

PR #50388 完整报告

vllm-project/vllm

[Core] Fix ValueError on KV load failure with a hybrid KV cache

合并时间: 2026-09-13 21:03

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/50388>

执行摘要

- 一句话: 修复混合 KV 缓存加载失败崩溃, 改为整请求重算
- 推荐动作: 值得精读。推荐关注三点: 一是 `_update_requests_with_invalid_blocks` 中「单组按 block 索引截断 / 多组整请求重算」双路径分支的判定与 `continue` 时机, 这是理解 V1 调度器失效恢复语义的最佳切入; 二是 `null block` 在 Mamba align 模式下的共享语义 (为什么它可以「既不能失败也不能被驱逐」); 三是 NickLucche 提出的 `connector` 层请求级失败方案——如果后续 #42461 推进, 本 PR 的多组分支很可能被再次改写, 建议在读代码时同步留意这条演进线。若团队在跑混合模型 + 远端 KV connector, 建议补充一个真实模型 (如 Kimi-Linear-48B-A3B) 的端到端失败注入验证。

功能与动机

PR body 指出: `_update_requests_with_invalid_blocks` 里残留的 `# TODO (davidb): add support for hybrid memory allocator` 与 `(req_block_ids,) = self.kv_cache_manager.get_block_ids(req_id)` 组合, 使混合模型上「第一个被上报的无效 block」就会触发 `ValueError: too many values to unpack (expected 1)`; 异常没有任何捕获点, `EngineCore` 调用 `_send_engine_dead()` 退出, 于是「one recoverable read failure takes down every request on the instance, not just the one that failed」。此外 hybrid 模型「groups sit on different block grids, so the same block index means a different token position in each and there is no single longest valid prefix to truncate at」, 因此选择整请求重算这一粒度。issue 评论中 wzhao18 补充该修复对 Kimi-K3 + Mooncake 场景是必需项, xijiaat 则用 DeepSeek-V4-Flash-0731 的五组布局复现了 #50687。

实现拆解

1. 修复解包入口 (`vllm/v1/core/sched/scheduler.py` → `Scheduler._update_requests_with_invalid_blocks`): 不再写 `(req_block_ids,) = self.kv_cache_manager.get_block_ids(req_id)`, 改为保留 `per-group` 列表 `req_block_ids_per_group`, 并在 docstring 中说明「混合模型一个 token span 跨越 block size 不一致的多个 KV cache group, 没有单一最长有效前缀」。这是整个 PR 的根因修复点。
2. 新增多组 (hybrid) 分支: 当 `len(req_block_ids_per_group) > 1` 时, 跨所有 group 收集命中 `invalid_block_ids` 的 block (排除 `null block`), 把结果并入 `marked_invalid_block_ids` (保证共享同一 block 的其它请求也会被重新调度), 累加

`total_affected_tokens += req_num_computed_tokens`, 将

`request.num_computed_tokens` 直接置 0, 并在 `evict_blocks` 为真时把该请求所有非 null block 收进 `blocks_to_evict`, 最后加入 `affected_req_ids` 并 `continue`。这样避免了为不同 block 网格推导统一截断点这一本质上不可解的问题。

