PR #49146 完整报告

vllm-project/vllm

[Bugfix][KV Offloading] Handle queued request aborts without allocated KV blocks

合并时间：2026-07-21 11:16

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49146

执行摘要

- 一句话：修复 KV offload 中止排队请求时的 assertion 崩溃
- 推荐动作：该 PR 是一个针对性强的 Bugfix，代码简洁，逻辑清晰。建议精读 storable_chunks 方法的改动，理解其在 offloading 调度中的核心作用。值得注意的设计决策是将分配块数检查融入现有方法，而非独立成新函数，避免了调用处的重复校验。

功能与动机

当客户端断开连接时，若请求仅处于排队状态（从未被调度），其 KV 块未分配。_build_store_jobs 中 storable_chunks 返回的 chunk 数超过了实际已分配的 block 数，导致 assert len(offload_keys) == len(offload_chunk_ids) 断言失败，进而杀死整个引擎，影响所有正在处理的请求。详见 Issue #49118。

实现拆解

1. 修改 storable_chunks 方法签名和内部逻辑：在 scheduler.py 中将 storable_chunks 方法新增 group_state 参数，并添加 num_allocated_chunks 计算，将返回值限制为 min(num_chunks, num_allocated_chunks)，确保不会返回超过实际已分配 block 数的 chunk 数。
2. 级联调用更新：将 advance_stored_idx 和 _build_store_jobs 中调用 storable_chunks 的地方同步传入 group_state 参数。
3. 新增单元测试：在 test_scheduler.py 中新增 test_abort_queued_request_does_not_build_store_job 测试用例，模拟创建请求、将其放入 waiting 队列后立即中止，并断言 _build_store_jobs 不会为该请求生成 store job，且请求的 _req_status 被正确清理。

关键文件：

- vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py（模块 KV 连接器；类别 source；类型 core-logic；符号 storable_chunks, advance_stored_idx, _build_store_jobs）：核心修改文件；修复 storable_chunks 边界条件，使其返回的 chunk 数量不超过实际已分配的 block 数，从而避免对未分配 KV 块的请求执行 offload storage 操作。
- tests/v1/kv_connector/unit/offloading_connector/test_scheduler.py（模块 测试；类别 test；类型 test-coverage；符号 test_abort_queued_request_does_not_build_store_job）：新增单元测试 test_abort_queued_request_does_not_build_store_job，验证中止从未被

调度的请求时，_build_store_jobs 不会生成 store job，且请求状态被正确清理。

关键符号：storable_chunks, advance_stored_idx, _build_store_jobs, test_abort_queued_request_does_not_build_store_job

关键源码片段

vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py

核心修改文件；修复 storable_chunks 边界条件，使其返回的 chunk 数量不超过实际已分配的 block 数，从而避免对未分配 KV 块的请求执行 offload storage 操作。

```
# vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py

class RequestOffloadState:

    def storable_chunks(
        self,
        group_config: "GroupOffloadConfig",
        group_state: RequestGroupState, # 新增 group_state 参数，用于获取实际分配的 block 数量
        num_offloadable_tokens: int,
    ) -> int:
        """返回可存储的 chunk 数量，受限于实际已分配的 block 数。

        原先的实现在请求从未被调度（无 allocated blocks）时仍返回基于
        num_offloadable_tokens 计算的 chunk 数，导致后续断言失败。
        现在通过 min(num_chunks, num_allocated_chunks) 确保返回值
        不会超过实际分配的 chunk 数。
        """
        num_chunks = num_offloadable_tokens // group_config.tokens_per_chunk
        is_decoding = num_offloadable_tokens > self.req.num_prompt_tokens
        if group_config.is_eagle_group and is_decoding:
            num_chunks = max(0, num_chunks - 1)
        # 使用 group_state.block_ids 计算实际已分配的 chunk 数量
        num_allocated_chunks = (
            len(group_state.block_ids) // self.config.blocks_per_chunk
        )
        return min(num_chunks, num_allocated_chunks) # 取较小值，避免过高估计

    def advance_stored_idx(self, num_offloadable_tokens: int) -> None:
        for group_config, group_state in zip(
            self.config.kv_group_configs, self.group_states
        ):
            group_state.next_stored_chunk_idx = max(
                group_state.next_stored_chunk_idx,
                self.storable_chunks(group_config, group_state, num_offloadable_tokens),
            )
```

