本文由 AINLP 公众号整理翻译，更多 LLM 资源请扫码关注!

Insights into DeepSeek-V3: Scaling Challenges and Reflections on Hardware for AI Architectures

Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Huazuo Gao, Jiashi Li, Liyue Zhang, Panpan Huang, Shangyan Zhou, Shirong Ma, Wenfeng Liang, Ying He, Yuqing Wang, Yuxuan Liu, Y.X. Wei
DeepSeek-AI
Beijing, China

Abstract

The rapid scaling of large language models (LLMs) has unveiled critical limitations in current hardware architectures, including constraints in memory capacity, computational efficiency, and interconnection bandwidth. DeepSeek-V3, trained on 2,048 NVIDIA H800 GPUs, demonstrates how hardware-aware model co-design can effectively address these challenges, enabling cost-efficient training and inference at scale. This paper presents an in-depth analysis of the DeepSeek-V3/R1 model architecture and its AI infrastructure, highlighting key innovations such as Multi-head Latent Attention (MLA) for enhanced memory efficiency, Mixture of Experts (MoE) architectures for optimized computation-communication trade-offs, FP8 mixed-precision training to unlock the full potential of hardware capabilities, and a Multi-Plane Network Topology to minimize cluster-level network overhead. Building on the hardware bottlenecks encountered during DeepSeek-V3's development, we engage in a broader discussion with academic and industry peers on potential future hardware directions, including precise low-precision computation units, scale-up and scale-out convergence, and innovations in low-latency communication fabrics. These insights underscore the critical role of hardware and model co-design in meeting the escalating demands of AI workloads, offering a practical blueprint for innovation in next-generation AI systems.

CCS Concepts

• **Computer systems organization** → **Architectures**.

Keywords

Large Language Model, Mixture-of-Experts, Deep Learning, FP8 Mixed-Precision Training, Multi-Plane Network, Co-Design

ACM Reference Format:

Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Huazuo Gao, Jiashi Li, Liyue Zhang, Panpan Huang, Shangyan Zhou, Shirong Ma, Wenfeng Liang, Ying He, Yuqing Wang, Yuxuan Liu, Y.X. Wei . 2025. Insights into DeepSeek-V3: Scaling Challenges and Reflections on Hardware for AI Architectures. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25), June 21–25, 2025, Tokyo, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3695053.3731412

ISCA '25, June 21–25, 2025, Tokyo, Japan

ACM ISBN 979-8-4007-1261-6/2025/06
https://doi.org/10.1145/3695053.3731412

1 Introduction

1.1 Background

Large Language Models (LLMs) have undergone rapid evolution in recent years, driven by iterative advancements in model design, computational power, and data availability. In 2024, groundbreaking models such as GPT4o [59], LLaMa-3 [3], Claude 3.5 Sonnet [8], Grok-2 [73], Qwen2.5 [75], Gemini-2 [37] and our DeepSeek-V3 [26] have showcased remarkable progress, further narrowing the gap towards Artificial General Intelligence (AGI). As the Scaling Laws [45] shows, increasing model size, training data, and computational resources leads to substantial improvements in model performance, underscoring the pivotal role of scaling in advancing AI capabilities. Collectively, these developments have ushered in an era where scaling model size and computational power is seen as the key to unlocking higher levels of intelligence.

Recent developments, reasoning models such as OpenAI's o1/o3 series models [60, 61], DeepSeek-R1 [28], Claude-3.7 Sonnet [9], Gemini 2.5 Pro [38], Seed1.5-Thinking [68] and Qwen3 [71] have demonstrated not only the benefits conferred by large-scale architectures, but also the necessity of improving inference efficiency, particularly in handling longer contexts and achieving greater reasoning depth. These advancements underscore the need for faster and more efficient inference, consequently placing ever-increasing demands on computational resources.

To meet these challenges, industry leaders such as Alibaba, ByteDance, Google, xAI and Meta have deployed colossal training clusters [33, 42, 43, 56, 62, 74], featuring tens or even hundreds of thousands of GPUs or TPUs. While such massive infrastructures have enabled the development of state-of-the-art models, their exorbitant costs present significant barriers for smaller research teams and organizations. Despite these barriers, open-source startups such as DeepSeek [23–26, 28] and Mistral [41, 55] are also striving to develop state-of-the-art models. Among them, DeepSeek has especially demonstrated that effective software-hardware co-design can enable cost-efficient training of large models, leveling the playing field for smaller teams.

Building on this tradition, DeepSeek-V3 [26] represents a new milestone in cost-effective training. By leveraging just 2,048 NVIDIA H800 GPUs, DeepSeek-V3 achieves state-of-the-art performance. This achievement aligns with the commitment to advance AI through practical and scalable solutions, as previously demonstrated in the cost-effective architecture of Fire-Flyer AI-HPC [7]. The practices and insights derived from DeepSeek-V3 demonstrate how existing hardware resources can be harnessed to their fullest potential, offering valuable lessons for the broader AI and HPC communities.

Authors are listed in alphabetical order of their first names. Yuqing Wang and Liyue Zhang are the corresponding authors of this paper (e-mail: research@deepseek.com).

深入了解DeepSeek-V3：扩展挑战与硬件在AI架构中的反思

Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Huazuo Gao, Jiashi Li, Liyue Zhang, Panpan Huang, Shangyan Zhou, Shirong Ma, Wenfeng Liang, Ying He, Yuqing Wang, Yuxuan Liu, Y.X. Wei DeepSeek-AI 北京，中国

摘要

大规模语言模型（LLMs）的快速扩展揭示了当前硬件架构的关键限制，包括内存容量、计算效率和互连带宽的约束。DeepSeek-V3在2048个NVIDIA H800 GPU上进行训练，展示了硬件感知的模型协同设计如何有效应对这些挑战，实现大规模的成本高效训练和推理。本文对DeepSeek-V3/R1模型架构及其AI基础设施进行了深入分析，突出显示了多头潜在注意力（MLA）以提升内存效率、专家混合（MoE）架构以优化计算-通信权衡、FP8混合精度训练以充分发挥硬件能力，以及多平面网络拓扑以最小化集群级网络开销等关键创新。在DeepSeek-V3开发过程中遇到的硬件瓶颈基础上，我们与学术界和产业界同行展开更广泛的讨论，探讨未来硬件的潜在方向，包括精确的低精度计算单元、规模扩展与缩减的融合，以及低延迟通信架构的创新。这些见解强调了硬件与模型协同设计在满足AI工作负载不断增长的需求中的关键作用，为下一代AI系统的创新提供了实用的蓝图。

CCS 概念

• 计算机系统组织 → 架构。

关键词

大型语言模型，专家混合模型，深度学习，FP8 混合精度训练，多平面网络，共设计

ACM 参考格式：
赵成刚、邓成奇、阮冲、戴大迈、郜华佐、李家石、张丽悦、黄盼盼、周尚彦、马世荣、梁文峰、何颖、王玉清、刘玉轩、韦耀翔。2025年。关于DeepSeek-V3的见解：规模挑战与硬件在AI架构中的反思。收录于第52届国际计算机体系结构年会（ISCA ’25）论文集，2025年6月21日至25日，日本东京。ACM，纽约，美国，共14页。https://doi.org/10.1145/3695053.3731412

ISCA ’25, June 21–25, 2025, Tokyo, Japan

ACM ISBN 979-8-4007-1261-6/2025/06
https://doi.org/10.1145/3695053.3731412

1 引言

1.1 背景

大型语言模型（LLMs）在近年来经历了快速演变，推动因素包括模型设计的不断迭代、计算能力的提升以及数据可用性的增加。到2024年，诸如GPT4o [59]、LLaMa-3 [3]、Claude 3.5 Sonnet [8]、Grok-2 [73]、Qwen2.5 [75]、Gemini-2 [37] 和我们的DeepSeek-V3 [26]等突破性模型已展现出显著的进展，进一步缩小了向人工通用智能（AGI）迈进的差距。正如Scaling Laws [45]所显示，增加模型规模、训练数据和计算资源可以带来模型性能的显著提升，强调了规模在推动AI能力发展中的关键作用。总体而言，这些发展标志着一个时代的到来，即将模型规模和计算能力的扩展视为解锁更高层次智能的关键。

近期发展，诸如OpenAI的o1/o3系列模型[60, 61]、DeepSeek-R1[28]、Claude-3.7 Sonnet[9]、Gemini 2.5 Pro[38]、Seed1.5-Thinking[68]和Qwen3[71]等推理模型，不仅展示了大规模架构带来的优势，还凸显了提升推理效率的必要性，特别是在处理更长的上下文和实现更深的推理方面。这些进展强调了对更快、更高效推理的需求，因而对计算资源提出了日益增长的要求。

为了应对这些挑战，阿里巴巴、字节跳动、谷歌、xAI 和 Meta 等行业领袖部署了庞大的训练集群 [33, 42, 43, 56, 62, 74]，配备数万甚至数十万的 GPU 或 TPU。虽然如此庞大的基础设施推动了最先进模型的发展，但其高昂的成本对较小的研究团队和组织构成了重大障碍。尽管存在这些障碍，像 DeepSeek [23–26, 28] 和 Mistral [41, 55] 这样的开源创业公司也在努力开发最先进的模型。其中，DeepSeek 特别展示了有效的软件-硬件协同设计可以实现大模型的低成本训练，为较小的团队创造了公平竞争的条件。

在此基础上，DeepSeek-V3 [26] 代表了成本效益训练的一个新里程碑。通过仅使用 2,048 个 NVIDIA H800 GPU，DeepSeek-V3 实现了最先进的性能。这一成就与通过实用且可扩展的解决方案推动人工智能发展的承诺相一致，正如之前在 Fire-Flyer AI-HPC [7] 的成本效益架构中所展示的那样。从 DeepSeek-V3 中获得的实践经验和见解展示了如何充分利用现有硬件资源，为更广泛的人工智能和高性能计算社区提供了宝贵的经验教训。

Authors are listed in alphabetical order of their first names. Yuqing Wang and Liyue Zhang are the corresponding authors of this paper (e-mail: research@deepseek.com).

1Translated Text: 1

1.2 Objectives

This paper does not aim to reiterate the detailed architectural and algorithmic specifics of DeepSeek-V3, which are extensively documented in its technical report [26]. Instead, it adopts a dual perspective—spanning hardware architecture and model design—to explore the intricate interplay between them in achieving cost-efficient large-scale training and inference. By examining this synergy, we aim to provide actionable insights for scaling LLMs efficiently without sacrificing performance or accessibility.

Specifically, the paper focuses on:

- **Hardware-Driven Model Design:** Analyze how hardware features, such as FP8 low-precision computation and scale-up/scale-out network properties, informed the architectural choices in DeepSeek-V3.
- **Mutual Dependencies Between Hardware and Models:** Investigate how hardware capabilities shape model innovation and how the evolving demands of LLMs drive the need for next-generation hardware.
- **Future Directions for Hardware Development:** Derive actionable insights from DeepSeek-V3 to guide the co-design of future hardware and model architectures, paving the way for scalable, cost-efficient AI systems.

1.3 Structure of this Paper

The remainder of this paper is organized as follows. Section 2 explores the design principles underpinning DeepSeek-V3 model architecture, highlighting key innovations such as Multi-head Latent Attention, Mixture-of-Experts optimizations and Multi-Token Prediction Module. Section 3 illustrates how our model architecture pursues low-precision computation and communication. Section 4 includes scale-up interconnection optimizations, discusses scale-up/scale-out convergence, and explores how hardware features influence parallelism and expert selection strategies. Section 5 focuses on scale-out network optimizations, including multi-plane network co-designs and low-latency interconnects. Besides current limitations and future suggestions mentioned in Section 3∼5, Section 6 elaborates on more critical insights from DeepSeek-V3, and identifies directions for future hardware and model co-design.

2 Design Principles for DeepSeek Models

The development of **DeepSeek-V3** exemplifies a hardware-aware approach to scaling LLMs, where each design decision was carefully aligned with hardware constraints to optimize performance and cost efficiency.

As shown in Figure 1, DeepSeek-V3 employs the **DeepSeek-MoE** [27] and **Multi-head Latent Attention (MLA)** [25] architectures that have been proven effective in DeepSeek-V2 [25]. DeepSeek-MoE unlocks the potential of MoE architecture, while MLA drastically reduces memory consumption by compressing Key-Value (KV) caches. In addition, **DeepSeek-V3** incorporates **FP8 mixed-precision training**, significantly lowering computational costs and making large-scale training more practical without compromising model quality. To improve the inference speed, DeepSeek-V3 integrates speculative decoding based on its **Multi-Token Prediction Module**, which significantly increases the generation speed. Beyond model architecture, we also explored cost-efficient AI infrastructure by deploying a **Multi-Plane** two-layer Fat-Tree network to replace a traditional three-layer Fat-Tree topology, reducing cluster networking costs.

These innovations aim to address three core challenges in scaling LLMs—**memory efficiency**, **cost-effectiveness**, and **inference speed**—which are explored in detail in the following subsections.

2.1 Memory Efficiency

LLMs generally require significant memory resources, with memory demands increasing by more than 1000% per year. In contrast, the growth rate of high-speed memory (e.g., HBM) capacity is much slower, typically less than 50% per year [35]. While multi-node parallelism is a viable solution to address memory limitations, optimizing memory usage at the source remains a crucial and effective strategy.

2.1.1 Low-Precision Models. Compared to models that utilize BF16 for weights, FP8 significantly reduces memory consumption by half, effectively alleviating the AI memory wall challenge. A detailed discussion of low-precision techniques is provided in Section 3 Low-Precision Driven Design.

2.1.2 Reducing KV Cache with MLA. For LLM inference, user requests often involve multi-turn conversations. To handle these efficiently, the context from previous requests is cached in what is commonly referred to as the *KV cache*. KV cache addresses this challenge by caching the **Key** and **Value** vectors of previously processed tokens, eliminating the need to recompute them for subsequent tokens. During each inference step, the model only computes the Key and Value vectors for the current token and performs attention computation by combining them with the cached Key-Value pairs from the history. This incremental computation reduces the complexity of generating each token to $O(N)$, making it efficient when processing long sequences or multi-turn inputs. However, it introduces a memory-bound bottleneck because the computation shifts from GEMM to GEMV, which has a much lower compute-to-memory ratio. With modern hardware offering hundreds of TFLOPS, GEMV quickly becomes limited by memory bandwidth, making memory access the primary bottleneck.

To address this bottleneck, we employ **Multi-head Latent Attention (MLA)** [25] that compresses the KV representations of all attention heads into a smaller latent vector using a projection matrix, which is jointly trained with the model. During inference, only the latent vector needs to be cached, significantly reducing memory consumption compared to storing the KV cache for all attention heads.

In addition to MLA, several other approaches have been proposed to reduce the size of the KV cache. These methods are highly valuable and provide significant inspiration for advancements in memory-efficient attention mechanisms:

- **Shared KV (Grouped-Query Attention, GQA; Multi-Query Attention, MQA):** Instead of maintaining separate KV pairs for each attention head, multiple heads share a single set of KV pairs, significantly compressing KV storage. Representative methods include GQA [5] and MQA [70].
- **Windowed KV:** For long sequences, only a sliding window of KV pairs is retained in the cache, discarding results outside the window. While this reduces storage, it compromises long-context reasoning. Representative methods include Longformer [11] and related architectures.

1.2 目标

本文不打算重申DeepSeek-V3的详细架构和算法细节，这些内容已在其技术报告[26]中进行了详尽的描述。相反，本文采用双重视角——涵盖硬件架构和模型设计——来探讨它们在实现成本高效的大规模训练和推理中的复杂相互作用。通过研究这种协同作用，我们旨在提供可行的见解，以在不牺牲性能或可访问性的前提下高效扩展LLMs。

具体而言，本文关注于：

- 硬件驱动的模型设计：分析硬件特性，如FP8低精度计算和扩展/横向网络属性，如何影响DeepSeek-V3的架构选择。
- 硬件与模型之间的相互依赖：研究硬件能力如何影响模型创新，以及不断发展的LLMs需求如何推动下一代硬件的需求。
- 未来硬件开发的方向：从DeepSeek-V3中提取可行的见解，指导未来硬件和模型架构的协同设计，为构建可扩展、成本高效的AI系统铺平道路。

1.3 本文结构

本文的其余部分安排如下。第2节探讨支撑DeepSeek-V3模型架构的设计原则，重点介绍多头潜在注意力机制、专家混合优化和多Token预测模块等关键创新。第3节说明我们的模型架构如何追求低精度计算与通信。第4节包括规模扩展的互连优化，讨论规模扩展/扩展收敛性，以及硬件特性如何影响并行性和专家选择策略。第5节专注于规模扩展网络优化，包括多平面网络共同设计和低延迟互连。除了第3~5节提到的当前限制和未来建议外，第6节详细阐述了DeepSeek-V3的更多关键见解，并指出未来硬件与模型共同设计的方向。

2 深度搜索模型的设计原则

DeepSeek-V3 的开发体现了一种硬件感知的扩展大规模语言模型（LLMs）的方法，每一个设计决策都经过精心调整，以符合硬件限制，从而优化性能和成本效率。

如图1所示，DeepSeek-V3采用了DeepSeek-MoE [27]和多头潜在注意力（MLA）[25]架构，这些架构在DeepSeek-V2 [25]中已被证明是有效的。DeepSeek-MoE释放了MoE架构的潜力，而MLA通过压缩键值（KV）缓存，大幅度降低了内存消耗。此外，DeepSeek-V3还引入了FP8混合精度训练，显著降低了计算成本，使大规模训练变得更加实用，同时不影响模型质量。为了提高推理速度，DeepSeek-V3集成了基于其多Token预测模块的投机解码，大大提升了生成速度。除了模型架构外，我们还探索了成本高效的AI基础设施。

通过部署多平面两层胖树网络来实现结构用传统的三层Fat-Tree拓扑结构取代，降低集群网络成本。

这些创新旨在解决扩展大型语言模型（LLMs）中的三个核心挑战——内存效率、成本效益和推理速度，以下小节将详细探讨。

2.1 内存效率

LLMs 通常需要大量的内存资源，内存需求每年增长超过 1000 %。相比之下，高速内存（例如 HBM）的容量增长速度要慢得多，通常每年不到 50% [35]。虽然多节点并行是解决内存限制的可行方案，但在源头优化内存使用仍然是一种关键且有效的策略。

2.1.1 低精度模型。与使用 BF16 作为权重的模型相比，FP8 大幅度减少了一半的内存消耗，有效缓解了 AI 内存壁挑战。关于低精度技术的详细讨论，请参见第 3 节 低精度驱动设计。

2.1.2 使用MLA减少KV缓存。对于LLM推理，用户请求通常涉及多轮对话。为了高效处理这些请求，之前请求的上下文会被缓存，通常称为KV缓存。KV缓存通过缓存之前处理过的Token的Key和Value向量来应对这一挑战，避免了对后续Token的重复计算。在每次推理步骤中，模型只计算当前Token的Key和Value向量，并通过将它们与历史中的缓存Key-Value对结合，进行注意力计算。这种增量计算将生成每个Token的复杂度降低到$O(N)$，在处理长序列或多轮输入时非常高效。然而，它也引入了一个内存瓶颈，因为计算从GEMM转变为GEMV，而GEMV的计算与内存比率要低得多。随着现代硬件提供数百TFLOPS的性能，GEMV很快就会受到内存带宽的限制，使得内存访问成为主要瓶颈。

为了解决这个瓶颈，我们采用多头潜在注意力（MLA）[25]，它使用投影矩阵将所有注意力头的KV表示压缩成一个较小的潜在向量，该投影矩阵与模型共同训练。在推理过程中，只需缓存潜在向量，显著减少了与存储所有注意力头的KV缓存相比的内存消耗。

除了MLA之外，还提出了几种其他方法来减小KV缓存的大小。这些方法非常有价值，并为内存高效注意力机制的进步提供了重要的启示：

- 共享KV（分组查询注意力，GQA；多查询注意力，MQA）：不为每个注意力头单独维护KV对，而是多个头共享一组KV对，从而大大压缩了KV存储。代表性方法包括GQA [5]和MQA [70]。
- 窗口化KV：对于长序列，缓存中只保留KV对的滑动窗口，舍弃窗口外的结果。虽然这减少了存储，但会影响长上下文推理。代表性的方法包括Long-former [11]及相关架构。

2翻译文本：

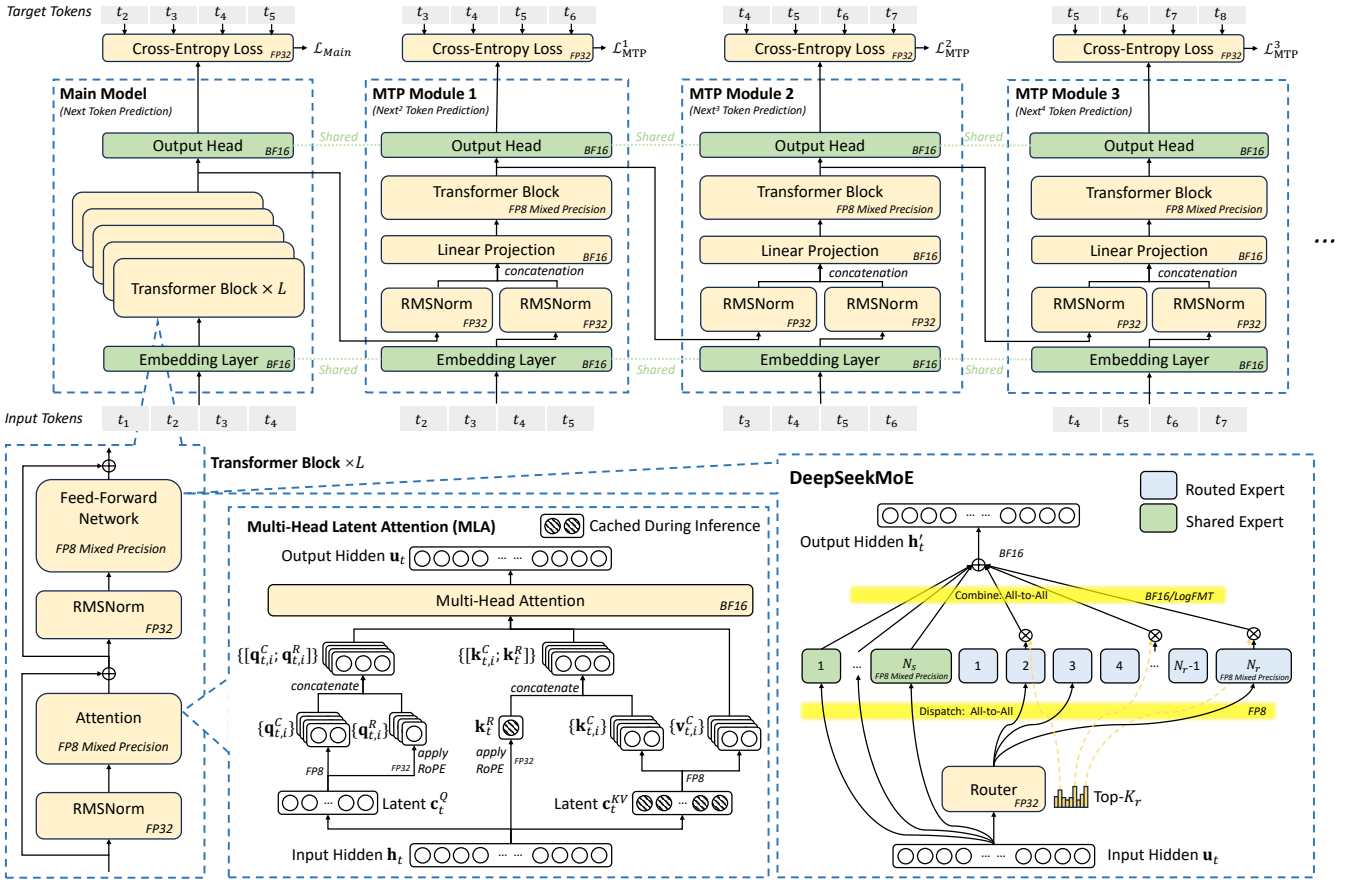

Figure 1: Basic architecture of DeepSeek-V3. Built upon DeepSeek-V2's MLA and DeepSeekMoE, a Multi-Token Prediction Module and FP8 mixed-precision training are introduced to enhance inference and training efficiency. The figure indicates the precision used for computations in different parts of the architecture. All components take inputs and outputs in BF16.

