PR #50859 完整报告

vllm-project/vllm

[ROCm][AITER] Hotfix for `memory access fault` errors in AITER triton MOE routing

合并时间：2026-08-04 10:07

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50859

执行摘要

- 一句话：AITER MoE 路由误处理 padding 行致内存访问错误的热修复。
- 推荐动作：值得精读。它示范了外部依赖回归下的三层应对：vLLM 侧临时 hotfix、针对性回归测试、指向上游修复的 TODO 清理机制。建议关注两点：一是 _get_padding_mask 对 Model Runner V1/V2 双版本的兼容处理，二是测试中的显存毒化技巧如何暴露越界读；同时应在 AITER 0.1.20+ pin 升级时及时验证并移除本补丁。

功能与动机

PR body 明确说明这是对 AITER 0.1.19 升级（vllm-project/vllm PR 49361）引入回归的热修复：ROCm CI 中 test_gpt_oss_attention_quantization[amd/gpt-oss-20b-MoE-Quant-W-MXFP4-A-FP8-KV-FP8-0.89-1] 开始以 Memory access fault by GPU node-2 失败，而 AITER 0.1.16.post5 时正常。根因是 AITER PR 4198 让 routing 错误处理含 inf/-inf/nan 的 token 行，HIP graph 重放时 padding 行恰好携带这类无效数据。

实现拆解

1. 定位根因

AITER 0.1.19 的 triton routing 将 padding 行视作有效 token，gating output 中的 inf/nan 导致非法 expert id 与越界内存访问。

2. 新增 padding mask 提取

在 vllm/model_executor/layers/fused_moe/experts/aiter_mxfp4_w4a8_moe.py 中实现 _get_padding_mask()，优先读取 forward context 的 is_padding（Model Runner V2），否则从 slot_mapping < 0（Model Runner V1）推导；两种 runner 的契约在注释中说明。

3. 新增 gating output 清理

patch_gating_output() 在 global_num_experts != 128 时，将 padding 行 gating output masked_fill 为 0.0，使 routing 的 top-k 不会选中 padding 行；128 专家场景暂不处理。

4. 挂钩两条前向路径

在 aiter_triton_kernel_w4a8_moe_forward 与 aiter_triton_kernel_w4a16_moe_forward 中、调用 aiter_routing 之前执行 patch_gating_output，并保留 TODO 指向 AITER 修复。

5. 测试配套

tests/models/quantization/test_gpt_oss.py 新增回归测试，构造 padding 垃圾行（inf/nan）并释放显存做内存毒化，在注入 is_padding 的 forward context 下调用两条前向，断言 unpadded 输出全部 isfinite；同时保留原端到端精度测试。注意测试函数体内 num_experts = 32 覆盖了参数化，使 64/128 专家参数实际未生效。

关键文件：

- vllm/model_executor/layers/fused_moe/experts/aiter_mxfp4_w4a8_moe.py（模块 专家内核；类别 source；类型 core-logic；符号 _get_padding_mask, patch_gating_output）：修复核心：新增 padding mask 提取与 gating output 清理，防止 AITER routing 误处理 inf/nan 导致 memory access fault。
- tests/models/quantization/test_gpt_oss.py（模块 量化测试；类别 test；类型 test-coverage；符号 on_gfx1250, test_aiter_mxfp4_moe_ignores_padded_rows）：新增回归测试，模拟 CUDA graph padding 垃圾行，验证未填充输出保持有限值；同时保留原端到端精度测试。

关键符号：_get_padding_mask, patch_gating_output, aiter_triton_kernel_w4a8_moe_forward, aiter_triton_kernel_w4a16_moe_forward, test_aiter_mxfp4_moe_ignores_padded_rows