tests/v1/kv_connector/unit/offloading_connector/test_scheduler.py

新增单元测试 test_abort_queued_request_does_not_build_store_job，验证中止从未被调度的请求时，_build_store_jobs 不会生成 store job，且请求状态被正确清理。

```
# tests/v1/kv_connector/unit/offloading_connector/test_scheduler.py

@pytest.mark.parametrize("async_scheduling", [True, False])
def test_abort_queued_request_does_not_build_store_job(
    request_runner, async_scheduling: bool
):
    """测试中止从未被调度的请求时，不会构建 store job，
    避免 assertion 错误导致引擎崩溃。"""
    block_size = 4
    runner = request_runner(
        block_size=block_size,
        num_gpu_blocks=8,
        async_scheduling=async_scheduling,
    )

    # 创建一个请求并将其调度出去，占用 GPU 块
    runner.new_request(token_ids=[0] * (block_size * 4))
    runner.scheduler.schedule()

    # 创建第二个请求，它应该进入 waiting 队列
    runner.new_request(token_ids=[1] * (block_size * 4))
    queued_req_id = str(runner.req_id)
    # 确认请求确实在 waiting 队列中
    assert any(
        request.request_id == queued_req_id for request in runner.scheduler.waiting
    )

    # 立即中止该请求 —— 此时它没有分配任何 KV 块
    runner.scheduler.finish_requests(queued_req_id, RequestStatus.FINISHED_ABORTED)
    req_status = runner.connector_scheduler._req_status[queued_req_id]
    # 验证 offload_keys 存在，但 block_ids 为空（因为没有分配块）
    assert all(group_state.offload_keys for group_state in req_status.group_states)
    assert all(not group_state.block_ids for group_state in req_status.group_states)

    # 执行调度 —— 这将触发 _build_store_jobs
    scheduler_output = runner.scheduler.schedule()

    metadata = scheduler_output.kv_connector_metadata
    assert isinstance(metadata, OffloadingConnectorMetadata)
    # 确保 store_jobs 中不包含已中止的请求
    assert all(job.req_id != queued_req_id for job in metadata.store_jobs.values())
    # 确保请求状态已被清理
    assert queued_req_id not in runner.connector_scheduler._req_status
```

评论区精华

Reviewer orozery 建议将独立的 storable_allocated_chunks 函数合并回 storable_chunks 中，以避免代码冗余。作者采纳建议，在 storable_chunks 内部直接添加分配块数限制。

- 合并 storable_allocated_chunks 为 storable_chunks (design): 作者采纳建议，直接在 storable_chunks 内部添加分配块数检查，移除单独的函数。

风险与影响

- 风险:
  1. 回归风险（低）：storable_chunks 返回值受限于已分配块数，可能影响正常结束请求的 offload 行为。但测试覆盖了多种场景，且逻辑上 num_allocated_chunks 在正常流程中通常大于或等于 num_chunks，因此影响很小。
  2. 未覆盖的边界情况：当前测试仅覆盖了 abort 场景，未测试 prefetch 或 preemption 等可能导致无分配块的场景。 - 影响：影响路径：修复了 OffloadingConnector 的关键崩溃 Bug，提升系统稳定性。影响用户：所有使用 OffloadingConnector 且启用了 prefix caching、chunked prefill 的用户，特别是大规模部署中客户端不稳定时。影响范围：仅涉及 offloading 调度模块，不影响无 KV offload 的场景。 - 风险标记：修复源于前一个 PR 的副作用，影响 KV offloading 稳定性的关键修复

关联脉络

- PR #48596 [Bugfix][KV Offloading] Fix assertion error on non block aligned requests: 该 PR 在 finished_req_ids 中引入了对 req.num_tokens 的全量计算，导致了本 PR 修复的问题。
- PR #49118 [Bug]: OffloadingConnector — aborting a queued (never-scheduled) request kills the engine with AssertionError in _build_store_jobs: 本 PR 修复的 Issue，提供了详细的复现步骤和根因分析。