- **Quantized Compression:** KV pairs are stored using low-bit representations [40, 44, 52], further reducing memory usage. Quantization achieves significant compression with minimal impact on model performance.

Table 1 compares the KV cache memory usage per token among DeepSeek-V3, Qwen-2.5 72B [75], and LLaMA-3.1 405B [4]. By adopting MLA, DeepSeek-V3 achieves a significant reduction in KV cache size, requiring only 70 KB per token, substantially less than LLaMA-3.1 405B's 516 KB and Qwen-2.5 72B's 327 KB. This reduction highlights the efficiency of MLA in compressing KV representations compared to GQA-based methods. The ability to achieve such a significant reduction in memory consumption makes DeepSeek-V3 particularly well-suited for scenarios involving long-context processing and resource-constrained environments, enabling more scalable and cost-effective inference.

2.1.3 Future Directions and Perspectives on Resource-Efficient Techniques. While reducing the size of the KV cache is a promising method for improving memory efficiency, the quadratic complexity inherent in Transformer-based autoregressive decoding remains a formidable challenge, especially for extremely long contexts. Recent research efforts, such as Mamba-2 [21] and Lightning Attention[63], investigate linear-time alternatives that offer new possibilities for balancing computational cost and model performance. In addition, approaches such as sparse attention [76], which seek to compress and sparsely activate attention keys and values, represent another attempt at overcoming the computational challenges associated with attention. We look forward to collaborative progress with the broader community toward breakthroughs in this area.

2.2 Cost-Effectiveness of MoE Models

For sparse computing, we have developed DeepSeekMoE, an advanced **Mixture of Experts (MoE)** architecture, which is illustrated in the lower right part of Figure 1. The advantages of MoE models lie in two folds.

2.2.1 Reducing Computational Requirements for Training. The primary advantage of the MoE architecture lies in its ability to significantly reduce training costs. By selectively activating only a subset of expert parameters, MoE models allow the total parameter count to scale up dramatically while keeping computational requirements modest. For example, **DeepSeek-V2** features 236B parameters, but only 21B parameters are activated per token. Similarly, **DeepSeek-V3** expands to 671B parameters—nearly three times the size of V2—while keeping the activation per token at just 37B. In comparison, dense models such as Qwen2.5-72B and LLaMa3.1-405B require all parameters to be active during training.

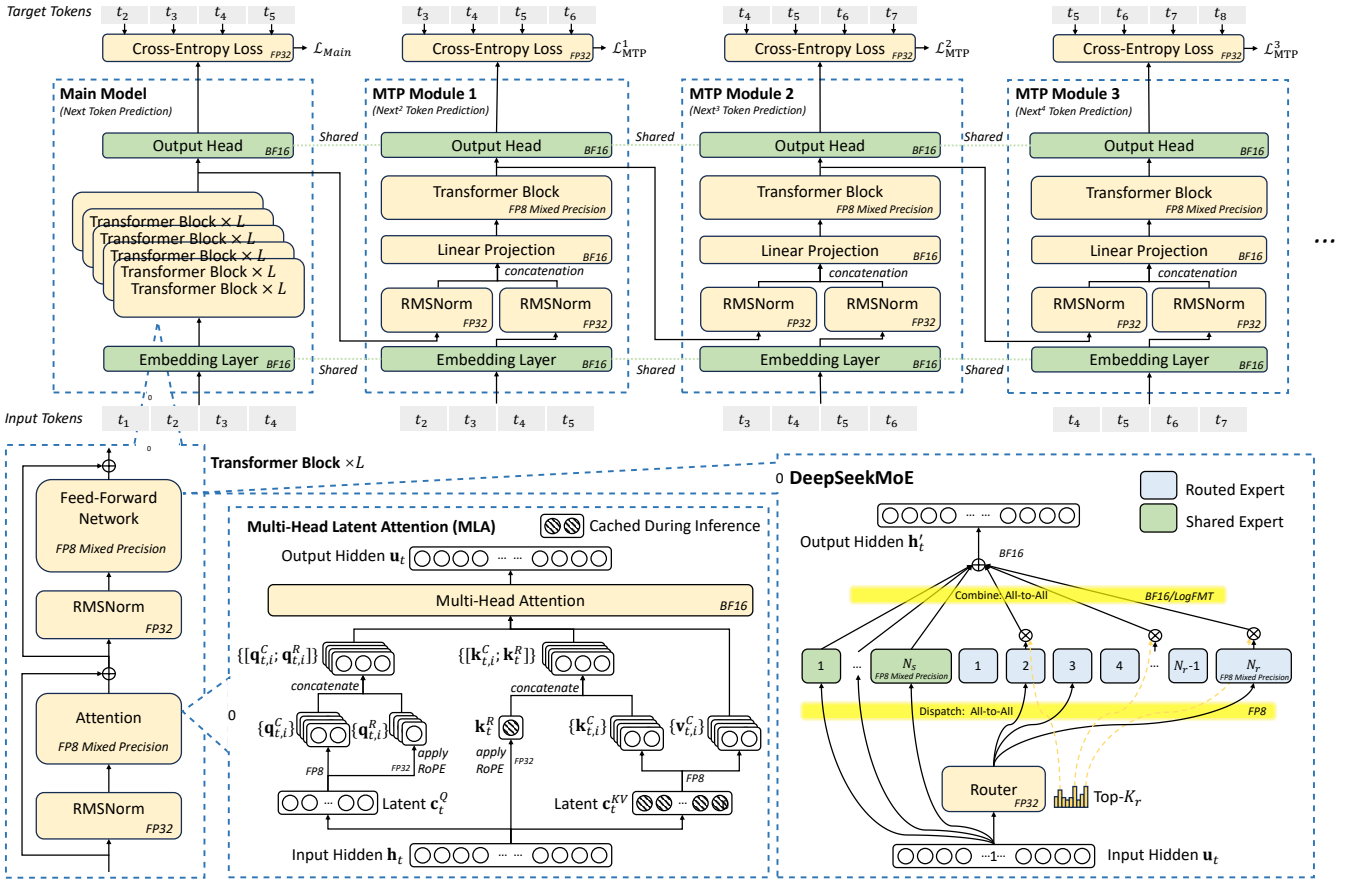

图1：DeepSeek-V3的基本架构。基于DeepSeek-V2的MLA和DeepSeekMoE，引入多Token预测模块和FP8混合精度训练，以提升推理和训练效率。图中显示了架构中不同部分所使用的计算精度。所有组件的输入和输出均为BF16。

- 量化压缩：KV对使用低位表示存储【40, 44, 52】，进一步减少内存使用。量化在对模型性能影响最小的情况下实现了显著的压缩。

表1比较了DeepSeek-V3、Qwen-2.5 72B [75]和LLaMA-3.1 405B [4]在每个token的KV缓存内存使用情况。通过采用MLA，DeepSeek-V3在KV缓存大小方面实现了显著的减少，每个token仅需70 KB，远低于LLaMA-3.1 405B的516 KB和Qwen-2.5 72B的327 KB。这一减少突显了MLA在压缩KV表示方面相较于基于GQA的方法的高效性。实现如此显著的内存消耗降低，使得DeepSeek-V3特别适合处理长上下文和资源受限环境的场景，从而实现更具扩展性和成本效益的推理。

2.1.3 未来方向与资源高效技术的展望。虽然缩小 KV 缓存的规模是一种提高内存效率的有前景的方法，但基于 Transformer 的自回归解码中固有的二次复杂度仍然是一个巨大的挑战，尤其是在处理极长的上下文时。近期的研究工作，如 Mamba-2 [21] 和 Lightning Attention[63]，探索了线性时间的替代方案，为平衡计算成本和模型性能提供了新的可能性。 此外，诸如稀疏注意力[76]之类的方法，旨在压缩和稀疏激活注意力的键和值，代表了克服与注意力相关的计算挑战的另一种尝试。我们期待与更广泛的社区合作，共同在这一领域取得突破。

2.2 MoE模型的性价比

对于稀疏计算，我们开发了DeepSeekMoE，一种先进的专家混合（MoE）架构，如图1右下角所示。MoE模型的优势体现在两个方面。

2.2.1 降低训练的计算需求。MoE 架构的主要优势在于其能够显著降低训练成本。通过有选择地只激活一部分专家参数，MoE 模型允许总参数数量大幅增加，同时保持计算需求适中。例如，DeepSeek-V2 拥有 236B 个参数，但每个 token 只激活 21B 个参数。类似地，DeepSeek-V3 扩展到 671B 个参数——几乎是 V2 的三倍——同时每个 token 的激活参数仅为 37B。相比之下，像 Qwen2.5-72B 和 LLaMa3.1-405B 这样的密集模型在训练过程中需要激活所有参数。

Table 1: KV cache size comparison (BF16 precision): DeepSeek-V3 (MLA) largely reduces KV cache size compared to other models using GQA.

Model	KV Cache Per Token	Multiplier
DeepSeek-V3 (MLA)	70.272 KB	1x
Qwen-2.5 72B (GQA)	327.680 KB	4.66x
LLaMA-3.1 405B (GQA)	516.096 KB	7.28x

As shown in Table 2, the total computational cost for DeepSeek-V3 is approximately 250 GFLOPS per token, whereas the 72B dense model requires 394 GFLOPS and the 405B dense model requires 2448 GFLOPS. This demonstrates that MoE models achieve comparable or even superior performance to dense models while consuming an order of magnitude less computational resources.

2.2.2 Advantages for Personal Use and On-Premises Deployment. In a future where personalized LLM agents [53] become ubiquitous, MoE models offer unique advantages in single-request scenarios. Because only a subset of parameters is activated per request, memory and computational demands are greatly reduced. For example, **DeepSeek-V2** (236B parameters) activates just 21B parameters during inference. This enables PCs with AI SoC chips [6, 10, 58] to achieve nearly 20 tokens per second (TPS), or even twice that speed, which is more than sufficient for personal use. In contrast, dense models of similar capability (e.g., 70B parameters) typically reach only single-digit TPS on similar hardware.

Notably, the increasingly popular KTransformers [39] inference engine allows the complete DeepSeek-V3 model to run on a low-cost server equipped with a consumer GPU (costing approximately $10,000), while still achieving nearly 20 TPS.

This efficiency makes MoE architectures suitable for local deployments and single-user scenarios, where hardware resources are often limited. By minimizing memory and computational overhead, MoE models can deliver high-quality inference performance without requiring expensive infrastructure.

2.3 Increasing Inference Speed

2.3.1 Overlapping Computation and Communication: Maximizing Throughput. Inference speed encompasses both system-wide maximum throughput and single-request latency. To maximize throughput, our model is architected from the outset to leverage dual micro-batch overlap [31, 78], intentionally overlapping communication latency with computation. As demonstrated in our online inference system and supported by open-source profiling data [31], we decouple the computation of MLA and MoE into two distinct stages. While one micro-batch executes a portion of MLA or MoE computation, the other micro-batch simultaneously performs the corresponding dispatch communication. Conversely, during the computation phase of the second micro-batch, the first micro-batch undergoes the combine communication step. This pipelined approach enables seamless overlap of all-to-all communication with ongoing computation, ensuring that the GPU remains fully utilized at all times. Moreover, in production, we adopt a prefill and decode disaggregation architecture [80], assigning large batch size prefill and latency-sensitive decode requests to different expert parallelism group sizes. This strategy ultimately maximizes system throughput under real-world service conditions.

Table 2: Comparison of computational costs for training MoE and dense models: Computational cost per token is measured, assuming a sequence length of 4096.

Model	Size	Training Cost
DeepSeek-V2 MoE	236B	155 GFLOPS/Token
DeepSeek-V3 MoE	671B	250 GFLOPS/Token
Qwen-72B Dense	72B	394 GFLOPS/Token
LLaMa-405B Dense	405B	2448 GFLOPS/Token

2.3.2 Inference Speed Limits. This section focuses on the decode output speed of LLM services, typically measured in **Time Per Output Token (TPOT)**. TPOT is a critical metric for user experience, and it also directly impacts the responsiveness of reasoning models such as OpenAI's o1/o3 and DeepSeek-R1, which rely on the inference length to enhance their intelligence.

For MoE models, achieving high inference speed relies on efficiently deploying expert parameters across computing devices. To achieve the fastest possible inference speed, each device should ideally perform computations for a single expert (or multiple devices should collaboratively compute a single expert if necessary). However, **Expert Parallelism (EP)** requires routing tokens to the appropriate devices, which involves `all-to-all` communication across the network. As a result, the upper limit of MoE inference speed is dictated by interconnection bandwidth.

Consider a system where each device holds one expert's parameters and processes approximately 32 tokens at a time. This token count strikes a balance between compute-to-memory ratio and communication latency. And this token count ensures that each device processes an equal batch size during expert parallelism, allowing the communication time to be easily calculated.

For a system interconnected with CX7 400Gbps InfiniBand (IB) NICs, the time required for the two `all-to-all` communications in EP is calculated as follows:

$$\text{Comm. Time} = (1\text{Byte} + 2\text{Bytes}) \times 32 \times 9 \times 7\text{K}/50\text{GB/s} = 120.96\mu s$$

Here, dispatch uses FP8 (1 byte), while combine uses BF16 (2 bytes), and the hidden size of each token is approximately 7K. The factor 9 indicates that each token is transferred to 8 routed experts and 1 shared expert.

As discussed in Section 2.3.1, maximizing throughput necessitates the use of dual micro-batch overlap. In this strategy, our theoretical best-case analysis assumes that computation overhead is minimized, so the upper bound on performance is determined by communication latency. In practical inference workloads, however, request contexts are often much longer, and MLA computations typically dominate execution time. Thus, this analysis represents an idealized scenario under dual micro-batch overlap. Under this assumption, the total time per layer can be formulated as:

$$\text{Total Time Per Layer} = 2 \times 120.96\mu s = 241.92\mu s$$

With 61 layers in DeepSeek-V3, the total inference time is:

$$\text{Total Inference Time} = 61 \times 241.92\mu s = 14.76\text{ms}$$

Thus, the theoretical upper limit for this system is approximately **14.76 ms TPOT**, equivalent to **67 tokens per second**. However, in practice, factors such as communication overhead, latency, incomplete bandwidth utilization, and computational inefficiencies reduce this number.

表1：KV缓存大小比较（BF16精度）：DeepSeek-V3（MLA）在使用GQA的其他模型中大大减少了KV缓存大小。

Model	KV Cache Per Token	Multiplier
DeepSeek-V3 (MLA)	70.272 KB	1x
Qwen-2.5 72B (GQA)	327.680 KB	4.66x
LLaMA-3.1 405B (GQA)	516.096 KB	7.28x

如表2所示，DeepSeek-V3的总计算成本大约为每个标记250 GFLOPS，而72B密集模型需要394 GFLOPS，405B密集模型则需要2448 GFLOPS。这表明，MoE模型在性能方面与密集模型相当甚至更优，同时消耗的计算资源少一个数量级。

2.2.2 个人使用和本地部署的优势。在未来个性化的LLM代理[53]变得无处不在的情况下，MoE模型在单请求场景中提供了独特的优势。因为每次请求只激活一部分参数，内存和计算需求大大降低。例如，DeepSeek-V2（236B参数）在推理过程中只激活21B参数。这使得配备AI SoC芯片的PC[6, 10, 58]能够实现接近每秒20个Token（TPS），甚至是两倍的速度，这对于个人使用来说已经绰绰有余。相比之下，具有类似能力的密集模型（例如70B参数）在类似硬件上通常只能达到单数字的TPS。

值得注意的是，越来越受欢迎的 KTransformers [39] 推理引擎允许完整的 DeepSeek-V3 模型在配备消费级 GPU（价格约为 10,000 美元）的低成本服务器上运行，同时仍能实现近 20 TPS。

这种效率使得 MoE 架构适用于本地部署和单用户场景，在这些场景中硬件资源通常有限。通过最小化内存和计算开销，MoE 模型可以在不需要昂贵基础设施的情况下提供高质量的推理性能。

2.3 提升推理速度

2.3.1 重叠计算与通信：最大化吞吐量。推理速度既包括系统范围内的最大吞吐量，也包括单请求的延迟。为了最大化吞吐量，我们的模型从一开始就设计为利用双微批次重叠[31, 78]，有意将通信延迟与计算重叠。如我们的在线推理系统所示，并得到开源性能分析数据[31]的支持，我们将MLA和MoE的计算解耦为两个不同的阶段。当一个微批次执行部分MLA或MoE计算时，另一个微批次同时进行相应的调度通信。相反，在第二个微批次的计算阶段，第一微批次进行合并通信步骤。这种流水线方法实现了全通信（all-to-all）与持续计算的无缝重叠，确保GPU始终保持充分利用。此外，在生产环境中，我们采用预填充和解码解耦架构[80]，将大批量预填充和对延迟敏感的解码请求分配到不同的专家并行组规模。这一策略最终在实际服务条件下最大化系统吞吐量。

表2：训练MoE和密集模型的计算成本比较：假设序列长度为4096，测量每个标记的计算成本。

Model	Size	Training Cost
DeepSeek-V2 MoE	236B	155 GFLOPS/Token
DeepSeek-V3 MoE	671B	250 GFLOPS/Token
Qwen-72B Dense	72B	394 GFLOPS/Token
LLaMa-405B Dense	405B	2448 GFLOPS/Token

2.3.2 推理速度限制。本节重点讨论大型语言模型（LLM）服务的解码输出速度，通常以每输出标记的时间（TPOT）衡量。TPOT 是衡量用户体验的关键指标，也直接影响诸如 OpenAI 的 o1/o3 和 DeepSeek-R1 等推理模型的响应速度，这些模型依赖推理长度来提升其智能水平。

对于 MoE 模型，实现高推理速度依赖于在计算设备上高效部署专家参数。为了实现尽可能快的推理速度，理想情况下每个设备应仅执行单个专家的计算（或者如果必要，多个设备应协作计算单个专家）。然而，专家并行（EP）需要将令牌路由到合适的设备，这涉及到网络中的全互联通信。因此，MoE 推理速度的上限由互连带宽决定。

考虑一个系统，其中每个设备持有一个专家的参数，并且一次处理大约32个标记。这个标记数在计算与内存比和通信延迟之间取得了平衡。并且，这个标记数确保每个设备在专家并行中处理相等的批次大小，从而可以轻松计算通信时间。

对于与CX7 400Gbps InfiniBand (IB) NICs互连的系统，EP中两次全对全通信所需的时间计算如下：

$$\text{通信时间} = (1\text{字节} + 2\text{字节}) \times 32 \times 9 \times 7\text{K}/50\text{GB/s} = 120.96\mu s$$

这里，dispatch 使用 FP8（1字节），而 combine 使用 BF16（2字节），每个标记的隐藏大小大约为7K。因子9表示每个标记被传输到8个路由专家和1个共享专家。

如第2.3.1节所述，最大化吞吐量需要使用双微批重叠。在这种策略中，我们的理论最佳情况分析假设计算开销被最小化，因此性能的上限由通信延迟决定。然而，在实际推理工作负载中，请求上下文通常要长得多，MLA计算通常占据主导执行时间。因此，这个分析代表了在双微批重叠下的理想化场景。在此假设下，每层的总时间可以表示为：

$$\text{每层总时间} = 2 \times 120.96\mu s = 241.92\mu s$$

DeepSeek-V3拥有61层，总推理时间为：

$$\text{总推理时间} = 61 \times 241.92\mu s = 14.76\text{毫秒}$$

因此，该系统的理论上限大约为14.76毫秒TPOT，等同于每秒67个标记。然而，在实际中，通信开销、延迟、不完全的带宽利用率以及计算效率低下等因素会降低这个数值。

By contrast, if a high-bandwidth interconnect like GB200 NVL72 (900GB/s unidirectional bandwidth across 72 GPUs) were used, the communication time per EP step drops to:

$$\text{Comm. Time} = (1\text{Byte} + 2\text{Bytes}) \times 32 \times 9 \times 7\text{K}/900\text{GB/s} = 6.72\mu s$$

Assuming the computation time is equal to the communication time, this reduces the total inference time significantly, enabling a theoretical upper limit of over **0.82 ms TPOT**, approximately **1200 tokens per second**. While this figure is purely theoretical and has not been empirically validated, it vividly illustrates the transformative potential of high-bandwidth scale-up networks in accelerating large-scale model inference.

