PR #39568 完整报告

vllm-project/vllm

[RFC] Replace shared-memory routed experts with ModelRunnerOutput transfer and HTTP support

合并时间：2026-05-14 22:12

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39568

执行摘要

- 一句话：重构 MoE 路由捕获传输层，移除共享内存，使用异步 D2H，支持 HTTP 返回。
- 推荐动作：值得精读。该 PR 设计了一套从 GPU 到 Scheduler 的完整异步数据传输管道，对 ModelRunnerOutput 扩展具有参考价值。特别是 pre-free capture 解决异步调度中数据竞争的手法，以及复用现有 IPC 路径实现零额外同步的设计，可推广到其他需要返传 GPU 状态的场景。但需密切关注外部 KV 块问题的后续修复。

功能与动机

原有的 enable_return_routed_experts 实现依赖 SharedMemory + fcntl.flock + /tmp 锁文件，Worker 每步执行 cpu().numpy() 同步 CUDA 调用，Scheduler 对每个完成请求执行阻塞的 flock 和共享内存读取。在 DP+EP 环境下（Issue #38079），这导致频繁的 NCCL 超时和吞吐量骤降。PR body 还列举了 SP + modular 内核下的 AssertionError，以及 abort/preemption 场景下数据丢失错位等问题。

实现拆解

1. 移除共享内存层，改用 ModelRunnerOutput 管道 - RoutedExpertsCapturer 简化为纯 GPU int32 缓冲区，仅保留 capture / clear / get_device_buffer 接口；所有 fcntl、tempfile、shared_memory 代码全部删除（routed_experts_capturer.py）。 - 新增 RoutedExpertsTensors / RoutedExpertsLists（outputs.py），作为设备侧和 CPU 侧的定型数据传输格式，复用现有的 ShmRingBuffer（同节点）和 zmq（跨节点）IPC 路径，零额外同步开销。 - 新增 RoutedExpertsManager（scheduler.py），在 Scheduler 端按物理 KV 槽索引 (slot_id = block_id * block_size + offset) 组织专家路由数据，使 prefix-caching 重用自动生效。
2. 适配 Sequence Parallelism 下的 topk_ids 形状 - 在 capture() 中增加 SP 分支：通过 TP 组的 all_gather 还原完整张量，然后按 DP rank 的起始位置切片写入设备缓冲区；丢弃 ceil-div 填充行，并统一使用 int32 类型避免 NCCL 类型依赖。
3. 实现异步 D2H 传输 - 在 GPUModelRunner 中预分配两个固定 CPU 缓冲区（routed_experts_cpu 和 routed_experts_slot_mapping_cpu）和一个私有 GPU 拷贝（routed_experts_slot_mapping_device）。 - 同步调度路径：copy_(non_blocking=True) 入列，后续 _to_list 中的 event.synchronize() 自然覆盖。 - 异步调度路径：在 execute_model 末尾通过 AsyncGPUModelRunnerOutput 携带 RoutedExpertsTensors，

在拷贝流上执行非阻塞 D2H，get_output() 时同步并转为 RoutedExpertsLists。

4. 处理提前中止与被抢占场景的数据完整性 - 引入 RoutedExpertsCache（scheduler.py），在释放 KV 块前通过 cache.capture() 保存当前路由数据快照。 - 采用"先存储后刷新"顺序：store_batch() 先行写入当前步生成的路由，再调用 cache.refresh_pending() 重读。 - _get_routed_experts 结合 cache.get_best() 比较快照长度，返回更完整的数据，确保不会因提前释放块而丢失行。
5. HTTP 输出支持 - 在 disagg/serving.py 中将 RoutedExpertsLists 用 np.save 编码为二进制，通过 SIMD 加速的 pybase64 转为 ASCII 字符串，嵌入 GenerateResponseChoice.routed_experts 字段。
6. 配置与兼容性检查 - 在 vllm/config/vllm.py 的 __post_init__ 中增加断言：enable_return_routed_experts 与 pipeline_parallel_size > 1、CPU offloading 互斥。 - 测试文件同步适配新接口并验证 SP 与 async pipeline 场景。

