DeepSeek-V3 / R1 推理系统概览

DeepSeek
已认证账号

目录

DeepSeek-V3⁺ / R1 推理系统⁺的优化目标是：更大的吞吐，更低的延迟。

为了实现这两个目标，我们的方案是使用大规模跨节点专家并行⁺（Expert Parallelism / EP）。首先 EP 使得 batch size 大大增加，从而提高 GPU 矩阵乘法的效率，提高吞吐。其次 EP 使得专家分散在不同的 GPU 上，每个 GPU 只需要计算很少的专家（因此更少的访存需求），从而降低延迟。

但 EP 同时也增加了系统的复杂性。复杂性主要体现在两个方面：

1. EP 引入跨节点的传输。为了优化吞吐，需要设计合适的计算流程使得传输和计算可以同步进行。
2. EP 涉及多个节点，因此天然需要 Data Parallelism⁺（DP），不同的 DP 之间需要进行负载均衡。

因此，本文的主要内容是如何使用 EP 增大 batch size，如何隐藏传输的耗时，如何进行负载均衡。

大规模跨节点专家并行（Expert Parallelism / EP）

由于 DeepSeek-V3 / R1 的专家数量众多，并且每层 256 个专家中仅激活其中 8 个。模型的高度稀疏性决定了我们必须采用很大的 overall batch size，才能给每个专家提供足够的 expert batch size，从而实现更大的吞吐、更低的延时。需要大规模跨节点专家并行（Expert Parallelism / EP）。

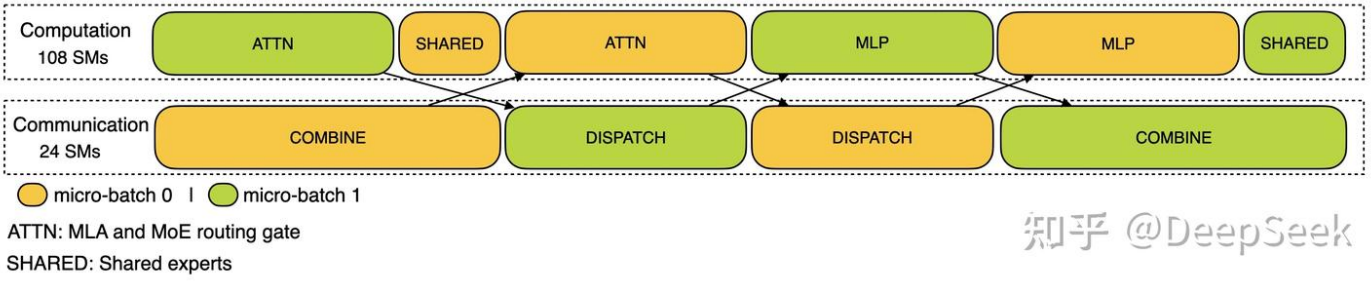
我们采用多机多卡间的专家并行策略来达到以下目的：

- **Prefill**：路由专家 EP32、MLA 和共享专家 DP32，一个部署单元是 4 节点，32 个冗余路由专家，每张卡 9 个路由专家和 1 个共享专家
- **Decode**：路由专家 EP144、MLA 和共享专家 DP144，一个部署单元是 18 节点，32 个冗余路由专家，每张卡 2 个路由专家和 1 个共享专家