While MoE models exhibit good scalability, achieving high inference speeds by increasing hardware resources alone is cost-prohibitive. Therefore, software and algorithms must also contribute to improving inference efficiency.

2.3.3 Multi-Token Prediction. Inspired by Gloeckle et al. [36], DeepSeek-V3 introduces a **Multi-Token Prediction (MTP)** framework, which simultaneously enhances model performance and improves inference speed. During inference, traditional autoregressive models generate one token at a decoding step, leading to sequential bottlenecks. MTP mitigates this issue by enabling the model to generate additional candidate tokens at a lower cost and verify them in parallel, similar to previous self-drafting-based speculative decoding approaches [14, 48]. This framework significantly accelerates inference without compromising accuracy.

As illustrated in the top part of Figure 1, each MTP module uses a single layer, which is much more lightweight than the full model, to predict additional tokens, enabling parallel verification of multiple candidate tokens. Although slightly hurting the throughput, this approach significantly improves the end-to-end generation latency. The real world practice data demonstrates that an MTP module achieves an acceptance rate of 80% to 90% for predicting the second subsequent token, which increases the generation TPS by 1.8x compared to the scenario without the MTP module.

Moreover, by predicting multiple tokens per step, MTP increases the inference batch size, which is crucial for boosting EP computational intensity and hardware utilization. Such algorithmic innovations are vital for fast and cost-effective inference in DeepSeek-V3.

2.3.4 High Inference Speed for Reasoning Models and Test-Time Scaling. Test-time scaling in LLMs, exemplified by OpenAI's o1/o3 series [60, 61], has enabled significant advances in mathematical reasoning, programming, and general reasoning by dynamically adjusting computational resources during inference. Subsequent models—including DeepSeek-R1 [28], Claude-3.7 Sonnet [9], Gemini 2.5 Pro [38], Seed1.5-Thinking [68], and Qwen3 [71]—have adopted similar strategies and achieved notable improvements in these tasks.

For these reasoning models, high token output speed is of paramount importance. In reinforcement learning (RL) workflows—such as PPO [67], DPO [64] and GRPO [69]—the necessity to rapidly generate large numbers of samples makes inference throughput a critical bottleneck. Likewise, prolonged reasoning sequences can increase user wait times, reducing the practical usability of such models. As a result, optimizing inference speed through synergistic hardware and software innovations is indispensable for advancing the efficiency of reasoning models. However, effective strategies for accelerating inference and expediting RL training remain active areas of investigation, as discussed in Section 2.1.3. We encourage the broader community to collaboratively explore and develop novel solutions to these ongoing challenges.

2.4 Technique Validation Methodology

Each acceleration technique undergoes rigorous empirical validation to evaluate its accuracy impact, including MLA, FP8 mixed-precision computation, and network co-designed MoE gate routing. Given the prohibitive cost of exhaustive ablation on full-scale models, we adopt a hierarchical and resource-efficient validation pipeline. Each technique is first validated extensively on small-scale models, followed by minimal large-scale tuning, and finally integrated in a single, comprehensive training run.

For instance, we first conducted fine-grained FP8 training ablation studies on both 16B and 230B DeepSeek-V2 models before final integration. Under these controlled settings, the relative accuracy loss compared to BF16 remains below 0.25%, attributable to our use of high-precision accumulation and fine-grained quantization strategies.

3 Low-Precision Driven Design

3.1 FP8 Mix-Precision Training

Quantization techniques such as GPTQ [32] and AWQ [51] have been widely used to reduce bit-widths to 8-bit, 4-bit, or even lower, significantly reducing memory requirements. However, these techniques are primarily applied during inference to save memory, rather than in the training phase. NVIDIA's Transformer Engine has supported FP8 mixed-precision training for some time, but prior to DeepSeek-V3, there were no open-source large models leveraging FP8 for training. Through deep collaboration between our infrastructure and algorithm teams, and after extensive experimentation and innovation, we developed an FP8-compatible training framework for MoE models. Figure 1 shows the computational components where FP8-precision forward and backward processes are utilized in the training pipeline. Fine-grained quantization is applied, i.e., tile-wise 1x128 quantization for activations and block-wise 128x128 quantization for model weights. Further technical details of our FP8 framework are documented in the DeepSeek-V3 technical report [26], and our fine-grained FP8 GEMM implementation has been open-sourced in DeepGEMM [77].

3.1.1 Limitations: While FP8 has great potential for accelerating training, several hardware limitations need to be addressed to fully exploit its capabilities:

- **FP8 Accumulation Precision:** FP8 uses constrained accumulation precision in Tensor Cores, affecting the stability for training large models, particularly on NVIDIA Hopper GPUs. After aligning 32 mantissa products by right-shifting based on the maximum exponent, the Tensor Core only maintains their highest 13 fraction bits for addition, and truncates bits exceeding this range. Addition results are accumulated to FP22 registers (1 sign bit, 8 exponent bits, and 13 mantissa bits).
- **Fine-Grained Quantization Challenges:** Fine-grained quantization such as tile-wise and block-wise quantization introduces large dequantization overhead in transporting the partial results

相比之下，如果使用像 GB200 NVL72 这样具有高带宽的互连（在 72 个 GPU 之间单向带宽为 900GB/s），每个 EP 步的通信时间将降至：

$$\text{通信时间} = (1\text{字节} + 2\text{字节}) \times 32 \times 9 \times 7\text{K}/900\text{GB/s} = 6.72\mu s$$

假设计算时间等于通信时间，这大大缩短了总推理时间，使得理论上的上限超过 0.82 ms TPOT，约为每秒 1200 个 token。虽然这个数字纯粹是理论上的，尚未经过实证验证，但它生动地展示了高带宽扩展网络在加速大规模模型推理方面的变革潜力。

虽然 MoE 模型表现出良好的可扩展性，但仅通过增加硬件资源来实现高推理速度是成本过高的。因此，软件和算法也必须为提高推理效率做出贡献。

2.3.3 多词预测。受到 Gloeckle 等人 [36] 的启发，DeepSeek-V3 引入了多词预测（MTP）框架，该框架同时提升了模型性能并改善了推理速度。在推理过程中，传统的自回归模型在每个解码步骤生成一个词，导致序列瓶颈。MTP 通过使模型能够以较低的成本生成额外的候选词并进行并行验证，从而缓解了这一问题，类似于之前基于自我草拟的推测解码方法 [14, 48]。该框架显著加快了推理速度，同时不影响准确性。

如图1的上部所示，每个MTP模块使用单层结构，这比完整模型要轻量得多，用于预测额外的标记，从而实现多个候选标记的并行验证。虽然略微影响吞吐量，但这种方法显著提高了端到端的生成延迟。实际应用数据表明，一个MTP模块在预测第二个后续标记时的接受率达到80%到90%，相比没有MTP模块的场景，生成TPS提高了1.8倍。

此外，通过每步预测多个标记，MTP 增加了推理批量大小，这对于提升 EP 计算强度和硬件利用率至关重要。这些算法创新对于在 DeepSeek-V3 中实现快速且成本高效的推理至关重要。

2.3.4 高推理速度的推理模型和测试时扩展。LLMs中的测试时扩展，以OpenAI的o1/o3系列[60, 61]为例，已在数学推理、编程和一般推理方面实现了显著的进步，通过在推理过程中动态调整计算资源。随后的模型——包括DeepSeek-R1[28]、Claude-3.7 Sonnet[9]、Gemini 2.5 Pro[38]、Seed1.5-Thinking[68]和Qwen3[71]——都采用了类似的策略，并在这些任务中取得了显著的改进。

对于这些推理模型来说，高速的标记输出速度至关重要。在强化学习（RL）工作流程中——如PPO [67]、DPO [64]和GRPO [69]——快速生成大量样本的需求使推理吞吐量成为一个关键瓶颈。同样，延长的推理序列会增加用户等待时间，降低此类模型的实际可用性。因此，通过硬件和软件的协同创新来优化推理速度，对于推动推理模型的效率提升是必不可少的。然而，有效的策略对于加快推理速度和加快强化学习训练仍然是研究的活跃领域，如第2.1.3节所讨论的。我们鼓励更广泛的社区共同探索和开发新颖的解决方案，以应对这些持续的挑战。

2.4 技术验证方法

每种加速技术都经过严格的实证验证，以评估其对准确性的影响，包括MLA、FP8混合精度计算和网络共同设计的MoE门路路由。鉴于对全规模模型进行全面消融的成本过高，我们采用分层且资源高效的验证流程。每项技术首先在小规模模型上进行广泛验证，然后进行最小限度的大规模调优，最后在一次完整的训练中进行集成。

例如，我们首先在16B和230B的DeepSeek-V2模型上进行了细粒度的FP8训练消融研究，然后才进行最终集成。在这些受控设置下，与BF16相比的相对准确率损失保持在0.25%以下，这归功于我们采用的高精度累积和细粒度量化策略。

3 低精度驱动设计

3.1 FP8 混合精度训练

量化技术如GPTQ [32]和AWQ [51]已被广泛应用于将比特宽度降低到8位、4位甚至更低，从而显著减少内存需求。然而，这些技术主要在推理阶段应用以节省内存，而非在训练阶段。在NVIDIA的Transformer引擎中，FP8混合精度训练已经支持了一段时间，但在DeepSeek-V3之前，还没有开源的大型模型利用FP8进行训练。通过我们基础设施团队和算法团队的深入合作，以及大量的实验和创新，我们开发了一个兼容FP8的MoE模型训练框架。图1显示了在训练流程中使用FP8精度的前向和后向计算组件。采用细粒度量化，即激活值的块状1x128量化和模型权重的块状128x128量化。我们FP8框架的更多技术细节已在DeepSeek-V3技术报告[26]中记录，我们的细粒度FP8 GEMM实现已在DeepGEMM [77]中开源。

3.1.1 限制：虽然 FP8 在加速训练方面具有巨大潜力，但需要解决一些硬件限制，才能充分发挥其能力：

- FP8 累积精度：FP8 在 Tensor Cores 中使用受限的累积精度，影响大型模型训练的稳定性，特别是在 NVIDIA Hopper GPU 上。在根据最大指数通过右移对 32 个尾数乘积进行对齐后，Tensor Core 仅保留它们的最高 13 位分数位用于相加，并截断超出此范围的位。相加结果被累积到 FP22 寄存器（1 个符号位、8 个指数位和 13 个尾数位）。

- 细粒度量化挑战：细粒度量化，如块级和块块级量化，在传输部分结果时引入了大量的反量化开销

from Tensor Cores to CUDA Cores for scaling factor multiplication. This incurs frequent data movements, reducing computational efficiency and complicating hardware utilization.

3.1.2 Suggestions: To address the limitations of existing hardware, we have the following suggestions for future designs:

- **Increased Accumulation Precision:** Hardware should improve the accumulation register precision to an appropriate value (e.g. FP32), or support a configurable accumulation precision, enabling a trade-off between performance and accuracy for different requirements of training and inference in various models.
- **Native Support for Fine-Grained Quantization:** Hardware should natively support fine-grained quantization, enabling Tensor Cores to receive scaling factors and implement matrix multiplication with group scaling. In this way, the whole partial sum accumulation and dequantization can be completed directly inside Tensor Cores until the final result is produced, avoiding frequent data movements to reduce dequantization overhead. A notable industrial implementation of this approach is NVIDIA Blackwell's support for **microscaling data format** [66], which exemplifies the practical benefits of native quantization at scale.

3.2 LogFMT: Communication Compression

In the current DeepSeek-V3 architecture, we employ low-precision compression for network communication. During EP parallelism, tokens are dispatched using fine-grained FP8 quantization, reducing communication volume by 50% compared to BF16. This significantly lowers communication time. While the `combine` stage still uses higher precision (e.g., BF16) due to accuracy requirements, we are actively testing FP8, custom precision formats (e.g., E5M6) and mixing FP8-BF16 for further reductions.

Besides these traditional floating point formats, we also tried a new data type, named **Logarithmic Floating-Point Formats (LogFMT-nBit)**, where n is the number of bits with the leading 1 bit as the sign bit S. By mapping the activations from the original `Linear` space to the `Log` space, the distribution of the activations is more uniform. To be specific, given a tile of elements, $[x_1, \cdots, x_m]$, which is 1x128 in our implementation, we take the absolute values and compute the logarithm of all the elements, and find the minimum $min = log(abs(x_i))$ and maximum $max = log(abs(x_j))$. The minimum is encoded as $S.00\cdots01$ and the maximum is encoded as $S.11\cdots11$, with an interval representing $Step = \frac{max-min}{2^{n-1}-2}$. Zero values are represented by $S.00\cdots00$, specially. The left values are rounded to the nearest integer K multiples of $Step$. The decoding process is simple by combining the sign bit and $exp^{min+Step\times(K-1)}$.

By locally calculating the min and $Step$, this data type supports dynamic representation range for different blocks, covering larger ranges or providing more precision, compared to static floating point formats. Besides, we find it is important to round in the original `Linear` space, instead of the `Log` space, for the unbiased activation quantization. We also constrain the min to be larger than $max - log(2^{32})$, which means that the max representation range is similar to E5, a floating point with 5 exponents. We validate our LogFMT-nBit on dense language models with around 7 billion parameters, by quantifying the output of the residual branch to simulate the `combine` stage in MoE models. When setting $n = 8$, sharing the same bits with FP8, the LogFMT-8Bit shows superior

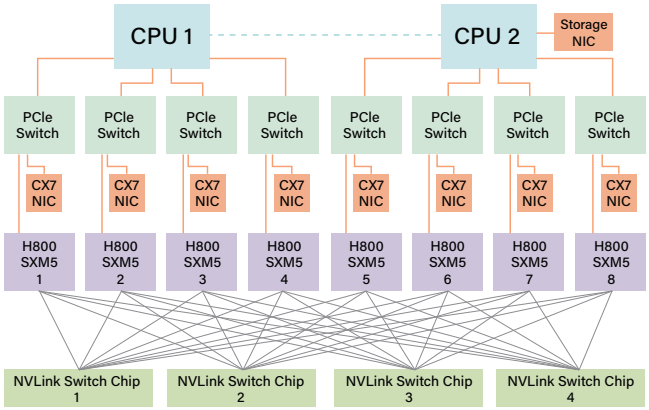

Figure 2: H800 node interconnection.

training accuracy compared to E4M3 or E5M2. After increasing the n to 10 bits, we find it's similar to the BF16 `combine` stage.

3.2.1 Limitations: The initial purpose of using LogFMT is to apply it to activations during transmission or near activation functions, as it offers higher precision than FP8 with the same bit width. However, subsequent computations require reconversion to BF16 or FP8 to accommodate the Hopper GPU tensor cores' data type. Due to insufficient GPU bandwidth for log/exp operations and excessive register pressure during encode/decode, if encode/decode operations are fused with all-to-all communication, the overhead can be substantial (50%~100%). Therefore, although experimental results validate the effectiveness of this format, we do not employ it eventually.

3.2.2 Suggestions: Providing native support for compression and decompression units tailored to FP8 or custom precision formats represents a viable approach for future hardware. This could help minimize bandwidth requirements and streamline communication pipelines. The reduced communication overhead is particularly helpful in bandwidth-intensive tasks like MoE training.

4 Interconnection Driven Design

4.1 Current Hardware Architecture

The NVIDIA H800 GPU SXM architecture we currently use, illustrated in Figure 2, is built on the Hopper architecture, similar to the H100 GPU. However, it features reduced FP64 computational performance and NVLink bandwidth for regulatory compliance. Specifically, the NVLink bandwidth in H800 SXM nodes is reduced from 900 GB/s to 400 GB/s. This significant reduction in intra-node scale-up bandwidth presents a challenge for high-performance workloads. To compensate, each node is equipped with eight 400G Infiniband (IB) CX7 NICs, enhancing scale-out capabilities to mitigate the bandwidth deficit.

To address these hardware constraints, the DeepSeek-V3 model incorporates several design considerations that align with the hardware's strengths and limitations.

4.2 Hardware-Aware Parallelism

To align with the constraints of the H800 architecture, the following parallelism strategies were considered to optimize the performance of DeepSeek-V3:

从张量核心到CUDA核心进行缩放因子乘法。这会导致频繁的数据移动，降低计算效率并增加硬件利用的复杂性。

3.1.2 建议：为了解决现有硬件的局限性，我们对未来的设计提出以下建议：

- 提高累积精度：硬件应将累积寄存器的精度提高到适当的值（例如 FP32），或支持可配置的累积精度，以在不同模型的训练和推理的性能与精度之间进行权衡。

- 对细粒度量化的原生支持：硬件应原生支持细粒度量化，使Tensor Cores能够接收缩放因子并实现带有组缩放的矩阵乘法。通过这种方式，整个部分和累积和反量化可以直接在Tensor Cores内部完成，直到产生最终结果，从而避免频繁的数据移动，减少反量化的开销。这种方法的一个显著工业实现是NVIDIA Blackwell对微缩放数据格式的支持[66]，它展示了大规模原生量化的实际益处。

3.2 LogFMT：通信压缩

在当前的 DeepSeek-V3 架构中，我们采用低精度压缩进行网络通信。在 EP 并行中，令牌使用细粒度的 FP8 量化进行调度，与 BF16 相比，通信量减少了 50%。这显著降低了通信时间。虽然合并阶段仍然使用较高的精度（例如 BF16）以满足精度要求，但我们正在积极测试 FP8、定制的精度格式（例如 E5M6）以及混合使用 FP8-BF16 以实现进一步的减少。

除了这些传统的浮点格式外，我们还尝试了一种新型数据类型，称为对数浮点格式（LogFMT-nBit），其中 n 表示以最高位为符号位的比特数 S。通过将激活值从原始线性空间映射到对数空间，激活值的分布更加均匀。具体来说，给定一组元素块，$[x_1, \cdots, x_m]$，在我们的实现中为 1x128，我们取其绝对值并计算所有元素的对数，找到最小值 $min = log(abs(x_i))$ 和最大值 $max = log(abs(x_j))$。最小值编码为 $S.00\cdots01$，最大值编码为 $S.11\cdots11$，区间表示为 $Step = \frac{max-min}{2^{n-1}-2}$。零值特别用 $S.00\cdots00$ 表示。其余值四舍五入到最接近的整数倍数 K 的 $Step$。解码过程通过结合符号位和 $exp^{min+Step\times(K-1)}$ 简单实现。

通过局部计算 min 和 $Step$，这种数据类型支持不同块的动态表示范围，覆盖更大的范围或提供更高的精度，优于静态浮点格式。此外，我们发现对原始线性空间进行舍入，而非对数空间，对于无偏激活量化非常重要。我们还限制 min 大于 $max - log(232)$，这意味着最大表示范围类似于 E5，一种具有5个指数的浮点数。我们在参数约7亿的密集语言模型上验证了我们的 LogFMT-nBit，通过量化残差分支的输出，模拟 MoE 模型中的合并阶段。当设置 n = 为8时，与 FP8 共享相同的比特数，LogFMT-8Bit 展示出更优的性能

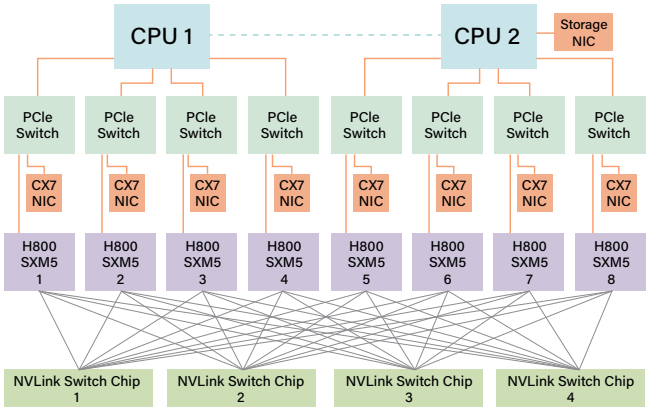

图2：H800节点互连。

训练准确率与 E4M3 或 E5M2 相比。在将 n 增加到 10 位后，我们发现它与 BF16 组合阶段类似。

3.2.1 限制：使用 LogFMT 的最初目的是将其应用于传输过程中的激活或接近激活函数的位置，因为它在相同位宽下提供比 FP8 更高的精度。然而，后续的计算需要重新转换为 BF16 或 FP8，以适应 Hopper GPU 张量核心的数据类型。由于 log/exp 操作的 GPU 带宽不足，以及在编码/解码过程中寄存器压力过大，如果将编码/解码操作与全所有通信融合，开销可能会非常大（50%~100%）。因此，尽管实验结果验证了该格式的有效性，但我们最终并未采用它。

3.2.2 建议：为专门针对FP8或自定义精度格式的压缩和解压缩单元提供原生支持，是未来硬件的可行方案。这有助于最小化带宽需求并简化通信流程。在带宽密集型任务如MoE训练中，减少通信开销尤为重要。

4 互联驱动设计

4.1 当前硬件架构

我们目前使用的 NVIDIA H800 GPU SXM 架构，如图 2 所示，基于 Hopper 架构，类似于 H100 GPU。然而，它在满足法规要求方面，具有较低的 FP64 计算性能和 NVLink 带宽。具体而言，H800 SXM 节点中的 NVLink 带宽从 900 GB/s 降低到 400 GB/s。这一显著的节点内扩展带宽的降低，为高性能工作负载带来了挑战。为此，每个节点配备了八个 400G Infiniband (IB) CX7 NIC，以增强扩展能力，弥补带宽的不足。

为了解决这些硬件限制，DeepSeek-V3模型结合了多项设计考虑，以充分发挥硬件的优势并规避其局限性。

4.2 硬件感知的并行性

为了符合H800架构的约束，考虑了以下并行策略以优化DeepSeek-V3的性能：

- **Avoidance of Tensor Parallelism (TP):** Tensor Parallelism is avoided during training due to its inefficiency under limited NVLink bandwidth. However, during inference, TP can still be selectively used to reduce latency and improve TPOT performance.
- **Enhanced Pipeline Parallelism (PP):** DualPipe [29] is employed to overlap attention and MoE computation with MoE communication. This also reduces pipeline bubbles and balances memory usage across GPUs, improving overall throughput. Additional details are available in the technical report [26].
- **Accelerated Expert Parallelism (EP):** With eight 400Gbps InfiniBand (IB) NICs, the system achieves all-to-all communication at speeds exceeding 40GB/s. Notably, our all-to-all EP implementation, DeepEP [78], is open-sourced, enabling highly efficient expert parallelism as discussed in the following subsection.