3. null block 守卫: 循环外新增 `null_block_id = self.kv_cache_manager.block_pool.null_block.block_id`。PR body 解释其必要性: Mamba align 模式下, 组内 block 列表大部分是共享 null block (「a 9-chunk request looks like [0, 0, 0, 0, 0, 0, 0, 0, 10], only the aligned snapshot is backed」), 若不排除, 被上报的 null block 会重启无关请求, `blocks_to_evict` 还会收集到不属于任何请求的 block。
4. 单组路径零行为变更: 多组分支之后恢复 `(req_block_ids,) = req_block_ids_per_group`, 原有的「按 `idx * self.block_size` 截断到首个无效 block」逻辑、共享 block 的 `marked_invalid_block_ids` 语义与「sync 加载才支持 block 共享」的注释都原样保留, 保证纯 attention 模型不受影响。
5. 测试与工具配套: `tests/v1/kv_connector/unit/utils.py::create_scheduler` 新增 `hash_block_size` 形参并透传给 `scheduler_cls(...)`, 让测试能构造非默认 hash 块大小的调度器; `tests/v1/kv_connector/unit/test_kv_load_failure_recovery.py` 新增 `hybrid_scheduler` fixture (基于 `make_kv_cache_config(mamba_enabled=True)`)、辅助函数 `_schedule_hybrid_async_load / _report_invalid_blocks`, 以及 `test_hybrid_load_failure_recomputes_whole_request` (`parametrize("failed_group_idx", [0, 1])`)、`test_hybrid_load_failure_ignores_null_block`; 进一步用 `_make_deepseek_v4_five_group_config` (1 个 `MLAAttentionSpec` + 4 个不同 block size 的 `SlidingWindowMLASpec`, 模拟 DeepSeek-V4-Flash 的 5 个物理组) 与 `test_deepseek_v4_load_failure_recomputes_whole_request` (`parametrize("failed_group_idx", range(5))`) 覆盖 issue #50687 的复现路径。测试与源码同步提交, 但全部为 mock connector 的调度器单元测试, 没有真实模型或端到端回归。

关键文件:

- `vllm/v1/core/sched/scheduler.py` (模块 调度器; 类别 source; 类型 core-logic; 符号 `Scheduler._update_requests_with_invalid_blocks`): 根因修复所在文件: `_update_requests_with_invalid_blocks` 从单组解包改为 per-group 列表, 并新增多组整请求重算分支与 null block 守卫, 是决定混合模型失效恢复语义的核心改动。
- `tests/v1/kv_connector/unit/test_kv_load_failure_recovery.py` (模块 失败恢复测试; 类别 test; 类型 test-coverage; 符号 `hybrid_scheduler, _schedule_hybrid_async_load, _report_invalid_blocks, test_hybrid_load_failure_recomputes_whole_request`): 新增 340 行测试: hybrid 双组 fixture 与整请求重算 / 忽略 null block 两个用例, 并用 DeepSeek-V4-Flash 五组布局配置逐组参数化, 直接对应 issue #50687 的复现路径。
- `tests/v1/kv_connector/unit/utils.py` (模块 测试工具; 类别 test; 类型 test-utility; 符号 `create_scheduler`): 测试基建适配: `create_scheduler` 新增 `hash_block_size` 形参并透传, 使 DeepSeek-V4 五组用例能构造与生产一致的 hash 块大小。

关键符号: `Scheduler._update_requests_with_invalid_blocks`, `create_scheduler`, `_schedule_hybrid_async_load`, `_report_invalid_blocks`, `test_hybrid_load_failure_recomputes_whole_request`,

```
test_hybrid_load_failure_ignores_null_block,  
_make_deepseek_v4_five_group_config,_create_deepseek_v4_scheduler,  
_schedule_deepseek_v4_async_load,test_deepseek_v4_load_failure_recomputes_whole_  
request
```