计算通信重叠

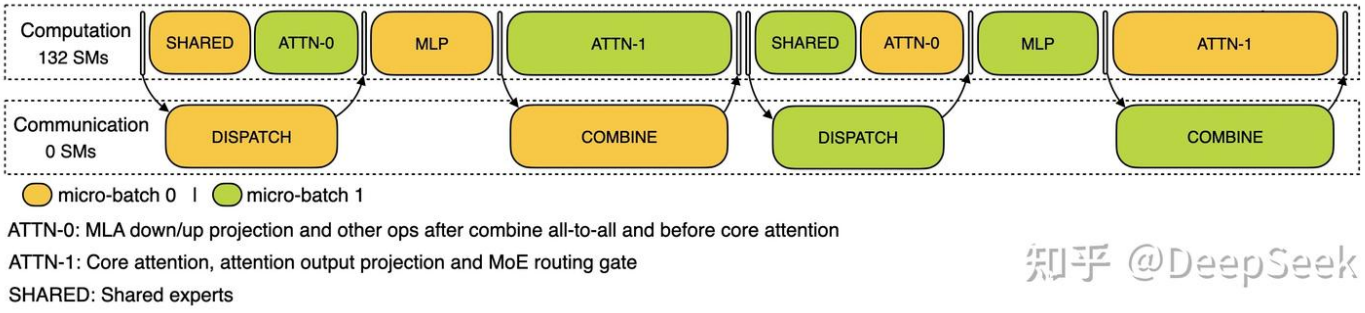
多机多卡的专家并行会引入比较大的通信开销，所以我们使用了双 batch 重叠来掩盖通信开销，提高整体吞吐。

对于 prefill 阶段，两个 batch 的计算和通信交错进行，一个 batch 在进行计算的时候可以去掩盖另一个 batch 的通信开销；



Prefill 阶段的双 batch 重叠

对于 decode 阶段，不同阶段的执行时间有所差别，所以我们把 attention 部分拆成了两个 stage，共计 5 个 stage 的流水线来实现计算和通信的重叠。



Decode 阶段的双 batch 重叠

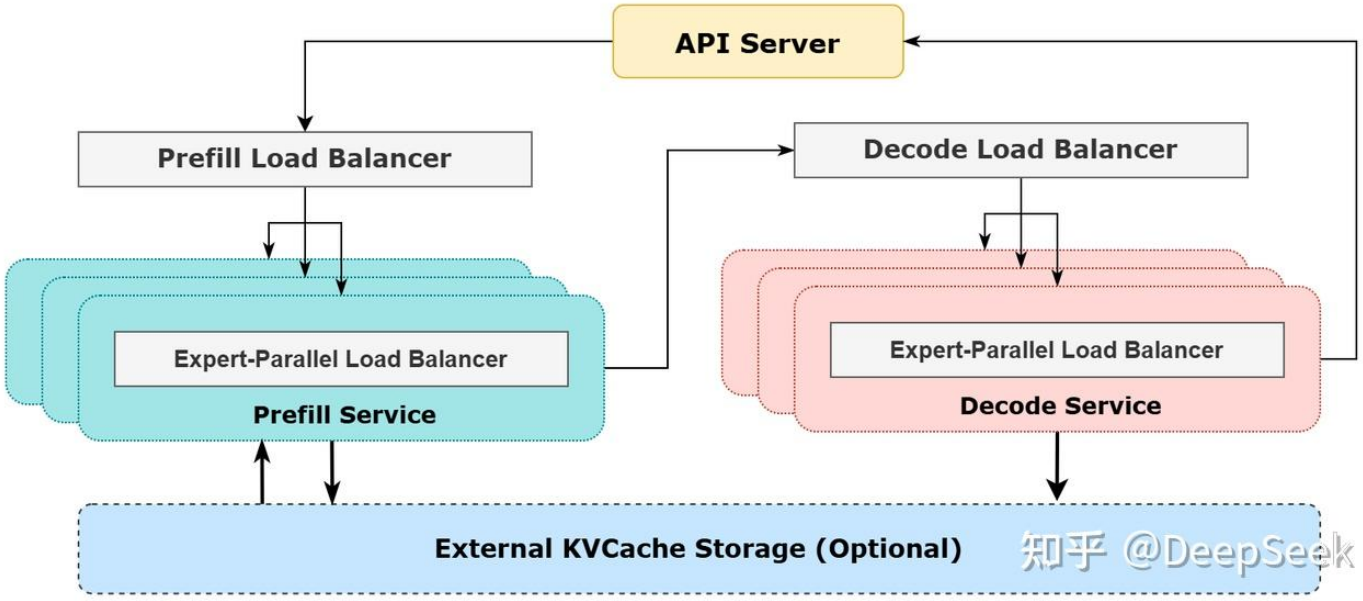
关于更多双 batch 重叠的细节，可以参考我们的 profiling 数据的 GitHub 仓库：github.com/deepseek-ai/...。

