PR #50517 完整报告

vllm-project/vllm

[ROCm][CI] Update Transformers AR+RMS fusion expectation

合并时间：2026-07-31 18:16

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50517

执行摘要

- 一句话：更新 ROCm Transformers AR+RMS 融合测试期望
- 推荐动作：值得快速浏览，属于测试期望与编译器 pass 行为联动的典型小样本。对 torch.compile / fusion pass 与端到端测试期望设计感兴趣的工程师可以精读：关注 #48757 的 canonicalization 如何改变图结构、_replace 覆盖为何不可省略，以及将 CUDA 扩展推迟到单独 PR 的范围管理思路。

功能与动机

PR body 说明：git bisect 定位 commit 59e831c09a22（vLLM #48757）为首个引入测试失败的 revision。该 PR 在 AITER AR+RMS 融合 pass 之前对分离的 residual-add 与 RMSNorm 操作做规范化（canonicalize），使可见的合法融合点从变更前的 1 处变为全部 9 处。AMD CI build 11504 的 Distributed Compile Unit Tests 因旧测试硬编码期望值 1 而失败，需要同步更新测试期望。

实现拆解

1. 定位回归源头：通过 git bisect 定位到 commit 59e831c09a22（vLLM #48757）为首次引入 CI 失败的 revision，明确失败与 Transformers 模型路径的 fusion 计数变化相关。
2. 理解行为变化：#48757 开启 AddRMSNormFusionPass，在 AITER all-reduce/RMSNorm 融合之前，将 8 处此前分离的 aten.add + rms_norm 站点规范化（canonicalize），加上原本可见的 1 处最终 norm，合法融合点总数变为 9 处；因此测试中硬编码的 aiter_ar_rms_fusion=1 期望失配。
3. 更新测试期望：在 tests/compile/fusions_e2e/test_tp2_ar_rms.py 的 test_tp2_ar_rms_fusions 中，将 matches._replace(aiter_ar_rms_fusion=1) 改为 matches._replace(aiter_ar_rms_fusion=matches.ar_rms_fusion)，使期望值跟随通用 AR+RMS 融合计数（该计数由 matches_fn 按模型层数计算）。
4. 保留 if 块的必要性：review 中讨论了直接删除整个 if 块的方案，但 matches_fn(n_layers) 对 Transformers 返回的默认 aiter_ar_rms_fusion 计数与规范化后的真实融合数不一致，直接删除会导致测试再次失败，故保留覆盖逻辑。
5. 配套改动：无源码、配置或部署配套；单文件 +3/−3，仅更新注释与期望取值来源。

关键文件：

- tests/compile/fusions_e2e/test_tp2_ar_rms.py（模块 融合测试；类别 test；类型 test-coverage；符号 test_tp2_ar_rms_fusions）：唯一变更文件：测试期望从硬编码 aiter_ar_rms_fusion=1 改为跟随 matches.ar_rms_fusion，以匹配 #48757 规范化后暴露的九个 AR+RMS 融合点，解除 AMD CI 阻塞。

