode [11] achieves  $EP=72$ , yielding a 35% performance gain. In contrast, WSCs enable  $EP=256$ , delivering a further 39% improvement and demonstrating significant potential for hosting huge MoE models. However, fully exploiting this potential requires addressing two critical challenges.

### B. Challenge One: Imbalanced Communication Pressure

Communication latency comprises two components: data transfer time (determined by data volume and bandwidth) and link latency (governed by physical implementation and protocols) [12]. This relationship is approximated by Eq. 1,

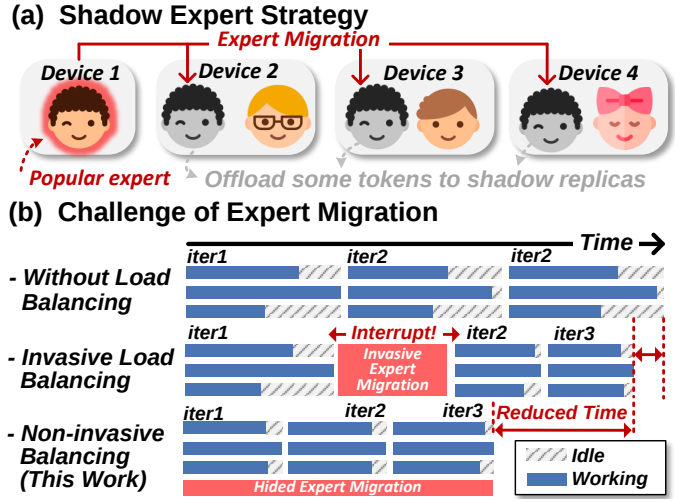


Fig. 7. (a) Shadow expert load balancing strategy. (b) The Comparison of without balancing, invasive-balancing, and non-invasive balancing strategies.

where both components are summed and multiplied by hop count. Consequently, longer distances increase communication overhead—even if individual links are fast.

$$latency = \left( \frac{volume}{bandwidth} + link\_latency \right) \times hops \quad (1)$$

Fig. 5 illustrates LLM deployment on a multi-WSC system using  $DP+TP$  for attention layers and  $EP$  for MoE layers. Inputs are partitioned into segments processed by distinct  $TP$  groups. After attention computation, ring all-reduce communication [18] aggregates results within each  $TP$  group. With hop count being one and WSC's high-performance network, this incurs minimal latency.

In contrast to the localized all-reduce, the subsequent MoE layer requires tokens to be dispatched and combined across the entire cluster via all-to-all communications. These exhibit greater complexity: tokens may reside on remote devices relative to their assigned experts, resulting in prevalent multi-hop cross-wafer transfers. Increased hop counts amplify both data transfer time and link latency, extending communication delays. Furthermore, stochastic token gating creates unpredictable point-to-point patterns that challenge orchestration, leading to concurrent transmissions congesting shared links. Expert load imbalance additionally induces traffic asymmetry that exacerbates congestion. Together, these factors make all-to-all communications count for significant time.

Fig. 6 compares both communications. Due to high data volume, latency is dominated by transfer time, though link latency contributes a portion in small-batch-size decoding. As WSCs scale from single 4x4 platforms to multi-wafer systems, all-reduce remains trivial while all-to-all latency surges dramatically. Consequently, WSCs' high-performance network only marginally reduces all-reduce latency—yielding limited practical benefit since baseline is already low enough to overlap with computation. Conversely, WSCs' mesh topol-

ogy makes all-to-all communications prohibitively expensive, establishing it as the system bottleneck. **The severe imbalance in communication pressure between all-reduce and all-to-all constitutes a critical challenge.**

### C. Challenge Two: Expert Load Balancing

The load of each expert fluctuates randomly, causing load imbalance and device underutilization—an issue exacerbated under large-scale *EP*. However, given the temporal continuity of load changes [65] and temporal similarity in expert selection, load prediction based on historical data is feasible. Prior works [6], [23], [65] have proposed balancing strategies for training systems. As shown in Fig. 7(a), devices reserve shadow slots beyond their native experts. The system predicts future popular experts using historical loads and dynamically replicates them to shadow slots on other devices. These replicas process portions of tokens for popular experts, achieving load balance.

Applying similar strategies to WSC inference systems presents unique challenges. In GPU systems, shadow slot reallocation copies expert weights from local disks via dedicated channels that avoid network contention [3]. However, WSC lacks on-wafer disks, forcing weight access through either wafer-edge connectors to external disks or on-wafer memory copies from devices hosting corresponding experts—both requiring high-volume multi-hop transfers across an already congested network. Furthermore, inference steps have short time spans, demanding agile balancing strategies that necessitate frequent expert migration. Yet inference serving imposes strict latency constraints. As Fig. 7(b) demonstrates, exposing migration on the critical path interrupts inference iterations and causes latency violations that negate balancing benefits. **Thus, ensuring balancing strategy agility without incurring latency overhead constitutes the primary challenge.**

## IV. ENTWINED RING MAPPING

As discussed in Section III-B, on WSCs, the communication latency is dominated by all-to-all while all-reduce contributes minimally. Considering that all-reduce can be overlapped with attention computation and has spare capacity, this raises a question: **can we leverage spare all-reduce capacity to alleviate all-to-all pressure?** In this section, we first explore their interaction, then propose a co-designed mapping strategy that balances communication pressure across them.

### A. The Definition of Full Token Domain

As displayed in Fig. 9, all-reduce consists of a reduce-scatter followed by an all-gather (AG). While prior works [31], [41] often omit AG to reduce overhead, we find that in mesh networks, AG shortens token-fetch distances and provides routing flexibility—especially critical in all-to-all-heavy MoE workloads (evaluated in Section VI-B6). Thus, we retain AG in all-reduce first.

The mapping of *TP* group in attention layers determines initial token distribution before gating, impacting MoE-layer

all-to-all overhead. To figure out this interaction, we innovatively propose the **Full Token Domain (FTD)** denoting the minimal set of devices that collectively hold tokens from all *TP* groups. Let  $D_{x,y}^s$  denote the device at coordinate  $(x, y)$  in the  $s^{th}$  *TP* group. With AG, each *TP* group device holds all group tokens. As Fig. 8(a) shows, the set  $\{D_{1,1}^1, D_{1,3}^2, D_{3,1}^3, D_{3,3}^4\}$  forms an FTD by including devices from all *TP* groups. Within an FTD, any device can access all required tokens, confining communication to this domain. Thus, the geometry of FTDs determines all-to-all overhead. As shown in Fig. 8(b), using FTD, we analyze all-to-all pressure from three perspectives:

- **Hops:** Assuming devices tend to access tokens from the nearest device of each *TP* group, we can find four  $3 \times 3$  area FTDs. Ignoring load imbalance, each FTD device has uniform probability (1/3) of accessing tokens from any of the other three devices. Summing probability-distance products yields an ideal average of 2.7 hops, implying  $2.7 \times$  longer data transfer time and link latency.
- **Congestion:** Accurate congestion analysis is challenging as it depends on the specific routing algorithm used. We adopt an intuitive approximation: links within an FTD experience similar utilization probabilities without inherent traffic imbalance. However, under baseline mapping, all FTDs overlap at the central four devices, causing links between central devices to be shared across FTDs. This overlap induces link congestion exacerbating latency.
- **Imbalance:** The impact of load imbalance on congestion depends on popular expert locations. Populer experts in FTD-intersection regions intensify congestion on central links, while edge-located popular experts reduce central traffic. However, for communication-computation overlap, worst-case analysis is necessary. Thus, with FTD intersections, load imbalance increases expected latency.

### B. Trade-off between All-to-all and All-reduce

Based on our analysis, all-to-all overhead is directly correlated with FTD area—larger FTDs increase average hop count and FTD intersection probability, exacerbating link congestion. To mitigate these costs, we propose Entwined Ring Mapping (ER-Mapping) minimizing the size of FTD. As shown in Fig. 8(c)(d), ER-Mapping preserves existing parallelism configurations while co-designing attention and MoE layer mappings to balance communication pressure.

