PR #42434 完整报告

vllm-project/vllm

Revert "[Core] Replace routing replay with device cache and async D2H pipeline" (#39917)

合并时间：2026-05-14 14:49

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/42434

执行摘要

- 一句话：回退 MoE 路由捕获机制到共享内存方案
- 推荐动作：建议密切关注被回退的 device cache 方案与后续 #39568 的演进关系。核心设计决策（共享内存 vs. device pipeline）值得深入阅读 routed_experts_capturer.py 中的注释和实现差异。对于直接使用 routed_experts API 的客户，需评估移除字段的影响。

功能与动机

根据 PR body，原作者（aoshen02）来自 nemo RL 有紧急发布需求，无法很快支持 #39917 方案中缺失的 prefix caching 等特性。经双方沟通，同意暂时还原 #39917，回退到之前的共享内存设计，并准备使用 #39568 作为后续更完善的方案。

实现拆解

1. 核心模块重写（vllm/model_executor/layers/fused_moe/routed_experts_capturer.py）：从自定义 CUDA op + _RoutedExpertsDeviceCache / _RoutedExpertsHostCache 完全切换到 SharedMemory 方案，引入 _file_lock、_create_or_attach_shared_memory 等辅助函数，RoutedExpertsCapturer 和 RoutedExpertsReader 作为全局单例管理设备缓冲区和跨进程共享内存。
2. 模型运行器解耦（vllm/v1/worker/gpu_model_runner.py）：移除异步 D2H 拷贝触发和路由数据提取调用（issue_routing_d2h_copy、extract_routed_experts_for_current_batch），改用 RoutedExpertsCapturer.get_instance().clear_buffer() 替换 finalize_pending_copy；导入简化，仅依赖 RoutedExpertsCapturer。
3. 调度器直连共享内存（vllm/v1/core/sched/scheduler.py）：Scheduler.__init__ 中创建 RoutedExpertsReader 单例并 attach_buffer，_get_routed_experts 直接通过共享内存的 numpy 视图获取路由数据，不再依赖 ModelRunnerOutput.routed_experts_dict。
4. API 响应精简（vllm/entrypoints/openai/chat_completion/serving.py 等）：移除 prompt_routed_experts 和 completion 级别的专家路由信息输出，output_processor.py 不再调用 split_routed_experts。
5. 配置与文档清理（vllm/config/vllm.py、docs/training/routed_experts_replay.md）：删除对 enable_return_routed_experts 的并行限制校验；整个文档文件随方案删除。

关键文件：

- vllm/model_executor/layers/fused_moe/routed_experts_capturer.py（模块 路由捕获；类别 source；类型 data-contract；符号 RoutedExpertsCapturer, RoutedExpertsReader, init_buffer, capture）：核心实现文件，从 device cache + custom op 重写为共享内存 + 文件锁，是整个 revert 的主要载体。
- vllm/v1/worker/gpu_model_runner.py（模块 模型运行器；类别 source；类型 data-contract；符号 _bind_routed_experts_capturer, _capture_fn, init_routed_experts_capturer, execute_model）：模型运行器集成方式变化，移除异步 D2H 相关调用，改用新的 capturer 单例。
- tests/model_executor/test_routed_experts_capture.py（模块 测试；类别 test；类型 test-coverage；符号 test_bind_routing_capture_to_model_sets_layer_view, _DummyMoEConfig, _capturer_with_buffer, _DummyQuantMethod）：测试重构，适配新的基于 BaseRouter 的 capture API。
- vllm/v1/core/sched/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 _get_routed_experts, Scheduler.init）：调度器新增直接访问共享内存的 RoutedExpertsReader，在初始化时 attach buffer，移除之前通过 ModelRunnerOutput 传递路由数据的间接方式。
- vllm/config/vllm.py（模块 配置校验；类别 source；类型 core-logic；符号 _validate_return_routed_experts）：删除对 enable_return_routed_experts 的并行限制校验，因为回退后该配置不再支持约束。
- vllm/v1/engine/output_processor.py（模块 输出处理；类别 source；类型 dependency-wiring）：简化 routed_experts 处理，不再拆分为 prompt/gen 两部分，直接传递。
- docs/training/routed_experts_replay.md（模块 文档；类别 docs；类型 deletion）：文档被完全删除，反映该方案不再推荐使用。

关键符号：RoutedExpertsCapturer.create, RoutedExpertsCapturer.init_buffer, RoutedExpertsCapturer.capture, RoutedExpertsCapturer.clear_buffer, RoutedExpertsReader.create, RoutedExpertsReader.attach_buffer, RoutedExpertsReader.get_routed_experts, _get_num_experts_per_tok, _file_lock, _create_or_attach_shared_memory, GPUModelRunner._bind_routed_experts_capturer, GPUModelRunner.init_routed_experts_capturer, Scheduler._get_routed_experts