4.3 Model Co-Design: Node-Limited Routing

The bandwidth disparity between scale-up (intra-node) and scale-out (inter-node) communication in the H800 architecture is approximately 4:1. Specifically, NVLink provides 200GB/s bandwidth (of which about 160GB/s can actually be achieved), while each 400Gbps IB NIC delivers only 50GB/s bandwidth (we consider small message size and latency influence, use 40GB/s for effective bandwidth). To balance and fully utilize the higher intra-node bandwidth, the model architecture is co-designed with hardware, particularly in the **TopK Expert Selection Strategy**.

Consider a setup with 8 nodes (64 GPUs in total) and 256 routed experts (4 experts per GPU). For DeepSeek-V3, each token is routed to one shared expert and 8 routed experts. If its 8 target experts are distributed across all 8 nodes, the communication time over IB would be $8t$, where t represents the time to send one token over IB. However, by leveraging the higher NVLink bandwidth, tokens routed to the same node can be sent once over IB and then forwarded via NVLink to other intra-node GPUs. The NVLink forwarding enables deduplication of the IB traffic. When the target experts for a given token are distributed across M nodes, the deduplicated IB communication cost will be reduced to Mt ($M < 8$).

Since the IB traffic depends on only M, DeepSeek-V3 introduces a **Node-Limited Routing** for the TopK expert selection strategy. Specifically, we group 256 routed experts into 8 groups, with 32 experts per group, and deploy each group on a single node. On top of this deployment, we algorithmically ensure that each token will be routed to up to 4 nodes. This approach mitigates the bottleneck of IB communication and enhances the effective communication bandwidth during training.

4.4 Scale-Up and Scale-Out Convergence

4.4.1 Limitations of Current Implementations. While the Node-Limited Routing strategy reduces communication bandwidth requirements, it complicates communication pipeline kernel implementations due to the disparity in bandwidth between intra-node (NVLink) and inter-node (IB) interconnects. In practice, GPU Streaming Multiprocessors (SM) threads are used for both network message handling (e.g., filling QPs and WQEs) and data forwarding over NVLink, consuming computational resources. For example, during training, up to 20 of the SMs on the H800 GPU are allocated for communication-related operations, leaving fewer resources available for actual computation. To maximize throughput in online inference, we perform EP all-to-all communication entirely through NIC RDMA, avoiding SM resource contention and improving compute efficiency. This highlights the advantage of RDMA's asynchronous communication model in overlapping computation and communication.

The following are key tasks currently performed by SMs during EP communication, particularly for the `combine` stage's `reduce` operations and data type conversions. Offloading these tasks to dedicated communication hardware could free up SMs for computation kernels, significantly improving overall efficiency:

- **Forwarding Data:** Aggregating IB traffic destined for multiple GPUs within the same node between the IB and NVLink domains.
- **Data Transport:** Moving data between RDMA buffers (registered GPU memory regions) and input/output buffers.
- **Reduce Operations:** Executing `reduce` operations required for EP `all-to-all` `combine` communications.
- **Managing Memory Layouts:** Handling fine-grained memory layouts for chunked data transfers across the IB and NVLink domains.
- **Data Type Cast**: Converting data type before and after `all-to-all` communications.

4.4.2 Suggestions: To address these inefficiencies, we strongly recommend that future hardware should integrate intra-node (scale-up) and inter-node (scale-out) communication into a unified framework. By incorporating dedicated co-processors for network traffic management and seamless forwarding between NVLink and IB domains, such designs can reduce software complexity and maximize bandwidth utilization. For example, node-limited routing strategies employed in DeepSeek-V3 can be further optimized with hardware support for dynamic traffic deduplication.

We also recognize emerging interconnect protocols such as the Ultra Ethernet Consortium (UEC) [17, 18], Ultra Accelerator Link (UALink) [16], both of which are poised to drive advancements in scale-up and scale-out communication. More recently, Unified Bus (UB) [49] has introduced a novel approach to scale-up and scale-out convergence. Section 6 further explores several technical innovations proposed by UEC and UALink. However, in this section, our primary focus is on achieving scale-up and scale-out convergence at the programming framework level.:

(1) **Unified Network Adapter:** Design NICs (Network Interface Cards) or I/O Dies that are connected to unified scale-up and scale-out networks. These adapters should also support basic switch functionality, such as forwarding packets from the scale-out network to specific GPUs within the scale-up network. This could be achieved using a single LID (Local Identifier) or IP address with policy-based routing.
(2) **Dedicated Communication Co-Processor:** Introduce a dedicated co-processor or programmable component—such as an I/O die—for handling network traffic. This component would offload packet processing from GPU SMs, preventing performance degradation. Besides, it should include hardware-accelerated memory copy capabilities for efficient buffer management.
(3) **Flexible Forwarding, Broadcast and Reduce Mechanisms:** Hardware should support flexible forwarding, broadcast operations (for EP dispatch), and reduce operations (for EP combine)

- 避免张量并行（TP）：在训练过程中，由于在有限的NVLink带宽下效率较低，避免使用张量并行。然而，在推理过程中，仍然可以选择性地使用TP，以减少延迟并提高TPOT性能。
- 增强的流水线并行（PP）：DualPipe [29] 被用来将注意力和 MoE 计算与 MoE 通信重叠。这也减少了流水线空洞并在多个 GPU 之间平衡了内存使用，从而提高了整体吞吐量。更多细节请参见技术报告 [26]。
- 加速专家并行（EP）：配备八个400Gbps的In-finiBand（IB）网卡，系统实现了所有到所有的通信，速度超过40GB/s。值得注意的是，我们的全到全EP实现DeepEP [78]是开源的，支持高效的专家并行，如下节所述。

4.3 模型协同设计：节点限制路由

在H800架构中，规模扩展（节点内）与规模扩展（节点间）通信之间的带宽差异大约为4:1。具体而言，NVLink提供200GB/s的带宽（其中大约160GB/s实际上可以实现），而每个400Gbps的IB NIC仅提供50GB/s的带宽（考虑到小消息大小和延迟影响，使用40GB/s作为有效带宽）。为了平衡并充分利用更高的节点内带宽，模型架构与硬件进行了协同设计，特别是在TopK专家选择策略中。

考虑一个由8个节点（总共64个GPU）和256个路由专家（每个GPU 4个专家）组成的设置。对于DeepSeek-V3，每个令牌被路由到一个共享专家和8个路由专家。如果其8个目标专家分布在所有8个节点上，IB上的通信时间将是$8t$，其中t表示通过IB发送一个令牌的时间。然而，通过利用更高的NVLink带宽，路由到同一节点的令牌可以一次性通过IB发送，然后通过NVLink转发到其他节点内的GPU。NVLink转发实现了IB流量的去重。当给定令牌的目标专家分布在M个节点上时，重复的IB通信成本将减少到Mt ($M < 8$)。

由于 IB 流量仅依赖于 M，DeepSeek-V3 引入了节点限制路由（Node-Limited Routing）以实现 TopK 专家选择策略。具体而言，我们将 256 个路由专家分成 8 个组，每组 32 个专家，并在每个节点上部署一个组。在此基础上，我们通过算法确保每个 token 最多会被路由到 4 个节点。这种方法缓解了 IB 通信的瓶颈，并在训练过程中提升了有效的通信带宽。

4.4 扩展与横向扩展的融合

4.4.1 当前实现的局限性。虽然节点限制路由策略减少了通信带宽需求，但由于节点内（NVLink）和节点间（IB）互连带宽的差异，它使通信管道内核的实现变得复杂。在实际应用中，GPU 流多处理器（SM）线程同时用于网络消息处理（例如填充 QP 和 WQE）和通过 NVLink 进行数据转发，消耗计算资源。例如，在训练过程中，H800 GPU 上最多有 20 个 SM 被分配用于与通信相关的操作，留下更少的资源用于实际计算。为了最大化在线推理的吞吐量，我们完全通过NIC RDMA执行EP全对全通信，避免了SM资源的竞争并提高了计算效率。这突显了RDMA异步通信模型在重叠计算和通信方面的优势。

以下是在EP通信过程中，SMs当前执行的关键任务，特别是针对合并阶段的归约操作和数据类型转换。将这些任务卸载到专用通信硬件上，可以释放SMs用于计算核，从而显著提高整体效率：

- 转发数据：在IB和NVLink域之间聚合面向同一节点内多个GPU的IB流量。
- 数据传输：在RDMA缓冲区（已注册的GPU内存区域）和输入/输出缓冲区之间移动数据。
- 减少操作：执行EP全对全合并通信所需的reduce操作。
- 管理内存布局：处理跨 IB 和 NVLink 域的块状数据传输的细粒度内存布局。
- 数据类型转换：在所有到所有通信之前和之后进行数据类型转换。

4.4.2 建议：为了解决这些低效问题，我们强烈建议未来的硬件应将节点内（横向扩展）和节点间（纵向扩展）通信集成到一个统一的框架中。通过引入专用的协处理器，用于网络流量管理以及在 NVLink 和 IB 域之间实现无缝转发，这样的设计可以降低软件复杂性并最大化带宽利用率。例如，DeepSeek-V3 中采用的节点限制路由策略可以通过硬件支持的动态流量去重进一步优化。

我们还认识到新兴的互连协议，例如超以太网联盟（UEC）[17, 18]、超加速器链路（UALink）[16]，它们都准备推动在扩展和横向扩展通信方面的进步。最近，统一总线（UB）[49] 引入了一种用于扩展和横向扩展融合的创新方法。第6节进一步探讨了由UEC和UALink提出的几项技术创新。然而，在本节中，我们的主要关注点是实现编程框架层面的扩展和横向扩展融合。

(1) 统一网络适配器：设计连接到统一扩展和横向扩展网络的 NIC（网络接口卡）或 I/O 芯片。这些适配器还应支持基本的交换机功能，例如将数据包从横向扩展网络转发到扩展网络中的特定 GPU。这可以通过使用单一的 LID（本地标识符）或 IP 地址结合策略路由来实现。

(2) 专用通信协处理器：引入一个专用的协处理器或可编程组件——例如 I/O 芯片，用于处理网络流量。该组件将卸载 GPU SM 的数据包处理，防止性能下降。此外，它还应包括硬件加速的内存复制功能，以实现高效的缓冲区管理。

(3) 灵活的转发、广播和归约机制：硬件应支持灵活的转发、广播操作（用于EP调度）以及归约操作（用于EP合并）

across scale-up and scale-out networks—mirroring our current GPU SM-based implementation. This would not only improve effective bandwidth but also reduce the computational complexity of network-specific operations.

(4) **Hardware Synchronization Primitives:** Provide fine-grained hardware synchronization instructions to handle memory consistency issues or out-of-order packet arrivals at the hardware level. This would eliminate the need for software-based synchronization mechanisms like RDMA completion events, which introduce extra latency and increase programming complexity. Memory-semantic communication with an acquire/release mechanism is a promising implementation.

By implementing these recommendations, future hardware designs can significantly enhance the efficiency of large-scale distributed AI systems while simplifying software development.

4.5 Bandwidth Contention and Latency

4.5.1 Limitations: Besides, current hardware lacks the flexibility to dynamically allocate bandwidth between different types of traffic on NVLink and PCIe. For example, during inference, transferring KV cache data from CPU memory to GPU can consume tens of GB/s, saturating PCIe bandwidth. If the GPU simultaneously uses IB for EP communication, this contention between KV cache transfers and EP communication can degrade overall performance and cause latency spikes.

4.5.2 Suggestions:

- **Dynamic NVLink/PCIe Traffic Prioritization:** Hardware should support dynamic prioritization of traffic based on its type. For example, traffic related to EP, TP, and KV cache transfers should be assigned different priorities to maximize interconnect efficiency. For PCIe, exposing the traffic class (TC) to user-level programming would suffice.
- **I/O Die Chiplet Integration:** Integrating NICs directly into the I/O die and connecting them to the compute die in the same package, rather than through conventional PCIe, would substantially reduce communication latency and alleviate PCIe bandwidth contention.
- **CPU–GPU Interconnects within the Scale-Up Domain:** To further optimize intra-node communication, CPUs and GPUs should be interconnected using NVLink or similar dedicated high-bandwidth fabrics, rather than relying solely on PCIe. Similar to the benefits provided by integrating NICs into the I/O die, this approach can significantly improve scenarios such as offloading parameters or KV cache between GPU and CPU memory during training and inference.

5 Large Scale Network Driven Design

5.1 Network Co-Design: Multi-Plane Fat-Tree

During the training of DeepSeek-V3, we deployed a **Multi-Plane Fat-Tree (MPFT)** scale-out network, as shown in Figure 3. Each node is equipped with eight GPUs and eight IB NICs, with each GPU–NIC pair assigned to a distinct network plane. Additionally, each node has a 400 Gbps Ethernet RoCE NIC connected to a separate storage network plane for accessing the 3FS [30] distributed file system. In the scale-out network, we used 64-port 400G IB switches, enabling the topology theoretically supports up to 16,384 GPUs

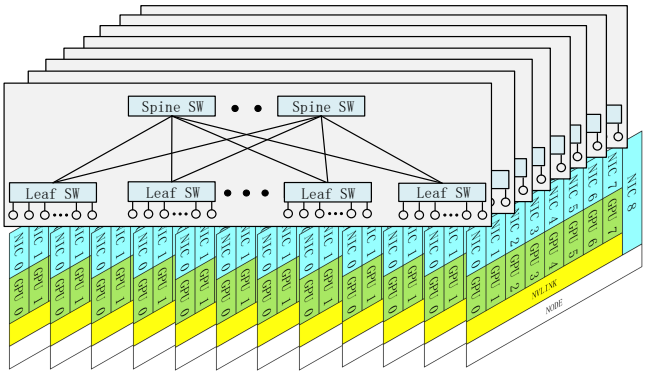

Figure 3: Eight-plane two-layer fat-tree scalue-out network: Each GPU and IB NIC pair belongs to one network plane. Cross-plane traffic must use another NIC and PCIe or NVLink for intra-node forwarding.

while retaining the cost and latency advantages of a two-layer network. However, due to policy and regulatory constraints, just over two thousand GPUs were ultimately deployed.

Furthermore, due to the current limitations of IB ConnectX-7, our deployed MPFT network does not fully realize the envisioned architecture. Ideally, as depicted in Figure 4, each NIC would feature multiple physical ports, each connected to a separate network plane, yet collectively exposed as a single logical interface to the user through port bonding. From a user perspective, a single Queue Pair (QP) could seamlessly transmit and receive messages across all available ports, akin to packet spraying. As a consequence, packets originating from the same QP may traverse distinct network paths and arrive at the receiver out of order, thereby necessitating native support for out-of-order placement within the NIC to guarantee message consistency and preserve the correct ordering semantics. For example, InfiniBand ConnectX-8 natively supports four plane. It would be advantageous for future NICs to fully support advanced multi-plane capabilities, allowing two-tier fat-tree networks to scale effectively to much larger AI clusters. Overall, the multi-plane architecture offers significant advantages in fault isolation, robustness, load balancing, and large-scale system scalability.

5.1.1 Advantages of Multi-Plane Fat-Tree Network.

- **Subset of Multi-Rail Fat-Tree (MRFT):** The MPFT topology constitutes a specific subset of the broader MRFT architecture. As a result, existing optimizations developed by NVIDIA and NCCL for Multi-Rail networks can be seamlessly leveraged within Multi-Plane network deployments. Furthermore, NCCL's support for PXN [54] technology addresses the inherent challenge of inter-plane isolation, enabling efficient communication even when direct interconnectivity between planes is absent.
- **Cost Efficiency:** As shown in Table 3, the multi-plane network enables over 10k endpoints using a two-layer fat-tree (FT2) topology, significantly reducing network costs compared to a three-layer fat tree (FT3). The cost per endpoint is even slightly more competitive than the cost-efficient Slim Fly (SF) topology [12].
- **Traffic Isolation:** Each plane operates independently, ensuring that congestion in one plane does not affect others. This isolation improves overall network stability and prevents cascading performance degradation.

跨越规模扩展和规模外扩网络——与我们当前基于GPU SM的实现相呼应。这不仅会提高有效带宽，还会降低网络特定操作的计算复杂度。

(4) 硬件同步原语：提供细粒度的硬件同步指令，以在硬件层面处理内存一致性问题或包的无序到达。这将消除对基于软件的同步机制（如 RDMA 完成事件）的需求，这些机制会引入额外的延迟并增加编程复杂性。具有获取/释放机制的内存语义通信是一种有前景的实现方式。

通过实施这些建议，未来的硬件设计可以显著提高大规模分布式AI系统的效率，同时简化软件开发。

4.5 带宽争用与延迟

4.5.1 限制：此外，当前硬件缺乏在 NVLink 和 PCIe 上动态分配不同类型流量带宽的灵活性。例如，在推理过程中，将 KV 缓存数据从 CPU 内存传输到 GPU 可能会消耗数十 GB/s，导致 PCIe 带宽饱和。如果 GPU 同时使用 IB 进行 EP 通信，KV 缓存传输与 EP 通信之间的竞争可能会降低整体性能并引起延迟峰值。

4.5.2 建议：

- 动态NVLink/PCIe流量优先级调度：硬件应支持根据流量类型进行动态优先级调度。例如，与EP、TP和KV缓存传输相关的流量应分配不同的优先级，以最大化互连效率。对于PCIe，向用户级编程暴露流量类别（TC）即可。

- I/O Die Chiplet 集成：将 NIC 直接集成到 I/O die 中，并将其连接到同一封装中的计算 die，而不是通过传统的 PCIe，将大大减少通信延迟并缓解 PCIe 带宽争用。

- 在扩展域内的 CPU–GPU 互连：为了进一步优化节点内通信，应使用 NVLink 或类似的专用高带宽互连技术将 CPU 和 GPU 连接，而不是仅依赖 PCIe。类似于将 NIC 集成到 I/O 芯片上的优势，这种方法可以显著改善在训练和推理过程中 GPU 和 CPU 内存之间进行参数卸载或 KV 缓存的场景。

5 大规模网络驱动设计

5.1 网络协同设计：多平面胖树

在DeepSeek-V3的训练过程中，我们部署了一个多平面胖树（MPFT）扩展网络，如图3所示。每个节点配备了八个GPU和八个IB NIC，每个GPU–NIC对分配到不同的网络平面。此外，每个节点还配备了一个400 Gbps以太网RoCE NIC，连接到一个单独的存储网络平面，用于访问3FS [30] 分布式文件系统。在扩展网络中，我们使用了64端口的400G IB交换机，理论上支持最多16,384个GPU的拓扑结构。

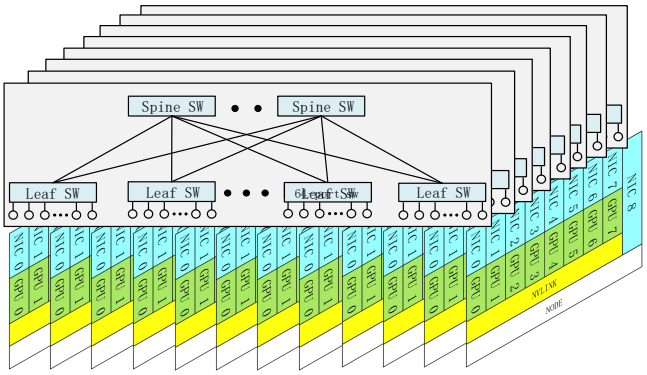

图 3：八平面两层脂树扩展网络：每个 GPU 和 IB NIC 对属于一个网络平面。跨平面流量必须使用另一个 NIC 和 PCIe 或 NVLink 进行节点内转发。

同时保持两层网络的成本和延迟优势。然而，由于政策和监管限制，最终部署的GPU数量刚刚超过两千。

此外，由于当前 IB ConnectX-7 的限制，我们部署的 MPFT 网络尚未完全实现设想的架构。理想情况下，如图 4 所示，每个 NIC 应具有多个物理端口，每个端口连接到不同的网络平面，但通过端口绑定，整体作为单一逻辑接口暴露给用户。从用户的角度来看，一个队列对（QP）可以无缝地在所有可用端口上传输和接收消息，类似于包喷射。因此，来自同一 QP 的数据包可能会经过不同的网络路径到达接收端，导致到达顺序错乱，从而需要 NIC 原生支持乱序放置，以保证消息的一致性和正确的排序语义。例如，InfiniBand ConnectX-8 原生支持四个平面。未来的 NIC 若能全面支持先进的多平面功能，将有助于两层胖树网络有效扩展到更大规模的 AI 集群。总体而言，多平面架构在故障隔离、鲁棒性、负载均衡和大规模系统扩展方面具有显著优势。

5.1.1 多平面胖树网络的优点

- 多轨胖树（MRFT）的子集：MPFT 拓扑结构构成了更广泛的 MRFT 架构的一个特定子集。因此，NVIDIA 和 NCCL 为多轨网络开发的现有优化可以在多平面网络部署中无缝利用。此外，NCCL 对 PXN [54] 技术的支持解决了平面间隔离的固有挑战，即使在平面之间没有直接互连的情况下，也能实现高效通信。

- 成本效率：如表3所示，多平面网络在使用两层胖树（FT2）拓扑结构时，支持超过10k个端点，显著降低了网络成本，相较于三层胖树（FT3）拓扑结构。每个端点的成本甚至比成本高效的Slim Fly（SF）拓扑[12]还要具有一定的竞争力。
- 交通隔离：每个平面独立运行，确保一个平面的拥堵不会影响其他平面。这种隔离提高了整体网络的稳定性，防止了级联性能下降。

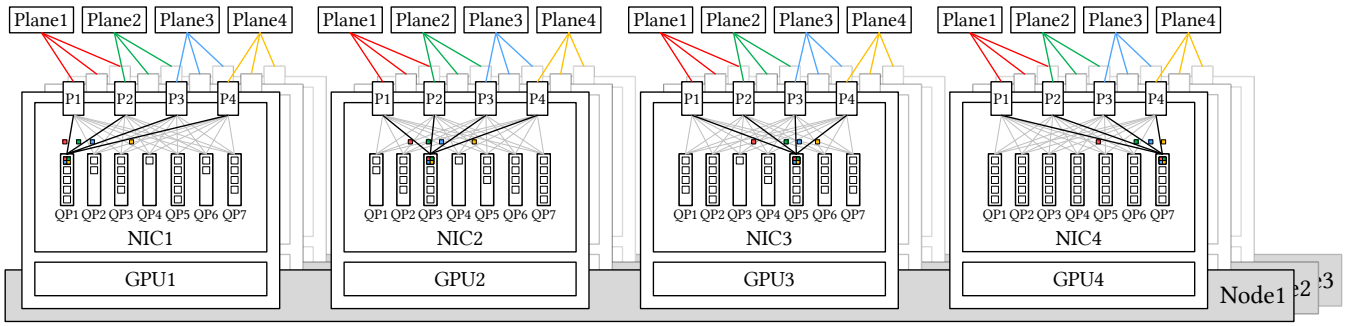

Figure 4: Ideal Multi-Plane Network: Each NIC is equipped with multiple physical ports, each connected to a distinct network plane. A single queue pair (QP) can simultaneously utilize all available ports for transmitting and receiving packets, which necessitates native support for out-of-order placement within the NIC.