关键文件：

- vllm/model_executor/layers/fused_moe/routed_experts_capturer.py（模块 路由捕获器；类别 source；类型 data-contract；符号 _file_lock, _create_or_attach_shared_memory, get_num_experts, RoutedExpertsCapturer）：核心重构文件，移除共享内存和文件锁，将全局单例简化为纯 GPU 设备缓冲区，新增 SP 分支和 int32 传输。是整个变更的入口。
- vllm/v1/outputs.py（模块 输出模型；类别 source；类型 core-logic；符号 RoutedExpertsTensors, to_cpu_nonblocking, tolists, RoutedExpertsLists）：新增 RoutedExpertsTensors 和 RoutedExpertsLists 数据类，定义异步 D2H 转换和 tolists 方法，是数据管道的核心契约。
- vllm/v1/core/sched/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _get_routed_experts）：引入 RoutedExpertsManager 替换原有的 RoutedExpertsReader，管理按物理槽索引的缓冲区和执行 store_batch；添加 RoutedExpertsCache 处理 pre-free capture。
- vllm/v1/worker/gpu_model_runner.py（模块 模型运行器；类别 source；类型 data-contract）：集成异步 D2H 拷贝：预分配固定 CPU 缓冲区和私有 GPU 副本，在 sync/async 路径中发射非阻塞拷贝，通过 AsyncGPUModelRunnerOutput 传递 RoutedExpertsTensors。
- vllm/config/vllm.py（模块 配置层；类别 source；类型 core-logic）：在 __post_init__中增加 enable_return_routed_experts 与 PP、CPU offload 的兼容性检查，防止配置冲突。
- tests/model_executor/test_routed_experts_capture.py（模块 测试；类别 test；类型 test-coverage）：更新测试以匹配新的 capturer 接口并覆盖 SP 和 async pipeline 场景。
- vllm/entrypoints/serve/disagg/serving.py（模块 服务端；类别 source；类型 dependency-wiring）：在 disagg 服务中添加 routed_experts 的 base64 .npy 编码，供 HTTP 客户端使用。

关键符号：RoutedExpertsCapturer.capture, RoutedExpertsTensors.to_cpu_nonblocking, RoutedExpertsTensors.tolists, RoutedExpertsManager.store_batch, RoutedExpertsCache.capture, AsyncGPUModelRunnerOutput.get_output,

Scheduler._get_routed_experts

关键源码片段

vllm/v1/outputs.py

新增 RoutedExpertsTensors 和 RoutedExpertsLists 数据类，定义异步 D2H 转换和 tolists 方法，是数据管道的核心契约。

```
class RoutedExpertsTensors(NamedTuple):
    # (num_scheduled_tokens, num_layers, num_experts_per_tok)
    routing_data: torch.Tensor
    # (num_scheduled_tokens,)
    slot_mapping: torch.Tensor

    def to_cpu_nonblocking(self) -> "RoutedExpertsTensors":
        # 在拷贝流上执行非阻塞 D2H；确保当前流等待默认流后执行
        if self.routing_data.device.type == "cpu":
            return self
        return RoutedExpertsTensors(
            self.routing_data.to("cpu", non_blocking=True),
            self.slot_mapping.to("cpu", non_blocking=True),
        )

    def tolists(self) -> "RoutedExpertsLists":
        # 转为 numpy 数组给 Scheduler 消费；已位于 CPU 时无开销
        return RoutedExpertsLists(
            self.routing_data.cpu().numpy(),
            self.slot_mapping.cpu().numpy(),
        )


class RoutedExpertsLists(NamedTuple):
    # 每步所有 token 的路由数据，按 slot_mapping 索引到物理 KV 槽
    routing_data: np.ndarray # (num_scheduled_tokens, num_layers, num_experts_per_tok)
    slot_mapping: np.ndarray # (num_scheduled_tokens,)
```