尽可能地负载均衡

由于采用了很大规模的并行（包括数据并行和专家并行），如果某个 GPU 的计算或通信负载过重，将成为性能瓶颈，拖慢整个系统；同时其他 GPU 因为等待而空转，造成整体利用率下降。因此我们需要尽可能地为每个 GPU 分配均衡的计算负载、通信负载。

1. **Prefill Load Balancer**
    1. 核心问题：不同数据并行（DP）实例上的请求个数、长度不同，导致 core-attention 计算量、dispatch 发送量也不同
    2. 优化目标：各 GPU 的计算量尽量相同（core-attention 计算负载均衡）、输入的 token 数量也尽量相同（dispatch 发送量负载均衡），避免部分 GPU 处理时间过长
2. **Decode Load Balancer**
    1. 核心问题：不同数据并行（DP）实例上的请求数量、长度不同，导致 core-attention 计算量（与 KVCache 占用量相关）、dispatch 发送量不同
    2. 优化目标：各 GPU 的 KVCache 占用量尽量相同（core-attention 计算负载均衡）、请求数量尽量相同（dispatch 发送量负载均衡）
3. **Expert-Parallel Load Balancer**
    1. 核心问题：对于给定 MoE 模型，存在一些天然的高负载专家（expert），导致不同 GPU 的专家计算负载不均衡
    2. 优化目标：每个 GPU 上的专家计算量均衡（即最小化所有 GPU 的 dispatch 接收量的最大值）

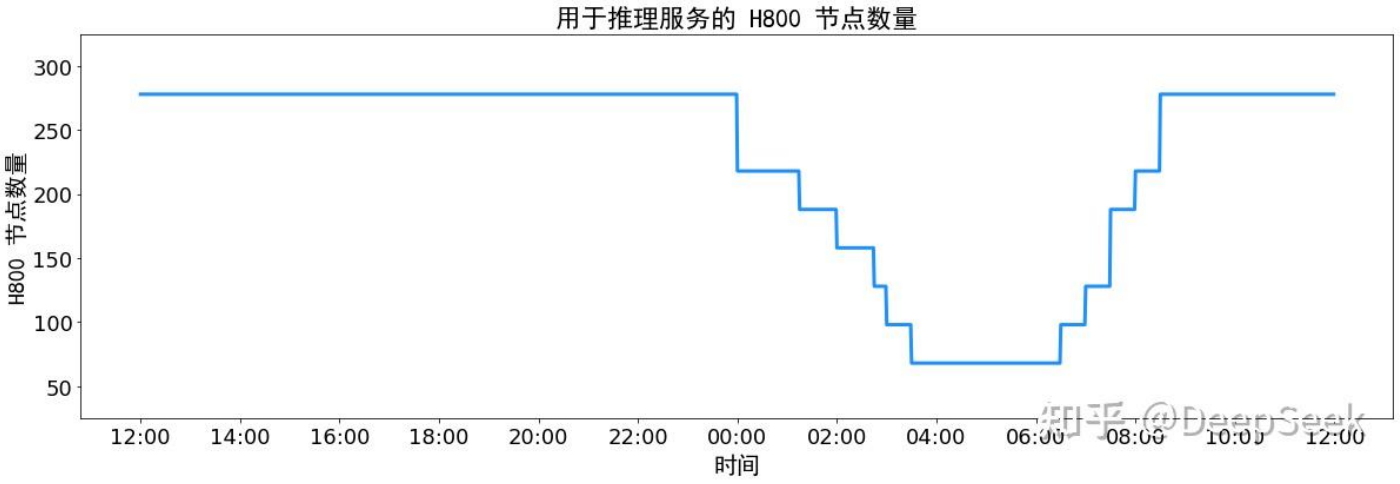
参考架构图



线上系统的实际统计数据

DeepSeek V3 和 R1 的所有服务均使用 H800 GPU⁺，使用和训练一致的精度，即矩阵计算和 dispatch 传输采用和训练一致的 FP8 格式，core-attention 计算和 combine 传输采用和训练一致的 BF16，最大程度保证了服务效果。