1) **All-to-all Overhead Reduction:** An FTD must contain devices from all *TP* groups. In baseline mapping, *TP* groups are spaced apart, each located in a separate corner of the mesh, which results in a large FTD area. In contrast, ER-Mapping entwines *TP* groups by locating devices from different *TP* groups closely together at each corner, forming compact FTDs. As Fig. 8(c) demonstrates, the device set  $\{D_{1,1}^1, D_{1,2}^2, D_{2,1}^3, D_{2,2}^4\}$  forms an independent  $2 \times 2$  FTD. Though devices host different experts, all required tokens remain accessible within this compact domain. This configuration reduces average hops by  $2 \times$ . It also eliminates FTD intersections, mitigating congestion, thereby reducing all-to-all latency by more than  $2 \times$ .

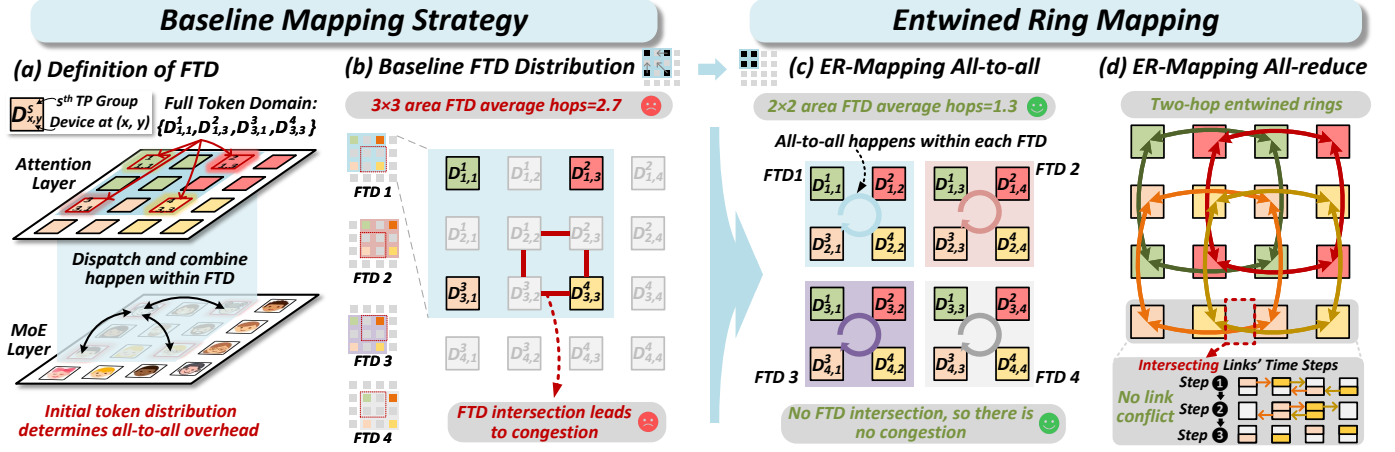


Fig. 8. (a) Definition of full token domain (FTD) and interaction of attention and MoE layers. (b) FTD distribution under baseline mapping with all FTDs intersect in the center area. (c) FTD distribution under ER-Mapping which eliminates all FTD intersections. (d) Diagram of entwined ring all-reduce.

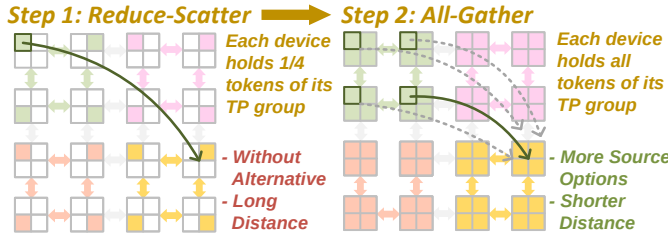


Fig. 9. Benefit of retaining all-gather (AG). When a yellow device requires the highlighted portion of tokens from the green TP group during MoE-layer all-to-all, AG provides more source options and shorter paths.

2) **All-reduce Latency Trade-off**: ER-Mapping achieves all-to-all latency reduction at the trade-off of all-reduce spare capacity. As Fig. 8(d) shows, after moving devices from different TP groups to neighboring position, the all-reduce is transformed into four entwined two-hop rings. Packages are sent bi-directionally, step by step. Although these rings have intersecting links, the transfers are time-staggered, so there is no link conflict. Consequently, while two-hop doubles the all-reduce latency, the intersection does not worsen the latency.

This trade-off remains advantageous. Since the base all-reduce latency is significantly shorter than all-to-all, a modest increase in all-reduce yields a substantial reduction in the more costly all-to-all. Furthermore, the increasing input sequence [36] and chain-of-thought lengths [17] in evolving LLMs dramatically expand attention computation time. This creates ample slack, allowing the longer all-reduce communication to be effectively overlapped. Consequently, the increased all-reduce latency is unlikely to degrade overall performance.

3) **Extend to General Cases**: Beyond the exemplary case, ER-Mapping can be readily extended to broader configurations using similar entwined-ring principles. We formalize the mapping algorithm in Fig. 10(a), which takes the parallelism and WSC scale and as inputs, then returns device sets for FTDs and

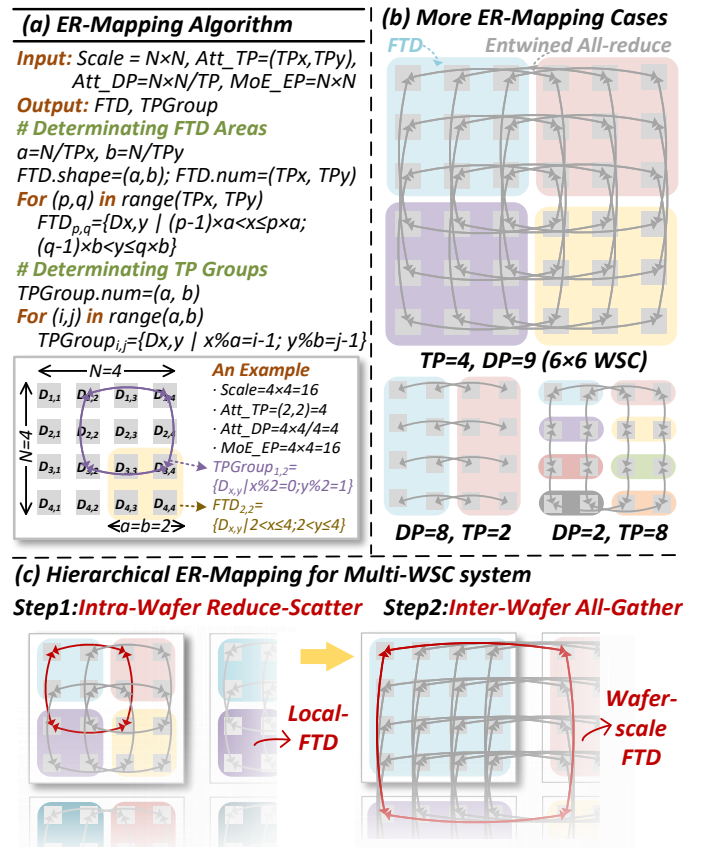


Fig. 10. (a) The ER-Mapping algorithm. (b) More mapping illustrations. (c) Hierarchical ER-Mapping for Multi-WSC System.

TP groups. These sets define the communication domains for all-to-all and all-reduce operations, respectively. As demonstrated in Fig. 10(b), ER-Mapping universally reduces the area of FTDs and eliminates their intersections, thereby balancing

communication load between all-reduce and all-to-all.

4) **Hierarchical ER-Mapping:** In larger-scale systems (e.g., multi-WSC), token distribution across multiple wafers makes single entwined-ring passes prohibitively expensive. Therefore, as shown in Fig. 10(c), the process splits into two hierarchical steps. First, intra-WSC reduce-scatter gathers tokens within each wafer into local FTDs—after this step, each device holds distinct token portions, enabling the entire wafer to function as a unified FTD. Second, inter-WSC all-gather aggregates tokens across wafers using these wafer-scale FTDs. Following both steps, each WSC contains tokens from all wafers, confining subsequent all-to-all exchanges within individual WSCs.

