PR #35020 完整报告

sgl-project/sglang

[Fix] Correct dense FP8 Marlin bias ordering

合并时间：2026-08-17 11:43

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35020

执行摘要

- 一句话：修正 dense FP8 Marlin 的 bias 通道顺序
- 推荐动作：值得精读。改动虽小，却清晰揭示了一个容易踩坑的契约：Marlin 内核的 tile 布局只约束 weight 和 scales，bias 是否置换取决于上层 wrapper 的调用方式。建议关注 prepare_fp8_layer_for_marlin 与 dense 层 forward 的约定关系，以及测试中用 patch 实现 CPU-only 验证前置逻辑的手法，这种模式适合推广到其他依赖 GPU kernel 的预处理函数。

功能与动机

PR body 明确指出：Dense FP8 Marlin preparation 把 bias 置换进 Marlin tile 布局，但 dense wrapper 不在 Marlin GEMM 里传 bias，而是在 kernel 返回后加 bias，因此 post-GEMM 加法需要原始逻辑输出通道序；旧行为会产生错误的有偏投影，包括 Qwen2 QKV projection。这是面向所有带 bias 的 dense FP8 Marlin 层的内部正确性修复。

实现拆解

1. 定位缺陷：在 python/sglang/srt/layers/quantization/marlin_utils_fp8.py 的 prepare_fp8_layer_for_marlin 中，bias 分支原先执行 marlin_permute_bias(layer.bias)，把 bias 从逻辑输出通道序置换为 Marlin tile 布局序，与 dense wrapper 的 post-GEMM 加法约定冲突。
2. 核心修复：删除 bias 置换，改为 torch.nn.Parameter(layer.bias.detach(), requires_grad=False)，仅重新包装参数、保持原通道顺序；同时新增注释说明“只有 scales 需要 Marlin tile 置换，bias 必须保持逻辑顺序”。scales 路径（marlin_permute_scales + fp8_fused_exponent_bias_into_scales）保持不变。
3. 测试配套：新增 test/registered/unit/layers/quantization/test_marlin_utils_fp8.py，构造仿 dense FP8 Marlin 层的 Module，用非均匀 bias（torch.arange）验证 prepare_fp8_layer_for_marlin 处理后 bias 与原始值完全一致；测试通过 patch 掉 marlin_make_workspace 和 gptq_marlin_repack 实现 CPU-only 运行，并注册到 base-a-test-cpu CI 套件。
4. 边界确认：MoE 路径 prepare_moe_fp8_layer_for_marlin 不改动，因为 fused MoE bias 走独立 kernel 路径；本次修复仅影响 dense FP8 Marlin 层。

关键文件：

- python/sglang/srt/layers/quantization/marlin_utils_fp8.py（模块 量化层；类别 source；类型 core-logic；符号 prepare_fp8_layer_for_marlin）：核心修复点。prepare_fp8_layer_for_marlin 中 bias 分支不再调用 marlin_permute_bias，改为直接 detach 重包装回 Parameter，保持逻辑输出通道顺序，并补充注释说明 dense wrapper 与内核的契约。
- test/registered/unit/layers/quantization/test_marlin_utils_fp8.py（模块 量化测试；类别 test；类型 test-coverage；符号 TestFp8MarlinBias, test_dense_bias_remains_in_logical_output_order）：新增 CPU-only 回归测试，用非均匀 bias 验证准备阶段不改变通道顺序，并注册到 base-a-test-cpu CI 套件，弥补了无 GPU 环境下对量化预处理逻辑的覆盖空白。

关键符号：prepare_fp8_layer_for_marlin, test_dense_bias_remains_in_logical_output_order

关键源码片段

python/sglang/srt/layers/quantization/marlin_utils_fp8.py

核心修复点。prepare_fp8_layer_for_marlin 中 bias 分支不再调用 marlin_permute_bias，改为直接 detach 重包装回 Parameter，保持逻辑输出通道顺序，并补充注释说明 dense wrapper 与内核的契约。

```
# prepare_fp8_layer_for_marlin 的核心尾部：
# 在完成 weight 重打包与 scale 的 Marlin 布局置换后处理 bias。
# 关键点：dense FP8 Marlin wrapper 不会把 bias 传给 GEMM 内核，
# 而是在 kernel 返回后按原始输出通道顺序做逐通道加法，
# 因此 bias 绝不能执行 Marlin tile 置换（marlin_permute_bias）。
if hasattr(layer, "bias") and layer.bias is not None:
    assert layer.bias.shape == (part_size_n,)
    # 只重新包装为不可训练 Parameter，保持与旧逻辑一致的参数形态，
    # 同时保留逻辑输出通道顺序，供 wrapper 在 GEMM 之后使用。
    layer.bias = torch.nn.Parameter(layer.bias.detach(), requires_grad=False)
```