关键源码片段

vllm/model_executor/layers/fused_moe/routed_experts_capturer.py

核心实现文件，从 device cache + custom op 重写为共享内存 + 文件锁，是整个 revert 的主要载体。

```
# vllm/model_executor/layers/fused_moe/routed_experts_capturer.py

import fcntl
import os
import tempfile
from contextlib import contextmanager
```

```
from multiprocessing import shared_memory
from typing import Generator

_TMP_DIR = tempfile.gettempdir()
_LOCK_FILE_PREFIX = os.path.join(_TMP_DIR, "vllm_routed_experts")
_BUFFER_PREFIX = "vllm_routed_experts_buffer"


@contextmanager
def _file_lock(lock_file: str, mode: str = "wb+") -> Generator[None, None, None]:
    """跨进程文件锁，确保共享内存创建和附加的原子性。"""
    with open(lock_file, mode) as fp:
        fcntl.flock(fp, fcntl.LOCK_EX)
        try:
            yield
        finally:
            fcntl.flock(fp, fcntl.LOCK_UN)


def _create_or_attach_shared_memory(
    name: str, size: int, lock_file: str
) -> shared_memory.SharedMemory:
    """创建或附加到已存在的共享内存块，通过文件锁处理竞态。"""
    # 确保锁文件存在
    with open(lock_file, "wb"):
        pass

    with _file_lock(lock_file):
        try:
            shm = shared_memory.SharedMemory(name=name, create=True, size=size)
        except FileExistsError:
            shm = shared_memory.SharedMemory(name=name, create=False, size=size)

        # 如果 size 不匹配，重建共享内存
        if shm.size != size:
            shm.close()
            shm.unlink()
            try:
                shm = shared_memory.SharedMemory(name=name, create=True, size=size)
            except FileExistsError:
                shm = shared_memory.SharedMemory(name=name, create=False, size=size)

    return shm
```

评论区精华

自动代码审查工具 gemini-code-assist[bot] 指出了两个潜在问题：

- 还原移除了 moe_layer_id 赋值，可能引发 AttributeError。作者回复验证 FusedMoE.layer_id 已通过 @property 从 layer_name 解析，无错误。
- 绑定阶段使用 module.layer_id 可能超出 buffer 边界。作者回复 buffer 维度为 num_hidden_layers（全局层数），与全局层索引一致，不会越界。 两个问题均被作者详细回复并标记为误报，未遗留未解决担忧。
- 移除 moe_layer_id 可能导致 AttributeError (correctness): aoshen02 回复验证 FusedMoE.layer_id 已通过 @property 从 layer_name 解析，不存在错误。
- 使用全局 layer_id 索引可能超出 buffer 边界 (correctness): aoshen02 回复 buffer 维度为 (num_hidden_layers, ...)，与全局层索引一致，不会越界。

风险与影响

- 风险：
  1. 性能回归：从自定义 CUDA op + 预分配 device buffer 切回共享内存 + 文件锁，可能增加跨进程同步开销，但典型批量场景影响有限。
  2. 兼容性风险：OpenAI API 响应中移除了 routed_experts 字段，依赖该字段的客户端会失效，属于破坏性变更。
  3. 配置安全性：VllmConfig 中对 enable_return_routed_experts 的并行校验被整段删除，若用户在共享内存方案下启用未支持的并行（如 PP > 1），可能因未验证路径抛出意外错误。
  4. 并发风险：共享内存使用文件锁，高并发场景下可能成为瓶颈。 - 影响：对用户：所有使用 --enable-return-routed-experts 的推理请求将不再在 API 响应中获得 routed_experts 信息。对系统：路由专家数据不再通过 device->host 异步 pipeline 传输，转而通过操作系统共享内存，可能轻微增加延迟但减小 GPU 内存占用。对团队：该 revert 为后续基于 #39568 的重构铺平了道路，但短期内需要维护两个版本的 capturer 实现。 - 风险标记：核心路径变更，API 响应字段移除，共享内存并发瓶颈，配置校验移除

关联脉络

- PR #39917 [Core] Replace routing replay with device cache and async D2H pipeline: 被回退的原始 PR，本次 revert 的核心对象。
- PR #41055 [MoE Refactor] EPLB refactoring for FusedMoE: MoE 重构导致测试需要适配新的 BaseRouter API，影响 revert 后的测试更新。