## V. NON-INVASIVE BALANCER

As discussed in Section III-C, it is critical to design an agile expert load balancing strategy while avoiding the introduction of extra latency. We therefore propose Non-invasive Balancer (NI-Balancer), which hides expert migration on the already-busy network while still delivering a satisfying and agile load balance during inference.

### A. Hiding Expert Migration Overhead

During inference, the network is constantly saturated by extensive all-reduce and all-to-all traffic. To conceal expert migration overhead, identifying idle links in the busy network is essential. Fig. 11(a) revisits the ER-Mapping design, revealing that neither the attention nor MoE layers achieve full link utilization. For all-reduce operations, links at ring intersections maintain constant activity while the others work for one cycle and then remain idle for the next cycle. Surprisingly, when flagging “hot” and “cold” links, all intra-FTD links are cold with hot links confined exclusively to FTD connection areas. This exposes spare intra-FTD bandwidth during all-reduce execution, permitting concurrent intra-FTD expert migration. Regarding MoE-layer all-to-all (Fig. 11(b)), communication occurs strictly within non-overlapping FTDs, leaving inter-FTD connection links entirely idle and thus available for simultaneous inter-FTD migration.

Fig. 11(c) further illustrates the heatmaps for more cases, which present similar complementary distribution of cold/hot links in these two communications. Consequently, expert migration decomposes into two operations: Local Migration (within FTDs) executed during all-reduce, and Global Migration (between FTDs) executed during all-to-all. Fig. 11(d) exemplifies a longest-distance migration decomposed into three stages: Local  $\rightarrow$  Global  $\rightarrow$  Local.

**Pipelining Strategy:** To prevent communication latency from being exposed on the critical path, it is common to overlap communication with computation. In this way, as long as the communication latency is shorter than the computation time, it is acceptable. As illustrated in Fig. 11(e), our kernel design leverages ER-Mapping’s balanced communication pressure to separately overlap all-reduce with attention computation and all-to-all with MoE computation. Inputs are split into

micro-batches pipelined through computation and communication streams. In addition, there is an independent migration stream, operating when expert migration is triggered. Local and Global migrations alternately occupy idle links during each layer’s attention and MoE phases, enabling zero-overhead expert migration without disrupting regular network traffic.

### B. Exploiting Temporal Locality of Expert Selection

During training, an auxiliary balance loss [21], [47] encourages uniform token distribution across experts to ensure sufficient training. However, this fails to guarantee satisfactory load balance during inference [28]. As profiled in Fig. 12, when experts are distributed across 8 devices, significant imbalance persists across all scenarios—peak device loads reach  $2.9\times$  the average, causing significant device underutilization.

However, further analysis reveals that while absolute loads remain imbalanced, device load ratios stabilize in fixed scenarios (e.g., Math-only) after initial inference iterations. This stability originates from two mechanisms: certain intrinsically popular experts consistently receive more tokens due to expert popularity bias [3], and fixed scenarios persistently activate corresponding domain-specific experts across token generations [65]. This presents a balancing opportunity where expert placement can be optimized once ratios stabilize post-warmup. However, production serving encounters cyclically evolving scenario mixtures [55], where request pools gradually transition between domains, inducing slow-varying load ratios. Consequently, dynamic load balancing that continuously adapts to shifting ratios is essential.

$$\begin{cases} \sum_{i=1}^L \frac{\max(\text{load}_i) - \mu(\text{load}_i)}{\mu(\text{load}_i)} > \alpha \\ \Delta t_{\text{mig}} > \beta \quad (\beta = 0 \text{ for non-invasive}) \end{cases} \quad (2)$$

We propose a template for balancing strategy in Eq. 2, where  $\text{load}_i$  represents device loads at layer  $i$ , and  $\mu$  denotes average load. The layer imbalance degree quantifies maximum load deviation from average. Balancing triggers when the cumulative imbalance across  $L$  layers exceeds threshold  $\alpha$  and the time since last migration  $\Delta t_{\text{mig}}$  exceeds  $\beta$ . Both  $\alpha$  and  $\beta$  are tuning parameters. For invasive balancing, token iteration interrupts to replicate popular experts to the slots of underutilized devices, with  $\beta$  preventing excessive interruptions. For non-invasive balancing ( $\beta = 0$ ), migration overhead is concealed, enabling continuous fine-tuning of slots assignments.

### C. Enhancing Balancing Agility

To determine migration sources and destinations, prior works [6], [23], [65] employ greedy algorithms that reassign shadow slots by directly copying the hottest expert to the coldest device. While achieving balanced load, these approaches neglect migration overhead—which can negate benefits—as selecting remote slots incurs substantially higher latency than choosing neighboring ones. Therefore, topology awareness is essential for maintaining algorithm agility.

Algorithm 1 presents our topology-aware balancing strategy. We derive *Load* from historical iteration statistics to predict

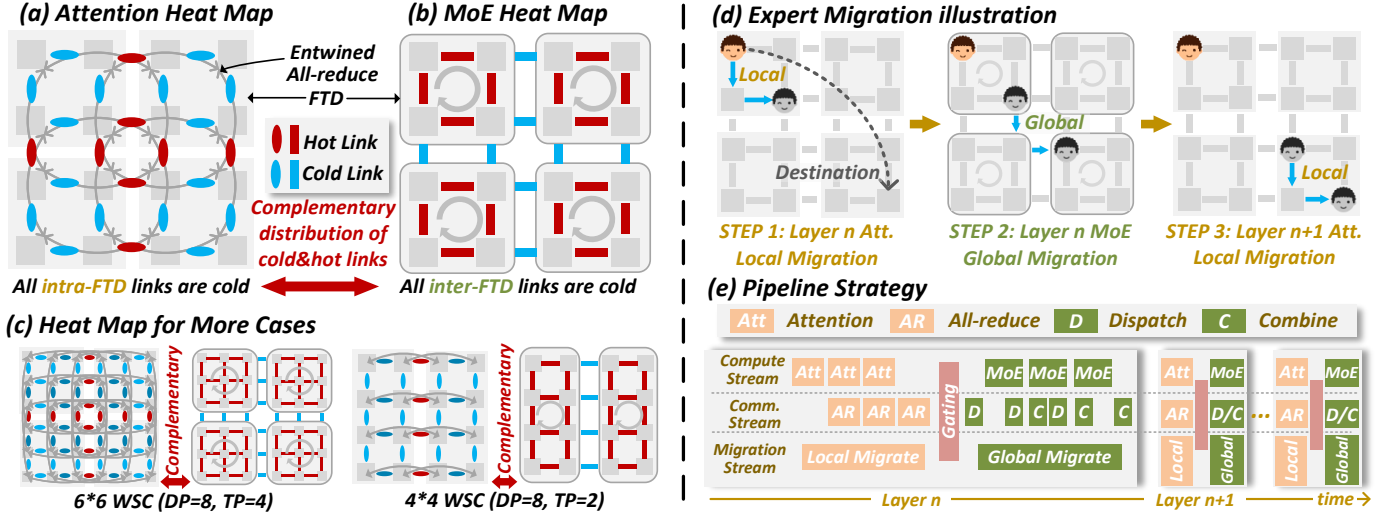


Fig. 11. Traffic heatmap for (a) the all-reduce of attention layer and (b) the all-to-all of MoE layer. (c) Heatmap for more cases. The distribution of hot/cold links are complementary in all cases. (d) An illustration of expert migration. (e) The diagram inference kernels with a dedicated stream for expert migration.