test/registered/unit/layers/quantization/test_marlin_utils_fp8.py

新增 CPU-only 回归测试，用非均匀 bias 验证准备阶段不改变通道顺序，并注册到 base-a-test-cpu CI 套件，弥补了无 GPU 环境下对量化预处理逻辑的覆盖空白。

```
"""Tests for FP8 Marlin utilities."""

import unittest
from unittest.mock import patch

import torch

from sglang.srt.layers.quantization import marlin_utils_fp8
from sglang.test.ci.ci_register import register_cpu_ci
from sglang.test.test_utils import CustomTestCase

# 注册到 CPU CI：约 5 秒，属于 base-a-test-cpu 套件
```

```
register_cpu_ci(est_time=5, suite="base-a-test-cpu")


class TestFp8MarlinBias(CustomTestCase):
    def test_dense_bias_remains_in_logical_output_order(self):
        size_k = 32
        size_n = 32

        # 手工构造一个仿 dense FP8 Marlin 层的 Module，
        # 用 arange 构造非均匀 bias，保证通道序被打乱时能被测试发现。
        layer = torch.nn.Module()
        layer.input_size_per_partition = size_k
        layer.output_size_per_partition = size_n
        layer.orig_dtype = torch.float16
        layer.weight_block_size = None
        layer.weight = torch.nn.Parameter(
            torch.zeros((size_k, size_n), dtype=torch.float8_e4m3fn),
            requires_grad=False,
        )
        layer.weight_scale = torch.nn.Parameter(
            torch.ones((size_n,), dtype=torch.float32), requires_grad=False
        )
        original_bias = torch.arange(size_n, dtype=torch.float16)
        layer.bias = torch.nn.Parameter(original_bias.clone(), requires_grad=False)

        # CPU-only：patch 掉需要 GPU 的重打包函数，
        # 仅让 prepare_fp8_layer_for_marlin 走到我们关心的 bias 分支。
        with (
            patch.object(
                marlin_utils_fp8,
                "marlin_make_workspace",
                return_value=torch.empty(0, dtype=torch.int32),
            ),
            patch.object(
                marlin_utils_fp8,
                "gptq_marlin_repack",
                return_value=torch.empty(0, dtype=torch.int32),
                create=True,
            ),
        ):
            marlin_utils_fp8.prepare_fp8_layer_for_marlin(layer)

        # 关键断言：预处理后 bias 必须保持原始逻辑输出通道顺序
        torch.testing.assert_close(layer.bias, original_bias)


if __name__ == "__main__":
    unittest.main(verbosity=3)
```

评论区精华

本 PR 没有任何 review 评论线程。唯一审核来自 mmangkad 的 APPROVED，其评论为 “Looks like this has been broken a while”，即该缺陷已潜伏较长时间，修复获得认可，无需进一步改动。

- 缺陷存在时长确认 (other): 无异议，直接批准合并，无需额外修改。

风险与影响

- 风险：
  - 行为变化面：修复会改变所有带 bias 的 dense FP8 Marlin 层的输出（典型如 Qwen2 QKV projection），若用户此前基于错误输出做了评测或微调，结果会变化——这是预期修正，但需在发布说明中提示。
  - 覆盖盲区：新增测试是 CPU-only，且 patch 掉了 gptq_marlin_repack 等真实重打包函数，只验证准备好了“不置换 bias”，并未在 GPU 上跑真实 Marlin kernel 与 wrapper 的端到端加 bias 路径；若未来某 dense 调用方真的把 bias 传进内核，该测试无法拦截。
  - 参数语义：detach() 会切断与加载权重的图联系，但旧代码同样以 requires_grad=False 重新包装 Parameter，语义一致，无回归风险。
  - 性能影响：推理热路径零变化，准备阶段反而少一次 bias 置换，理论上启动开销略有下降。
- 影响：
  - 用户侧：使用 FP8 权重的 Qwen2 等模型在无原生 FP8 支持的 GPU 上走 Marlin 路径时，带 bias 投影的输出从错误修正为正确，影响面覆盖所有带 bias 的 dense FP8 Marlin 层。
  - 系统侧：推理路径完全不变，无性能或显存影响；模型加载时只少一次 bias 置换操作。
  - 团队侧：新增约 5 秒的 CPU CI 用例（base-a-test-cpu 套件），为量化层准备逻辑补上了回归保护，后续类似“布局约定”改动更容易被测试捕获。
  - 风险标记：影响所有带 bias 的 dense FP8 Marlin 层，测试为 CPU-only 模拟，未覆盖端到端 GPU 路径，修复历史错误行为可能改变既有模型输出

关联脉络

- PR #34962 [Quantization] Fix GPTQ scheme attachment broken by LinearBase.scheme default: 同属 sglang/srt/layers/quantization 模块的加载正确性修复，与本 PR 一起构成量化层近期的 bugfix 脉络，且都涉及 layer 参数形态与加载语义的约定。