

PR #49714 完整报告

vllm-project/vllm

[ROCm][Bugfix] Sanitize AITER paged-MQA logits before sparse top-k for DeepSeek-V4

合并时间: 2026-07-29 02:08

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/49714>

执行摘要

- 一句话: 修复 ROCm gfx950 稀疏 attention 的 NaN/Inf 导致 GPU 内存故障
- 推荐动作: 值得精读: 展示了在无法修改底层库时, 通过上层 sanitize 快速隔离问题并配备精准回归测试的典型模式。对类似跨库 bug 的定位与修复有参考价值。

功能与动机

在 8x gfx950 上并发推理 DeepSeek-V4 时触发 GPU 内存故障 (max_model_len 8192/9472/16384 均复现)。根因为 AITER `deepgemm_fp8_paged_mqa_logits` 核函数间歇性输出 NaN 或 Inf, 通用 histogram top-k 核假定输入有限, 导致部分共享内存槽未写入、全局索引缓冲区中出现未初始化值, 最终被稀疏注意力用作块表地址引发越界访问。详见外部 issue #13。

实现拆解

1. 核心 sanitize: 在 `vllm/v1/attention/ops/rocm_aiter_mla_sparse.py` 的 `rocm_fp8_paged_mqa_logits` 函数中, 于 AITER 核调用后立即对输出张量执行 `out_logits.nan_to_num(float('-inf'))`, 将 NaN/Inf 统一替换为 dtype 的有限最小值, 确保 top-k 消费者获得可比较输入。
2. 回归测试: 在 `tests/kernels/attention/test_rocm_triton_attn_dsv4.py` 中新增 `test_paged_mqa_logits_do_not_contain_nan`, 通过 monkeypatch 模拟 AITER 返回全 NaN logits, 验证 sanitize 后输出不含有有限值以外的元素。

关键文件:

- `vllm/v1/attention/ops/rocm_aiter_mla_sparse.py` (模块 稀疏注意力; 类别 source; 类型 core-logic; 符号 `rocm_fp8_paged_mqa_logits`): 核心修复: 在 AITER 核输出后添加一行 `sanitize`, 防止 NaN/Inf 传播到 top-k 步骤。
- `tests/kernels/attention/test_rocm_triton_attn_dsv4.py` (模块 稀疏注意力; 类别 test; 类型 test-coverage; 符号 `test_paged_mqa_logits_do_not_contain_nan`, `FakeWorkspaceManager`, `fake_paged_mqa_logits`): 新增回归测试, 验证 `sanitize` 逻辑在 monkeypatch 模拟 AITER 输出 NaN 时仍生效。

关键符号: `rocm_fp8_paged_mqa_logits`, `test_paged_mqa_logits_do_not_contain_nan`

关键源码片段

vllm/v1/attention/ops/rocm_aiter_mla_sparse.py

核心修复：在 AITER 核输出后添加一行 sanitize，防止 NaN/Inf 传播到 top-k 步骤。

```
def rocm_fp8_paged_mqa_logits(
    q_fp8: torch.Tensor,
    kv_cache_fp8: torch.Tensor,
    weights: torch.Tensor,
    context_lens: torch.Tensor,
    block_tables: torch.Tensor,
    max_seq_len: int,
    ...
) -> torch.Tensor:
    # ... 调用 AITER 内核 ...
    deepgemm_fp8_paged_mqa_logits(
        q_fp8, kv_cache_fp8, weights, out_logits,
        context_lens, block_tables, max_seq_len,
        KVCacheDtype="fp8", WavePerEU=2
    )
    # 将 NaN/Inf 替换为有限最小值，保证 top-k 排序不产生未定义行为
    out_logits.nan_to_num_(float("-inf"))
    return out_logits
```