关键符号：test_tp2_ar_rms_fusions

关键源码片段

tests/compile/fusions_e2e/test_tp2_ar_rms.py

唯一变更文件：测试期望从硬编码 aiter_ar_rms_fusion=1 改为跟随 matches.ar_rms_fusion，以匹配 #48757 规范化后暴露的九个 AR+RMS 融合点，解除 AMD CI 阻塞。

```
@pytest.mark.parametrize(
    "attn_backend",
    [TRITON_ATTN, FLASHINFER_ATTN, ROCM_ATTN, ROCM_AITER_UNIFIED_ATTN],
)
@pytest.mark.parametrize("n_layers", [4])
@pytest.mark.skipif(not current_platform.is_cuda_alike(), reason="Only test CUDA/ROCm")
def test_tp2_ar_rms_fusions(
    model_name: str,
    matches_fn: Callable[[int], Matches],
    model_kwargs: dict,
    hf_overrides: Callable[[int], dict],
    model_impl: str,
    n_layers: int,
    run_e2e_fusion_test,
):
    # Transformers 3D AR+RMS 回归目前只在 ROCm 上跑
    if model_impl == "transformers" and not current_platform.is_rocm():
        pytest.skip("Transformers 3D AR+RMS regression is ROCm-only")

    matches = matches_fn(n_layers)
    if model_impl == "transformers":
        # #48757 的 AddRMSNormFusionPass 会在 AITER 融合前把 8 处分离的
        # aten.add + rms_norm 规范化，使全部 9 个通用 AR+RMS 融合点
        # （含最终 norm）对 AITER pass 可见；期望值因此必须跟随
        # matches.ar_rms_fusion（CUDA 侧同一份计数），而不是硬编码 1。
        # 直接删除该 if 块会让 matches_fn 默认返回的 Transformers 计数
        # 与实际融合数不一致，测试将再次失败。
        matches = matches._replace(aiter_ar_rms_fusion=matches.ar_rms_fusion)

    # 缩小模型规模、跳过权重加载以加速测试
    model_kwargs["hf_overrides"] = hf_overrides(n_layers)
    model_kwargs["load_format"] = "dummy"
    model_kwargs["model_impl"] = model_impl
    model_kwargs["max_model_len"] = 1024
    model_kwargs["kernel_config"] = {"enable_flashinfer_autotune": False}
    model_kwargs["disable_custom_all_reduce"] = False
```

```
compilation_config = dict(
    use_inductor_graph_partition=inductor_graph_partition,
    custom_ops=custom_ops.split(","),
    pass_config=PassConfig(
        enable_qk_norm_rope_fusion=True,
        fuse_allreduce_rms=True,
    ),
)

# 按平台选择要校验的融合计数键：ROCm 走 AITER，CUDA 走通用 AR+RMS
matches_check = ["norm_rope_fusion"]
if current_platform.is_rocm():
    matches_check.append("aiter_ar_rms_fusion")
else:
    matches_check.append("ar_rms_fusion")
```

评论区精华

1. 是否删除整个 if 块：BadrBasowid 建议直接移除 if model_impl == "transformers" 块，认为覆盖已无必要；AndreasKaratzas 回应删除会导致测试再次失败，因为 matches_fn 返回的 Transformers 默认计数与真实融合数不一致。结论：保留 if 块与 _replace 覆盖，仅更新取值来源。
2. 是否借机为 CUDA 启用 model_impl：BadrBasowid 询问是否趁此修改为 CUDA 开启该参数；AndreasKaratzas 认为单独开 PR 更合适，优先快速修复回归以解除 CI 阻塞。结论：CUDA 侧覆盖留待后续单独 PR。

- 是否可以直接删除 Transformers 期望覆盖块 (correctness): 保留 if 块与 _replace 调用；仅更新注释与取值来源为 matches.ar_rms_fusion。
- 是否借机为 CUDA 启用 model_impl 参数 (question): CUDA 侧的 Transformers 融合覆盖留待后续单独 PR，本次仅修复 ROCm 回归。

风险与影响

- 风险：
  1. 测试期望与上游 pass 行为耦合：新期望依赖 matches.ar_rms_fusion 与 AITER 实际融合数一致的前提，该一致性由 #48757 的 canonicalization 保证；未来若 Transformers 路径的规范化范围再次变化（如部分层不参与），测试会变红，但这是预期中的有效信号。
  2. 断言粒度为数量而非逐点核对：测试只比较融合计数，个别融合点错位可能被总数掩盖，属于该测试框架的既有设计限制。
  3. CUDA 侧无覆盖：model_impl == "transformers" 在非 ROCm 平台被跳过，CUDA 侧同类融合行为变化不会被该测试捕获，需依赖后续单独 PR 补上覆盖。 - 影响：对用户无任何运行时影响。对系统而言，本 PR 恢复 AMD CI（build 11504，Distributed Compile Unit Tests）绿色，解除 ROCm 相关 PR 的合并阻塞。对团队而言，测试期望语义更准确地反映真实融合行为，降低后续 ROCm 融合 pass 演进的误报风险，并明确了 CUDA 侧覆盖扩展的后续工作项。 - 风险标记：测试期望与上游 pass 行为耦合，数

量断言粒度较粗，仅 ROCm 覆盖

关联脉络

- PR #48757 Canonicalize residual-add and RMSNorm before AITER AR+RMS fusion: PR body 中 git bisect 定位为首次引入本测试失败的 commit 所属 PR（标题为基于 body 描述的归纳）；其 AddRMSNormFusionPass 规范化行为直接导致融合计数由 1 变 9。
- PR #50242 K3 DSpark AR fusion: 同一 AR+RMS 融合功能线（vllm/models/common/ops/fused_allreduce_rms_norm.py），说明融合 pass 持续演进，未来类似行为变化仍可能波及本测试期望。