Table 3: Network topology comparison. Cost estimates are derived from the methodology in the Slim Fly (SF) paper [12]. DF denotes the canonical dragonfly topology [22, 46, 65].

Metric	FT2	MPFT	FT3	SF	DF
Endpoints	2,048	16,384	65,536	32,928	261,632
Switches	96	768	5,120	1,568	16,352
Links	2,048	16,384	131,072	32,928	384,272
Cost [M$]	9	72	491	146	1,522
Cost/Endpoint [k$]	4.39	4.39	7.5	4.4	5.8

- **Latency Reduction:** The two-layer topology achieves lower latency than three-layer fat trees, as demonstrated in our experiments. This makes it particularly suitable for latency-sensitive applications such as MoE-based training and inference.
- **Robustness:** As shown in Figure 4, multi-port NICs provide multiple uplinks, so single-port failures do not disrupt connectivity and rapid, transparent fault recovery is possible.

It is important to note that, due to current 400G NDR InfiniBand limitations, cross-plane communication requires intra-node forwarding, which introduces additional latency during inference. If future hardware can achieve scale-up and scale-out network convergence as discussed earlier, this latency can be significantly reduced, further enhancing the viability of multi-plane networks.

5.1.2 Performance Anlaysis. To verify the effectiveness of the Multi-Plane Network design, we conducted real-world experiments on our cluster, modifying the cluster's network topology to compare the performance of the **Multi-Plane Two-Layer Fat Tree (MPFT)** and the **Single-Plane Multi-Rail Fat Tree (MRFT)**. Below are the key findings from our experiments:

1. **All-to-All Communication and EP Scenarios**: As illustrated in Figure 5, the all-to-all performance of the multi-plane network is very similar to that of the single-plane multi-rail network. This performance parity can be attributed to NCCL's PXN [54] mechanism, which optimizes traffic forwarding via NVLink in multi-rail topologies. The multi-plane topology also benefits from this mechanism. As shown in Figure 6, the results of all-to-all communication tests conducted on 16 GPUs reveal negligible differences in latency between the MPFT and MRFT topologies.

To evaluate MPFT's performance of all-to-all communication in practical training scenarios, we tested the EP communication patterns commonly used during training. As shown in Figure 7, each

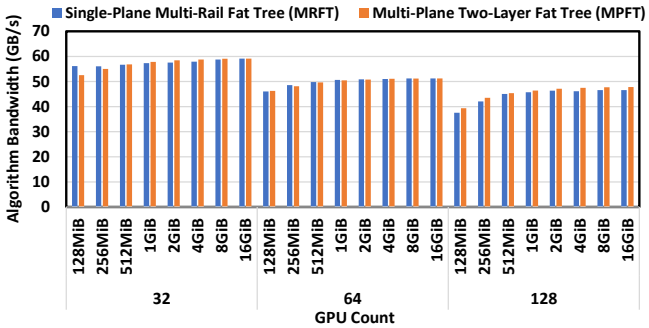

Figure 5: NCCL all-to-all performance from 32 to 128 GPUs for MRFT and MPFT networks.

GPU achieves a high bandwidth exceeding 40GB/s in a multi-plane network, providing reliable performance that meets the demands of training.

2. **Training Throughput for DeepSeek-V3 Model**: We also compare the training metrics of the DeepSeek-V3 model between MPFT and MRFT in Table 4. MFU (Model Flops Utilization) is calculated based on BF16 peak performance. Causal MFU only takes into account the flops of the lower triangle of the attention matrix (in line with FlashAttention[19, 20]), while non-causal MFU includes the flops of the whole attention matrix (in line with Megatron [47]). 1F, 1B, and 1W denote forward time, input backward time, and weight backward time, respectively. When training the V3 model on 2048 GPUs, the performance of MPFT is nearly identical to that of MRFT, with observed differences falling within normal fluctuations and measurement error.

5.2 Low Latency Networks

In our model inference, large-scale EP relies heavily on `all-to-all` communication, which is highly sensitive to both bandwidth and latency. Consider a typical scenario discussed in Section 2.3.2, with a network bandwidth of 50GB/s, the data transfer should ideally take approximately 120 μs . Therefore, the intrinsic network latencies on the order of microseconds can critically impact system performance, making their effects non-negligible.

5.2.1 IB or RoCE. As shown in Table 5, IB consistently achieves lower latency, making it the preferred choice for latency-sensitive

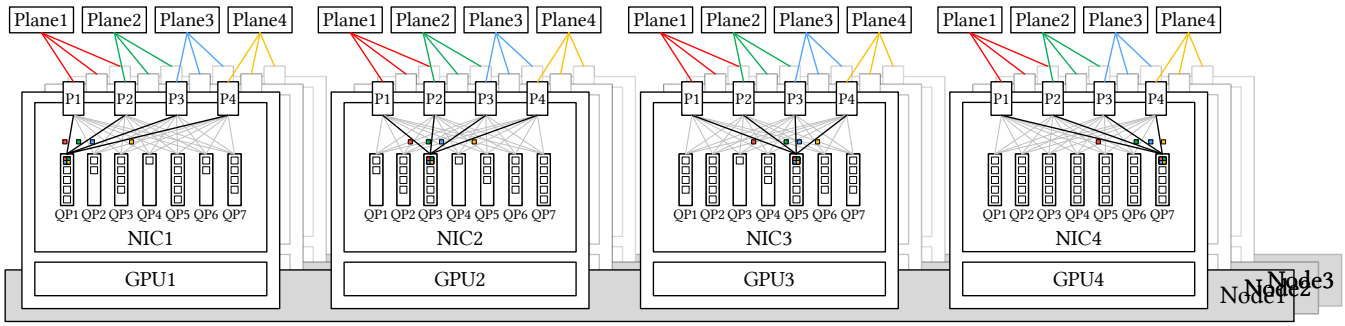

图4：理想的多平面网络：每个NIC配备多个物理端口，每个端口连接到不同的网络平面。单个队列对（QP）可以同时利用所有可用端口进行数据包的传输和接收，这需要NIC原生支持乱序放置。

表 3：网络拓扑结构比较。成本估算基于 Slim Fly (SF) 论文 [12] 中的方法。DF 表示典型的蜻蜓拓扑结构 [22, 46, 65]。

Metric	FT2	MPFT	FT3	SF	DF
Endpoints	2,048	16,384	65,536	32,928	261,632
Switches	96	768	5,120	1,568	16,352
Links	2,048	16,384	131,072	32,928	384,272
Cost [M$]	9	72	491	146	1,522
Cost/Endpoint [k$]	4.39	4.39	7.5	4.4	5.8

- 延迟降低：两层拓扑结构比三层胖树实现更低的延迟，正如我们的实验所示。这使其特别适用于对延迟敏感的应用，如基于MoE的训练和推理。
- 鲁棒性：如图4所示，多端口NIC提供多个上行链路，因此单端口故障不会中断连接，并且可以实现快速、透明的故障恢复。

需要注意的是，由于当前400G NDR InfiniBand的限制，跨平面通信需要节点内转发，这在推理过程中会引入额外的延迟。如果未来的硬件能够实现前面讨论的规模扩展和规模扩展网络融合，这种延迟可以显著减少，进一步提高多平面网络的可行性。

5.1.2 性能分析。为了验证多平面网络设计的有效性，我们在我们的集群上进行了实际测试，修改了集群的网络拓扑，以比较多平面两层胖树（MPFT）和单平面多轨胖树（MRFT）的性能。以下是我们实验的主要发现：

1. 全对全通信与EP场景：如图5所示，多平面网络的全对全性能与单平面多轨网络非常相似。这种性能的等同性归因于NCCL的PXN [54]机制，该机制通过NVLink优化多轨拓扑中的流量转发。多平面拓扑也从该机制中受益。如图6所示，在16个GPU上进行的全对全通信测试结果显示，MPFT和MRFT拓扑在延迟方面几乎没有差异。

为了评估MPFT在实际训练场景中全对全通信的性能，我们测试了训练中常用的EP通信模式。如图7所示，每个

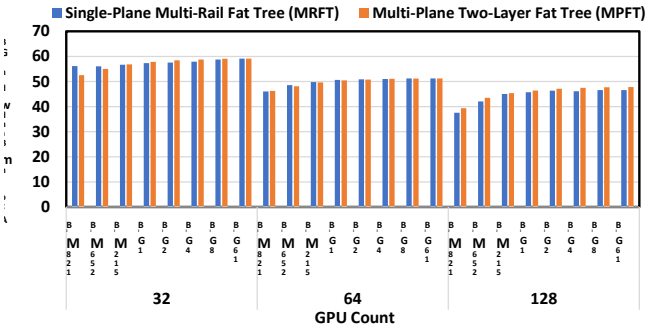

图 5：MRFT 和 MPFT 网络在 32 到 128 GPU 上的 NCCL 全对全性能。

GPU在多平面网络中实现了超过40GB/s的高带宽，提供了可靠的性能，满足训练的需求。

2. DeepSeek-V3 模型的训练吞吐量：我们还在表4中比较了DeepSeek-V3 模型在 MPFT 和 MRFT 之间的训练指标。MFU（模型 Flops 利用率）是基于 BF16 峰值性能计算的。因果 MFU 仅考虑注意力矩阵的下三角部分的 flops（与 FlashAttention [19, 20] 一致），而非因果 MFU 则包括整个注意力矩阵的 flops（与 Megatron[47] 一致）。1F、1B 和 1W 分别表示前向时间、输入反向时间和权重反向时间。当在 2048 个 GPU 上训练 V3 模型时，MPFT 的性能几乎与 MRFT 相同，观察到的差异在正常波动和测量误差范围内。

5.2 低延迟网络

在我们的模型推理中，大规模的EP在很大程度上依赖于全互通信，这对带宽和延迟都非常敏感。考虑第2.3.2节中讨论的一个典型场景，网络带宽为50GB/s，数据传输理想情况下应大约需要120 μ秒。因此，微秒级的固有网络延迟可能会对系统性能产生关键影响，使其影响不可忽视。

5.2.1 IB 或 RoCE。如表5所示，IB 一贯实现更低的延迟，成为对延迟敏感的首选。

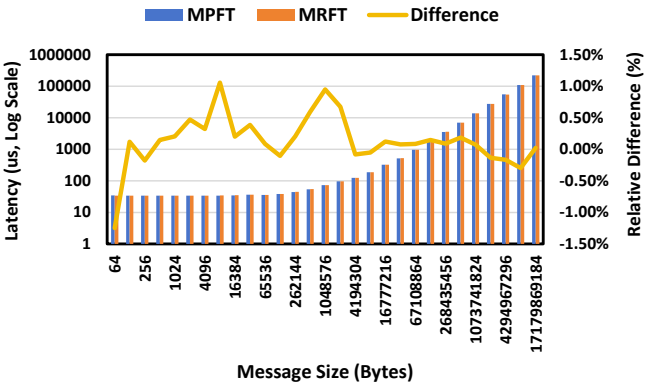

Figure 6: Latency comparison between MPFT and MRFT networks in NCCL all-to-all test under different message sizes, showing that their performance is nearly identical.

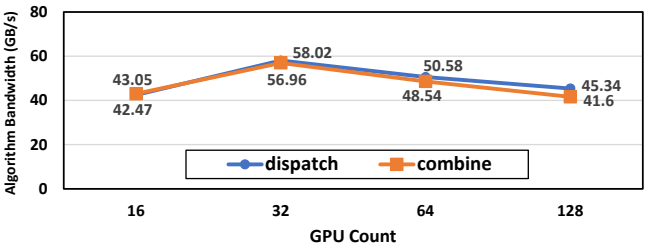

Figure 7: DeepEP performance on MPFT: The EP dispatch and combine kernel communicates across 16 to 128 GPUs using all-to-all. Each GPU processes 4096 tokens. The observed throughput nearly saturates the 400Gps NIC bandwidth.

workloads such as distributed training and inference. Although IB has superior latency performance compared to RDMA over Converged Ethernet (RoCE), it comes with certain limitations:

- **Cost:** IB hardware is significantly more expensive than RoCE solutions, which limits its widespread adoption.
- **Scalability:** IB switches typically support only 64 ports per switch, compared to the 128 ports commonly found in RoCE switches. This restricts the scalability of IB-based clusters, particularly for large-scale deployments.

5.2.2 Recommendations for RoCE Improvements. While RoCE has the potential to be a cost-effective alternative to IB, its current limitations in latency and scalability prevent it from fully meeting the demands of large-scale AI systems. Below, we outline specific recommendations for improving RoCE:

(1) **Specialized Low-Latency RoCE Switches:** We recommend that Ethernet vendors develop RoCE switches specifically optimized for RDMA workloads by removing unnecessary Ethernet features. The Slingshot architecture [22] exemplifies how Ethernet-based designs can achieve latency performance comparable to IB. Similarly, recent innovations from Broadcom [13], including the AI Forwarding Header (AIFH) and upcoming low-latency Ethernet switches, demonstrate the feasibility of high-performance Ethernet fabrics tailored for AI. We are looking forward to continuing innovation in this direction.

Table 4: Training metric comparison between MPFT and MRFT networks.

Metric	MPFT	MRFT
tokens/day (B)	272.80	272.52
time/step (s)	19.926	19.946
1F (s)	1.13	1.13
bubble (s)	2.06	2.03
1B (s)	1.99	1.99
1W (s)	0.48	0.48
1F1B (s)	13.95	14.00
opt (s)	0.29	0.31
TFLOPS (non-causal)	432	432
TFLOPS (causal)	385	385
MFU (non-causal)	43.73%	43.68%
MFU (causal)	38.94%	38.90%

Table 5: CPU side end-to-end latency comparison between IB, RoCE, and intra-node NVLink for 64B data transmission.

Link Layer	Same Leaf	Cross Leaf
RoCE	3.6us	5.6us
InfiniBand	2.8us	3.7us
NVLink	3.33us	-

(2) **Optimized Route Policy:** As shown in Figure 8, the default Equal-Cost Multi-Path (ECMP) routing policy in RoCE struggles to distribute traffic efficiently across interconnects, leading to severe congestion performance degradation in NCCL collective communication tests. LLM training traffic, such as in DP (Data Parallelism), tends to lack randomness, causing multiple flows to converge on the same interconnect link. In contrast, Adaptive Routing (AR) [34] can significantly enhance network performance by dynamically spraying packets across multiple paths. While static routing—based on manually configured route tables—can avoid link conflicts for specific destinations, it lacks flexibility. For large-scale all-to-all communication, adaptive routing offers superior performance and scalability.

(3) **Improved Traffic Isolation or Congestion Control Mechanisms:** Current RoCE switches support only a limited number of priority queues, which are insufficient for complex AI workloads involving concurrent communication patterns such as EP's all-to-all and DP's all-reduce. In such mixed workloads, all-to-all traffic can cause incast congestion due to bursty many-to-one transfers, potentially degrading overall network performance. To address incast's influence on other traffic, one approach is to adopt virtual output queuing (VOQ), assigning a dedicated virtual queue to each QP to isolate traffic flows. Alternatively, more effective congestion control (CC) mechanisms such as RTT-based CC (RTTCC) or user-programmable CC (PCC) can be employed, enabling NIC–switch co-optimization to maintain low latency and high throughput under dynamic traffic conditions.

5.2.3 InfiniBand GPUDirect Async (IBGDA). We utilize IBGDA [2, 57] to reduce latency in network communications. Traditionally, network communication involves the creation of a CPU proxy thread: once the GPU has prepared the data, it must notify the CPU proxy, which then populates the control information for the

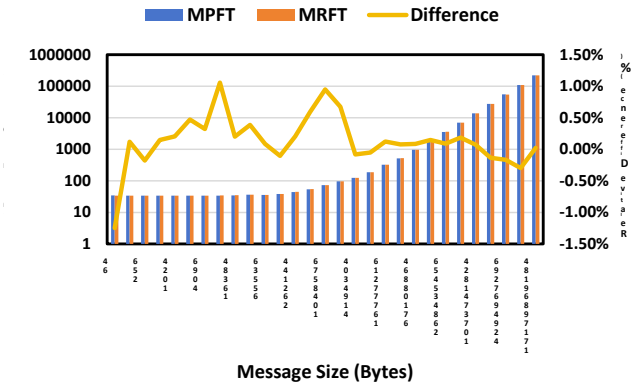

图6：在不同消息大小下，NCCL全对全测试中MPFT和MRFT网络的延迟对比，显示它们的性能几乎相同。

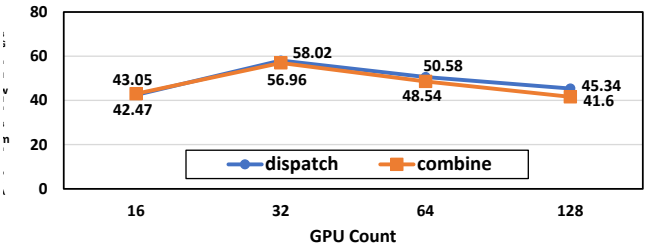

图7：DeepEP在MPFT上的性能：EP调度和合并内核通过全对全通信在16到128个GPU之间进行通信。每个GPU处理4096个令牌。观察到的吞吐量几乎达到了400Gbps NIC带宽的饱和。

工作负载，例如分布式训练和推理。虽然 IB 在延迟性能方面优于基于收敛以太网（RoCE）的 RDMA，但它也存在一些限制：

- 成本：IB硬件的价格明显高于RoCE解决方案，这限制了其广泛采用。
- 可扩展性：IB交换机通常仅支持每个交换机64个端口，而RoCE交换机通常支持128个端口。这限制了基于IB的集群的可扩展性，尤其是在大规模部署中。

5.2.2 改进RoCE的建议。虽然RoCE有潜力成为IB的具有成本效益的替代方案，但其在延迟和可扩展性方面的当前限制阻碍了其完全满足大规模AI系统的需求。以下，我们列出了一些改进RoCE的具体建议：

(1) 专用低延迟RoCE交换机：我们建议以太网供应商开发专门针对RDMA工作负载优化的RoCE交换机，通过去除不必要的以太网功能。Slingshot架构[22]展示了基于以太网的设计如何实现与IB相当的延迟性能。类似地，Broadcom最近的创新[13]，包括AI转发头（AIFH）和即将推出的低延迟以太网交换机，展示了为AI量身定制的高性能以太网网络的可行性。我们期待在这一方向上持续创新。

表4：MPFT和MRFT网络的训练指标比较。

Metric	MPFT	MRFT
tokens/day (B)	272.80	272.52
time/step (s)	19.926	19.946
1F (s)	1.13	1.13
bubble (s)	2.06	2.03
1B (s)	1.99	1.99
1W (s)	0.48	0.48
1F1B (s)	13.95	14.00
opt (s)	0.29	0.31
TFLOPS (non-causal)	432	432
TFLOPS (causal)	385	385
MFU (non-causal)	43.73%	43.68%
MFU (causal)	38.94%	38.90%

表5：64B数据传输中，IB、RoCE和节点内NVLink在CPU端端到端延迟的比较。

Link Layer	Same Leaf	Cross Leaf
RoCE	3.6us	5.6us
InfiniBand	2.8us	3.7us
NVLink	3.33us	-

(2) 优化路径策略：如图8所示，RoCE中的默认等成本多路径（ECMP）路由策略在跨互连分配流量方面存在困难，导致在NCCL集体通信测试中出现严重的拥塞和性能下降。LLM训练流量，例如在DP（数据并行）中，往往缺乏随机性，导致多个流在同一互连链路上汇聚。相比之下，自适应路由（AR）[34]通过动态将数据包分散到多条路径上，可以显著提升网络性能。虽然基于手动配置路由表的静态路由可以避免特定目的地的链路冲突，但缺乏灵活性。对于大规模的全对全通信，自适应路由提供了更优的性能和扩展性。

(3) 改进的流量隔离或拥塞控制机制：当前的RoCE交换机仅支持有限数量的优先级队列，无法满足涉及并发通信模式的复杂AI工作负载，例如EP的全对全和DP的全规约。在这种混合工作负载中，全对全的流量可能会由于突发的多对一传输而引发拥塞，潜在地降低整体网络性能。为了解决拥塞对其他流量的影响，一种方法是采用虚拟输出队列（VOQ），为每个QP分配一个专用的虚拟队列以隔离流量。或者，可以采用更有效的拥塞控制（CC）机制，例如基于RTT的CC（RTTCC）或用户可编程的CC（PCC），实现NIC与交换机的协同优化，在动态流量条件下保持低延迟和高吞吐量。

5.2.3 InfiniBand GPUDirect 异步（IBGDA）。我们利用 IBGDA [2, 57] 来减少网络通信中的延迟。传统上，网络通信涉及创建一个 CPU 代理线程：一旦 GPU 准备好数据，它必须通知 CPU 代理，然后由其填充控制信息以进行后续处理。

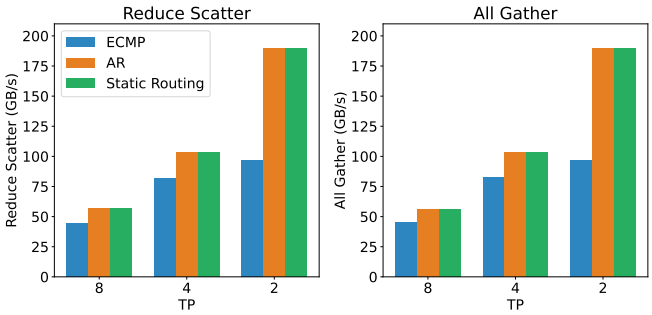

Figure 8: RoCE network bandwidth of AllGather and ReduceScatter communication primitives under different routing methods (ECMP, AR, Static Routing) and TP dimensions.

work request (WR) and signals the NIC via a doorbell mechanism to initiate data transmission. This process introduces additional communication overhead.

