# PR #43979 完整报告

vllm-project/vllm

[ROCm][Bugfix] Fix GPT-OSS Quark MXFP4 MoE loading - emulation buffer not block-aligned

合并时间：2026-07-18 11:49

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43979

## 执行摘要

- 一句话：修复 Quark MXFP4 emulation MoE 维度 block 对齐
- 推荐动作：该 PR 修复了实际 bug 且重构方向正确，值得精读。review 中的共享 helper 设计思路体现了代码复用和统一管理的重要性。

## 功能与动机

Quark-quantized MXFP4/FP8 GPT-OSS loading 的 emulation 路径在 TP 下缺失 OCP block 对齐，导致 RuntimeError: The size of tensor a (1440) must match the size of tensor b (1472)。需要修复以使 emulation 后端正常工作。

## 实现拆解

1. 在 vllm/model_executor/layers/fused_moe/oracle/mxfp4.py 中导入 OCP_MX_BLOCK_SIZE，并在 mxfp4_round_up_hidden_size_and_intermediate_size 函数中增加 Mxfp4MoeBackend.EMULATION 分支，将 hidden_size 和 intermediate_size 向上 round up 到 OCP_MX_BLOCK_SIZE（32），确保 scale 缓冲区维度不会被截断。
2. 在 vllm/model_executor/layers/quantization/quark/quark_moe.py 的 maybe_roundup_sizes 方法中，移除原先针对 emulation 后端的排除条件 self.mxfp4_backend != Mxfp4MoeBackend.EMULATION，使所有非 None 后端都调用共享对齐函数，保持逻辑统一。
3. 在 tests/kernels/moe/test_ocp_mx_moe.py 中新增参数化测试 test_mxfp4_emulation_rounds_up_to_block_size，覆盖多种 TP 分片后维度组合（如 2880/720, 2880/360, 2880/2880, 90/90），验证 roundup 结果与预期一致且能被 OCP_MX_BLOCK_SIZE 整除。测试通过 @pytest.mark.skipif(not ROCM_AVAILABLE) 限定仅在 ROCm 下执行。

关键文件：

- vllm/model_executor/layers/fused_moe/oracle/mxfp4.py（模块 MoE 量化；类别 source；类型 data-contract；符号 mxfp4_round_up_hidden_size_and_intermediate_size）：核心文件：在共享对齐函数中添加 EMULATION 分支，并导入 OCP_MX_BLOCK_SIZE
- vllm/model_executor/layers/quantization/quark/quark_moe.py（模块 量化层；类别 source；类型 data-contract；符号 maybe_roundup_sizes）：移除排除 emulation 后端的条件，使 emulation 也调用共享对齐函数，确保一致性。

- tests/kernels/moe/test_ocp_mx_moe.py（模块 测试；类别 test；类型 test-coverage；符号 test_mxfp4_emulation_rounds_up_to_block_size）：新增单元测试，验证 emulation 后端维度对齐的 roundup 逻辑，覆盖关键 shard 场景。

关键符号：mxfp4_round_up_hidden_size_and_intermediate_size, maybe_roundup_sizes, test_mxfp4_emulation_rounds_up_to_block_size