关键源码片段

vllm/v1/core/sched/scheduler.py

根因修复所在文件： `_update_requests_with_invalid_blocks` 从单组解包改为 per-group 列表，并新增多组整请求重算分支与 null block 守卫，是决定混合模型失效恢复语义的核心改动。

```
# vllm/v1/core/sched/scheduler.py  
# Scheduler._update_requests_with_invalid_blocks (节选：null block 守卫 + 单组 / 多组双路径)
```

```
# 所有组都不承载 KV 的位置——被跳过的 sliding-window block，以及 Mamba  
# 除对齐快照以外的全部 block——都指向同一个 null block，被所有请求共享。  
# 它不携带任何请求的数据，既不能替某个请求“失败”，也不能替它被驱逐。  
null_block_id = self.kv_cache_manager.block_pool.null_block.block_id
```

```
for request in requests:  
    is_affected = False  
    marked_invalid_block = False  
    req_id = request.request_id  
  
    # 每个 KV cache group 一个 block-id 列表；单组模型长度为 1，hybrid 模型 > 1  
    req_block_ids_per_group = self.kv_cache_manager.get_block_ids(req_id)  
    # 只统计可能含外部计算 token 的 block  
    req_num_computed_tokens = (  
        request.num_computed_tokens - num_scheduled_tokens.get(req_id, 0)  
    )  
  
    if len(req_block_ids_per_group) > 1:  
        # hybrid: 各组 block 网格不同，同一 block index 对应不同 token 位置，  
        # 不存在统一的最长有效前缀，因此整请求重算（与 #42461 的目标粒度一致）  
        request_invalid_block_ids = {  
            block_id  
            for req_block_ids in req_block_ids_per_group  
            for block_id in req_block_ids  
            # null block 必须排除，否则会重启无关请求  
            if block_id != null_block_id and block_id in invalid_block_ids  
        }  
        if request_invalid_block_ids:  
            # 仍然登记这些 block，让共享同一 block 的其它请求也被重新调度  
            marked_invalid_block_ids |= request_invalid_block_ids  
            total_affected_tokens += req_num_computed_tokens  
            request.num_computed_tokens = 0  
            if evict_blocks:  
                for req_block_ids in req_block_ids_per_group:
```

```

        blocks_to_evict.update(
            block_id
            for block_id in req_block_ids
            if block_id != null_block_id
        )
    affected_req_ids.add(req_id)
    continue

```

单组（纯 attention）模型：保持原有的 block 级截断语义不变
 (req_block_ids,) = req_block_ids_per_group

tests/v1/kv_connector/unit/test_kv_load_failure_recovery.py

新增 340 行测试：hybrid 双组 fixture 与整请求重算 / 忽略 null block 两个用例，并用 DeepSeek-V4-Flash 五组布局配置逐组参数化，直接对应 issue #50687 的复现路径。

tests/v1/kv_connector/unit/test_kv_load_failure_recovery.py

```

def _schedule_hybrid_async_load(
    scheduler: Scheduler, num_prompt_blocks: int, num_external_computed_blocks: int
) -> Request:
    """把一个请求置入 hybrid 模型的异步外部加载状态。"""
    request = create_request(num_tokens=num_prompt_blocks * scheduler.block_size)
    scheduler.add_request(request=request)

    # mock connector: 报告命中的外部 token 数，并以异步方式加载
    scheduler.connector = Mock()
    scheduler.connector.get_num_new_matched_tokens.side_effect = (
        _make_get_num_new_matched_tokens(
            {request.request_id: num_external_computed_blocks * scheduler.block_size},
            async_load=True,
        )
    )
    scheduler.connector.take_events.return_value = ()
    return request

```

```

def _report_invalid_blocks(
    scheduler: Scheduler, scheduler_output, invalid_block_ids: set[int]
) -> None:
    """模拟 model runner 在输出里上报无效 block。"""
    scheduler.update_from_output(
        scheduler_output,
        create_model_runner_output(
            reqs=[],
            finished_recving=set(),
            invalid_block_ids=invalid_block_ids,
            use_eos=True,
        ),
    )

```

```

@pytest.mark.parametrize("failed_group_idx", [0, 1])
def test_hybrid_load_failure_recomputes_whole_request(
    hybrid_scheduler: Scheduler, failed_group_idx: int
):
    """hybrid 模型必须能恢复而不是崩溃。

    它的各组位于不同的 block 网格上，不存在单一截断点，因此请求从头重算。
    """
    num_prompt_blocks, num_external_computed_blocks = 10, 9
    request = _schedule_hybrid_async_load(
        hybrid_scheduler, num_prompt_blocks, num_external_computed_blocks
    )
    scheduler_output = hybrid_scheduler.schedule()

    block_ids_per_group = hybrid_scheduler.kv_cache_manager.get_block_ids(
        request.request_id
    )
    assert len(block_ids_per_group) == 2, "expected a hybrid (multi-group) layout"
    null_block_id = hybrid_scheduler.kv_cache_manager.block_pool.null_block.block_id
    # Mamba 组只用真实 block 支撑对齐快照，因此挑一个确实属于该请求的 block
    failed_block = next(
        block_id
        for block_id in block_ids_per_group[failed_group_idx]
        if block_id != null_block_id
    )

    _report_invalid_blocks(hybrid_scheduler, scheduler_output, {failed_block})

    # 整请求重算：计算进度归零，并重新排队等待远端 KV
    assert request.num_computed_tokens == 0
    assert hybrid_scheduler.failed_recving_kv_req_ids == {request.request_id}
    assert request.status == RequestStatus.WAITING_FOR_REMOTE_KVS

# 同文件另有 _make_deepseek_v4_five_group_config(): 用 1 个 MLAAttentionSpec
# (block_size=256) 加上 4 个 block_size 分别为 4 / 8 / 64 / 64 的
# SlidingWindowMLASpec，复现 DeepSeek-V4-Flash 的 5 个物理 KV 组，并以
# parametrize("failed_group_idx", range(5)) 逐组验证失败恢复（#50687 的复现）。