IBGDA addresses this issue by allowing the GPU to directly fill the WR content and write to the RDMA doorbell MMIO address. By managing the entire control plane within the GPU, IBGDA eliminates the significant latency overhead associated with GPU-CPU communication. Moreover, when sending a large number of small packets, the control plane processor can easily become a bottleneck. Since GPUs have multiple parallel threads, the sender can leverage these threads to distribute the workload, thereby avoiding such bottlenecks. A range of works—including our DeepEP [78]—have leveraged IBGDA and reported substantial performance gains [1, 15, 79]. We therefore advocate for such capabilities to be widely supported across accelerator devices.

6 Discussion and Insights for Future Hardware Architecture Design

Building on the previous sections, we summarize key architectural insights and outline future directions for hardware design tailored to large-scale AI workloads.

Section 2.3.2 highlighted the importance of large-scale scale-up networks for accelerating model inference. Section 3 discussed the necessity of efficient support for low-precision computation and communication. Section 4 explored the convergence of scale-up and scale-out architectures, along with several proposed enhancements. Section 5 focused on multi-plane network topologies and identified key improvements needed for Ethernet-based interconnects.

Together, these sections identify hardware limitations in concrete application contexts and offer corresponding suggestions. Building on that foundation, this section expands the discussion to broader considerations and proposes forward-looking directions for future hardware architecture design.

6.1 Robustness Challenges

6.1.1 Limitations:

- **Interconnect Failures:** High-performance interconnects (e.g., IB and NVLink) are prone to intermittent disconnections, which can disrupt node-to-node communication. This is especially harmful in communication-heavy workloads like EP, where even brief interruptions may lead to significant performance drops or job failures.
- **Single Hardware Failures:** Node crashes, GPU failures, or ECC (Error-Correcting Code) memory errors can compromise long-running training jobs, often requiring costly restarts. The impact of such failures escalates in large-scale deployments, where the probability of a single-point failure increases proportionally with system size.
- **Silent Data Corruption:** Errors undetected by ECC mechanisms, such as multi-bit memory flips or computational inaccuracies, pose a significant risk to model quality. These errors are particularly insidious in long-running tasks, as they can propagate undetected and corrupt downstream computations. Current mitigation strategies rely on application-level heuristics, which are insufficient for ensuring system-wide robustness.

6.1.2 Suggestions for Advanced Error Detection and Correction. To mitigate risks associated with silent corruption, hardware must incorporate advanced error detection mechanisms beyond traditional ECC. Techniques such as checksum-based validation or hardware-accelerated redundancy checks can provide higher reliability for large-scale deployments.

Furthermore, hardware vendors should deliver comprehensive diagnostic toolkits to end users, empowering them to rigorously verify the integrity of their systems and proactively identify any latent silent data corruption. Such toolkits, when embedded as part of the standard hardware package, foster transparency and enable continuous validation throughout the operational lifecycle, thereby bolstering overall system trustworthiness.

6.2 CPU Bottlenecks and Interconnects

While accelerator design often takes center stage, CPUs remain essential for coordinating computation, managing I/O, and sustaining system throughput. However, current architectures face several critical bottlenecks:

First, as discussed in Section 4.5, the PCIe interface between CPUs and GPUs often becomes a bandwidth bottleneck, particularly during large-scale parameter, gradient, or KV cache transfers. To mitigate this, future systems should adopt direct CPU–GPU interconnects—such as NVLink or Infinity Fabric—or integrate both CPUs and GPUs into the scale-up domain, thereby eliminating intra-node bottlenecks.

In addition to PCIe limitations, sustaining such high data transfer rates also requires exceptionally high memory bandwidth. For example, saturating 160 lanes of PCIe 5.0 demands over 640 GB/s per node, translating to a memory bandwidth requirement of approximately 1 TB/s per node—posing a significant challenge for conventional DRAM architectures.

Lastly, latency-sensitive tasks such as kernel launches and network processing demand high single-core CPU performance, typically requiring base frequencies above 4 GHz. Furthermore, modern AI workloads require sufficient CPU cores per GPU to prevent control-side bottlenecks. For chiplet-based architectures, additional cores are needed to support cache-aware workload partitioning and isolation.

6.3 Toward Intelligent Networks for AI

To meet the demands of latency-sensitive workloads, future interconnects must prioritize both low latency and intelligent networks:

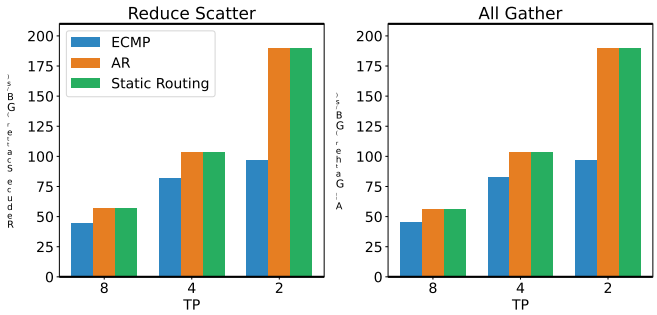

图8：在不同路由方式（ECMP、AR、静态路由）和TP维度下，AllGather和ReduceScatter通信原语的RoCE网络带宽。

工作请求（WR）通过门铃机制向NIC发出信号以启动数据传输。此过程会引入额外的通信开销。

IBGDA 通过允许 GPU 直接填充 WR 内容并写入 RDMA 门铃 MMIO 地址来解决此问题。通过在 GPU 内管理整个控制平面，IBGDA 消除了与 GPU-CPU 通信相关的显著延迟开销。此外，在发送大量小包时，控制平面处理器很容易成为瓶颈。由于 GPU 具有多个并行线程，发送方可以利用这些线程分配工作负载，从而避免此类瓶颈。一系列工作——包括我们的 DeepEP [78]——已经利用 IBGDA 并报告了显著的性能提升 [1, 15, 79]。因此，我们倡导在加速器设备中广泛支持此类功能。

6 未来硬件架构设计的讨论与见解

在前几节的基础上，我们总结了关键的架构见解，并概述了面向大规模AI工作负载的硬件设计的未来方向。

第2.3.2节强调了大规模扩展网络在加速模型推理中的重要性。第3节讨论了对低精度计算和通信的高效支持的必要性。第4节探讨了扩展和横向扩展架构的融合，以及几项提出的增强措施。第5节关注多平面网络拓扑结构，并指出了基于以太网互连所需的关键改进。

这些部分共同识别了具体应用场景中的硬件限制，并提出了相应的建议。在此基础上，本节将讨论范围扩大到更广泛的考虑因素，并提出面向未来的硬件架构设计方向。

6.1 鲁棒性挑战

6.1.1 限制：

- 互连故障：高性能互连（例如 IB 和 NVLink）容易出现间歇性断开，可能会中断节点间通信。这在像 EP 这样以通信为重的工作负载中特别有害，即使是短暂的中断也可能导致性能显著下降或作业失败。
- 单点硬件故障：节点崩溃、GPU故障或ECC（错误更正码）内存错误可能会影响长时间运行的训练任务，通常需要昂贵的重启。在大规模部署中，这类故障的影响会加剧，因为单点故障的概率与系统规模成正比增加。
- 静态数据损坏：由ECC机制未检测到的错误，例如多位内存翻转或计算不准确，构成对模型质量的重大风险。这些错误在长时间运行的任务中尤为隐蔽，因为它们可能在未被检测到的情况下传播并破坏下游计算。目前的缓解策略依赖于应用层的启发式方法，但不足以确保系统的整体鲁棒性。

6.1.2 高级错误检测与纠正建议。为了减轻与静默损坏相关的风险，硬件必须采用超出传统ECC的先进错误检测机制。诸如基于校验和的验证或硬件加速的冗余检查等技术，可以为大规模部署提供更高的可靠性。

此外，硬件供应商应向终端用户提供全面的诊断工具包，赋予他们严格验证系统完整性和主动识别潜在静默数据损坏的能力。当这些工具包作为标准硬件包的一部分嵌入时，能够促进透明度并实现整个运行周期中的持续验证，从而增强整体系统的可信度。

6.2 CPU 瓶颈与互连

虽然加速器设计常常占据核心位置，但CPU仍然在协调计算、管理I/O和维持系统吞吐量方面发挥着关键作用。然而，当前的架构面临几个关键瓶颈：

首先，正如第4.5节所讨论的，CPU与GPU之间的PCIe接口经常成为带宽瓶颈，尤其是在进行大规模参数、梯度或KV缓存传输时。为了缓解这一问题，未来的系统应采用直接的CPU–GPU互连方式——如NVLink或Infinity Fabric——或将CPU和GPU集成到扩展域中，从而消除节点内的瓶颈。

除了 PCIe 的限制之外，维持如此高的数据传输速率还需要极高的内存带宽。例如，饱和160条 PCIe 5.0 通道每个节点需要超过 640 GB/s 的带宽，转化为每个节点大约 1 TB/s 的内存带宽需求——这对传统的 DRAM 架构提出了重大挑战。

最后，延迟敏感的任务，如内核启动和网络处理，要求高单核CPU性能，通常需要基础频率超过4 GHz。此外，现代AI工作负载需要每个GPU配备足够的CPU核心，以防止控制端瓶颈。对于基于芯片块的架构，还需要额外的核心来支持缓存感知的工作负载划分和隔离。

6.3 迈向人工智能的智能网络

为了满足对延迟敏感的工作负载的需求，未来的互连必须同时优先考虑低延迟和智能网络：

- **Co-Packaged Optics:** Incorporating silicon photonics enables scalable higher bandwidth scalability and enhanced energy efficiency, both are critical for large-scale distributed systems.
- **Lossless Network**: Credit-Based Flow Control (CBFC) mechanisms ensures lossless data transmission, yet naively triggering flow control can induce severe head-of-line blocking. Therefore, it is imperative to deploy advanced, endpoint-driven congestion control (CC) algorithms that proactively regulate injection rates and avert pathological congestion scenarios.
- **Adaptive Routing:** As underscored in Section 5.2.2, future network should standardize the adoption of dynamic routing schemes—such as packet spraying and congestion-aware path selection—that continuously monitor real-time network conditions and intelligently redistribute traffic. These adaptive strategies are particularly effective in alleviating hotspots and mitigating bottlenecks during collective communication workloads, including all-to-all and reduce-scatter operations.
- **Efficient Fault-Tolerant Protocols:** Robustness against failures can be significantly enhanced through the deployment of self-healing protocols, redundant ports, and rapid failover techniques. For instance, link-layer retry mechanisms and selective retransmission protocols prove indispensable in scaling reliability across large networks, minimizing downtime and ensuring seamless operation despite intermittent failures.
- **Dynamic Resource Management:** To handle mixed workloads effectively, future hardware should enable dynamic bandwidth allocation and traffic prioritization. For example, inference tasks should be isolated from training traffic in unified clusters, ensuring responsiveness for latency-sensitive applications.

6.4 Discussion on Memory-Semantic Communication and Ordering Issue

Inter-node communication using load/store memory semantics is efficient and programmer-friendly, but current implementations are hampered by memory ordering challenges. For example, after writing data, the sender must issue an explicit memory barrier (fence) before updating a flag to notify the receiver, ensuring data consistency. This strict ordering introduces additional round-trip time (RTT) latency and can stall the issuing thread, impeding in-flight stores and reducing throughput. Similar out-of-order synchronization issues arise in message-semantic RDMA; for instance, performing RDMA atomic add operations with packet spraying after regular RDMA writes on InfiniBand or NVIDIA BlueField-3 can incur additional RTT latency.

To address these, we advocate for hardware support that offers built-in ordering guarantees for memory-semantic communication. Such consistency should be enforced both at the programming level (e.g., via acquire/release semantics) and by hardware at the receiver, enabling in-order delivery without added overhead.

Several approaches are possible. For instance, the receiver could buffer atomic messages and use packet sequence numbers for in-order processing. However, an acquire/release mechanism is both more elegant and efficient. We suggest a simple conceptual mechanism, `Region Acquire/Release (RAR)` mechanism, wherein receiver hardware maintains a bitmap to track the state of the RNR memory region, and acquire/release operations are scoped to the RAR address range. With minimal bitmap overhead, this enables efficient, hardware-enforced ordering, eliminating explicit sender-side fences and delegating ordering to hardware—ideally on the NIC or I/O die. Importantly, the RAR mechanism benefits not only memory-semantic operations but also message-semantic RDMA primitives, thus broadening its practical applicability.

6.5 In-Network Computation and Compression

EP involves two critical `all-to-all` stages—**dispatch** and **combine**—that present significant opportunities for in-network optimization. The **dispatch** stage resembles a small-scale multicast operation, where a single message must be forwarded to multiple target devices. A hardware-level protocol enabling automatic packet replication and forwarding to multiple destinations could drastically reduce communication overhead and improve efficiency.

The **combine** stage, acting as a small-scale reduction operation, could benefit from in-network aggregation techniques. However, due to the small reduction scope and imbalanced workload in EP combine, implementing in-network aggregation in a flexible manner is challenging.

Moreover, as highlighted in Section 3.2, LogFMT enables low-precision token transmission with minimal impact on model performance. Incorporating LogFMT natively within network hardware could further optimize communication by increasing entropy density and reducing bandwidth usage. Hardware-accelerated compression and decompression would allow seamless integration of LogFMT into distributed systems, enhancing overall throughput.

6.6 Memory-Centric Innovations

6.6.1 Limitations of Memory Bandwidth. The exponential growth in model sizes has outpaced advancements in high-bandwidth memory (HBM) technology. This disparity creates a memory bottleneck, particularly in attention-heavy architectures like Transformers.

6.6.2 Suggestions:

- **DRAM-Stacked Accelerators:** Leveraging advanced 3D stacking technologies, DRAM dies can be vertically integrated atop a logic die, thereby enabling exceptionally high memory bandwidth, ultra-low latency, and a practical memory capacity (though stack-limited). This architectural paradigm proves remarkably advantageous for ultra-fast inference in MoE models, where memory throughput is a critical bottleneck. Architectures such as SeDRAM[72] exemplify the potential of this approach, delivering unprecedented performance for memory-bound workloads.
- **System-on-Wafer (SoW):** Wafer-scale integration [50] can maximize computational density and memory bandwidth, addressing the needs of ultra-large-scale models.

7 Conclusion

DeepSeek-V3 exemplifies the transformative potential of hardware-software co-design in advancing the scalability, efficiency, and robustness of large-scale AI systems. By addressing the limitations of current hardware architectures and proposing actionable recommendations, this paper provides a roadmap for the next generation of AI-optimized hardware. These innovations will be critical as AI workloads continue to grow in complexity and scale, driving the future of intelligent systems.

- 共封装光学：集成硅光子技术实现可扩展的更高带宽扩展性和增强的能量效率，这两者对于大规模分布式系统都至关重要。
- 无损网络：基于信用的流量控制（CBFC）机制确保无损数据传输，但天真地触发流量控制可能引发严重的队头阻塞。因此，部署先进的端点驱动的拥塞控制（CC）算法以主动调节注入速率并防止病理性拥塞场景是势在必行的。

- 自适应路由：如第5.2.2节所强调，未来的网络应标准化采用动态路由方案——例如包喷射和拥塞感知路径选择——这些方案能够持续监测实时网络状况并智能地重新分配流量。这些自适应策略在缓解热点和减轻瓶颈方面特别有效，尤其是在集体通信工作负载中，包括全对全和归约-散布操作。

- 高效的容错协议：通过部署自愈协议、冗余端口和快速故障切换技术，可以显著增强系统对故障的鲁棒性。例如，链路层重试机制和选择性重传协议在扩大大规模网络的可靠性方面发挥着不可或缺的作用，最大限度地减少停机时间，确保在间歇性故障情况下的无缝运行。

- 动态资源管理：为了有效应对混合工作负载，未来的硬件应支持动态带宽分配和流量优先级。例如，推理任务应与训练流量在统一集群中隔离，确保对延迟敏感应用的响应能力。

6.4 关于记忆-语义通信与排序问题的讨论

使用加载/存储内存语义的节点间通信高效且对程序员友好，但当前的实现受到内存顺序问题的限制。例如，在写入数据后，发送方必须在更新标志以通知接收方之前发出显式的内存屏障（fence），以确保数据一致性。这种严格的顺序会引入额外的往返时间（RTT）延迟，并可能阻塞发出线程，阻碍在飞存储操作并降低吞吐量。在消息语义的RDMA中也会出现类似的乱序同步问题；例如，在InfiniBand或NVIDIA BlueField-3上进行常规RDMA写入后，使用数据包喷射执行RDMA原子加操作可能会引入额外的RTT延迟。

为了解决这些问题，我们倡导提供内置排序保证的硬件支持，用于内存语义通信。这种一致性应在编程层面（例如，通过 acquire/release 语义）以及硬件在接收端得到强制执行，从而实现有序传递而不增加额外开销。

有几种方法是可行的。例如，接收方可以缓冲原子消息，并使用数据包序列号进行有序处理。然而，获取/释放机制既更优雅又更高效。我们建议一种简单的概念机制，即区域获取/释放（RAR）机制，其中接收硬件维护一个位图以跟踪 RNR 内存区域的状态，获取/释放操作的范围限定于 RAR 地址范围。以最小的位图开销实现，这使得高效的硬件强制排序成为可能，消除了显式的发送端屏障，并将排序委托给硬件——理想情况下是在 NIC 或 I/O 芯片上。重要的是，RAR 机制不仅有益于内存语义操作，还对消息语义的 RDMA 原语有益，从而扩大了其实际应用范围。

6.5 网络内计算与压缩

EP 包含两个关键的全对全阶段——调度和合并——这为网络内优化提供了重要的机会。调度阶段类似于一个小规模的多播操作，其中一条消息必须转发到多个目标设备。一种硬件级协议，能够实现自动的数据包复制和转发到多个目的地，可能会大幅度降低通信开销并提高效率。

合并阶段，作为一种小规模的归约操作，可以受益于网络内聚合技术。然而，由于归约范围较小且在EP合并中的工作负载不平衡，灵活实现网络内聚合具有一定的挑战性。

此外，正如第3.2节所强调的，LogFMT实现了低精度的标记传输，对模型性能的影响最小。在网络硬件中原生集成LogFMT还可以通过增加熵密度和减少带宽使用来进一步优化通信。硬件加速的压缩和解压缩将使LogFMT能够无缝集成到分布式系统中，从而提升整体吞吐量。

6.6 以内存为中心的创新

6.6.1 内存带宽的限制。模型规模的指数级增长已超过高带宽内存（HBM）技术的进步。这种差异造成了内存瓶颈，尤其是在像Transformer这样依赖注意力机制的架构中。

6.6.2 建议：

- DRAM堆叠加速器：利用先进的3D堆叠技术，DRAM芯片可以垂直集成在逻辑芯片之上，从而实现极高的内存带宽、超低延迟以及实际的内存容量（尽管受堆叠限制）。这种架构范式在MoE模型的超快推理中表现出极大的优势，因为内存吞吐量是一个关键瓶颈。诸如SeDRAM[72]的架构就是这种方法潜力的典范，为内存受限的工作负载提供了前所未有的性能。

- 晶圆系统（SoW）：晶圆级集成[50]可以最大化计算密度和内存带宽，满足超大规模模型的需求。

7 结论

DeepSeek-V3 体现了硬件-软件协同设计在推动大规模人工智能系统的可扩展性、效率和鲁棒性方面的变革潜力。通过解决当前硬件架构的局限性并提出可行的建议，本文为下一代AI优化硬件提供了路线图。这些创新将在AI工作负载持续增长的复杂性和规模中发挥关键作用，推动智能系统的未来。

References

[1] Elena Agostini, Davide Rossetti, and Sreeram Potluri. 2017. Offloading Communication Control Logic in GPU Accelerated Applications. In *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. 248–257. https://doi.org/10.1109/CCGRID.2017.29

[2] E. Agostini, D. Rossetti, and S. Potluri. 2018. GPUDirect Async: Exploring GPU synchronous communication techniques for InfiniBand clusters. *J. Parallel and Distrib. Comput.* 114 (2018), 28–45. https://doi.org/10.1016/j.jpdc.2017.12.007

[3] AI@Meta. 2024. Llama 3 Model Card. https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[4] AI@Meta. 2024. Llama 3.1 Model Card. https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md

[5] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. *arXiv preprint arXiv:2305.13245* (2023).