评论区精华

- 竞争条件风险（gemini-code-assist[bot]）：指出在 batch_queue_size > 1 时，共享缓冲区 self.routed_experts_cpu 会被并发步覆写，且元数据陈旧。作者通过 RoutedExpertsCache 和 pre-free capture 机制解决了此问题。
- 阻塞式同步瓶颈（gemini-code-assist[bot]）：原 slot_mapping[...].cpu().numpy() 是阻塞同步点，改版后使用非阻塞 D2H 和独立拷贝流来消除延迟。
- 外部填充 KV 块的正确性（S1ro1 in PR comment）：当请求通过 P/D 拆分或 CPU offload 获得外部填充的 KV 块时，本方案按物理槽索引可能读到未计算的过期数据。作者回复该场景已在 pre-free capture 中通过缓存截止到释放时刻的快照来规避，但未完全禁止外部填充的 KV 块进入此功能（待后续跟进）。

- 配置 guard 建议（aoshen02 in review）：建议在 config 中明确阻止与 CPU offload 和 KV connector 的兼容性问题，作者已在 vllm/config/vllm.py 中添加了 PP 和 offload 的检查。
- 共享缓冲区并发覆写风险 (correctness): 作者通过引入 RoutedExpertsCache 和 pre-free capture 机制，在释放块前保存快照，并通过 take-the-best 策略确保完整性。已解决。
- 阻塞式同步瓶颈 (performance): 改为非阻塞 D2H 拷贝到固定 CPU 缓冲区，利用 async_copy_stream 和 event.synchronize() 实现异步，不再阻塞默认流。已解决。
- 外部填充 KV 块的数据正确性 (correctness): 作者回应当前方案依赖本地计算的 KV 槽，当块来自外部时不会写入路由数据；pre-free capture 仅缓解部分场景，整体问题未完全解决，需后续跟进。
- 配置兼容性 guard (design): 作者在 vllm/config/vllm.py 的 __post_init__中增加了相应 ValueError 检查。已解决。

风险与影响

- 风险：
  1. 外部KV块数据一致性：当请求使用了来自KVconnector（如P/D拆分、CPUoffload）的外部计算块时，按物理槽索引可能读取到不存在的路由数据，导致静默错误。当前实现并未阻断此类组合，存在潜在误用风险。
  2. 共享缓冲区并发覆写：虽然添加了 pre-free capture，但在极端高并发调度下如果 update_from_output 与后续 schedule() 交错过于复杂，仍不排除数据竞争。
  3. 仅支持 int32 传输：路由数据类型固定为 int32，若未来模型使用更大数量的 expert（超过 65535），Scheduler 侧窄数据类型 uint8/uint16 可能溢出。
  4. HTTP base64 负载膨胀：.npy + base64 编码使数据大小增加约 33%，对于大 batch 高频调用可能带来网络和序列化开销。
  5. 不完整测试覆盖：测试以单元测试为主，缺少对 DP+EP 跨节点、多步 async pipeline 的集成测试。
- 影响：
  - 用户 / 开发者：启用 --enable-return-routed-experts 后，MoE 路由数据可用于训练或分析管道，且不再引发 NCCL 超时。但需要显式禁用 CPU offload 和 PP。
  - 系统性能：移除了每步的同步 CUDA 调用和每次请求的阻塞文件锁，预期在 DP+EP 场景下大幅提升吞吐量，对于单步延迟也有减小。
  - 团队维护：接口从全局单例 RoutedExpertsReader / RoutedExpertsCapturer 改为无状态的 RoutedExpertsManager 和纯设备 Capturer，架构更清晰，但需确保外部 KV 块场景的处理方案不被遗漏。
  - 风险标记：外部 KV 块数据一致性，共享缓冲区并发覆写，int32 类型局限性，base64 负载膨胀，缺少集成测试覆盖

关联脉络

- PR #42434 Revert "[Core] Replace routing replay with device cache and async D2H pipeline" (#39917): 回退了旧版异步路由捕获方案（#39917），与本 PR 解决同一问题但采用不同架构。本 PR 是新方案，从 42434 回退后的状态出发重新设计。
- PR #38079 [RFC] Redesign enable_return_routed_experts to avoid blocking EngineCore event loop: 关联 issue，详细描述了原有共享内存方案的挂起问题和根本原因分析，是本 PR 的动机来源。