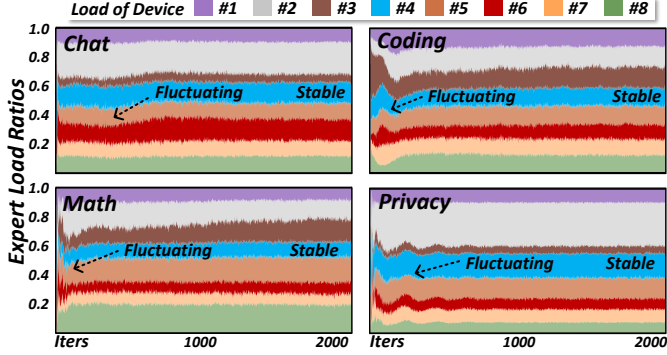


Fig. 12. The expert load trace across Chat [7], Coding [1], Math [9], and Privacy [8] scenarios. We employ Qwen3-234B with  $EP=8$ . Each color represents one of the device load ratios. Within all scenarios, the load ratios achieve stable after a brief warm-up.

expert loads.  $Num_e$  denotes the number of devices hosting expert  $e$  (initialized to 1), and  $Load_e/Num_e$  represents the per-device load when shared. Device  $Heat_d$  is defined as the sum of  $Load_e/Num_e$  for all experts on device  $d$ . Unlike training systems aiming for uniform token distribution, inference optimization focuses solely on reducing peak device load. Rather than targeting the globally hottest expert, we select the most popular expert on the highest- $Heat$  device as the migration source. Devices whose  $Heat_d$  would not exceed the current maximum after hosting this expert constitute the  $cold\_d$  set. If  $cold\_d$  is empty or lacks available shadow slots, the algorithm terminates. Since any  $cold\_d$  device equally reduces peak  $Heat$ , we select the **topologically nearest** device to minimize migration latency. After copying the source expert to the target's shadow slot, we increment  $Num_e$  and update device heats. The process repeats until termination.

#### Algorithm 1: Topology-aware Balancing

**Data:**  $Load_e$ , historical average load of  $e^{th}$  expert  
**Data:**  $Num_e$ , number of device hosting  $e^{th}$  expert  
**Data:**  $Device_d$ , experts hosted in  $d^{th}$  device  
**Data:**  $Heat_d$ , cumulative load of  $d^{th}$  device

- 1  $Num \leftarrow \{1\}$ ;  $Heat_d \leftarrow \sum \frac{Load_e}{Num_e}$  for  $e \in Device_d$ ;
- 2 **while** True **do**
- 3    $hottest\_d \leftarrow \max\{Heat_d\}$ ;
- 4    $src\_e \leftarrow \max\{\frac{Load_e}{Num_e}\}$  for  $e \in Device_{hottest\_d}$ ;
- 5    $cold\_d \leftarrow d$  for  $Heat_d < Heat_{hottest\_d} - \frac{Load_{src\_e}}{Num_{src\_e}}$ ;
- 6   **Break if**  $cold\_d$  is empty **or** no\_slots in  $cold\_d$ ;
- 7    $des\_d \leftarrow nearest\{d \text{ for } d \in cold\_d\}$ ;
- 8   Copy  $src\_e$  to  $des\_d$ ;
- 9    $Num_{src\_e} + 1$ ; Update  $Device_d$  and  $Heat_d$ ;

## VI. EVALUATION

### A. Evaluation Setup

1) **Platform Setup:** To ensure fairness, we assume each device in the WSC is equivalent to an NVIDIA B200 GPU [10] capable of 2250 TFLOPS@FP16, equipped with 180GB HBM featuring 8TB/s access bandwidth. According to Tesla Dojo [56], the bidirectional communication bandwidth of a single die and one-border of cross-wafer bandwidth are set at 8TB/s and 9TB/s respectively. For attention layers and all communications, we employ FP16 precision, while other linear operations utilize INT8 quantization. A minimal 4x4 WSC configuration delivers 35 PFLOPS and 2.8TB memory capacities, sufficient for these huge MoE models.

2) **Methodology:** We employ a profile-and-simulate methodology for experimentation of both WSC and GPU clusters. The evaluator is built upon ASTRA-sim 2.0 [60]—a widely recognized open-source distributed ML simulator

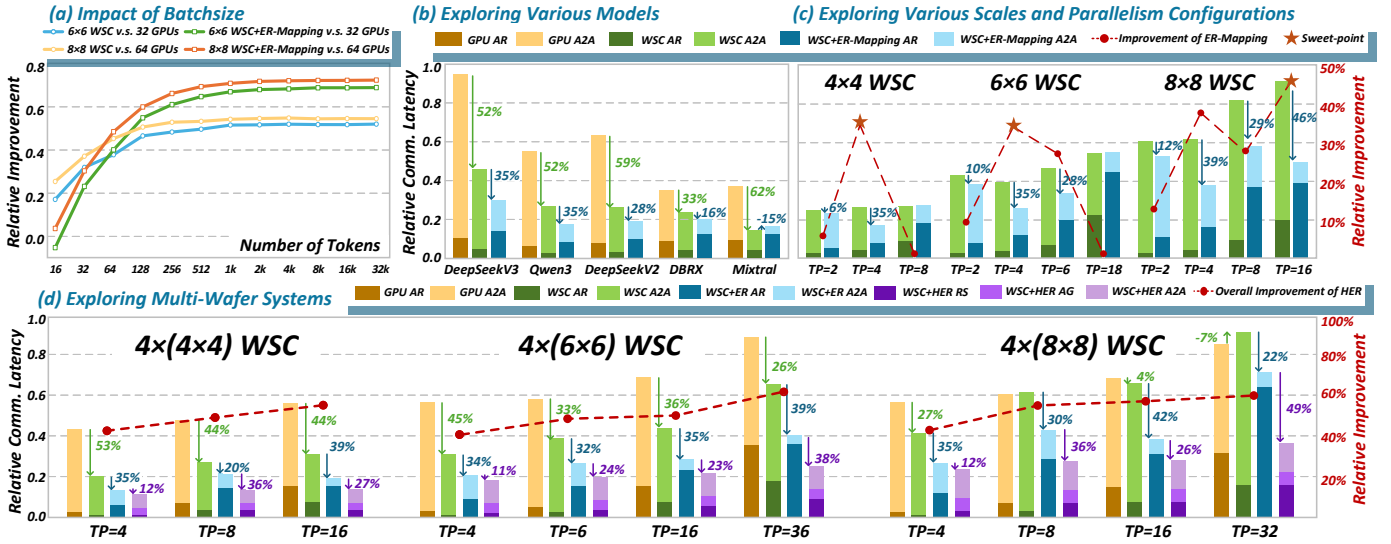


Fig. 13. (a) Communication improvement of WSC over DGX under different token counts. (b) Performance of ER-Mapping under various models. (c) Exploring the impact of WSC scales and parallelism. (d) Performance of Hierarchical ER-Mapping.

featuring a dedicated analytical backend for network simulation. To capture dynamic inference characteristics, we profile all benchmark requests on the B200 using the vLLM framework [32], recording input/output lengths and expert selection traces. For alignment with GPU baselines, we profile FlashInfer kernels [63] across diverse input shapes on the B200, compiling a dataset of computation and memory access performance. Regarding communication performance, we first enhance ASTRA-sim’s network backend with mesh-topology support. Additionally, we extend its system layer with multi-hop ring collective and point-to-point communication capabilities to support ER-Mapping and NI-Balance.

3) **MoE Models:** To validate our optimization across various all-to-all communication overheads and parallelism configurations, as listed in Table I, we selected SOTA MoE models with different activated expert numbers and expert sizes. This two parameters determine the magnitude of all-to-all overhead and the optimal parallel configuration, respectively.

TABLE I  
PARAMETERS OF EVALUATION MOE MODELS

| Models             | Size | Layers<br>Sparse/Total | Single<br>Expert Size | Experts<br>Activated/Total |
|--------------------|------|------------------------|-----------------------|----------------------------|
| DeepSeek-V3 [38]   | 671B | 58 / 61                | 42MB                  | 8 / 256                    |
| Qwen3 [61]         | 235B | 94 / 94                | 18MB                  | 8 / 128                    |
| DeepSeek-V2 [19]   | 236B | 59 / 60                | 23MB                  | 6 / 160                    |
| DBRX [2]           | 132B | 40 / 40                | 189MB                 | 4 / 16                     |
| Mixtral-8x22B [30] | 141B | 56 / 56                | 288MB                 | 2 / 8                      |

### B. Performance of Entwined Ring Mapping

We first explore the communication benefits of ER-Mapping across various scenarios to demonstrate its generality. To clearly isolate the sources of benefits, we initially disregard expert load imbalance. By adjusting the gating function of the

MoE layer to equalize the probability of each expert being selected, we ensure balanced loads. The baseline system uses DGX B200 GPU nodes, each equipped with 8 devices, accelerated with hierarchical network communication optimization [46]. Both GPU and WSC employ optimizations similar to PipeMoE [49] to determine the optimal pipeline stages for communication-computation fusion.