[6] AMD. 2025. AMD Ryzen AI Max+ PRO 395: Designed to power a new generation of compact Copilot+ PC workstations. https://www.amd.com/en/products/processors/laptop/ryzen-pro/ai-max-pro-300-series/amd-ryzen-ai-max-plus-pro-395.html

[7] Wei An, Xiao Bi, Guanting Chen, Shanhuang Chen, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Wenjun Gao, Kang Guan, Jianzhong Guo, Yongqiang Guo, Zhe Fu, Ying He, Panpan Huang, Jiashi Li, Wenfeng Liang, Xiaodong Liu, Xin Liu, Yiyuan Liu, Yuxuan Liu, Shanghao Lu, Xuan Lu, Xiaotao Nie, Tian Pei, Junjie Qiu, Hui Qu, Zehui Ren, Zhangli Sha, Xuecheng Su, Xiaowen Sun, Yixuan Tan, Minghui Tang, Shiyu Wang, Yaohui Wang, Yongji Wang, Ziwei Xie, Yiliang Xiong, Yanhong Xu, Shengfeng Ye, Shuiping Yu, Yukun Zha, Liyue Zhang, Haowei Zhang, Mingchuan Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, and Yuheng Zou. 2024. Fire-Flyer AI-HPC: A Cost-Effective Software-Hardware Co-Design for Deep Learning. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–23. https://doi.org/10.1109/SC41406.2024.00089

[8] Anthropic. 2024. Claude 3.5 Sonnet. https://www.anthropic.com/news/claude-3-5-sonnet

[9] Anthropic. 2025. Claude 3.7 Sonnet and Claude Code. https://www.anthropic.com/news/claude-3-7-sonnet

[10] Apple. 2024. Apple introduces M4 Pro and M4 Max. https://www.apple.com/newsroom/2024/10/apple-introduces-m4-pro-and-m4-max/

[11] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv:2004.05150* (2020).

[12] Nils Blach, Maciej Besta, Daniele De Sensi, Jens Domke, Hussein Harake, Shigang Li, Patrick Iff, Marek Konieczny, Kartik Lakhotia, Ales Kubicek, Marcel Ferrari, Fabrizio Petrini, and Torsten Hoefler. 2025. A high-performance design, implementation, deployment, and evaluation of the slim fly network. In *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation* (Santa Clara, CA, USA) *(NSDI'24)*. USENIX Association, USA, Article 57, 20 pages.

[13] Broadcom. 2025. Scale Up Ethernet Framework. https://docs.broadcom.com/doc/scale-up-ethernet-framework

[14] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. https://openreview.net/forum?id=PEpbUobfJv

[15] Shaoyuan Chen, Wencong Xiao, Yutong Lin, Mingxing Zhang, Yingdi Shan, Jinlei Jiang, Kang Chen, and Yongwei Wu. 2025. Efficient Heterogeneous Large Language Model Decoding with Model-Attention Disaggregation. arXiv:2405.01814 [cs.LG] https://arxiv.org/abs/2405.01814

[16] ULTRA ACCELERATOR LINK CONSORTIUM. 2025. Introducing UALink 200G 1.0 Specification. https://ualinkconsortium.org/wp-content/uploads/2025/04/UALink-1.0-White_Paper_FINAL.pdf

[17] Ultra Ethernet Consortium. 2023. Overview of and Motivation for the Forthcoming Ultra Ethernet Consortium Specification. https://ultraethernet.org/wp-content/uploads/sites/20/2023/10/23.07.12-UEC-1.0-Overview-FINAL-WITH-LOGO.pdf

[18] Ultra Ethernet Consortium. 2024. UEC Progresses Towards v1.0 Set of Specifications. https://ultraethernet.org/uec-progresses-towards-v1-0-set-of-specifications/

[19] Tri Dao. 2023. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning.

[20] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems*.

[21] Tri Dao and Albert Gu. 2024. Transformers are SSMs: generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning* (Vienna, Austria) *(ICML'24)*. JMLR.org, Article 399, 31 pages.

[22] Daniele De Sensi, Salvatore Di Girolamo, Kim H. McMahon, Duncan Roweth, and Torsten Hoefler. 2020. An In-Depth Analysis of the Slingshot Interconnect. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–14. https://doi.org/10.1109/SC41405.2020.00039

[23] DeepSeek-AI. 2024. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. *CoRR* abs/2406.11931 (2024). https://doi.org/10.48550/arXiv.2406.11931

[24] DeepSeek-AI. 2024. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism. *CoRR* abs/2401.02954 (2024). https://doi.org/10.48550/arXiv.2401.02954

[25] DeepSeek-AI. 2024. DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model. *CoRR* abs/2405.04434 (2024). https://doi.org/10.48550/arXiv.2405.04434

[26] DeepSeek-AI. 2024. DeepSeek-V3 Technical Report. (2024). arXiv:2412.19437 [cs.CL] https://arxiv.org/abs/2412.19437

[27] DeepSeek-AI. 2024. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models. *CoRR* abs/2401.06066 (2024). https://doi.org/10.48550/arXiv.2401.06066

[28] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/2501.12948

[29] DeepSeek-AI. 2025. DualPipe: A bidirectional pipeline parallelism algorithm for computation-communication overlap in V3/R1 training. https://github.com/deepseek-ai/dualpipe.

[30] DeepSeek-AI. 2025. Fire-Flyer File System. https://github.com/deepseek-ai/3FS

[31] DeepSeek-AI. 2025. Profiling Data in DeepSeek Infra. https://github.com/deepseek-ai/profile-data?tab=readme-ov-file#inference

[32] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323* (2022).

[33] Adithya Gangidi, Rui Miao, Shengbao Zheng, Sai Jayesh Bondu, Guilherme Goes, Hany Morsy, Rohit Puri, Mohammad Riftadi, Ashmitha Jeevaraj Shetty, Jingyi Yang, Shuqiang Zhang, Mikel Jimenez Fernandez, Shashidhar Gandham, and Hongyi Zeng. 2024. RDMA over Ethernet for Distributed Training at Meta Scale. In *Proceedings of the ACM SIGCOMM 2024 Conference* (Sydney, NSW, Australia) *(ACM SIGCOMM '24)*. Association for Computing Machinery, New York, NY, USA, 57–70. https://doi.org/10.1145/3651890.3672233

[34] Patrick Geoffray and Torsten Hoefler. 2008. Adaptive Routing Strategies for Modern High Performance Networks. In *2008 16th IEEE Symposium on High Performance Interconnects*. 165–172. https://doi.org/10.1109/HOTI.2008.21

[35] Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W. Mahoney, and Kurt Keutzer. 2024. AI and Memory Wall . *IEEE Micro* 44, 03 (May 2024), 33–39. https://doi.org/10.1109/MM.2024.3373763

[36] Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & Faster Large Language Models via Multi-token Prediction. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. https://openreview.net/forum?id=pEWAcejiU2

[37] Google. 2024. Introducing Gemini 2.0: our new AI model for the agentic era. https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024

[38] Google. 2025. Gemini 2.5: Our most intelligent AI model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/

[39] MADSys group and Approaching.AI. 2025. A Flexible Framework for Experiencing Cutting-edge LLM Inference Optimizations. https://github.com/kvcache-ai/ktransformers

[40] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2024. KVQuant: Towards 10 Million Context Length LLM Inference with KV Cache Quantization. *arXiv preprint arXiv:2401.18079* (2024).

[41] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[42] Ziheng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, Yulu Jia, Sun He, Hongmin Chen, Zhihao Bai, Qi Hou, Shipeng Yan, Ding Zhou, Yiyao Sheng, Zhuo Jiang, Haohan Xu, Haoran Wei, Zhang Zhang, Pengfei Nie, Leqi Zou, Sida Zhao, Liang Xiang, Zherui Liu, Zhe Li, Xiaoying Jia, Jianxi Ye, Xin Jin, and Xin Liu. 2024. MegaScale: Scaling Large Language Model Training to More Than 10,000 GPUs. http://arxiv.org/abs/2402.15627 arXiv:2402.15627 [cs].

[43] Norm Jouppi, George Kurian, Sheng Li, Peter Ma, Rahul Nagarajan, Lifeng Nai, Nishant Patil, Suvinay Subramanian, Andy Swing, Brian Towles, Clifford Young, Xiang Zhou, Zongwei Zhou, and David A Patterson. 2023. 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* (Orlando, FL, USA) *(ISCA '23)*. Association for Computing Machinery, New York, NY, USA, Article 82, 14 pages. https://doi.org/10.1145/

参考文献

[1] Elena Agostini, Davide Rossetti 和 Sreeram Potluri. 2017. 在GPU加速应用中卸载通信控制逻辑。收录于2017年第17届IEEE/ACM国际集群、云计算与网格计算研讨会（CCGRID）。第248–257页。https://doi.org/10.1109/CCGRID.2017.29 [2] E. Agostini, D. Rossetti 和 S. Potluri. 2018. GPUDirect Async：探索用于InfiniBand集群的GPU同步通信技术。J. Parallel and Distrib. Comput. 114 (2018), 28–45。https://doi.org/10.1016/j.jpdc.2017.12.007 [3] AI@Meta. 2024. Llama 3模型卡。https://github.com/meta-llama/llama3/ blob/main/MODEL_CARD.md [4] AI@Meta. 2024. Llama 3.1模型卡。https://github.com/meta-llama/llama- models/blob/main/models/llama3_1/MODEL_CARD.md [5] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón 和 Sumit Sanghai. 2023. GQA：从多头检查点训练通用多查询Transformer模型。arXiv预印本 arXiv:2305.13245（2023年）。 [6] AMD. 2025. AMD Ryzen AI Max+ PRO 395：旨在为新一代紧凑型Copilot+ PC工作站提供动力。https://www.amd.com/en/products/processors/laptop/ryzen-pro/ai-max-pro-300-series/amd-ryzen-ai-max-plus-pro-395.html [7] Wei An, Xiao Bi, Guanting Chen, Shanhuang Chen, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Wenjun Gao, Kang Guan, Jianzhong Guo, Yongqiang Guo, Zhe Fu, Ying He, Panpan Huang, Jiashi Li, Wenfeng Liang, Xiaodong Liu, Xin Liu, Yiyuan Liu, Yuxuan Liu, Shanghao Lu, Xuan Lu, Xiaotao Nie, Tian Pei, Junjie Qiu, Hui Qu, Zehui Ren, Zhangli Sha, Xuecheng Su, Xiaowen Sun, Yixuan Tan, Minghui Tang, Shiyu Wang, Yaohui Wang, Yongji Wang, Ziwei Xie, Yiliang Xiong, Yanhong Xu, Shengfeng Ye, Shuiping Yu, Yukun Zha, Liyue Zhang, Haowei Zhang, Mingchuan Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou 和 Yuheng Zou. 2024. Fire-Flyer AI-HPC：一种具有成本效益的软件-硬件协同设计用于深度学习。在SC24：高性能计算、网络、存储与分析国际会议。第1–23页。https://doi.org/10.1109/SC41406.2024.00089 [8] Anthropic. 2024. Claude 3.5 Sonnet。https://www.anthropic.com/news/claude-3-5-sonnet [9] Anthropic. 2025. Claude 3.7 Sonnet 和 Claude 代码。https://www.anthropic.com/news/claude-3-7-sonnet [10] Apple. 2024. Apple推出M4 Pro和M4 Max。https://www.apple.com/newsroom/2024/10/apple-introduces-m4-pro-and-m4-max/ [11] Iz Beltagy, Matthew E. Peters 和 Arman Cohan. 2020. Longformer：长文档Transformer。arXiv:2004.05150（2020年）。 [12] Nils Blach, Maciej Besta, Daniele De Sensi, Jens Domke, Hussein Harake, Shigang Li, Patrick Iff, Marek Konieczny, Kartik Lakhotia, Ales Kubicek, Marcel Ferrari, Fabrizio Petrini 和 Torsten Hoefler. 2025. 高性能设计、实现、部署与评估的slimfly网络。在第21届USENIX网络系统设计与实现研讨会（Santa Clara, CA, USA）（NSDI'24）上。USENIX协会，美国，第57篇文章，20页。 [13] Broadcom. 2025. 规模化以太网框架。https://docs.broadcom.com/doc/scale-up-ethernet-framework [14] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen 和 Tri Dao. 2024. Medusa：具有多解码头的简单双向推理加速框架。在第41届国际机器学习会议（ICML 2024），奥地利维也纳，2024年7月21-27日。OpenReview.net。https://openreview.net/forum?id=PEpbUobfJv [15] Shaoyuan Chen, Wencong Xiao, Yutong Lin, Mingxing Zhang, Yingdi Shan, Jinlei Jiang, Kang Chen 和 Yongwei Wu. 2025. 利用模型注意力解离实现高效异构大规模语言模型解码。arXiv:2405.01814 [cs.LG] https://arxiv.org/abs/2405.01814 [16] ULTRA ACCELERATOR LINK联盟. 2025. UALink 200G 1.0规范介绍。https://ualinkconsortium.org/wp-content/uploads/2025/04/UALink-1.0-White_Paper_FINAL.pdf [17] Ultra Ethernet联盟. 2023.即将到来的Ultra Ethernet联盟规范的概述与动机。https://ultraethernet.org/wp-content/uploads/sites/20/2023/10/23.07.12-UEC-1.0-Overview-FINAL-WITH-LOGO.pdf [18] Ultra Ethernet联盟. 2024. UEC朝着v1.0规范集迈进。https://ultraethernet.org/uec-progresses-towards-v1-0-set-of- specifications/ [19] Tri Dao. 2023. FlashAttention-2：更快的注意力机制，具有更好的并行性和工作划分。 [20] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra 和 Christopher Ré. 2022. FlashAttention：具有IO感知的快速且内存高效的精确注意力机制。在神经信息处理系统的进展中。 [21] Tri Dao 和 Albert Gu. 2024. 变换器是SSMs：通过结构化状态空间对偶实现的广义模型和高效算法。在第41届国际机器学习会议（维也纳，奥地利）（ICML'24）上。JMLR.org，第399篇文章，31页。

[22] Daniele De Sensi, Salvatore Di Girolamo, Kim H. McMahon, Duncan Roweth 和 Torsten Hoefler. 2020. 对 Slingshot 互连的深入分析。收录于 SC20：高性能计算、网络、存储与分析国际会议。1–14。https://doi.org/10.1109/SC41405.2020.00039 [23] DeepSeek-AI. 2024. DeepSeek-Coder-V2：突破代码智能中闭源模型的壁垒。CoRR abs/2406.11931 (2024)。https://doi.org/ 10.48550/arXiv.2406.11931 [24] DeepSeek-AI. 2024. DeepSeek LLM：以长远主义扩展开源语言模型。CoRR abs/2401.02954 (2024)。https://doi.org/10.48550/ arXiv.2401.02954 [25] DeepSeek-AI. 2024. DeepSeek-V2：一种强大、经济且高效的专家混合语言模型。CoRR abs/2405.04434 (2024)。https://doi.org/10.48550/ arXiv.2405.04434 [26] DeepSeek-AI. 2024. DeepSeek-V3 技术报告。 (2024)。arXiv:2412.19437 [cs.CL] https://arxiv.org/abs/2412.19437 [27] DeepSeek-AI. 2024. DeepSeekMoE：迈向终极专家专业化的专家混合语言模型。CoRR abs/2401.06066 (2024)。https://doi.org/10.48550/arXiv.2401.06066 [28] DeepSeek-AI. 2025. DeepSeek-R1：通过强化学习激励LLMs的推理能力。arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/ 2501.12948 [29] DeepSeek-AI. 2025. DualPipe：一种用于V3/R1训练中计算-通信重叠的双向流水线并行算法。https://github.com/ deepseek-ai/dualpipe。 [30] DeepSeek-AI. 2025. Fire-Flyer 文件系统。https://github.com/deepseek-ai/3FS [31] DeepSeek-AI. 2025. DeepSeek基础设施中的性能分析数据。https://github.com/ deepseek-ai/profile-data?tab=readme-ov-file#inference [32] Elias Frantar, Saleh Ashkboos, Torsten Hoefler 和 Dan Alistarh. 2022. Gptq：生成式预训练变换器的精确后训练量化。arXiv预印本 arXiv:2210.17323 (2022)。 [33] Adithya Gangidi, Rui Miao, Shengbao Zheng, Sai Jayesh Bondu, Guilherme Goes, Hany Morsy, Rohit Puri, Mohammad Riftadi, Ashmitha Jeevaraj Shetty, Jingyi Yang, Shuqiang Zhang, Mikel Jimenez Fernandez, Shashidhar Gandham 和 Hongyi Zeng. 2024. 以以太网实现的RDMA用于Meta规模的分布式训练。在ACM SIGCOMM 2024会议论文集（悉尼，新南威尔士，澳大利亚）（ACM SIGCOMM '24）。计算机协会，纽约，纽约州，美国，57–70。https://doi.org/10.1145/3651890.3672233 [34] Patrick Geoffray 和 Torsten Hoefler. 2008. 现代高性能网络的自适应路由策略。在第16届IEEE高性能互连研讨会，165–172。https://doi.org/10.1109/HOTI.2008.21 [35] Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W. Mahoney 和 Kurt Keutzer. 2024. AI与内存墙。IEEE Micro 44, 03 (2024年5月)，33–39。https://doi.org/10.1109/MM.2024.3373763 [36] Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz 和 Gabriel Synnaeve. 2024. 通过多令牌预测实现更好更快的大型语言模型。在第41届国际机器学习会议（ICML 2024），奥地利维也纳，7月21-27日。OpenReview.net。https://openreview.net/ forum?id=pEWAcejiU2 [37] Google. 2024. 介绍 Gemini 2.0：我们面向代理时代的全新AI模型。https://blog.google/technology/google-deepmind/google-gemini-ai- update-december-2024 [38] Google. 2025. Gemini 2.5：我们最智能的AI模型。https://blog.google/ technology/google-deepmind/gemini-model-thinking-updates-march-2025/ [39] MADSys团队与Approaching.AI. 2025. 一种体验前沿LLM推理优化的灵活框架。https://github.com/kvcache- ai/ktransformers [40] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer 和 Amir Gholami. 2024. KVQuant：迈向拥有1000万上下文长度的LLM推理的KV缓存量化。arXiv预印本 arXiv:2401.18079 (2024)。 [41] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier 等人。2023. Mistral 7B。arXiv预印本 arXiv:2310.06825 (2023)。 [42] Zihe ng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, Yulu Jia, Sun He, Hongmin Chen, Zhihao Bai, Qi Hou, Shipeng Yan, Ding Zhou, Yiyao Sheng, Zhuo Jiang, Haohan Xu, Haoran Wei, Zhang Zhang, Pengfei Nie, Leqi Zou, Sida Zhao, Liang Xiang, Zherui Liu, Zhe Li, Xiaoying Jia, Jianxi Ye, Xin Jin 和 Xin Liu. 2024. MegaScale：将大型语言模型训练扩展到超过10,000个GPU。http://arxiv.org/abs/2402.15627 arXiv:2402.15627 [cs]。 [43] Norm Jouppi, George Kurian, Sheng Li, Peter Ma, Rahul Nagarajan, Lifeng Nai, Nishant Patil, Suvinay Subramanian, Andy Swing, Brian Towles, Clifford Young, Xiang Zhou, Zongwei Zhou 和 David A Patterson. 2023. TPU v4：一种具有硬件支持嵌入的光可重构超算，用于机器学习。在第50届国际计算机体系结构研讨会（Orlando, FL, USA）（ISCA '23）上发表。计算机协会，纽约，纽约州，美国，第82篇文章，14页。https://doi.org/10.1145/

