PR #29918 完整报告

sgl-project/sglang

[AMD] Gate broken CK block-FP8 GEMM shapes to aiter-triton-GEMM to fix ROCm 7.0 Qwen3.5 accuracy

合并时间：2026-07-03 09:18

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29918

执行摘要

- 一句话：修复 ROCm 7.0 上 FP8 GEMM 的 NaN 问题
- 推荐动作：该 PR 是一个针对特定硬件 / 软件组合的精准修复，设计思路清晰——按形状和 M 动态选择后端，而不是全局弃用 CK。值得关注其 优雅的降级策略，可作为后续同类硬件差异修复的参考模式。如果团队在生产 AMD gfx950 + ROCm 7.0，建议立即合并。

功能与动机

Qwen3.5-397B-A17B-FP8 在 AMD gfx950 + ROCm 7.0 上准确率从正常 0.95 跌至 0.24，且约 67% 输出无效。PR body 中详细分析了根因：由于 ROCm 7.0 hipcc 编译问题，bpreshuffle CK GEMM 被禁用，fallback 至非 bpreshuffle 的 ck_gemm_a8w8_blockscale，该实现在特定形状和大 M 值下返回 NaN。已在 PR body 附上形状与 NaN 阈值的对照表，并在 ROCm 7.2 上验证 CK 行为正确。

实现拆解

1. 定义安全 M 上限字典：在 python/sglang/srt/layers/quantization/fp8_utils.py 中新增模块级字典 _AITER_GFX95_CK_W8A8_MAX_SAFE_M，记录受影响的 (n, k) 形状及其确认安全的 M 上限。
2. 修改 GEMM 选择逻辑：在 aiter_w8a8_block_fp8_linear 函数的 elif _use_aiter_gfx95: 分支中，当 input_2d.shape[0] > _ck_safe_m 时，强制 use_triton=True，从而回退到数值正确的 Triton 实现。
3. 保留小 M 性能：对于低于安全 M 上限的场景，仍使用更快的 CK 路径，避免性能退化。
4. 不改动其他路径：对 ROCm ≥ 7.2、非 gfx95 硬件、或未在字典中列出的形状，逻辑保持不变。

关键文件：

- python/sglang/srt/layers/quantization/fp8_utils.py（模块 量化层；类别 source；类型 core-logic）：核心更改文件：定义安全 M 字典并修改 GEMM 选择逻辑，直接修复精度问题。

关键符号：未识别

关键源码片段

python/sglang/srt/layers/quantization/fp8_utils.py

核心更改文件：定义安全 M 字典并修改 GEMM 选择逻辑，直接修复精度问题。

```
# 位于 _use_aiter_bpreshuffle_gfx95 定义之后
# gfx95 + ROCm < 7.2: bpreshuffle CK 被禁用（如上），非 bpreshuffle 的
# ck_gemm_a8w8_blockscale 后备路径在某些形状下对大 M 返回 NaN
# （实测 NaN 起始点：(2560,4096)@M>=4096, (4096,1024)@M>=8192），
# 这会导致 prefill 阶段被污染。
# 这里将每个受影响的 (n, k) 映射到确认安全的 CK 最大 M 值
# （保守取最后一个安全 M）。在不超过该 M 时保留较快的 CK 路径，
# 超过时回退到数值正确的 Triton FP8 GEMM。该问题在 ROCm 7.2 中已修复。
_AITER_GFX95_CK_W8A8_MAX_SAFE_M = {
    (2560, 4096): 2048,
    (4096, 1024): 4096,
}

# ... 省略其他代码 ...

def aiter_w8a8_block_fp8_linear(
    input: torch.Tensor,
    weight: torch.Tensor,
    block_size: List[int],
    weight_scale: torch.Tensor,
    input_scale: Optional[torch.Tensor] = None,
    bias: Optional[torch.Tensor] = None,
) -> torch.Tensor:
    input_2d = input.view(-1, input.shape[-1])
    output_shape = [*input.shape[:-1], weight.shape[0]]
    n, k = weight.shape
    # 原有分支：当 bpreshuffle 可用则优先使用 tuned gfx950
    if _use_aiter_bpreshuffle_gfx95:
        use_triton = use_aiter_triton_gemm_w8a8_tuned_gfx950(n, k)
    elif _use_aiter_gfx95:
        # gfx95 + ROCm < 7.2: 在安全 M 以下保留较快 CK 路径；高于阈值时
        # ck_gemm_a8w8_blockscale 返回 NaN，故改用 Triton。未列出的形状保持原有决策。
        _ck_safe_m = _AITER_GFX95_CK_W8A8_MAX_SAFE_M.get((n, k))
        use_triton = use_aiter_triton_gemm_w8a8_tuned_gfx950(n, k) or (
            _ck_safe_m is not None and input_2d.shape[0] > _ck_safe_m
        )
    else:
        use_triton = True
    # 后续根据 use_triton 选择具体算子
```

评论区精华

该 PR 的 review 讨论很简单，仅包含自动机器人评论和 HaiShaw 的批准，没有实质性质疑。HaiShaw 直接批准了变更。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险低：变更仅作用于 gfx95 + ROCm < 7.2 + 特定（n, k）形状 + 大 M，且保留了小 M 的 CK 路径；其他硬件和 ROCm 版本不受影响。
  2. 性能风险：大 M 场景切换到 Triton 实现，可能比 CK 慢，但保证了正确性；对于 Qwen3.5 的 prefill 阶段，优先级是正确性。
  3. 测试覆盖不足：PR 描述提到已通过本地和 CI 测试验证精度恢复，但未包含直接的单元测试文件，手动构造的 fp8 GEMM 测试未纳入回归套件。 - 影响：直接影响：修复 Qwen3.5-397B-A17B-FP8 在 AMD gfx950 + ROCm 7.0 上的推理精度，GSM8K 从 0.244 恢复到 0.950，达到 ROCm 7.2 参考水平。影响范围：仅限使用 AMD gfx950 且 ROCm 版本低于 7.2，同时运行时触发受影响的 GEMM 形状（(2560,4096) 和 (4096,1024)）且 M 值大于 2048 或 4096 的场景。性能：大 M 场景下 Triton 代替 CK，预计 prefill 用时略有增加，但 *decode* 等小 M 路径不受影响。 - 风险标记：缺少测试覆盖

关联脉络

- PR #23319 Unknown (mentioned as PR#23319): PR body 中提及的 ROCm 7.0 hipcc miscompile 问题，是导致 bpreshuffle 被禁用的根本原因，本 PR 在此基础上进一步修复了 fallback 路径的 NaN bug。