PR #30900 完整报告

sgl-project/sglang

[AMD] [Quantization] [Bugfix] Fix bug related to fp8 max on gfx95x for per-token-group quant (ROCm)

合并时间：2026-08-16 10:54

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30900

执行摘要

- 一句话：gfx95x fp8 量化改用设备真实上限 448，修复静默精度损失
- 推荐动作：值得快速精读：这是一个小而典型的“设备相关常量硬编码”bug 修复，教科书式展示了用模块级解析常量替代魔法数字，并用非 mock 的真实内核回归测试在目标硬件上钉住行为。对关注 ROCm 量化精度、硬件差异处理或硬件门控测试策略的工程师有参考价值；且已获 AMD 维护者批准合并。

功能与动机

PR body 明确指出：_per_token_group_quant_8bit_raw 对所有 ROCm 设备硬编码 224.0，但 fp8 范围是设备相关的：gfx94x (MI300) 为 e4m3fnuz（max 224.0），gfx95x (MI355X) 为 e4m3fn（max 448.0）。模块本身已经用 fp8_max 正确解析了这一差异，但该函数忽略了它。在 gfx95x 上 per-group scale 被算成 absmax / 224 而不是 absmax / 448，虽然不会崩溃或产生 NaN，但只用到了 ±448 范围中的 [-224, 224] 子集，浪费约 1 bit 精度，造成 MI355X 上静默的精度退化。gfx94x 与 CUDA 不受影响。

实现拆解

1. 核心修复（python/sglang/kernels/ops/quantization/fp8_kernel.py）：在 _per_token_group_quant_8bit_raw 的 ROCm fp8 分支中，把 bit8_max = 224.0 改为 bit8_max = fp8_max。fp8_max 是该模块已按设备解析好的常量（gfx94x = 224.0、gfx95x = 448.0），因此 gfx94x 与 CUDA 行为保持不变；int8 分支（含既有 # TODO incorrect for int8）原样保留，bit8_min = -bit8_max 不变。
2. 新增回归测试（test/registered/unit/layers/quantization/test_fp8_kernel_hip_max.py）：不 mock Triton 内核，直接在真实 GPU 上调用 _per_token_group_quant_8bit_raw，构造一个 absmax 恰好等于 448.0（e4m3fn 上限）的 group，断言输出的 per-group scale 为 1.0（448/448）；在旧代码上 scale 为 2.0（448/224），测试必失败。测试用 is_hip() and is_gfx95_supported() 门控，并通过 register_amd_ci(est_time=20, suite="stage-b-test-1-gpu-small-amd-mi35x") 注册进 MI355X CI 套件。
3. 硬件验证：作者在真实 MI355X (gfx950) 上验证——旧代码测试失败（scale=2.0），修复后通过（scale=1.0）；gfx94x 与 CUDA 路径未受影响。

关键文件：

- python/sglang/kernels/ops/quantization/fp8_kernel.py（模块 量化内核；类别 source；类型 core-logic；符号 _per_token_group_quant_8bit_raw）：本次 bug 修复的唯一生产代码变更所在：在 _per_token_group_quant_8bit_raw 的 ROCm fp8 分支中，将写死的 224.0 替换为设备解析常量 fp8_max，恢复 gfx95x 上 e4m3fn 的 ±448 全精度范围；gfx94x、CUDA 与 int8 路径不受影响。
- test/registered/unit/layers/quantization/test_fp8_kernel_hip_max.py（模块 回归测试；类别 test；类型 test-coverage；符号 TestPerTokenGroupQuant8BitHipMax, test_hip_fp8_scale_uses_e4m3fn_max_448）：新增的 MI355X (gfx950) 回归测试：不 mock 真实 Triton 量化内核，构造 absmax=448 的输入组并断言 scale=1.0，旧代码下 scale=2.0 必失败；已注册到 stage-b-test-1-gpu-small-amd-mi35x CI 套件，是本修复的硬件级保障。

关键符号：_per_token_group_quant_8bit_raw, test_hip_fp8_scale_uses_e4m3fn_max_448

关键源码片段

python/sglang/kernels/ops/quantization/fp8_kernel.py

本次 bug 修复的唯一生产代码变更所在：在 _per_token_group_quant_8bit_raw 的 ROCm fp8 分支中，将写死的 224.0 替换为设备解析常量 fp8_max，恢复 gfx95x 上 e4m3fn 的 ±448 全精度范围；gfx94x、CUDA 与 int8 路径不受影响。

```
# python/sglang/kernels/ops/quantization/fp8_kernel.py
# `_per_token_group_quant_8bit_raw` 的 ROCm fp8 分支核心片段（函数其余部分省略）：

if dtype == torch.int8:
    # int8 路径保持原样：上限 127.0，min 取负（既有 TODO 未处理）。
    bit8_max = 127.0
else:
    # fp8 的 clamp/scale 上限随设备变化：
    # gfx94x (MI300) 为 e4m3fnuz，max 224.0；
    # gfx95x (MI355X) 为 e4m3fn，max 448.0。
    # 必须使用模块内已按平台解析好的 fp8_max，而不是写死 gfx94x 的 224.0，
    # 否则 gfx95x 上 scale 会被算成 absmax / 224，丢掉 e4m3fn 一半动态范围。
    bit8_max = fp8_max

bit8_min = -bit8_max
# 上一行对 int8 的对称取负并不严格正确（见既有 TODO），本次改动不触碰该问题。
```

