PR #43433 完整报告

vllm-project/vllm

Keep scheduler alive for delayed KV connector frees

合并时间：2026-05-23 14:23

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43433

执行摘要

- 一句话：修复 KV connector 延迟释放时调度器活性中断
- 推荐动作：值得精读，展示了调度器活性设计与异步连接器的交互边界。

功能与动机

某些 KV connector 会在请求完成后延迟释放块，直到异步传输清理完成。原逻辑中 has_finished_requests() 仅判断 finished_req_ids 非空，当 finished_req_ids 清空后，即使调度器仍持有已完成请求，也会报告无已完成工作，导致引擎停止无转发步骤的轮询，connector 完成事件无法被消费，KV 块可能残留。(PR body)

实现拆解

1. 扩展 has_finished_requests 方法（vllm/v1/core/sched/scheduler.py）：在原检查 finished_req_ids 的基础上，增加对 connector 非空时队列内请求数的判断。如果总请求数大于等待、跳过等待和运行中队列之和，说明存在等待延迟释放的已完成请求，返回 True。
2. 调整引擎让步条件（vllm/v1/engine/core.py）：将 _process_engine_step 中的让步条件从 has_unfinished_requests() 改为 has_requests()，使得即使没有未完成的请求（但有待清理的请求时），引擎仍然持续进行无转发步轮询，从而能够消费 connector 完成事件。
3. 新增单元测试（tests/v1/core/test_scheduler.py）：添加 test_delayed_kv_connector_free_keeps_scheduler_active，模拟 connector 延迟释放场景，验证 has_finished_requests 和 has_requests 的正确行为。
4. 调整 connector 测试断言（tests/v1/kv_connector/unit/test_multi_connector.py）：引入辅助函数 _ignore_event_collection 过滤掉因引擎主动轮询而新增的 take_events 调用，使断言更健壮。

关键文件：

- vllm/v1/core/sched/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 has_finished_requests）：核心变更：扩展 has_finished_requests 方法以检测等待延迟释放的已完成请求。
- vllm/v1/engine/core.py（模块 引擎；类别 source；类型 core-logic；符号 _process_engine_step）：引擎让步条件变更：从 has_unfinished_requests 改为 has_requests，确保持续轮询。

- tests/v1/core/test_scheduler.py（模块 测试；类别 test；类型 test-coverage；符号 test_delayed_kv_connector_free_keeps_scheduler_active）：新增测试 test_delayed_kv_connector_free_keeps_scheduler_active，覆盖延迟释放生命周期。
- tests/v1/kv_connector/unit/test_multi_connector.py（模块 测试；类别 test；类型 test-coverage；符号 _ignore_event_collection）：引入 _ignore_event_collection 辅助函数，使断言更健壮，避免因新增的 take_events 调用干扰。

关键符号：has_finished_requests, _process_engine_step, test_delayed_kv_connector_free_keeps_scheduler_active, _ignore_event_collection

关键源码片段

vllm/v1/core/sched/scheduler.py

核心变更：扩展 has_finished_requests 方法以检测等待延迟释放的已完成请求。

```
def has_finished_requests(self) -> bool:
    # 如果 finished_req_ids 非空，直接返回 True
    if self.finished_req_ids:
        return True
    # 如果没有 connector，只依赖 finished_req_ids
    if self.connector is None:
        return False
    # 有 connector 时，检查是否还有请求等待延迟释放
    # 这些请求已从调度队列移除，但仍保留在 self.requests 中
    num_in_queues = (
        len(self.waiting) + len(self.skipped_waiting) + len(self.running)
    )
    # 如果总请求数大于队列内请求数，说明有已完成请求等待清理
    return len(self.requests) > num_in_queues
```

vllm/v1/engine/core.py

引擎让步条件变更：从 has_unfinished_requests 改为 has_requests，确保持续轮询。

```
def _process_engine_step(self) -> bool:
    """Called only when there are unfinished local requests."""
    outputs, model_executed = self.step_fn()
    for output in outputs.items() if outputs else ():
        self.output_queue.put_nowait(output)
    self.post_step(model_executed)

    # 如果没有模型执行但有调度器工作（如 WAITING_FOR_REMOTE_KVS 或延迟的 KV connector 释放），
    # 短暂让步 GIL 以允许后台传输线程推进。
    if not model_executed and self.scheduler.has_requests():
        time.sleep(0.001)

    return model_executed
```

评论区精华

首次提问：njhill 询问是否实际遇到问题，作者 lucifer1004 解释在 P/D 分离中遇到 KV 块计数错误。 性能建议：gemini-code-assist 指出初始实现 O(N) 遍历所有请求，建议改为 O(1) 的队列长度比较；作者接受并修改为最终版本。 测试调整：作者调整了 test_multi_example_connector_consistency 以适配新增的 take_events 调用。

- 实际使用场景 (question): njhill 确认场景合理。
- 性能优化 (performance): 作者修改为队列长度比较，避免遍历。

风险与影响

- 风险：仅影响启用 KV connector 的场景，无 connector 时行为保持不变。最终实现采用 O(1) 判断，避免了性能风险。新增测试覆盖了延迟释放路径。
- 影响：修复了 P/D 分离中预填充实例闲置时 KV 缓存残留的真实问题；对未使用 connector 的用户无影响。
- 风险标记：调度器活性逻辑变更，低风险，有测试覆盖

关联脉络

- PR #42753 Unknown: 在 PR 评论中被 markmc 引用，可能相关