PR #50589 完整报告

vllm-project/vllm

[CI][Test] Seed the DeepEP v2 MoE workers, not just the parent

合并时间：2026-08-15 04:01

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50589

执行摘要

- 一句话：修复 DeepEP v2 MoE 测试 worker 未播种随机数导致的偶发失败
- 推荐动作：值得阅读，因为修复方式值得借鉴：不通过放宽容差掩盖问题，而是修复根本的不确定性；同时作者给出了清晰的根因分析和无法本地验证的诚实说明。建议有 GPU 权限的开发者运行该测试以进一步验证种子化后的数值行为。

功能与动机

关联 Issue #50184 报告 test_deep_ep_v2_moe 在多个 nightly CI 中长期偶发失败，断言 Only 98.2% of FP8 outputs are within tolerance。PR body 分析指出测试在父进程中调用 set_random_seed(7)，但实际参与断言的输入张量由 worker 中的 TestTensors.make(config)（torch.randn / torch.randperm）生成，而 torch.multiprocessing.spawn 启动的 worker 未播种 RNG，因此每次运行输入不同，导致 FP8 容差断言在临界情况下失败。

实现拆解

1. 定位根因：在 tests/kernels/moe/test_deepep_v2_moe.py 中，test_deep_ep_v2_moe 仅在父进程调用 set_random_seed(7)，覆盖 make_test_weights()；而 _deep_ep_v2_moe 中 TestTensors.make(config) 会在 worker 进程里用 torch.randn/torch.randperm 生成 rank_tokens、topk_weights、topk 等，worker 的 RNG 未播种导致每次运行输入不一致。
2. 修改 worker 入口：在 _deep_ep_v2_moe 和 _deep_ep_v2_moe_cudagraph 两个 worker 函数的 TestTensors.make(config) 调用前，添加 set_random_seed(7 + pgi.rank)，用 rank 做偏移确保各 rank 数据仍有差异，同时保证运行可复现。
3. 保持断言严格：刻意不改动 FP8 容差阈值（atol=rtol=2e-1）或 close_fraction > 0.99 的判定，避免掩盖真实数值问题。
4. 验证：作者无法在本地运行需 2 GPU + DeepEP v2 的测试，但通过不用 GPU 的父进程/worker 播种复现实验验证了机制；并运行 pre-commit 通过。期望 CI 中种子化后的运行结果可稳定复现，若仍失败则问题在 FP8 路径而非测试随机性。

关键文件：

- tests/kernels/moe/test_deepep_v2_moe.py（模块 MoE 测试；类别 test；类型 test-coverage；符号 _deep_ep_v2_moe, _deep_ep_v2_moe_cudagraph）：唯一修改的文件，为两个 worker 入口添加 RNG 播种，修复测试非确定性。

关键符号：_deep_ep_v2_moe, _deep_ep_v2_moe_cudagraph

关键源码片段

tests/kernels/moe/test_deepep_v2_moe.py

唯一修改的文件，为两个 worker 入口添加 RNG 播种，修复测试非确定性。

以下为 _deep_ep_v2_moe 中的修复片段：

```
def _deep_ep_v2_moe(
    pgi: ProcessGroupInfo,
    dp_size: int,
    config: TestConfig,
    w1: torch.Tensor,
    w2: torch.Tensor,
    w1_scale: torch.Tensor | None,
    w2_scale: torch.Tensor | None,
    use_fp8_dispatch: bool,
    per_act_token_quant: bool,
):
    device = torch.device(f"cuda:{pgi.local_rank}")
    init_workspace_manager(device)

    is_quantized = w1.dtype == torch.float8_e4m3fn
    device_idx = torch.accelerator.current_device_index()
    w1 = w1.to(device=device_idx)
    w2 = w2.to(device=device_idx)
    if is_quantized:
        assert w1_scale is not None and w2_scale is not None
        w1_scale = w1_scale.to(device=device_idx)
        w2_scale = w2_scale.to(device=device_idx)

    pg = torch.distributed.new_group(list(range(pgi.world_size)))

    # 父进程的 set_random_seed(7) 只覆盖 make_test_weights(),
    # spawn 出的 worker 拥有全新未播种 RNG，导致 TestTensors.make 每次运行结果不同。
    # 这里用 7 + pgi.rank 作为种子：既能复现，又让各 rank 的数据保持差异。
    set_random_seed(7 + pgi.rank)
    test_tensors = TestTensors.make(config)
```

评论区精华

维护者 tlrmchlsmth 批准该 PR，评论“looks good to fix flakey test（但保留 #50184 未关闭，因为容差还应被审计和调整）”。这体现了社区认可将测试稳定性与真实数值校验分离的做法。作者在 PR body 中声明“我无法运行测试本身”，并指出若 CI 中种子化后仍持续失败，将是比偶发失败更清晰的 bug 报告，建议后续聚焦 FP8 路径。Claude code review 因 fork PR 自动审查被禁用。

- 修复不稳定测试与容差审计 (testing): PR 被批准合并，issue #50184 保持打开以跟进 FP8 容差审计。

风险与影响

- 风险：变更仅涉及测试文件，不触碰生产代码，风险极低。由于作者未能在真实 GPU + DeepEP v2 环境运行测试，无法确认种子化后的数值是否完全稳定；如果种子化后的输入恰好落入更差的情况，测试可能稳定失败，但这反而有助于暴露真实 FP8 精度问题。引入 set_random_seed(7 + pgi.rank) 后，各 worker 的 RNG 与父进程不同，但生成的权重仍通过 torch.distributed.broadcast 同步，不会造成分布不一致。需注意 _deep_ep_v2_moe_cudagraph 中种子在 w1_bf16 生成之前，rank 0 使用种子 7，因此所有 worker 的权重仍一致，不影响 EP 校验。
- 影响：影响范围小，仅限测试稳定性和 CI 可靠性：test_deep_ep_v2_moe 从偶发失败变为确定性行为；若未来数值问题仍然存在，CI 会稳定报错，便于定位。对用户和生产系统无影响，团队可减少因误报而花费的排查时间。
- 风险标记：缺少真实 GPU 验证，仅测试代码变更

关联脉络

- 暂无明显关联 PR