1) **Impact of Token Count:** To investigate how token count affects communication performance, we compare a 6x6 wafer with a 4-node DGX and an 8x8 wafer with an 8-node DGX. As the number of tokens per TP group increases (Fig. 13(a)), link latency impact diminishes, and WSC’s advantage over DGX grows rapidly. Beyond 256 tokens, WSC consistently outperforms DGX by 54%, while ER-Mapping further extends this advantage to 73%. Since token counts exceeding 256 per group are achievable in both prefill and decode stages, subsequent communication experiments fix token counts at 256 without distinguishing stages.

2) **Exploring Various Models:** We compare a 6x6 WSC with a 4-node DGX to explore communication benefits under various models. Fig. 13(b) shows that, benefited from unified high-performance network, pure WSC outperforms DGX by an average of 56%. Additionally, in both DGX and WSC architectures, all-to-all latency is significantly higher than all-reduce, which remains minimal. ER-Mapping balances this communication pressure imbalance, substantially reducing all-to-all latency and delivering up to 35% additional performance gains. Furthermore, since all-to-all communication overhead scales directly with the number of activated experts, ER-Mapping’s benefits increase correspondingly. However, for models like Mixtral [30] that activate only two experts, all-to-all overhead remains relatively small while original all-reduce overhead is comparatively large. In such cases, naive

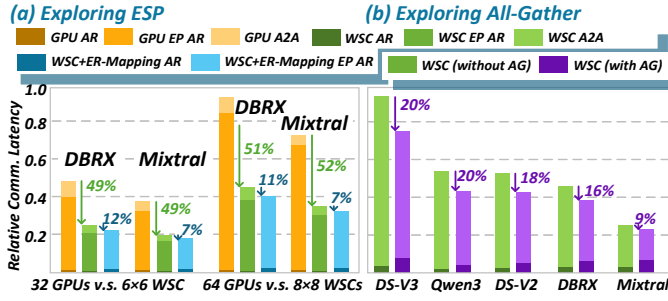


Fig. 14. (a) ER-Mapping performance under ESP parallelism. (b) Justifying the retaining of all-gather.

ER-Mapping may fail to yield benefits.

3) *Exploring Various Scales and Parallelism*: We further focus on the Qwen3 to explore the impact of different configurations. As shown in Fig. 13(c), ER-Mapping consistently outperforms the baseline, achieving improvements of up to 46%. As  $TP$  increases, the total token count grows, resulting in higher communication overhead. Moreover, ER-Mapping’s benefits do not scale linearly with parallelism; they are governed by the geometry of FTDs and entwined-rings, and the all-to-all/all-reduce ratio. Consequently, optimal configurations exist—for example, an  $8 \times 8$  WSC at  $TP=16$ —where the topology minimizes all-to-all latency while maintaining acceptable all-reduce overhead, yielding peak acceleration.

4) *Exploring Multi-WSC Systems*: Established in Section IV-B4, for large-scale WSC clusters, Hierarchical ER-Mapping (HER-Mapping) is introduced to reduce multi-hop all-reduce overhead across wafers. As Fig. 13(d) shows, HER-Mapping decouples all-reduce into two hierarchical phases—reduce-scatter and all-gather—thus further minimizing all-reduce overhead and delivering up to 62% performance gain. Unlike pure ER-Mapping, whose performance gains vary significantly across parallelism configurations, HER-Mapping achieves consistent improvement over the baseline mapping in all cases.

5) *Discussion for ESP Parallelism*: Some models employ few but large-size experts (e.g., DBRX [2] and Mixtral [30]), where the substantial expert size permits further slicing. This motivates *ESP* (Expert Sharding Parallelism), which further partitions individual experts across devices based on *EP*. *ESP* necessitates all-to-all communication to gather tokens across *EP* groups, followed by all-reduce operations to aggregate partial sums within *EP* groups. ER-Mapping remains effective in this context: each FTD hosts several experts while distributing their slices across devices. Crucially, because all tokens across  $TP$  groups reside within each FTD, the all-to-all communications is eliminated. As demonstrated in Fig. 14(a), WSC outperforms DGX by 50% on average, with ER-Mapping still surpassing the baseline. However, since latency is dominated by all-reduce operations within *EP* groups, ER-Mapping yields only a further 9% average improvement.

6) *Discussion for the Retaining of All-gather*: As established in Section IV-A, we retain the all-gather operation in

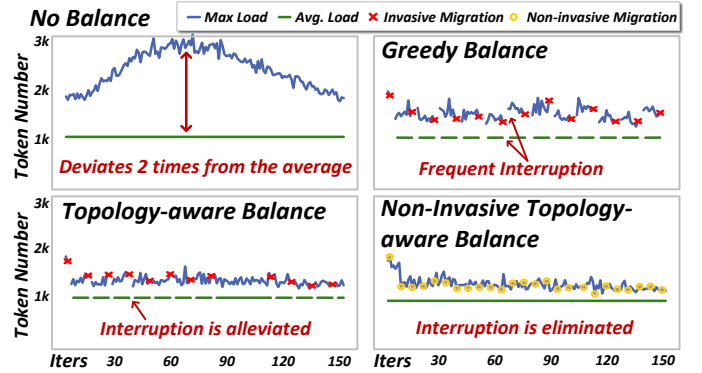


Fig. 15. Run-time trace of expert loads. The green line denotes ideal average load, and the intervals on it mean the interruptions.

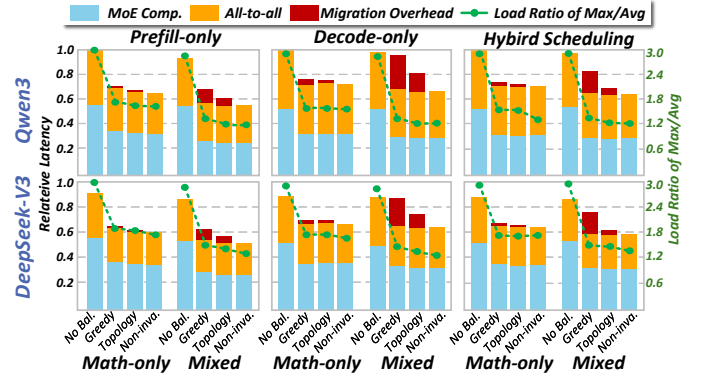


Fig. 16. Comparison of different balancing strategies.

the attention layer, which reduces communication distance and expands path diversity for subsequent all-to-all communications. Fig. 14(b) demonstrates that while this design doubles all-reduce latency, the overhead is not significant due to the inherently low all-reduce latency. Crucially, the latency reduction from all-to-all communication offsets this cost. Consequently, after introducing AG, the performance is even improved by average 17%, which builds the foundation for the subsequent ER-Mapping design.

### C. Performance of Non-invasive Balancer

From this section, we study the impact of load imbalance and dynamic input/output lengths. We evaluate both Disaggregated-LLM [26], [45], [67], which separates prefill and decode on distinct platforms, and Hybrid Scheduling [13], [14], [24], which mixes them in a batch. For workloads, we leverage Evidently AI’s open-source benchmark collection [4] covering four representative inference scenarios: Chat [7], Coding [1], Math [9], and Privacy Agent [8], where we construct single-scenario exclusively using the Math benchmark and generate mixed-scenario by integrating request arrival traces from Azure [5] to combine all four benchmarks. The baseline greedy balancer is from EPLB [6].

