PR #49427 完整报告

vllm-project/vllm

[Bugfix] Restore `gather_and_maybe_dequant_cache` OOB guard

合并时间：2026-07-23 04:47

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/49427

执行摘要

- 一句话：恢复 cache kernel 的越界保护
- 推荐动作：建议合并。这是一个明确的安全修复，且配合了测试验证。值得关注的设计决策是：在 kernel 中通过 continue 跳过越界 token 而非提前截断或报错，保持了简洁性。

功能与动机

PR body 指出，gather_and_maybe_dequant_cache 在 #28029 的重写是基于 #28760 之前的代码，并于后者合并五天后合并，从而静默恢复了 #28760 添加的边界检查。当 seq_starts 将 block index 推到 batch block table 行的末尾之后时，kernel 会越界读取 block_table（Issue #27909）。

实现拆解

1. CUDA kernel 保护恢复：在 csrc/libtorch_stable/cache_kernels.cu 的 gather_and_maybe_dequant_cache 核函数中，添加 if (block_table_id >= block_table_stride) continue; 检查，在 block index 越界时跳过该 token 的处理。
2. 测试适配：修改 tests/kernels/test_cache_kernels.py 中的 test_gather_cache_oob 测试用例，去除已废弃的 batch_size 参数，新增 token_to_seq 和 seq_len 参数以匹配重写后的 kernel API，并将 entry_size 从 128 改为 576（仅 MLA 支持的 entry size）。

关键文件：

- csrc/libtorch_stable/cache_kernels.cu（模块 CUDA 内核；类别 other；类型 core-logic；符号 gather_and_maybe_dequant_cache）：核心修改：在 gather_and_maybe_dequant_cache kernel 中添加越界保护
- tests/kernels/test_cache_kernels.py（模块 缓存内核；类别 test；类型 test-coverage；符号 test_gather_cache_oob）：适配测试用例以匹配重写后的 kernel API，并维持 OOB 场景覆盖

关键符号：gather_and_maybe_dequant_cache

关键源码片段

csrc/libtorch_stable/cache_kernels.cu

核心修改：在 gather_and_maybe_dequant_cache kernel 中添加越界保护

```
// ... existing code ...
batch_offset += offset;
int32_t block_table_id = batch_offset / block_size;
int32_t slot_id = batch_offset % block_size;
// 当 seq_starts 将 block index 推到 batch 的 block table 行末尾之后时，
// block_table_id 可能等于或超过 block_table_stride，导致后续读取越界。
// 这里跳过该 token（continue），避免越界访问。
if (block_table_id >= block_table_stride) continue;
int32_t block_table_offset = batch_id * block_table_stride + block_table_id;
int32_t block_id = block_table[block_table_offset];
int64_t cache_offset = ...
// ... existing code ...
```

tests/kernels/test_cache_kernels.py

适配测试用例以匹配重写后的 kernel API，并维持 OOB 场景覆盖

```
@pytest.mark.skipif(torch.accelerator.device_count() < 1, reason="Need CUDA device")
def test_gather_cache_oob():
    """
    Tests for OOB read in gather_and_maybe_dequant_cache (Issue #27909).
    This test constructs a boundary case identified in the issue where
    seq_starts causes the block_table offset to read out of bounds.
    """

    block_size = 64
    # 重写后的 kernel 只支持 MLA entry sizes.
    entry_size = 576

    block_table = torch.tensor([[1, 2]], dtype=torch.int32, device="cuda")

    # 这将导致 offset = 128 / block_size = 128 / 64 = 2
    # 使得 kernel 尝试读取 block_table[0, 2]，但其 size 仅为 2。
    seq_starts = torch.tensor([128], dtype=torch.int32, device="cuda")

    seq_len = 65
    cu_seq_lens = torch.tensor([0, seq_len], dtype=torch.int32, device="cuda")
    token_to_seq = torch.zeros(seq_len, dtype=torch.int32, device="cuda")

    # src_cache: [num_blocks, block_size, entry_size]
    num_blocks = 5
    src_cache = torch.randn(
        (num_blocks, block_size, entry_size), dtype=torch.float16, device="cuda"
    )

    dst = torch.empty((seq_len, entry_size), dtype=torch.float16, device="cuda")
    scale = torch.tensor([1.0], dtype=torch.float32, device="cuda")

    # 调用 C++ 函数 gather_and_maybe_dequant_cache
    ops.gather_and_maybe_dequant_cache(
```

```
        src_cache,
        dst,
        block_table,
        cu_seq_lens,
        token_to_seq,
        seq_len,
        "auto", # kv_cache_dtype
        scale,
        seq_starts,
    )

    torch.accelerator.synchronize()
    assert True
```

评论区精华

在 review 讨论中，开发者 cleonard530 澄清越界检查的丢失并非其迁移所致，而是 #28029 覆写的结果。njhill 更正了描述并感谢指出。

- 越界检查丢失的归因 (other): njhill 更正描述，确认错误归因。

风险与影响

- 风险：该变更是对越界读取的修复，仅添加条件判断和跳过逻辑，不改变正常路径行为。风险极低。测试已同步适配。
- 影响：影响范围小，仅涉及 gather_and_maybe_dequant_cache kernel。修复了潜在的内存越界读取，可能导致不正确的推理结果或崩溃（Issue #27909）。用户无需额外操作。
- 风险标记：核心路径变更

关联脉络

- PR #27909 Issue: gather_and_maybe_dequant_cache OOB read: 该 PR 修复的问题
- PR #28760 PR: Add bound check to gather_and_maybe_dequant_cache: 原始添加边界检查的 PR，被后续改写覆盖
- PR #28029 PR: Rewrite gather_and_maybe_dequant_cache: 基于旧代码的改写，无意中移除了边界检查
- PR #43717 PR: stable-ABI rewrite of gather_and_maybe_dequant_cache: 另一项改写，本 PR 修复的边界检查是否由该 rewrite 引入曾引起讨论