tests/kernels/attention/test_rocm_triton_attn_dsv4.py

新增回归测试，验证 sanitize 逻辑在 monkeypatch 模拟 AITER 输出 NaN 时仍生效。

```
@torch.inference_mode()
def test_paged_mqa_logits_do_not_contain_nan(monkeypatch) -> None:
    # 导入被测试模块
    from vllm._aiter_ops import rocm_aiter_ops
    from vllm.v1.attention.ops import rocm_aiter_mla_sparse as mod

    device = torch.device("cuda")

    class FakeWorkspaceManager:
        def get_simultaneous(self, *shapes_and_dtypes):
            return [torch.empty(s, dtype=d, device=device) for s, d in shapes_and_dtypes]

    def fake_paged_mqa_logits(
        q_fp8, kv_cache_fp8, weights, out_logits, context_lens,
        block_tables, max_seq_len, **kwargs
    ):
        # 忽略参数，直接将输出填充为 NaN，模拟 AITER 缺陷
        del (q_fp8, kv_cache_fp8, weights, context_lens, block_tables,
            max_seq_len, kwargs)
        out_logits.fill_(float("nan"))

    # 模拟 gfx950 环境并替换 AITER 实现
    monkeypatch.setattr(mod, "_ON_GFX942", False)
    monkeypatch.setattr(mod, "_ON_GFX950", True)
```

```

monkeypatch.setattr(rocm_aiter_ops, "is_enabled", lambda: True)
monkeypatch.setattr(
    mod, "paged_mqa_logits_module",
    lambda: SimpleNamespace(deepgemm_fp8_paged_mqa_logits=fake_paged_mqa_logits)
)
monkeypatch.setattr(
    mod, "current_workspace_manager", lambda: FakeWorkspaceManager()
)

# 调用被测试函数
logits = mod.rocm_fp8_paged_mqa_logits(
    torch.empty((1, 1, 1, 1), dtype=torch.uint8, device=device),
    torch.empty((1, 1, 1, 5), dtype=torch.uint8, device=device),
    torch.empty((1, 1), dtype=torch.float32, device=device),
    torch.ones(1, dtype=torch.int32, device=device),
    torch.zeros((1, 1), dtype=torch.int32, device=device),
    torch.empty(0, dtype=torch.int32, device=device),
    1,
)

# 断言 sanitize 后不存在 NaN
assert not torch.isnan(logits).any()

```

评论区精华

- [tjtanaa](#) 指出 `torch.finfo(dtype).min` 开销可能较大，[shen-shanshan](#) 确认该行不需要，最终实现仅保留 `nan_to_num_`。
- [Rohan138](#) 提到类似 NaN 问题在 AITER 上游已有修复（commit [a5bd6d9](#)），但 [shen-shanshan](#) 验证该上游提交未解决本场景问题，确认此 PR 作为 vLLM 侧临时 workaround。
- 性能对比显示吞吐量下降 0.31%、TTFT 增 0.39%、TPOT 增 0.26%，[tjtanaa](#) 认为可接受以修复功能性错误。
- 性能开销考量 (performance): 简化实现，仅使用 `nan_to_num_`。
- 临时 workaround vs 上游修复 (other): PR 作为 vLLM 侧临时 workaround，长期需 AITER 上游修复。

风险与影响

- 风险：风险集中在临时 workaround 对性能的轻微影响（~0.3%）以及仅覆盖 ROCm AITER 路径；根本缺陷在 AITER 上游，若上游后续修复，vLLM 侧该 `sanitize` 可作为防御性保留。修改未触及 NVIDIA 路径或 sampler 核，无跨平台风险。新增测试确保改动可被 CI 捕获。
- 影响：对 ROCm gfx950 + DeepSeek-V4 用户是功能性修复，解决 GPU 内存故障导致服务中断的问题；对其他硬件或模型无影响。性能下降 < 0.5%，在可接受范围内。

- 风险标记: 核心路径变更 (sparse top-k 前处理), 临时 workaround (需上游修复), 性能略微下降 (<0.5%)

关联脉络

- PR #50060 [CI] Fix CI error for PR #49714: 评论中提及该 PR 修复了本 PR 的 CI 错误, 确保测试正常通过。