PR #28237 完整报告

sgl-project/sglang

[AMD] fix(moe): correct fused shared-expert scaling on aiter/DeepEP path (mori all-to-all)

合并时间：2026-06-25 16:56

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28237

执行摘要

- 一句话：修复 AMD HIP 路径 fused shared-expert 权重缩放错误
- 推荐动作：该 PR 值得精读，尤其是理解 MoE 推理中 routed_scaling_factor 如何在不同路径下被应用，以及如何通过单元测试固定契约。设计上明确限定修复范围，避免引入不必要的风险。对 AMD 后端维护者有直接参考价值。

功能与动机

On the HIP aiter + DeepEP-class MoE path (e.g. MoRI all-to-all), the fused shared expert is under-weighted by routed_scaling_factor, corrupting every MoE layer and producing degenerate, non-converging generation on long outputs. (参见 PR body)

实现拆解

1. 修改 python/sglang/srt/layers/moe/topk.py 中的 _remap_topk_for_deepep 函数，在设置 fused shared-expert 权重时增加条件判断：当 _use_aiter 为 True 时，权重固定为 1.0；否则沿用原有逻辑（1/routed_scaling_factor）。同时添加详细注释说明两种路径的差异及为何这样设计。
2. 新增测试文件 test/registered/unit/layers/moe/test_fused_shared_expert_scaling.py，通过 mock _use_aiter 和并行参数，验证三种场景：aiter 路径（权重应为 1.0）、post-MoE scaling 路径（权重应为 1/rsf）、以及共享专家 ID 路由到 home rank 的正确性。
3. 测试注册为 CPU 测试 suite base-a-test-cpu，不依赖 GPU 硬件。在 CI 中通过 patch get_parallel 模拟并行环境，避免运行时依赖。
4. 经过 CI 测试失败（首次 patch 了不存在的函数）后，作者修正为 patch get_parallel，最终测试通过并完成合并。

关键文件：

- python/sglang/srt/layers/moe/topk.py（模块 MoE 路由；类别 source；类型 core-logic；符号 _remap_topk_for_deepep）：核心修复文件，修改了 _remap_topk_for_deepep 中 fused shared-expert 权重设置逻辑，增加条件分支区分 aiter 和默认路径。
- test/registered/unit/layers/moe/test_fused_shared_expert_scaling.py（模块 MoE 测试；类别 test；类型 test-coverage；符号 TestFusedSharedExpertScaling, _run_remap, test_aiter_path_uses_unit_shared_weight, test_post_moe_scaling_path_compensates_

with_inverse_rsf）：新增的单元测试固定了修复后的契约，覆盖 aiter、post-MoE scaling 和 shared expert ID 路由三种场景。

关键符号：_remap_topk_for_deepep, TestFusedSharedExpertScaling._run_remap, TestFusedSharedExpertScaling.test_aiter_path_uses_unit_shared_weight, TestFusedSharedExpertScaling.test_post_moe_scaling_path_compensates_with_inverse_rsf, TestFusedSharedExpertScaling.test_shared_expert_ids_route_to_home_rank