test/registered/unit/layers/quantization/test_fp8_kernel_hip_max.py

新增的 MI355X (gfx950) 回归测试：不 mock 真实 Triton 量化内核，构造 absmax=448 的输入组并断言 scale=1.0，旧代码下 scale=2.0 必失败；已注册到 stage-b-test-1-gpu-small-amd-mi35x CI 套件，是本修复的硬件级保障。

```
"""AMD/gfx950 (MI355X) 回归测试：per-token-group quant 必须使用设备相关 fp8 上限。

ROCm 的 fp8 格式随设备不同：gfx94x (MI300) 为 e4m3fnuz（max 224.0），
gfx95x (MI355X) 为 e4m3fn（max 448.0）。旧实现把 224.0 硬编码进
``_per_token_group_quant_8bit_raw``，导致 gfx95x 上 scale 被算成
```

```
``absmax / 224`` 而非 ``absmax / 448``，白白浪费 e4m3fn 的一个 binade 精度。
"""

import unittest

import torch

from sglang.srt.utils import is_gfx95_supported, is_hip
from sglang.test.ci.ci_register import register_amd_ci
from sglang.test.test_utils import CustomTestCase

register_amd_ci(est_time=20, suite="stage-b-test-1-gpu-small-amd-mi35x")

# e4m3fn（max 448.0）只存在于 gfx95x；gfx94x 上是 e4m3fnuz（max 224.0）。
_RUNNABLE = is_hip() and is_gfx95_supported()

# e4m3fn 可表示的最大值，也是内核在 gfx95x 上应采用的 scale/clamp 基准。
E4M3FN_MAX = 448.0


@unittest.skipUnless(_RUNNABLE, "requires HIP gfx950 (MI355X, e4m3fn fp8)")
class TestPerTokenGroupQuant8BitHipMax(CustomTestCase):
    def test_hip_fp8_scale_uses_e4m3fn_max_448(self):
        from sglang.kernels.ops.quantization.fp8_kernel import (
            _per_token_group_quant_8bit_raw,
            fp8_dtype,
            fp8_max,
        )

        # 前置检查：在 gfx95x 上模块必须解析为 e4m3fn / 448。
        self.assertIs(fp8_dtype, torch.float8_e4m3fn)
        self.assertEqual(fp8_max, E4M3FN_MAX)

        group_size = 8
        # 构造一个 absmax 恰好等于 e4m3fn 上限的 group，直接用最坏情况暴露 bug。
        x = torch.zeros((1, group_size), dtype=torch.bfloat16, device="cuda")
        x[0, 0] = E4M3FN_MAX

        # 运行真实 Triton 内核（不 mock），返回量化数据与 per-group scale。
        _, x_s = _per_token_group_quant_8bit_raw(
            x, group_size=group_size, dtype=fp8_dtype
        )

        # 正确实现 scale = absmax / 设备上限 = 448 / 448 = 1.0；
        # 旧实现 448 / 224 = 2.0，此断言会失败。
        scale = x_s.float().flatten()[0].item()
        self.assertAlmostEqual(scale, 1.0, places=5)

if __name__ == "__main__":
```

```
unittest.main()
```

评论区精华

Review 评论不多但结论明确：BowenBao 在 approval 中评价“LGTM @HaiShaw this is an important bug fix for fp8 accuracy on gfx950”，将修复定性为 gfx950 fp8 精度的重要修正；hubertlu-tw 与 HaiShaw 也均批准合并。作者在 issue 评论中补充：CI 失败 lane（AMD wait-for-stage-a-amd 超时、Base/NPU/XPU 失败）与本 ROCm 单行改动无关，且已在真实 gfx950 上本地验证测试行为（旧码失败、新码通过）。

- gfx950 fp8 精度修复的重要性 (correctness): 三位维护者（BowenBao、hubertlu-tw、HaiShaw）均批准合并。
- CI 失败是否与本次改动相关 (other): 确认失败与本 ROCm 单行改动无关，PR 获批合并；同时暴露了 MI355X 测试依赖 stage-a 串行门控的 CI 脆弱性。

风险与影响

- 风险：变更面极小：仅 _per_token_group_quant_8bit_raw 的 ROCm fp8 分支做常量替换，不影响 int8、CUDA 路径与 gfx94x。主要风险点：
  - fp8_max 的模块级解析正确性是本修复的前提，新测试只覆盖 gfx95x，gfx94x 依赖 AMD CI 套件间接回归；若某 ROCm 设备上 fp8_max 解析异常（如错误返回 0），会导致除法异常或 scale 错值，但该常量已在模块其他位置使用，风险较低。
  - 新测试门控在 MI355X 硬件上且注册到 stage-b-test-1-gpu-small-amd-mi35x 套件，若该套件因 CI 串行依赖（如本次 stage-a 超时）被跳过，回归保护会出现空窗。
  - 无运行时性能开销（编译期常量替换）。
  - 影响：用户 / 系统：MI355X (gfx950) 上启用 per-token-group fp8 量化的模型将恢复完整的 ±448 e4m3fn 动态范围，约提升 1 bit 有效精度，属于静默精度 bug 修复；MI300 (gfx94x)、CUDA 用户行为无变化。团队：为 AMD CI 新增一个 MI355X 真实硬件测试入口，后续 fp8 量化改动需保证该测试通过，同时测试可用性依赖 gfx950 硬件的持续供给。
  - 风险标记：设备相关常量修复，仅 gfx95x 有专门测试，MI355X CI 依赖硬件可用

关联脉络

- PR #30024 [AMD] perf(sgl-kernel): default block_quota=16 for MLA page_first KV gather…: 同为 AMD 侧内核 / 量化路径的性能与正确性优化（HiCache MLA gather），共享 ROCm 硬件与 sgl-kernel 优化上下文，但无直接代码依赖。
- PR #34883 [Kimi-K3] Use explicit SiTU activation for MegaMoE: 同属量化路径改动（Kimi-K3 显式激活与 mxfp4 量化），体现仓库对量化精度与 GPU 代际差异的持续关注（B300 vs MI355X）。