3579371.3589350
[44] Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. 2024. GEAR: An Efficient KV Cache Compression Recipe for Near-Lossless Generative Inference of LLM. arXiv:2403.05527 [cs.LG]
[45] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *CoRR* abs/2001.08361 (2020). arXiv:2001.08361 https://arxiv.org/abs/2001.08361
[46] John Kim, Wiliam J. Dally, Steve Scott, and Dennis Abts. 2008. Technology-Driven, Highly-Scalable Dragonfly Topology. In *2008 International Symposium on Computer Architecture*. 77–88. https://doi.org/10.1109/ISCA.2008.19
[47] Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Reducing activation recomputation in large transformer models. *Proceedings of Machine Learning and Systems* 5 (2023).
[48] Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024. EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. https://openreview.net/forum?id=1NdN7eXyb4
[49] Heng Liao, Bingyang Liu, Xianping Chen, Zhigang Guo, Chuanning Cheng, Jianbing Wang, Xiangyu Chen, Peng Dong, Rui Meng, Wenjie Liu, Zhe Zhou, Ziyang Zhang, Yuhang Gai, Cunle Qian, Yi Xiong, Zhongwu Cheng, Jing Xia, Yuli Ma, Xi Chen, Wenhua Du, Shizhong Xiao, Chungang Li, Yong Qin, Liudong Xiong, Zhou Yu, Lv Chen, Lei Chen, Buyun Wang, Pei Wu, Junen Gao, Xiaochu Li, Jian He, Shizhuan Yan, and Bill McColl. 2025. UB-Mesh: a Hierarchically Localized nD-FullMesh Datacenter Network Architecture. arXiv:2503.20377 [cs.AR] https://arxiv.org/abs/2503.20377
[50] Sean Lie. 2022. Cerebras Architecture Deep Dive: First Look Inside the HW/SW Co-Design for Deep Learning : Cerebras Systems. In *2022 IEEE Hot Chips 34 Symposium (HCS)*. 1–34. https://doi.org/10.1109/HCS55958.2022.9895479
[51] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. In *MLSys*.
[52] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024. KIVI: A Tuning-Free Asymmetric 2bit Quantization for KV Cache. *arXiv preprint arXiv:2402.02750* (2024).
[53] Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiao Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Meng Xiao, Chenwu Liu, Jingyang Yuan, Shichang Zhang, Yiqiao Jin, Fan Zhang, Xian Wu, Hanqing Zhao, Dacheng Tao, Philip S. Yu, and Ming Zhang. 2025. Large Language Model Agent: A Survey on Methodology, Applications and Challenges. *arXiv preprint arXiv:2503.21460* (2025).
[54] Karthik Mandakolathur and Sylvain Jeaugey. 2022. Doubling all2all Performance with NVIDIA Collective Communication Library 2.12. https://developer.nvidia.com/blog/doubling-all2all-performance-with-nvidia-collective-communication-library-2-12/
[55] Mistral. 2024. Cheaper, Better, Faster, Stronger: Continuing to push the frontier of AI and making it accessible to all. https://mistral.ai/news/mixtral-8x22b
[56] Dheevatsa Mudigere, Yuchen Hao, Jianyu Huang, Zhihao Jia, Andrew Tulloch, Srinivas Sridharan, Xing Liu, Mustafa Ozdal, Jade Nie, Jongsoo Park, Liang Luo, Jie Amy Yang, Leon Gao, Dmytro Ivchenko, Aarti Basant, Yuxi Hu, Jiyan Yang, Ehsan K. Ardestani, Xiaodong Wang, Rakesh Komuravelli, Ching-Hsiang Chu, Serhat Yilmaz, Huayu Li, Jiyuan Qian, Zhuobo Feng, Yinbin Ma, Junjie Yang, Ellie Wen, Hong Li, Lin Yang, Chonglin Sun, Whitney Zhao, Dimitry Melts, Krishna Dhulipala, K. R. Kishore, Tyler Graf, Assaf Eisenman, Kiran Kumar Matam, Adi Gangidi, Guoqiang Jerry Chen, Manoj Krishnan, Avinash Nayak, Krishnakumar Nair, Bharath Muthiah, Mahmoud khorashadi, Pallab Bhattacharya, Petr Lapukhov, Maxim Naumov, Ajit Mathews, Lin Qiao, Mikhail Smelyanskiy, Bill Jia, and Vijay Rao. 2023. Software-Hardware Co-design for Fast and Scalable Training of Deep Learning Recommendation Models. http://arxiv.org/abs/2104.05158 arXiv:2104.05158 [cs].
[57] NVIDIA. 2022. Improving Network Performance of HPC Systems Using NVIDIA Magnum IO NVSHMEM and GPUDirect Async. https://developer.nvidia.com/blog/improving-network-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-and-gpudirect-async/
[58] NVIDIA. 2025. NVIDIA DGX Spark: A Grace Blackwell AI supercomputer on your desk. https://www.nvidia.com/en-us/products/workstations/dgx-spark/
[59] OpenAI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/
[60] OpenAI. 2024. Introducing OpenAI o1. https://openai.com/o1/
[61] OpenAI. 2025. Introducing OpenAI o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/.
[62] Kun Qian, Yongqing Xi, Jiamin Cao, Jiaqi Gao, Yichi Xu, Yu Guan, Binzhang Fu, Xuemei Shi, Fangbo Zhu, Rui Miao, Chao Wang, Peng Wang, Pengcheng Zhang, Xianlong Zeng, Eddie Ruan, Zhiping Yao, Ennan Zhai, and Dennis Cai. 2024. Alibaba HPN: A Data Center Network for Large Language Model Training. In *Proceedings of the ACM SIGCOMM 2024 Conference* (Sydney, NSW, Australia) *(ACM SIGCOMM '24)*. Association for Computing Machinery, New York, NY, USA, 691–706. https://doi.org/10.1145/3651890.3672265
[63] Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. 2024. Various lengths, constant speed: efficient language modeling with lightning attention. In *Proceedings of the 41st International Conference on Machine Learning* (Vienna, Austria) *(ICML'24)*. JMLR.org, Article 1688, 19 pages.
[64] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290 [cs.LG] https://arxiv.org/abs/2305.18290
[65] Md Shafayat Rahman, Saptarshi Bhowmik, Yevgeniy Ryasnianskiy, Xin Yuan, and Michael Lang. 2019. Topology-custom UGAL routing on dragonfly. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '19)*. Association for Computing Machinery, New York, NY, USA, Article 17, 15 pages. https://doi.org/10.1145/3295500.3356208
[66] Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. 2023. Microscaling Data Formats for Deep Learning. arXiv:2310.10537 [cs.LG] https://arxiv.org/abs/2310.10537
[67] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347
[68] ByteDance Seed. 2025. Seed1.5-Thinking: Advancing Superb Reasoning Models with Reinforcement Learning. arXiv:2504.13914 [cs.CL] https://arxiv.org/abs/2504.13914
[69] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300 [cs.CL] https://arxiv.org/abs/2402.03300
[70] Noam Shazeer. 2019. Fast Transformer Decoding: One Write-Head is All You Need. *CoRR* abs/1911.02150 (2019). http://arxiv.org/abs/1911.02150
[71] Qwen Team. 2025. Qwen3: Think Deeper, Act Faster. https://github.com/QwenLM/Qwen3
[72] Song Wang, Bing Yu, Wenwu Xiao, Fujun Bai, Xiaodong Long, Liang Bai, Xuerong Jia, Fengguo Zuo, Jie Tan, Yixin Guo, Peng Sun, Jun Zhou, Qiong Zhan, Sheng Hu, Yu Zhou, Yi Kang, Qiwei Ren, and Xiping Jiang. 2023. A 135 GBps/Gbit 0.66 pJ/bit Stacked Embedded DRAM with Multilayer Arrays by Fine Pitch Hybrid Bonding and Mini-TSV. In *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. 1–2. https://doi.org/10.23919/VLSITechnologyandCir57934.2023.10185427
[73] xAI. 2024. Grok-2 Beta Release. https://x.ai/news/grok-2.
[74] xAI. 2024. Our Gigafactory of Compute:Colossus. https://x.ai/colossus.
[75] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115* (2024).
[76] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Y. X. Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. 2025. Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention. https://arxiv.org/abs/2502.11089
[77] Chenggang Zhao, Liang Zhao, Jiashi Li, and Zhean Xu. 2025. DeepGEMM: clean and efficient FP8 GEMM kernels with fine-grained scaling. https://github.com/deepseek-ai/DeepGEMM.
[78] Chenggang Zhao, Shangyan Zhou, Liyue Zhang, Chengqi Deng, Zhean Xu, Yuxuan Liu, Kuai Yu, Jiashi Li, and Liang Zhao. 2025. DeepEP: an efficient expert-parallel communication library. https://github.com/deepseek-ai/DeepEP.
[79] Size Zheng, Jin Fang, Xuegui Zheng, Qi Hou, Wenlei Bao, Ningxin Zheng, Ziheng Jiang, Dongyang Wang, Jianxi Ye, Haibin Lin, Li-Wen Chang, and Xin Liu. 2025. TileLink: Generating Efficient Compute-Communication Overlapping Kernels using Tile-Centric Primitives. arXiv:2503.20313 [cs.DC] https://arxiv.org/abs/2503.20313
[80] Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. 2024. DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*. USENIX Association, Santa Clara, CA, 193–210. https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin

3579371.3589350
[44] 郝康，张青如，Souvik Kundu，Jeong Geonhwa，刘昭兴，Krishna Tushar，赵拓。2024。GEAR：一种用于近乎无损生成推理的高效KV缓存压缩方案，适用于LLM。arXiv:2403.05527 [cs.LG] [45] Jared Kaplan，Sam McCandlish，Tom Henighan，Tom B. Brown，Benjamin Chess，Rewon Child，Scott Gray，Alec Radford，Jeffrey Wu，Dario Amodei。2020。神经语言模型的扩展规律。CoRR abs/2001.08361（2020）。arXiv:2001.08361 https://arxiv.org/abs/2001.08361 [46] John Kim，William J. Dally，Steve Scott，Dennis Abts。2008。技术驱动的高扩展性Dragonfly拓扑结构。在2008年国际计算机体系结构研讨会。77–88。https://doi.org/10.1109/ISCA.2008.19 [47] Vijay Anand Korthikanti，Jared Casper，Sangkug Lym，Lawrence McAfee，Michael Andersch，Mohammad Shoeybi，Bryan Catanzaro。2023。减少大型Transformer模型中的激活重计算。机器学习与系统会议论文集 5（2023）。 [48] Yuhui Li，Fangyun Wei，Chao Zhang，Hongyang Zhang。2024。EAGLE：投机采样需要重新思考特征不确定性。在第41届国际机器学习会议（ICML 2024），奥地利维也纳，2024年7月21-27日。OpenReview.net。https://openreview.net/forum?id=1NdN7eXyb4 [49] Heng Liao，Bingyang Liu，Xianping Chen，Zhigang Guo，Chuanning Cheng，Jianbing Wang，Xiangyu Chen，Peng Dong，Rui Meng，Wenjie Liu，Zhe Zhou，Ziyang Zhang，Yuhang Gai，Cunle Qian，Yi Xiong，Zhongwu Cheng，Jing Xia，Yuli Ma，Xi Chen，Wenhua Du，Shizhong Xiao，Chungang Li，Yong Qin，Liudong Xiong，Zhou Yu，Lv Chen，Lei Chen，Buyun Wang，Pei Wu，Junen Gao，Xiaochu Li，Jian He，Shizhuan Yan，Bill McColl。2025。UB-Mesh：一种分层本地化的nD全网数据中心网络架构。arXiv:2503.20377 [cs.AR] https://arxiv.org/abs/2503.20377 [50] Sean Lie。2022。Cerebras架构深度解析：深入了解深度学习的硬件/软件协同设计：Cerebras Systems。在2022年IEEE热芯片34研讨会（HCS）。1–34。https://doi.org/10.1109/HCS55958.2022.9895479 [51] Ji Lin，Jiaming Tang，Haotian Tang，Shang Yang，Wei-Ming Chen，Wei-Chen Wang，Guangxuan Xiao，Xingyu Dang，Chuang Gan，Song Han。2024。AWQ：面向LLM压缩与加速的激活感知权重量化。在MLSys会议。 [52] Zirui Liu，Jiayi Yuan，Hongye Jin，Shaochen Zhong，Zhaozhuo Xu，Vladimir Braverman，Beidi Chen，Xia Hu。2024。KIVI：一种无需调优的非对称2比特KV缓存量化方法。arXiv预印本 arXiv:2402.02750（2024）。 [53] Junyu Luo，Weizhi Zhang，Ye Yuan，Yusheng Zhao，Junwei Yang，Yiyang Gu，Bohan Wu，Binqi Chen，Ziyue Qiao，Qingqing Long，Rongcheng Tu，Xiao Luo，Wei Ju，Zhiping Xiao，Yifan Wang，Meng Xiao，Chenwu Liu，Jingyang Yuan，Shichang Zhang，Yiqiao Jin，Fan Zhang，Xian Wu，Hanqing Zhao，Dacheng Tao，Philip S. Yu，Ming Zhang。2025。大型语言模型代理：方法论、应用与挑战综述。arXiv预印本 arXiv:2503.21460（2025）。 [54] Karthik Mandakolathur和 Sylvain Jeaugey。2022。使用NVIDIA集体通信库2.12将all2all性能翻倍。https://developer.nvidia.com/blog/doubling-all2all-performance-with- nvidia-collective-communication-library-2-12/ [55] Mistral。2024。更便宜、更好、更快、更强：持续推动AI前沿，让其惠及所有人。https://mistral.ai/news/mixtral-8x22b [56] Dheevatsa Mudigere，郝宇辰，黄建宇，贾志豪，Andrew Tulloch，Srinivas Sridharan，Xing Liu，Mustafa Ozdal，Jade Nie，Jongsoo Park，Liang Luo，Jie Amy Yang，Leon Gao，Dmytro Ivchenko，Aarti Basant，Yuxi Hu，Jiyan Yang，Ehsan K. Ardestani，Wang Xiaodong，Rakesh Komuravelli，Ching-Hsiang Chu，Serhat Yilmaz，Li Huayu，Jiyuan Qian，Feng Zhuobo，Ma Yinbin，Junjie Yang，Ellie Wen，Hong Li，Lin Yang，Chonglin Sun，Whitney Zhao，Dimitry Melts，Krishna Dhulipala，K. R. Kishore，Tyler Graf，Assaf Eisenman，Kiran Kumar Matam，Adi Gangidi，Guoqiang Jerry Chen，Manoj Krishnan，Avinash Nayak，Krishnaku-mar Nair，Bharath Muthiah，Mahmoud Khorashadi，Pallab Bhattacharya，Petr Lapukhov，Maxim Naumov，Ajit Mathews，Lin Qiao，Mikhail Smelyanskiy，Bill Jia，Vijay Rao。2023。深度学习推荐模型的快速可扩展训练的软件-硬件协同设计。http://arxiv.org/abs/2104.05158 arXiv:2104.05158 [cs]。 [57] NVIDIA。2022。利用NVIDIA Magnum IO NVSHMEM和GPUDirect Async提升HPC系统网络性能。https://developer.nvidia.com/blog/improving-network-performance-of- hpc-systems-using-nvidia-magnum-io-nvshmem-and-gpudirect-async/ [58] NVIDIA。2025。NVIDIA DGX Spark：桌面上的Grace Blackwell AI超级计算机。https://www.nvidia.com/en-us/products/workstations/dgx-spark/ [59] OpenAI。2024。你好GPT-4o。https://openai.com/index/hello-gpt-4o/ [60] OpenAI。2024。介绍OpenAI o1。https://openai.com/o1/ [61] OpenAI。2025。介绍OpenAI o3和o4-mini。https://openai.com/index/ introducing-o3-and-o4-mini/ [62] 钱坤，席永庆，曹佳敏，高嘉琪，徐亦驰，关瑜，傅志昂，史学梅，朱芳博，苗锐，王超，王鹏，张鹏程，曾宪龙，阮爱迪，姚志鹏，翟恩南，蔡德胜。2024。阿里巴巴HPN：一种用于大规模语言模型训练的数据中心网络。在2024年ACM SIGCOMM会议论文集（悉尼，新南威尔士，澳大利亚）(ACM SIGCOMM '24). 计算机协会，纽约，纽约州，美国，691–706。https://doi.org/10.1145/3651890.3672265 [63] 秦震，孙伟高，李东，沈旭阳，孙伟轩，钟逸然。2024年。多长度，恒定速度：用闪电注意力实现高效语言建模。在第41届国际机器学习会议（奥地利维也纳）论文集（ICML'24）。JMLR.org，第1688篇，19页。 [64] 拉斐尔·拉菲洛夫，阿尔奇特·夏尔马，埃里克·米切尔，斯特凡诺·厄尔蒙，克里斯托弗·D·曼宁，切尔西·芬恩。2024年。直接偏好优化：你的语言模型实际上是奖励模型。arXiv:2305.18290 [cs.LG] https://arxiv.org/abs/2305.18290 [65] 马德·沙法亚特·拉赫曼，萨普塔尔希·博伊米克，叶夫根尼·里亚斯尼安斯基，辛恩·袁，迈克尔·朗。2019年。龙飞凤舞上的拓扑定制UGAL路由。在高性能计算、网络、存储与分析国际会议（丹佛，科罗拉多州）（SC'19）论文集。计算机协会，纽约，纽约州，美国，第17篇，15页。https://doi.org/10.1145/ 3295500.3356208 [66] 比塔·达尔维什·鲁哈尼，里奇·赵，安基特·莫尔，马修·霍尔，阿里雷扎·科哈多马拉迪，夏季·邓，德鲁夫·乔达里，马里乌斯·科尔内亚，埃里克·德林格，克里斯托弗·德诺尔夫，斯托西奇·杜桑，文穆吉尔·埃兰戈，马克西米连·戈卢布，亚历山大·海内克，菲尔·詹姆斯·罗克斯比，达尔梅什·贾尼，戈拉夫·科尔赫，马丁·朗汉默，阿达·李，利维·梅尔尼克，玛拉尔·梅斯马霍斯罗沙希，安德烈斯·罗德里格斯，迈克尔·舒尔特，拉苏尔·沙菲普尔，雷伊·邵，迈克尔·萧，迈克尔·邵，普拉迪普·杜贝，保留斯·米基凯维奇，马克西米连·瑙莫夫，科林·维雷利，拉尔夫·维蒂格，道格·伯格，埃里克·钟。2023年。深度学习的微型数据格式。arXiv:2310.10537 [cs.LG] https://arxiv.org/abs/2310.10537 [67] 约翰·舒尔曼，菲利普·沃尔斯基，普拉弗拉·达里沃尔，亚历克斯·拉德福德，奥列格·克里莫夫。2017年。近端策略优化算法。arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347 [68] 字节跳动种子。2025年。Seed1.5-Thinking：用强化学习推进卓越推理模型。arXiv:2504.13914 [cs.CL] https://arxiv.org/abs/ 2504.13914 [69] 邵志宏，王佩怡，朱启浩，徐润新，宋俊孝，毕晓，张浩伟，张明川，李耀坤，吴毅，郭大雅。2024年。DeepSeek- 数学：推动开放语言模型中的数学推理极限。arXiv:2402.03300 [cs.CL] https://arxiv.org/abs/2402.03300 [70] 诺姆·沙泽尔。2019年。快速变换器解码：只需一个写入头。CoRR abs/1911.02150（2019年）。http://arxiv.org/abs/1911.02150 [71] Qwen团队。2025年。Qwen3：思考更深，行动更快。https://github.com/ QwenLM/Qwen3 [72] 王松，余冰，肖文武，白福军，龙晓东，白亮，贾学荣，左丰国，谭杰，郭亦欣，孙鹏，周俊，詹琼，胡胜，周瑜，康毅，任启伟，鲍克勤，杨科新，尤雷，李梅，薛明峰，张佩，朱秦，门瑞，林润吉，李天浩，唐天翼，夏婷玉，任兴章，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟。2023年。一个135 GBps/Gbit 0.66 pJ/bit的堆叠嵌入式DRAM，采用精细间距混合键合和迷你TSV。2023 IEEE VLSI技术与电路研讨会，第1-2页。https://doi.org/10.23919/ VLSITechnologyandCir57934.2023.10185427 [73] xAI。2024年。Grok-2 Beta版本。https://x.ai/news/grok-2. [74] xAI。2024年。我们的计算巨型工厂：Colossus。https://x.ai/colossus。 [75] 杨安，杨宝松，张贝辰，惠彬源，郑博，余炳，李成远，刘大恒，黄飞，魏浩然，林欢，张建，朱建宏，张建新，杨佳熙，周景仁，林俊阳，邓凯，卢克明，李雷，李梅，薛明峰，张佩，朱秦，门瑞，林宣成，李天浩，唐天翼，夏婷玉，任兴章，任轩成，范杨，苏杨，张亦昌，周瑜，刘亦强，任启伟。2024年。Qwen2.5技术报告。arXiv预印本 arXiv:2412.15115（2024年）。 [76] 袁靖阳，高浩，戴大迈，罗俊宇，赵亮，张正彦，谢振达，魏晓宇，王利，肖志平，王玉清，阮冲，张明，梁文峰，曾旺丁。2025年。原生稀疏注意力：硬件对齐且本地可训练的稀疏注意力。https://arxiv.org/abs/2502.11089 [77] 赵成刚，赵亮，李家石，徐哲。2025年。DeepGEMM：具有细粒度缩放的干净高效FP8 GEMM内核。https://github.com/deepseek-ai/DeepGEMM。 [78] 赵成刚，周尚岩，张丽月，邓成奇，徐哲，刘宇轩，余快，李家石，赵亮。2025年。DeepEP：一种高效的专家并行通信库。https://github.com/deepseek-ai/DeepEP。 [79] 郑思泽，方金，郑学贵，侯启，鲍文雷，郑宁新，江子恒，王东阳，叶建西，林海滨，张立文，刘新。2025年。TileLink：使用以块为中心的原语生成高效的计算-通信重叠内核。arXiv:2503.20313 [cs.DC] https://arxiv.org/abs/ 2503.20313 [80] 钟银敏，刘胜宇，陈俊达，胡建波，朱亦博，刘轩，辛金，张浩。2024年。DistServe：为优化吞吐量的大型语言模型服务实现预填充与解码的拆分。第18届USENIX操作系统设计与实现研讨会（OSDI 24）。USENIX协会，圣克拉拉，加利福尼亚州，193–210。https://www.usenix.org/conference/osdi24/ presentation/zhong-yinmin