1) **Run-time Load Traces:** Fig. 15 presents run-time traces of device loads and expert migrations. Without load balancing, the maximum load deviates by  $2\times$  from the average, causing severe imbalance and low hardware utilization. Greedy balancing reduces this deviation to approximately  $0.4\times$ . However, as an invasive method, it frequently interrupts inference iterations to perform expert migrations—triggered on average every 10 iterations with overhead equivalent to 2 iterations. In contrast, topology-aware balancing reduces migration distance, thereby mitigating interruptions while improving load balance. Finally, topology-aware non-invasive balancing eliminates interruption overhead entirely, allowing the balancer to remain continuously active and migrate experts whenever minor adjustments to shadow slots are required, achieving satisfactory balance.

2) **Performance Improvement after Load Balancing:** We evaluate load balancing impacts across scenarios, result is displayed in 16. As discussed in Section V-B, fixed scenarios (e.g., Math-only) stabilize load ratios after warm-up, minimizing expert migrations. However, in more common mixed scenarios, fluctuating load ratios trigger frequent migrations. Under Prefill-only, migration overhead constitutes 22% per iteration. For Decode-only or Hybrid Scheduling, due to shorter iterations time, this surges to 45%, potentially offsetting balancing benefits and degrading performance. Topology-aware balancing reduces migration overhead by  $2.6\times$  on average. Non-invasive balancing eliminates overhead completely, achieving optimal load balance while reducing MoE computation by up to 54%. Additionally, balanced traffic decreases all-to-all communication time by 23% on average.

#### D. Ablation Study of Overall Performance

We select NVL72 [11], NVIDIA’s SOTA supernode integrating 72 devices with a unified high-performance network, as the baseline. Dedicated NVMe channels are adopted to hide expert migration overhead [3]. For WSC, we configure a multi-WSC system using four  $8\times 8$  wafers (256 devices total).

As illustrated in Fig. 17, NVL72 also exhibits load imbalance. Its  $EP=72$  setup (multiple experts per device) leads to memory access dominating execution time, restricting load balancing gains to just 26% computational enhancement. WSC uses  $EP=256$ . However, rather than alleviating memory access overhead, the single-expert-per-device allocation worsens load imbalance. Moreover, the mesh topology results in all-to-all latency greatly surpassing computation time.

ER-Mapping reduces all-to-all communication by 30%, while HER-Mapping amplifies this reduction to 71%, eliminating communication bottlenecks. Subsequent load balancing decreases computation and communication overhead by 49% and 20% respectively. However, expert migration overhead on the critical path degrades overall performance. Topology-aware balancing reduces this overhead by 67%, and non-invasive balancing eliminates it entirely. Ultimately, our optimizations remove both communication and migration bottlenecks. Compared to NVL72, WSC achieves a significantly larger  $EP$ , delivering an average 39% higher per-device MoE performance.

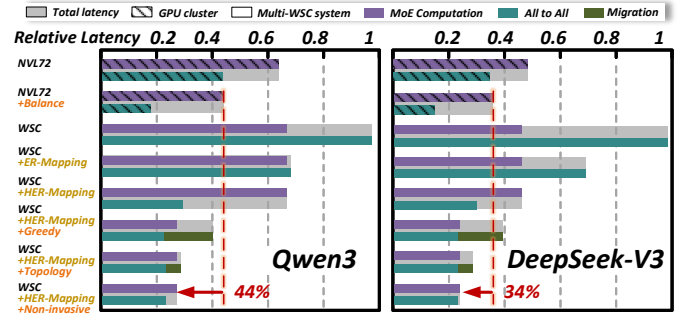


Fig. 17. Multi-WSC cluster v.s. NVL72 supernode.

## VII. RELATED WORK

**Communication Optimization:** Prior studies [38], [40], [41] have optimized all-to-all communication on GPU platforms. They primarily alleviate negative impact of low-bandwidth domains through hierarchical network utilization via combined or overlapped intra-node/inter-node communications. However, WSC fundamentally differs by employing a unified high-performance network where all traffic shares homogeneous links, rendering GPU-centric approaches unsuitable. Our work uniquely exploits WSC’s mesh topology to co-design all-to-all and all-reduce communications, distinguishing it from existing methods. While Lina [34] optimizes all-reduce for training gradients—inapplicable to our inference scenario—and Chimera [44] explores communication fusion, MoE’s gating network between all-reduce and all-to-all precludes such fusion.

**Topology Awareness:** To reduce the communication overhead, DeepSeek [38] and LocMoE [35] proposes Node-Limited Routing, which bounds the number of nodes a token can be routed to. TA-MoE [16] incorporates topology-aware communication-cost penalties into training loss. These GPU-focused methods disregard mesh networks and inevitably constrain model capacity through routing limitations. In contrast, our approach imposes no token routing restrictions, explicitly accounts for mesh topology, and establishes a flexible platform for general MoE models without compromising capacity.

**Expert Load Balance:** Prior works [21], [47] introduce an auxiliary loss to guarantee balanced training. However, expert loads remain imbalanced at inference time [28]. GShard [33] imposes a capacity threshold and directly drop tokens that exceed it, ensuring load balance yet incurring accuracy loss [65]. Consequently, dynamic load balancing at inference remains necessary. Existing balancing strategies like EPLB [6], FlexMoE [65], and FasterMoE [23] typically rely on greedy algorithms, which interrupt inference and introduce significant expert migration latency. In contrast, our work explicitly accounts for migration cost, preserving algorithmic agility and hiding the migration traffic within the already busy network without causing any interruption.

## CONCLUSION

WSC presents a promising platform for hosting huge MoE models, yet their architectural distinct from conventional GPU clusters. Directly porting prior techniques hinders full utilization of WSC potential. This work introduces ER-Mapping, significantly reducing all-to-all latency on mesh networks through balanced communication. Building upon this, NI-Balancer achieves optimal load balance while concealing expert migration overhead within existing network operations. Collectively, these innovations enable WSC to deliver 39% higher per-device performance compared to NVL72 supernodes.