关键源码片段

test/registered/unit/layers/moe/test_fused_shared_expert_scaling.py

新增的单元测试固定了修复后的契约，覆盖 aiter、post-MoE scaling 和 shared expert ID 路由三种场景。

```
class TestFusedSharedExpertScaling(CustomTestCase):
    # 256 个物理路由专家，ep_size=8 → 每 rank 32 个本地路由专家
    NUM_PHYSICAL_ROUTED = 256
    EP_SIZE = 8
    EP_RANK = 0
    ROUTED_SCALING_FACTOR = 2.5

    def _run_remap(self, *, use_aiter):
        # 构造一个包含路由和共享专家的 topk 张量，共享权重初始为 -123.0 用于断言被覆盖
        topk_ids = torch.tensor([[5, 40, 100, 999]], dtype=torch.int32)
        routed_weights = torch.tensor([1.0, 0.5, 0.25], dtype=torch.float32)
        topk_weights = torch.tensor([[1.0, 0.5, 0.25, -123.0]], dtype=torch.float32)
        topk_config = TopKConfig(top_k=4, num_fused_shared_experts=1,
                                 routed_scaling_factor=self.ROUTED_SCALING_FACTOR)
        with (
            patch.object(topk_module, "_use_aiter", use_aiter),
            patch.object(topk_module, "get_parallel",
                         return_value=SimpleNamespace(
                             moe_ep_size=self.EP_SIZE, moe_ep_rank=self.EP_RANK)),
        ):
            _out_ids, out_weights = topk_module._remap_topk_for_deepep(
                topk_ids.clone(), topk_weights.clone(),
                num_fused_shared_experts=1,
                num_physical_routed_experts=self.NUM_PHYSICAL_ROUTED,
                topk_config=topk_config)
        # 确保路由权重未被修改
        self.assertTrue(torch.equal(out_weights[0, :-1], routed_weights))
        return out_weights[0, -1].item()

    def test_aiter_path_uses_unit_shared_weight(self):
        # aiter 路径：共享权重应为 1.0
        shared_weight = self._run_remap(use_aiter=True)
        self.assertAlmostEqual(shared_weight, 1.0)

    def test_post_moe_scaling_path_compensates_with_inverse_rsf(self):
```

```
        # 默认路径：共享权重应为 1/rsf
        shared_weight = self._run_remap(use_aiter=False)
        self.assertAlmostEqual(shared_weight, 1.0 / self.ROUTED_SCALING_FACTOR)

    def test_shared_expert_ids_route_to_home_rank(self):
        # 验证共享专家 ID 被正确设置到 home rank 的交错位置
        topk_ids = torch.tensor([[5, 40, 100, 999]], dtype=torch.int32)
        topk_weights = torch.tensor([[1.0, 0.5, 0.25, 0.0]], dtype=torch.float32)
        topk_config = TopKConfig(top_k=4, num_fused_shared_experts=1,
                                 routed_scaling_factor=self.ROUTED_SCALING_FACTOR)
        with (
            patch.object(topk_module, "_use_aiter", True),
            patch.object(topk_module, "get_parallel",
                         return_value=SimpleNamespace(
                             moe_ep_size=self.EP_SIZE, moe_ep_rank=self.EP_RANK)),
        ):
            out_ids, _ = topk_module._remap_topk_for_deepep(
                topk_ids.clone(), topk_weights.clone(),
                num_fused_shared_experts=1,
                num_physical_routed_experts=self.NUM_PHYSICAL_ROUTED,
                topk_config=topk_config)
        num_local_routed = self.NUM_PHYSICAL_ROUTED // self.EP_SIZE # 32
        num_local_experts = num_local_routed + 1 # 33
        expected_shared_id = self.EP_RANK * num_local_experts + num_local_routed
        self.assertEqual(out_ids[0, -1].item(), expected_shared_id)
```

评论区精华

主要讨论包括：

- CI 测试失败：amd-bot 指出测试中 patch 了不存在的函数 *get_moe_expert_parallel_world_size* 等，要求修正。作者随后将 patch 对象改为 *get_parallel* 并调整测试逻辑，测试通过。
- 注册为 CPU 测试：评审者 yctseng0211 询问为何将测试注册为 CPU 测试（suite base-a-test-cpu），作者解释该测试是纯 CPU 单元测试，不依赖 GPU，因此放在 CPU suite 中合理。最终所有 CI 通过，PR 获批准。
  - 注册为 CPU 测试的原因 (question): 测试为纯 CPU 单元测试，不依赖 GPU，注册在 CPU suite 合理。
  - CI 测试失败与被修正 (correctness): 作者将 patch 对象从 *get_moe_expert_parallel_** 改为 *get_parallel*，测试通过。

风险与影响

- 风险：该修复范围明确限定于 *aiter* 路径，对于其他类似的后端（如 ModelOpt NVFP4、*cutlass/trtllm-routed* fp8）可能存在同样的缺陷但未处理，需各后端维护者自行验证并修复。修改位于核心 MoE 路由逻辑，虽然单元测试覆盖，但在生产环境中 AMD MI355X 以外的硬件上未经过端到端验证。修复本身较小，回归风险较低。

- 影响：影响主要面向使用 AMD GPU 且启用 aiter + DeepEP 的 DeepSeek-R1 等 MoE 模型用户：长文本推理质量从退化（重复循环）恢复到正常。对默认路径（非 aiter）无影响。新增的单元测试可作为未来重构的回归保障。其他后端用户可能仍然受类似 bug 影响，但不在此 PR 范围内。
- 风险标记：限定 aiter 路径，其他后端待验证，测试 mock 依赖

关联脉络

- 暂无明显关联 PR