PR #51083 完整报告

vllm-project/vllm

[ROCm] Relax MLA rope+cache test tolerances for bf16

合并时间：2026-08-06 06:50

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51083

执行摘要

- 一句话：放宽 ROCm bf16 下 MLA rope 融合测试容差
- 推荐动作：值得快速浏览，了解低精度数值测试中基于 ULP 的容差推导方法；无需深入精读。若团队维护 ROCm 测试矩阵，建议关注后续 bisect 结果，并在恢复严格容差后补一个注释说明根因。

功能与动机

CI 中 test_concat_and_cache_mla_rope_fused 在 ROCm gfx942 上对 bfloat16 的 48 个参数化用例失败。PR body 说明："The failures are ~1 bfloat16 ULP, not a kernel bug"且"One bf16 ULP at values ~2–3 is 0.0156–0.03125, i.e. 16–32× above the 0.001 tolerance"。目标是消除因精度分辨率低于容差阈值导致的假失败，同时保持对真实内核错误的敏感度。

实现拆解

1. 在 tests/kernels/core/test_rotary_embedding_mla_cache_fused.py 的 test_concat_and_cache_mla_rope_fused 中新增 rocm_bf16 = current_platform.is_rocm() and dtype == torch.bfloat16 标志，用于识别 ROCm 且低精度 bf16 的路径。
2. 对 fp8 kv-cache 分支：将 atol 从 0.001 调整为 0.004 if rocm_bf16 else 0.001，rtol 从 0.15 if rocm_neox else 0.1 扩展为 0.15 if rocm_neox or rocm_bf16 else 0.1，因为 bf16 在 e4m3 量化下约一个 ULP 12.5%。
3. 对非 fp8 的 kv-cache 分支：新增 elif rocm_bf16 分支，以 atol=0.02, rtol=1e-3 调用 assert_close，替代原来对 bf16 硬编码过紧的默认容差。
4. 对 query 断言：atol 调整为 0.04 if rocm_bf16 else get_default_atol(query)，因为 bf16 下 1 ULP 最大可达 0.03125，0.04 仍在一个 ULP 内。
5. 保持 fp16、fp32、CUDA 以及其他 ROCm 路径的原有容差，确保不会放宽非目标场景的检查。

关键文件：

- tests/kernels/core/test_rotary_embedding_mla_cache_fused.py（模块 内核测试；类别 test；类型 test-coverage；符号 test_concat_and_cache_mla_rope_fused）：唯一改动文件，通过对 ROCm bf16 路径的条件化容差设置，消除融合 MLA rope 内核测试在低精度下

的假失败，并保留其他路径的严格校验。

关键符号：test_concat_and_cache_mla_rope_fused

关键源码片段

tests/kernels/core/test_rotary_embedding_mla_cache_fused.py

唯一改动文件，通过对 ROCm bf16 路径的条件化容差设置，消除融合 MLA rope 内核测试在低精度下的假失败，并保留其他路径的严格校验。

```
# 仅针对 ROCm 平台放宽低精度（bf16/fp8）路径的容差，
# 原因：AITER Triton rope 与融合 C++ 内核各距 fp32 真值约 1 ULP，
# 而 CUDA 默认的硬编码容差低于 bf16 的分辨率。
rocm_neox = current_platform.is_rocm() and is_neox_style
rocm_bf16 = current_platform.is_rocm() and dtype == torch.bfloat16

if kv_cache_dtype == 'fp8':
    # fp8 路径先解量化到 fp16 再对比；
    # bf16 场景下 e4m3 量化误差可达约一个 ULP（12.5%），故同步上浮 atol/rtol。
    result_temp = torch.empty_like(kv_cache, dtype=torch.float16)
    ops.convert_fp8(result_temp, kv_cache.contiguous(),
                    kv_cache_scale.item(), kv_dtype=kv_cache_dtype)
    expected_temp = torch.empty_like(ref_kv_cache, dtype=torch.float16)
    ops.convert_fp8(expected_temp, ref_kv_cache,
                    kv_cache_scale.item(), kv_dtype=kv_cache_dtype)
    torch.testing.assert_close(
        result_temp,
        expected_temp,
        atol=0.004 if rocm_bf16 else 0.001, # bf16 路径四倍放宽
        rtol=0.15 if rocm_neox or rocm_bf16 else 0.1,
    )
elif rocm_bf16:
    # bf16 一个 ULP 约为 0.0156~0.03125，atol=0.02 仍保持在单 ULP 内
    torch.testing.assert_close(kv_cache, ref_kv_cache, atol=0.02, rtol=1e-3)
elif rocm_neox:
    torch.testing.assert_close(kv_cache, ref_kv_cache, atol=1e-3, rtol=1e-3)
else:
    torch.testing.assert_close(kv_cache, ref_kv_cache)

# query 部分同理，bf16 下放宽默认 atol 至 0.04
torch.testing.assert_close(
    query,
    ref_q_pe,
    atol=0.04 if rocm_bf16 else get_default_atol(query),
    rtol=get_default_rtol(query),
)
```

评论区精华

唯一的维护者讨论来自 AndreasKaratzas 的批准评论："LGTM Let's bisect Pytorch and see if we can restore the tolerance"。这表明放宽容差是权宜之计，后续希望通过二分定位 PyTorch 侧根因后再恢复严格阈值；同时 claude[bot] 指出 fork PR 默认不自动评审。

- 是否通过 bisect PyTorch 恢复更严格容差 (testing): 当前先以放宽容差解决 CI 假失败，并计划对 PyTorch 做二分定位根因后恢复严格阈值。

风险与影响

- 风险：主要风险在于放宽容差可能掩盖真实的数值回归，尤其 bf16 路径 atol=0.02 接近一个 ULP 上界，fp8 的 rtol=0.15 也仅略高于 e4m3 的 12.5% 量化误差；若未来内核引入超过 1 ULP 的误差，该测试可能无法及时捕获。另外，ROCm 与 CUDA 的容差不一致可能增加跨平台回归判断的复杂度，但测试代码未改动任何运行路径，风险整体可控。
- 影响：影响范围限于 ROCm CI 中该测试文件的通过率，预计从 96/144 恢复至 144/144；对模型推理、显存、性能无任何影响。对团队而言，减少了一次长期存在的 CI 红牌，但留下的容差设置需要后续 PyTorch bisect 的跟进才能恢复到更严格的检查。
- 风险标记：测试容差放宽，可能掩盖数值回归，ROCm 与 CUDA 容差不一致

关联脉络

- 暂无明显关联 PR