## REFERENCES

- [1] “cruzeval/data at main · facebookresearch/cruzeval — github.com,” <https://github.com/facebookresearch/cruzeval/tree/main/data>, [Accessed 21-07-2025].
- [2] “databricks/dbx-instruct · Hugging Face — huggingface.co,” <https://huggingface.co/databricks/dbx-instruct>, [Accessed 21-07-2025].
- [3] “Deploying DeepSeek with PD Disaggregation and Large-Scale Expert Parallelism on 96 H100 GPUs — LMSYS Org — lmsys.org,” <https://lmsys.org/blog/2025-05-05-large-scale-ep/#large-scale-expert-parallelism>, [Accessed 21-07-2025].
- [4] “Evidently AI - 200 LLM benchmarks and evaluation datasets — evidentlyai.com,” <https://www.evidentlyai.com/llm-evaluation-benchmarks-datasets>, [Accessed 22-07-2025].
- [5] “GitHub - Azure/AzurePublicDataset: Microsoft Azure Traces — github.com,” <https://github.com/Azure/AzurePublicDataset/tree/master>, [Accessed 22-07-2025].
- [6] “GitHub - deepseek-ai/EPLB: Expert Parallelism Load Balancer — github.com,” <https://github.com/deepseek-ai/EPLB>, [Accessed 21-07-2025].
- [7] “GitHub - ekwin117/multi-challenge — github.com,” <https://github.com/ekwin117/multi-challenge>, [Accessed 21-07-2025].
- [8] “GitHub - HowieHwong/TrustLLM: [ICML 2024] TrustLLM: Trustworthiness in Large Language Models — github.com,” <https://github.com/HowieHwong/TrustLLM/tree/main>, [Accessed 21-07-2025].
- [9] “GitHub - sarahmart/HARDMath: A new dataset of difficult graduate-level applied mathematics problems; evaluations demonstrate that leading LLMs currently exhibit low accuracy in solving these problems. — github.com,” <https://github.com/sarahmart/HARDMath/tree/main>, [Accessed 21-07-2025].
- [10] “NVIDIA DGX B200 — nvidia.com,” <https://www.nvidia.com/en-us/data-center/dgx-b200/>, [Accessed 31-07-2025].
- [11] “NVIDIA GB200 NVL72 — nvidia.com,” <https://www.nvidia.com/en-us/data-center/gb200-nvl72/>, [Accessed 18-04-2025].
- [12] “Scaling-out ethernet for the data center,” <https://network.nvidia.com/pdf/whitepapers/WP-ethernet%20scaleout-WEB.pdf>, [Accessed 21-07-2025].
- [13] A. Agrawal, J. Chen, Í. Goiri, R. Ramjee, C. Zhang, A. Tumanov, and E. Choukse, “Mnemosyne: Parallelization strategies for efficiently serving multi-million context length llm inference requests without approximations,” *arXiv preprint arXiv:2409.17264*, 2024.
- [14] A. Agrawal, N. Kedia, A. Panwar, J. Mohan, N. Kwatra, B. Gulavani, A. Tumanov, and R. Ramjee, “Taming throughput-latency tradeoff in llm inference with sarathi-serve,” in *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, 2024, pp. 117–134.
- [15] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *European conference on computer vision*. Springer, 2020, pp. 213–229.
- [16] C. Chen, M. Li, Z. Wu, D. Yu, and C. Yang, “Ta-moe: Topology-aware large scale mixture-of-expert training,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 22 173–22 186, 2022.
- [17] Q. Chen, L. Qin, J. Liu, D. Peng, J. Guan, P. Wang, M. Hu, Y. Zhou, T. Gao, and W. Che, “Towards reasoning era: A survey of long chain-of-thought for reasoning large language models,” *arXiv preprint arXiv:2503.09567*, 2025.
- [18] D. De Sensi, T. Bonato, D. Saam, and T. Hoefer, “Swing: Short-cutting rings for higher bandwidth allreduce,” in *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, 2024, pp. 1445–1462.
- [19] DeepSeek-AI, A. Liu, B. Feng, B. Wang, B. Wang, B. Liu, C. Zhao, C. Dengr, C. Ruan, D. Dai, D. Guo, D. Yang, D. Chen, D. Ji, E. Li, F. Lin, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Xu, H. Yang, H. Zhang, H. Ding, H. Xin, H. Gao, H. Li, H. Qu, J. L. Cai, J. Liang, J. Guo, J. Ni, J. Li, J. Chen, J. Yuan, J. Qiu, J. Song, K. Dong, K. Gao, K. Guan, L. Wang, L. Zhang, L. Xu, L. Xia, L. Zhao, L. Zhang, M. Li, M. Wang, M. Zhang, M. Zhang, M. Tang, M. Li, N. Tian, P. Huang, P. Wang, P. Zhang, Q. Zhu, Q. Chen, Q. Du, R. J. Chen, R. L. Jin, R. Ge, R. Pan, R. Xu, R. Chen, S. S. Li, S. Liu, S. Zhou, S. Chen, S. Wu, S. Ye, S. Ma, S. Wang, S. Zhou, S. Yu, S. Zhou, S. Zheng, T. Wang, T. Pei, T. Yuan, T. Sun, W. L. Xiao, W. Zeng, W. An, W. Liu, W. Liang, W. Gao, W. Zhang, X. Q. Li, X. Jin, X. Wang, X. Bi, X. Liu, X. Wang, X. Shen, X. Chen, X. Chen, X. Nie, X. Sun, X. Wang, X. Liu, X. Xie, X. Yu, X. Song, X. Zhou, X. Yang, X. Lu, X. Su, Y. Wu, Y. K. Li, Y. X. Wei, Y. X. Zhu, Y. Xu, Y. Huang, Y. Li, Y. Zhao, Y. Sun, Y. Li, Y. Wang, Y. Zheng, Y. Zhang, Y. Xiong, Y. Zhao, Y. He, Y. Tang, Y. Piao, Y. Dong, Y. Tan, Y. Liu, Y. Wang, Y. Guo, Y. Zhu, Y. Wang, Y. Zou, Y. Zha, Y. Ma, Y. Yan, Y. You, Y. Liu, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Huang, Z. Zhang, Z. Xie, Z. Hao, Z. Shao, Z. Wen, Z. Xu, Z. Zhang, Z. Li, Z. Wang, Z. Gu, Z. Li, and Z. Xie, “Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model,” 2024. [Online]. Available: <https://arxiv.org/abs/2405.04434>
- [20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” in *International Conference on Learning Representations*, 2021. [Online]. Available: <https://openreview.net/forum?id=YicbFdNTTy>
- [21] W. Fedus, B. Zoph, and N. Shazeer, “Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity,” *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022.
- [22] T. Gale, D. Narayanan, C. Young, and M. Zaharia, “Megablocks: Efficient sparse training with mixture-of-experts,” *Proceedings of Machine Learning and Systems*, vol. 5, pp. 288–304, 2023.
- [23] J. He, J. Zhai, T. Antunes, H. Wang, F. Luo, S. Shi, and Q. Li, “Fastermoe: modeling and optimizing training of large-scale dynamic pre-trained models,” in *PPoPP*, 2022.
- [24] C. Holmes, M. Tanaka, M. Wyatt, A. A. Awan, J. Rasley, S. Rajbhandari, R. Y. Aminabadi, H. Qin, A. Bakhtiari, L. Kurilenko *et al.*, “Deepspeed-fastgen: High-throughput text generation for llms via mii and deepspeed-inference,” *arXiv preprint arXiv:2401.08671*, 2024.
- [25] S. Hou, W. C. Chen, C. Hu, C. Chiu, K. Ting, T. Lin, W. Wei, W. Chiou, V. J. Lin, V. C. Chang *et al.*, “Wafer-level integration of an advanced logic-memory system through the second-generation cowos technology,” *IEEE Transactions on Electron Devices*, vol. 64, no. 10, pp. 4071–4077, 2017.
- [26] C. Hu, H. Huang, L. Xu, X. Chen, J. Xu, S. Chen, H. Feng, C. Wang, S. Wang, Y. Bao *et al.*, “Inference without interference: Disaggregate llm inference for mixed downstream workloads,” *arXiv preprint arXiv:2401.11181*, 2024.
- [27] Y. Hu, X. Lin, H. Wang, Z. He, X. Yu, J. Zhang, Q. Yang, Z. Xu, S. Guan, J. Fang *et al.*, “Wafer-scale computing: Advancements, challenges, and future perspectives [feature],” *IEEE Circuits and Systems Magazine*, vol. 24, no. 1, pp. 52–81, 2024.
- [28] H. Huang, N. Ardalani, A. Sun, L. Ke, H.-H. S. Lee, A. Sridhar, S. Bhosale, C.-J. Wu, and B. Lee, “Towards moe deployment: Mitigating inefficiencies in mixture-of-expert (moe) inference,” *arXiv preprint arXiv:2303.06182*, 2023.
- [29] C. Hwang, W. Cui, Y. Xiong, Z. Yang, Z. Liu, H. Hu, Z. Wang, R. Salas, J. Jose, P. Ram *et al.*, “Tutel: Adaptive mixture-of-experts at scale,” *Proceedings of Machine Learning and Systems*, vol. 5, pp. 269–287, 2023.
- [30] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, “Mixtral of experts,” 2024. [Online]. Available: <https://arxiv.org/abs/2401.04088>

