PR #48158 完整报告

vllm-project/vllm

[Refactor] Remove unused rocm kernel `combine_topk_swa_indices_ragged`

合并时间：2026-07-10 21:33

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/48158

执行摘要

- 一句话：删除未使用的 ROCm 稀疏注意力内核
- 推荐动作：值得作为死代码清理的参考示例，但无需深入审查代码细节。可以向团队推广类似的定期清理实践。

功能与动机

PR 说明：Remove unused rocm kernel combine_topk_swa_indices_ragged and related test。该内核及其辅助函数在现有代码路径中已无调用，为减少维护负担进行清理。

实现拆解

1. 在 vllm/models/deepseek_v4/amd/rocm.py 中删除两个 Triton JIT 内核函数定义（_compute_combined_lens_kernel、_combine_topk_swa_indices_ragged_kernel）和 Python 封装函数（combine_topk_swa_indices_ragged），共约 147 行。
2. 在 tests/kernels/attention/test_rocm_triton_attn_dsv4.py 中删除对应的参考实现 _ref_combine_topk_swa_ragged 和测试函数 test_combine_topk_swa_indices_ragged，约 89 行。
3. 经 grep 确认无其他模块引用这些符号，提交并合并。

关键文件：

- vllm/models/deepseek_v4/amd/rocm.py（模块 模型层；类别 source；类型 dead-code-removal；符号 _compute_combined_lens_kernel, _combine_topk_swa_indices_ragged_kernel, combine_topk_swa_indices_ragged）：核心源文件，删除了未使用的 combine_topk_swa_indices_ragged 函数及其 Triton 内核定义，减少 147 行 dead code。
- tests/kernels/attention/test_rocm_triton_attn_dsv4.py（模块 测试；类别 test；类型 test-removal；符号 _ref_combine_topk_swa_ragged, test_combine_topk_swa_indices_ragged）：测试文件，删除了与上述内核对应的参考实现和测试用例，避免测试引用已删除的代码。

关键符号：_compute_combined_lens_kernel, _combine_topk_swa_indices_ragged_kernel, combine_topk_swa_indices_ragged, _ref_combine_topk_swa_ragged, test_combine_topk_swa_indices_ragged

关键源码片段

vllm/models/deepseek_v4/amd/rocm.py

核心源文件，删除了未使用的 combine_topk_swa_indices_ragged 函数及其 Triton 内核定义，减少 147 行 dead code。

```
# 本 PR 移除了未使用的 `combine_topk_swa_indices_ragged` 函数及其 Triton 内核。
# 以下函数 `_copy_ragged_to_graph_buffers` 是之前保留的辅助函数，用于
# 将动态 ragged 元数据拷贝到 CUDA graph 持久化缓冲区中。

def _copy_ragged_to_graph_buffers(
    ragged_indices: torch.Tensor,
    ragged_indptr: torch.Tensor,
    ragged_indices_buffer: torch.Tensor,
    ragged_indptr_buffer: torch.Tensor,
    num_rows: int,
    max_entries_per_row: int,
) -> tuple[torch.Tensor, torch.Tensor]:
    """Copy dynamic ragged metadata into persistent CUDA graph buffers.

    FULL decode graphs capture kernel argument addresses. Keep the returned
    tensors backed by stable storage, while indptr continues to bound reads.
    """
    indptr_out = ragged_indptr_buffer[: num_rows + 1]
    indptr_out.copy_(ragged_indptr, non_blocking=True)

    max_entries = max(num_rows * max_entries_per_row, 1)
    ragged_out = ragged_indices_buffer[:max_entries]
    nnz = ragged_indices.numel()
    if nnz > 0:
        ragged_out[:nnz].copy_(ragged_indices, non_blocking=True)
    return ragged_out, indptr_out
```

tests/kernels/attention/test_rocm_triton_attn_dsv4.py

测试文件，删除了与上述内核对应的参考实现和测试用例，避免测试引用已删除的代码。

```
# 本 PR 移除了 `_ref_combine_topk_swa_ragged` 和 `test_combine_topk_swa_indices_ragged`。
# 以下函数 `test_decode_num_splits_heuristic` 保持不变，测试 split-K 解码路径的分片数启发式。

@requires_gfx950
@torch.inference_mode()
def test_decode_num_splits_heuristic(monkeypatch) -> None:
    """Split-count heuristic added with the flash-decode split-K decode path."""
    from vllm.v1.attention.ops import rocm_aiter_mla_sparse as mod

    # Pin the CU count so the heuristic is deterministic off-device.
    monkeypatch.setattr(mod, "_decode_cu_count", lambda: 256)

    # A batch that already fills the device should not be split.
```

```
    assert mod._decode_num_splits(256, 1, avg_main_len=128.0, avg_extra_len=0.0) == 1
    # A tiny batch on a large device should split to add parallelism.
    assert mod._decode_num_splits(2, 1, avg_main_len=256.0, avg_extra_len=0.0) > 1

    # The chosen count always stays within the searched [1, 16] range, and a
    # zero-length workload never splits (no work to parallelize).
    for num_queries in (1, 4, 24, 224, 1024):
        splits = mod._decode_num_splits(
            num_queries, 1, avg_main_len=512.0, avg_extra_len=128.0
        )
        assert 1 <= splits <= 16
    assert mod._decode_num_splits(2, 1, avg_main_len=0.0, avg_extra_len=0.0) >= 1
```

评论区精华

无实质性讨论。Claude bot 因来自 fork 而自动跳过审查，仅有 maintainer AndreasKaratzas 批准。该 PR 属于纯粹的清理操作，未产生设计争议。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。被删除的代码经 grep 确认在当前代码库中无任何调用点。唯一潜在风险是未来若需要此功能需重新实现，但该功能已不被需要的可能性极高。此外，vllm/models/deepseek_v4/amd/rocm.py 中新增的 _copy_ragged_to_graph_buffers 函数与删除的代码无关，保持稳定。
- 影响：无用户可见影响。团队减少 236 行需维护的代码，降低了认知负担。CI 中不再运行已删除的测试。
- 风险标记：低风险：删除未使用代码

关联脉络

- 暂无明显关联 PR