```

评论区精华

1. 最有分量的设计异议来自 NickLucche：「To provide more context, I already implemented something very similar here <https://github.com/vllm-project/vllm/pull/32204>, but we were not satisfied with the request-level failure back then and ended up reverting that commit.」他建议更彻底的方向——MC connector 不要上报 block failure，而是直接上报请求级失败（`self._invalid_block_ids.put(set(meta.local_block_ids[0])) + self._failed_recv_reqs.put(req_id)`），由 #42461 统一处理请求级失败，并 cc 了 @ZhanqiuHu。kebe7jun 的回应是「Modifying the connector api might take some

time. Could we merge this PR first to fix the error?」——即把本 PR 定位为止血方案。

2. null block 的语义分歧点在测试里被显式固化: `test_hybrid_load_failure_ignores_null_block` 断言 `mamba_block_ids[0] == null_block_id`, 说明 Mamba 组把未对齐位置全部停在 null block 上, 被上报时必须不触发任何重启。
3. DeepSeek-V4 五组布局的复现贡献: xijiaat 表示「I reproduced #50687 with the DeepSeek-V4-Flash-0731 five-group scheduler layout and validated this recovery path in an isolated B300 test pod」, 并在 fork 上提交了 tests-only 补充 (覆盖五组失败、共享 null block、`failure_policy=fail` 驱逐, 合计 24 passed)。kebe7jun 采纳了其中一部分 (五组配置与逐组参数化用例进入本 PR)。
4. 审批情况: claude[bot] 因 PR 来自 fork 未做自动审查; Dao007forever 与 ivanium 均直接 APPROVED, 最终由 DarkLight1337 合并。CI 侧 b200-miscellaneous-kernels 曾失败, kebe7jun 判断「maybefixedby<https://github.com/vllm-project/vllm/pull/56594>」。
- hybrid 恢复粒度: 调度器整请求重算 vs connector 上报请求级失败 (design): 本 PR 作为止血方案合并, 采用调度器内整请求重算; connector 层请求级失败的 API 改造推迟到后续 PR, 双方案存在短期重叠。
- null block 误报会重启无关请求 (correctness): 引入 `null_block_id = self.kv_cache_manager.block_pool.null_block.block_id` 守卫, 并新增 `test_hybrid_load_failure_ignores_null_block` 固化该语义 (断言 `mamba_block_ids[0] == null_block_id` 时 `num_computed_tokens` 不变、`failed_recving_kv_req_ids` 为空)。
- DeepSeek-V4-Flash 五组布局的复现与测试补充 (testing): 五组配置 `_make_deepseek_v4_five_group_config` 与逐组参数化用例被纳入本 PR; `failure_policy=fail` 的驱逐断言未进入主测试文件。
- Kimi-K3 + Mooncake 场景的合并诉求 (question): 该业务诉求推动了 PR 的合并节奏, Dao007forever 与 ivanium 先后 APPROVED, DarkLight1337 完成合并。