- [31] V. A. Korthikanti, J. Casper, S. Lym, L. McAfee, M. Andersch, M. Shoeybi, and B. Catanzaro, "Reducing activation recomputation in large transformer models," *Proceedings of Machine Learning and Systems*, vol. 5, pp. 341–353, 2023.
- [32] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.
- [33] D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen, "Gshard: Scaling giant models with conditional computation and automatic sharding," in *International Conference on Learning Representations*.
- [34] J. Li, Y. Jiang, Y. Zhu, C. Wang, and H. Xu, "Accelerating distributed {MoE} training and inference with lina," in *2023 USENIX Annual Technical Conference (USENIX ATC 23)*, 2023, pp. 945–959.
- [35] J. Li, Z. Sun, X. He, L. Zeng, Y. Lin, E. Li, B. Zheng, R. Zhao, and X. Chen, "Locmoe: A low-overhead moe for large language model training," *arXiv preprint arXiv:2401.13920*, 2024.
- [36] T. Li, G. Zhang, Q. D. Do, X. Yue, and W. Chen, "Long-context llms struggle with long in-context learning," *arXiv preprint arXiv:2404.02060*, 2024.
- [37] S. Lie, "Wafer-scale ai: Gpu impossible performance," in *2024 IEEE Hot Chips 36 Symposium (HCS)*. IEEE Computer Society, 2024, pp. 1–71.
- [38] 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.
- [39] S. Pal, J. Liu, I. Alam, N. Cebry, H. Suhail, S. Bu, S. S. Iyer, S. Pamarti, R. Kumar, and P. Gupta, "Designing a 2048-chiplet, 14336-core waferscale processor," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 1183–1188.
- [40] X. Pan, W. Lin, S. Shi, X. Chu, W. Sun, and B. Li, "Parm: Efficient training of large sparsely-activated models with dedicated schedules," in *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*. IEEE, 2024, pp. 1880–1889.
- [41] X. Pan, W. Lin, L. Zhang, S. Shi, Z. Tang, R. Wang, B. Li, and X. Chu, "Fsmoe: A flexible and scalable training system for sparse mixture-of-experts models," in *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, 2025, pp. 524–539.
- [42] S. Potluri, K. Hamidouche, A. Venkatesh, D. Bureddy, and D. K. Panda, "Efficient inter-node mpi communication using gpubdirect rdma for infiniband clusters with nvidia gpus," in *2013 42nd International Conference on Parallel Processing*. IEEE, 2013, pp. 80–89.
- [43] Y. Qian, F. Li, X. Ji, X. Zhao, J. Tan, K. Zhang, and X. Cai, "Eps-moe: Expert pipeline scheduler for cost-efficient moe inference," *arXiv preprint arXiv:2410.12247*, 2024.
- [44] L. Qin, J. Cui, W. Cai, and J. Huang, "Chimera: Communication fusion for hybrid parallelism in large language models," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, 2025, pp. 498–513.
- [45] R. Qin, Z. Li, W. He, J. Cui, F. Ren, M. Zhang, Y. Wu, W. Zheng, and X. Xu, "Mooncake: Trading more storage for less computation — a KVCache-centric architecture for serving LLM chatbot," in *23rd USENIX Conference on File and Storage Technologies (FAST 25)*. Santa Clara, CA: USENIX Association, Feb. 2025, pp. 155–170. [Online]. Available: <https://www.usenix.org/conference/fast25/presentation/qin>
- [46] S. Rajbhandari, C. Li, Z. Yao, M. Zhang, R. Y. Aminabadi, A. A. Awan, J. Rasley, and Y. He, "Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale," in *International conference on machine learning*. PMLR, 2022, pp. 18 332–18 346.
- [47] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017.
- [48] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: <https://openreview.net/forum?id=B1ckMDqlg>
- [49] S. Shi, X. Pan, X. Chu, and B. Li, "Pipemoe: Accelerating mixture-of-experts through adaptive pipelining," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.
- [50] P.-C. Shih, A.-J. Su, K.-H. Tam, T.-C. Huang, K. Chuang, and J. Yeh, "Sow-x: A novel system-on-wafer technology for next generation ai server application," in *2025 IEEE 75th Electronic Components and Technology Conference (ECTC)*. IEEE, 2025, pp. 1–6.
- [51] M. Song, Y. Hu, H. Chen, and T. Li, "Towards pervasive and user satisfactory cnn across gpu microarchitectures," in *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2017, pp. 1–12.
- [52] M. Song, X. Tang, F. Hou, J. Li, W. Wei, Y. Ma, R. Xiao, H. Si, D. Jiang, S. Yin *et al.*, "Tackling the dynamicity in a production llm serving system with sota optimizations via hybrid prefill/decode/verify scheduling on efficient meta-kernels," *arXiv preprint arXiv:2412.18106*, 2024.
- [53] M. Song, J. Zhao, Y. Hu, J. Zhang, and T. Li, "Prediction based execution on deep neural networks," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, 2018, pp. 752–763.
- [54] M. Song, K. Zhong, J. Zhang, Y. Hu, D. Liu, W. Zhang, J. Wang, and T. Li, "In-situ ai: Towards autonomous and incremental deep learning for iot systems," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2018, pp. 92–103.
- [55] J. Stojkovic, C. Zhang, Í. Goiri, J. Torrellas, and E. Choukse, "Dynamollm: Designing llm inference clusters for performance and energy efficiency," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2025, pp. 1348–1362.
- [56] E. Talpes, D. Williams, and D. D. Sarma, "Dojo: The microarchitecture of tesla's exa-scale computer," in *2022 IEEE Hot Chips 34 Symposium (HCS)*. IEEE Computer Society, 2022, pp. 1–28.
- [57] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.
- [58] J. Wang, E. Shi, H. Hu, C. Ma, Y. Liu, X. Wang, Y. Yao, X. Liu, B. Ge, and S. Zhang, "Large language models for robotics: Opportunities, challenges, and perspectives," *Journal of Automation and Intelligence*, vol. 4, no. 1, pp. 52–64, 2025.
- [59] Y. Wei, Y. C. Huang, H. Tang, N. Sankaran, I. Chadha, D. Dai, O. Oluwale, V. Balan, and E. Lee, "9.3 nvlink-c2c: A coherent off package chip-to-chip interconnect with 40gbps/pin single-ended signaling," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 2023, pp. 160–162.
- [60] 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)*. IEEE, 2023, pp. 283–294.
- [61] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, C. Zheng, D. Liu, F. Zhou, F. Huang, F. Hu, H. Ge, H. Wei, H. Lin, J. Tang, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Zhou, J. Lin, K. Dang, K. Bao, K. Yang, L. Yu, L. Deng, M. Li, M. Xue, M. Li, P. Zhang, P. Wang, Q. Zhu, R. Men, R. Gao, S. Liu, S. Luo, T. Li, T. Tang, W. Yin, X. Ren, X. Wang, X. Zhang, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Zhang, Y. Wan, Y. Liu, Z. Wang, Z. Cui, Z. Zhang, Z. Zhou, and Z. Qiu, "Qwen3 technical report," 2025. [Online]. Available: <https://arxiv.org/abs/2505.09388>
- [62] Q. Yang, T. Wei, S. Guan, C. Li, H. Shang, J. Deng, H. Wang, C. Li, L. Wang, Y. Zhang *et al.*, "Pd constraint-aware physical/logical topology co-design for network on wafer," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, 2025, pp. 49–64.
- [63] Z. Ye, L. Chen, R. Lai, W. Lin, Y. Zhang, S. Wang, T. Chen, B. Kasikci, V. Grover, A. Krishnamurthy, and L. Ceze, "Flashinfer: Efficient and customizable attention engine for llm inference serving," *arXiv preprint arXiv:2501.01005*, 2025. [Online]. Available: <https://arxiv.org/abs/2501.01005>
- [64] X. Yu, D. Jiang, J. Deng, J. Liu, C. Li, S. Yin, and Y. Hu, "Cramming a data center into one cabinet, a co-exploration of computing and hardware architecture of waferscale chip," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, 2025, pp. 631–645.
- [65] S. Yun, I. Choi, J. Peng, Y. Wu, J. Bao, Q. Zhang, J. Xin, Q. Long, and T. Chen, "Flex-moe: Modeling arbitrary modality combination via the flexible mixture-of-experts," *Advances in Neural Information Processing Systems*, vol. 37, pp. 98 782–98 805, 2024.

- [66] M. Zhai, J. He, Z. Ma, Z. Zong, R. Zhang, and J. Zhai, “{SmartMoE}: Efficiently training {Sparsely-Activated} models through combining offline and online parallelization,” in *2023 USENIX Annual Technical Conference (USENIX ATC 23)*, 2023, pp. 961–975.
- [67] Y. Zhong, S. Liu, J. Chen, J. Hu, Y. Zhu, X. Liu, X. Jin, and H. Zhang, “Distserve: Disaggregating prefill and decoding for goodput-optimized large language model serving,” in *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, 2024, pp. 193–210.