关键源码片段

vllm/model_executor/layers/fused_moe/experts/aiter_mxfp4_w4a8_moe.py

修复核心：新增 padding mask 提取与 gating output 清理，防止 AITER routing 误处理 inf/nan 导致 memory access fault。

```
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project

# 从 forward context 恢复 padding mask，兼容 Model Runner V1 / V2 两代约定。
def _get_padding_mask() -> torch.Tensor | None:
    """
    返回布尔掩码：非 padding 为 0，padding 为 1。

    `slot_mapping < 0` 的约定来自 gpu_model_runner.py：
    slot_mapping[num_tokens_unpadded:num_tokens_padded].fill_(-1)
    """
    forward_context = get_forward_context()

    # Model Runner V2 直接暴露 is_padding 字段。
    if forward_context.is_padding is not None:
        return forward_context.is_padding

    # Model Runner V1 从 slot_mapping 推断 padding token。
    slot_mapping = forward_context.slot_mapping
    if isinstance(slot_mapping, list):
        slot_mapping_dict = slot_mapping[0]
```

```
    else:
        slot_mapping_dict = slot_mapping

    if isinstance(slot_mapping_dict, dict):
        slot_mapping_sample = next(iter(slot_mapping_dict.values()), None)
    else:
        slot_mapping_sample = slot_mapping_dict

    # slot_mapping 为负即 padding token。
    if isinstance(slot_mapping_sample, torch.Tensor):
        return slot_mapping_sample < 0
    else:
        return None


# 将 padding 行路由分数清零，避免 AITER routing 误处理 inf / nan。
def patch_gating_output(gating_output, global_num_experts):
    # 128 专家场景暂不生效，与 AITER 侧行为差异有关。
    if global_num_experts != 128:
        is_padding = _get_padding_mask()

        if is_padding is not None:
            # 掩码行数与 gating_output 必须一致。
            assert is_padding.shape[0] == gating_output.shape[0]
            gating_output = gating_output.masked_fill(is_padding[:, None], 0.0)

    return gating_output
```

tests/models/quantization/test_gpt_oss.py

新增回归测试，模拟 CUDA graph padding 垃圾行，验证未填充输出保持有限值；同时保留原端到端精度测试。

```
@pytest.mark.parametrize('num_experts', [32, 64, 128])
def test_aiter_mxfp4_moe_ignores_padded_rows(mxfp4_backend, num_experts, monkeypatch,
dist_init):
    """
    cudagraph padding 行的垃圾数据不能污染 unpadded 输出。

    AITER 0.1.19 的回归：tokens_unpadded = 7、tokens_padded = 8 时，
    padding 行含 inf / -inf / nan，routing 误处理后会触发
    memory access fault。
    """
    TOKENS_PADDED = 8
    TOKENS_UNPADDED = 7

    # rocm_aiter_ops 在 import 时缓存环境变量，需手动刷新。
    monkeypatch.setenv('VLLM_ROCM_USE_AITER', '1')
    rocm_aiter_ops.refresh_env_variables()
```

```
# 构造带垃圾数据的 padding 行，并释放大块显存做内存毒化。
hidden_states[TOKENS_UNPADDED:, 0] = float('inf')
hidden_states[TOKENS_UNPADDED:, 1] = float('-inf')
hidden_states[TOKENS_UNPADDED:, 2] = float('nan')
gating_output[TOKENS_UNPADDED:, :] = float('-inf')
del blocks
torch.accelerator.synchronize()

# 通过 forward context 显式注入 is_padding，模拟 Model Runner V2。
with set_forward_context(None, get_current_vllm_config(), is_padding=is_padding):
    if is_w4a8:
        output = aiter_triton_kernel_w4a8_moe_forward(...)
    else:
        output = aiter_triton_kernel_w4a16_moe_forward(...)

# 核心断言：真实 token 输出必须是有限值。
assert torch.isfinite(output[:TOKENS_UNPADDED]).all()
```

评论区精华

PR 几乎没有实质技术讨论。claude[bot] 因 PR 来自 fork 自动跳过评审；AndreasKaratzas 以 LGTM 批准。作者在 body 中给出完整根因链（AITER PR 4198 回归 + HIP graph padding）与临时方案说明，Issue 评论确认 mi355_1 Quantized Models Test 通过。

- 来自 fork 的自动化审查与快速批准 (other): 无需额外修改，直接合并；作者在 body 中已说明根因与 TODO。

风险与影响

- 风险：临时补丁依赖上游 AITER 修复（ROCm/aiter PR 4530），若上游未及时发布或 pin 未更新，该 hack 会长期滞留，并可能与上游修复产生重复逻辑；patch_gating_output 仅对 global_num_experts != 128 生效，而失败用例恰好是 128 专家的 gpt-oss-20b，128 专家路径是否同样受影响需要确认；_get_padding_mask 依赖 forward context 中 is_padding 与 slot_mapping 的既有契约，未来 runner 变更可能让补丁静默失效；新增测试的参数化被函数体内 num_experts = 32 覆盖，覆盖范围有限。
- 影响：影响范围限于 ROCm + AITER + MXFP4 MoE（w4a8/w4a16）路径：修复 HIP graph padding 导致的 GPU memory access fault，恢复 gpt-oss-20b 量化精度测试与相关生产推理；非 ROCm 或非 AITER 路径无行为变化。对团队而言是 CI 稳定性修复，后续需在 AITER pin 升级时清理本补丁。
- 风险标记：临时依赖上游修复，128 专家路径未覆盖，测试参数化失效，forward context 契约耦合

关联脉络

- PR #49361 Bump AITER version from 0.1.16.post5 to 0.1.19: PR body 中引用的 AITER 版本升级，是本 PR 要修复的回归根因来源。