另外，由于白天的服务负荷高，晚上的服务负荷低，因此我们实现了一套机制，在白天负荷高的时候，用所有节点部署推理服务。晚上负荷低的时候，减少推理节点，以用来做研究和训练。在最近的 24 小时里（北京时间 2025/02/27 12:00 至 2025/02/28 12:00），DeepSeek V3 和 R1 推理服务占用节点总和，峰值占用为 278 个节点，平均占用 226.75 个节点（每个节点为 8 个 H800 GPU）。假定 GPU 租赁成本为 2 美金/小时，总成本为 $87,072/天。

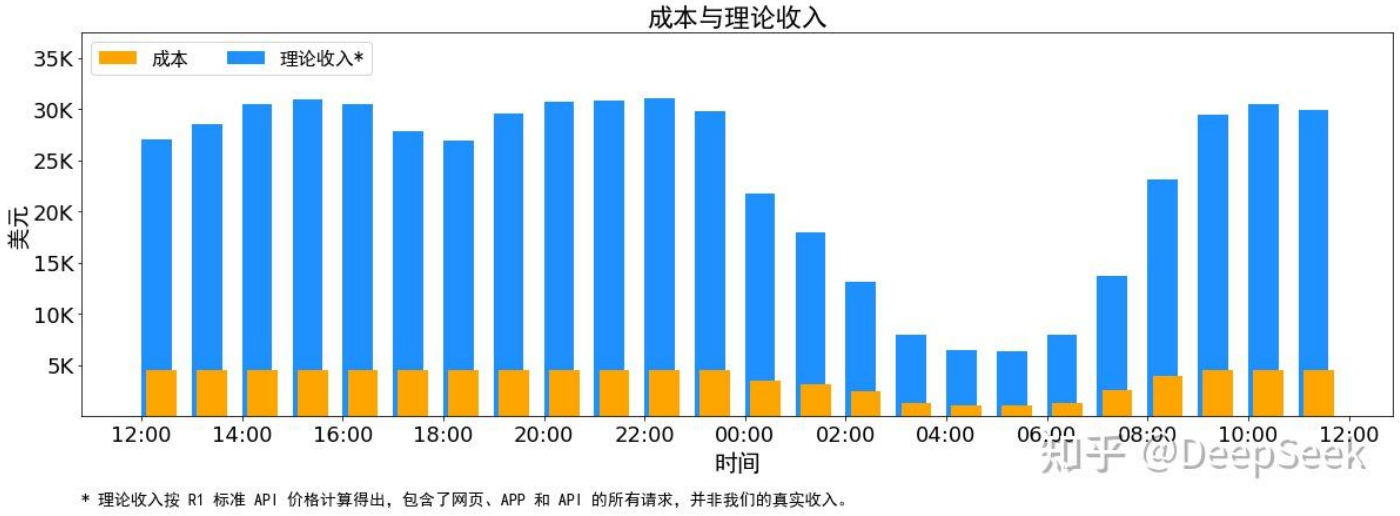


在 24 小时统计时段内，DeepSeek V3 和 R1：

- 输入 token 总数为 608B，其中 342B tokens（56.3%）命中 KVCache 硬盘缓存。
- 输出 token 总数为 168B。平均输出速率为 20~22 tps，平均每输出一个 token 的 KVCache 长度是 4989。
- 平均每台 H800 的吞吐量为：对于 prefill 任务，输入吞吐约 73.7k tokens/s（含缓存命中）；对于 decode 任务，输出吞吐约 14.8k tokens/s。

以上统计包括了网页、APP 和 API 的所有负载。如果所有 tokens 全部按照 DeepSeek R1 的定价[1]计算，理论上一天的总收入为 $562,027，成本利润率 545%。

> 当然我们实际上没有这么多收入，因为 V3 的定价更低，同时收费服务只占了一部分，另外夜间还会有折扣。



参考

1. ^ DeepSeek R1 的定价：$0.14 / 百万输入 tokens (缓存命中)，$0.55 / 百万输入 tokens (缓存未命中)，$2.19 / 百万输出 tokens。

发布于 2025-03-01 12:02 · IP 属地北京

DeepSeek