风险与影响

- 风险:
 1. 恢复粒度退化带来的重算成本: hybrid 分支把 `request.num_computed_tokens` 置 0, 等于放弃已经成功加载的外部 KV。对长 prompt (如 100K 级) 的混合模型请求, 一次远端读失败会触发整段 prefill 重算, 开销显著高于单组分支的「截断到最长有效前缀」, 在共享前缀场景下会放大 TPOT/TTFT 抖动。
 1. 驱逐范围扩大: 多组分支会把该请求所有非 null block 加入 `blocks_to_evict`, 而单组分支只在既有条件下按需收集。若这些 block 被 prefix caching 或其它请求共享, 失效范围会随之扩大 (虽然语义上必要, 但缺少对共享命中的细粒度校验)。
 2. null block 取值路径的隐含依赖: `self.kv_cache_manager.block_pool.null_block.block_id` 是新增硬依赖, 未看到对 `null_block` 缺失 / 为 None 时的兜底; 在非 hybrid 但 block pool 配置特殊的路径上是否安全, 材料中无法确认。
 3. 测试覆盖的边界: 340 行测试全部基于 mock connector 的调度器单测, DeepSeek-V4 五组用例只是「scheduler-side projection」, 与真实 model runner / backend layout 可能不同步; 没有真实 Mamba 或 DeepSeek-V4 端到端回归, `failure_policy=fail` 策略下的驱逐路径也未进入主测试。

4. CONNECTOR 层 API 分歧未闭合: NickLucche 指出的「block 级 vs 请求级失败上报」方向未在本 PR 解决, 后续 #42461 落地时本 PR 的整请求重算逻辑很可能被再次替换, 存在短期重复改动成本。- 影响: 用户侧: 混合模型 (Kimi-Linear / Kimi-K3 + Mooncake、DeepSeek-V4-Flash 等多组布局) 上, 一次可恢复的外部 KV 读失败不再导致 EngineCore 退出与实例级全量请求失败, 可用性从「单点失败打挂整个实例」改善为「单个请求整段重算」; 纯 attention 单组模型行为与性能完全不变。系统侧: `vllm/v1/core/sched/scheduler.py` 是 V1 调度主路径, 该文件任何改动都属于高风险区域, 但本次改动被严格限制在 `len(...) > 1` 分支内, 回归面可控。团队侧: 确立了「hybrid 模型以请求为恢复粒度」的临时契约, 与正在进行的 per-block 错误处理移除 (#42461) 方向一致, 后续替换成本较低。- 风险标记: 核心调度路径变更, hybrid 模型整请求重算成本放大, 驱逐范围扩大到请求全部 block, 仅 mock 单元测试、缺少端到端回归, connector 层失败上报方案待统一

关联脉络

- PR #42461 (历史 PR, 标题未在本次上下文中提供) 移除 per-block 错误处理、转向请求级失败: PR body 明确指出 per-block error handling 正在该 PR 中被移除, 「affected hybrid requests recompute in full, which is the granularity that replacement moves to anyway」; NickLucche 也建议由其统一处理请求级失败。
- PR #32204 (历史 PR, 标题未在本次上下文中提供) 早期的同类 hybrid/ 请求级失败恢复尝试: NickLucche 在 review 中直接引用: 「I already implemented something very similar here <https://github.com/vllm-project/vllm/pull/32204>, but we were not satisfied with the request-level failure back then and ended up reverting that commit.» 是本 PR 方案取舍的直接前史。
- PR #56594 [Bugfix][CI] Update CuTeDSL indexer Q sentinel test for migrated wrappers: PR 评论中 kebe7jun 提到 b200-miscellaneous-kernels CI 失败 「maybe fixed by <https://github.com/vllm-project/vllm/pull/56594>」, 属于同批 CI 修复链路。
- PR #56621 [Bugfix][KV Offload] Submit CPU stores on no-forward steps: 同为 V1 KV connector / offload 路径上的失败恢复类修复, 涉及调度步与 KV 状态判定的边界条件, 属于同一故障恢复脉络。
- PR #56629 [5/N] Share HiSparse host cache across TP ranks (reopens #52760): 同为 kv-connector + v1 的 KV 缓存共享与 block 归属语义改动, 与本 PR 的 block 标记 / 驱逐语义在同一片代码区域演进。