## 关键源码片段

### vllm/model_executor/layers/fused_moe/oracle/mxfp4.py

核心文件：在共享对齐函数中添加 EMULATION 分支，并导入 OCP_MX_BLOCK_SIZE

```
# 在文件开头导入 OCP_MX_BLOCK_SIZE（32）
from vllm.model_executor.layers.quantization.utils.ocp_mx_utils import (
    OCP_MX_BLOCK_SIZE,
)


def mxfp4_round_up_hidden_size_and_intermediate_size(
    backend: Mxfp4MoeBackend, hidden_size: int, intermediate_size: int
) -> tuple[int, int]:
    """
    Round up hidden_size and intermediate_size based on backend requirements.
    模拟后端只需要 OCP MX block 对齐，以防止 per-block scale 缓冲区被截断。
    """
    if backend == Mxfp4MoeBackend.EMULATION:
        # Emulation has no kernel tile; it only needs OCP MX block alignment.
        intermediate_size = round_up(intermediate_size, OCP_MX_BLOCK_SIZE)
        hidden_size = round_up(hidden_size, OCP_MX_BLOCK_SIZE)
    elif backend == Mxfp4MoeBackend.DEEPGEMM_MXFP4:
        # DeepGEMM requires M/N/K alignment
        intermediate_size = round_up(intermediate_size, 128)
        hidden_size = round_up(hidden_size, 128)
    # ... 其他后端分支（Marlin, TRTLLM, FlashInfer, ROCm, CPU 等）略
    else:
        # 默认对齐到 64
        intermediate_size = round_up(intermediate_size, 64)
    return hidden_size, intermediate_size
```

### tests/kernels/moe/test_ocp_mx_moe.py

新增单元测试，验证 emulation 后端维度对齐的 roundup 逻辑，覆盖关键 shard 场景。

```
# 参数化测试，验证 emulation 后端的 round up 行为
@pytest.mark.skipif(not ROCM_AVAILABLE, reason="emulation backend targets ROCm")
@pytest.mark.parametrize(
    "hidden_size,intermediate_size,expected_hidden,expected_intermediate",
    [
        (2880, 720, 2880, 736), # GPT-OSS TP=4 shard: 720 -> round_up(720, 32)
        (2880, 360, 2880, 384), # GPT-OSS TP=8 shard: 360 -> 384
        (2880, 2880, 2880, 2880), # 已经 block-aligned，保持不变
```

```
        (90, 90, 96, 96), # 两个维度都未对齐
    ],
)
def test_mxfp4_emulation_rounds_up_to_block_size(
    hidden_size: int,
    intermediate_size: int,
    expected_hidden: int,
    expected_intermediate: int,
):
    """确保 emulation 后端将 each per-partition dim 向上对齐到 OCP_MX_BLOCK_SIZE"""
    from vllm.model_executor.layers.fused_moe.oracle.mxfp4 import (
        Mxfp4MoeBackend,
        mxfp4_round_up_hidden_size_and_intermediate_size,
    )
    from vllm.model_executor.layers.quantization.utils.ocp_mx_utils import (
        OCP_MX_BLOCK_SIZE,
    )

    rounded_hidden, rounded_intermediate = (
        mxfp4_round_up_hidden_size_and_intermediate_size(
            Mxfp4MoeBackend.EMULATION, hidden_size, intermediate_size
        )
    )

    assert rounded_hidden == expected_hidden
    assert rounded_intermediate == expected_intermediate
    # scale 缓冲区必须不产生 floor-truncation
    assert rounded_hidden % OCP_MX_BLOCK_SIZE == 0
    assert rounded_intermediate % OCP_MX_BLOCK_SIZE == 0
```

## 评论区精华

1. 对齐逻辑位置：BowenBao 和 fxmarty-amd 在 review 中指出 emulation 的对齐逻辑应统一放入共享函数，而非留在 quark_moe.py。作者采纳，为 EMULATION 添加显式分支并移除排除条件。讨论后合并。（设计）
2. 测试平台限制：tjtanaa 询问测试是否应限于 ROCm，作者确认 emulation 后端当前仅在 ROCm 使用，追加 @pytest.mark.skipif(not ROCM_AVAILABLE)。（测试）

- Emulation 对齐逻辑应统一放入共享 helper (design): 作者为 EMULATION 添加显式分支，并移除 quark_moe.py 中的排除条件，所有后端 roundup 统一在共享函数中处理。
- 测试是否应限于 ROCm 平台 (testing): 作者确认 emulation 后端目前仅用于 ROCm，因此加入 @pytest.mark.skipif(not ROCM_AVAILABLE) 限定。

## 风险与影响

- 风险：
  - 回归风险：修改了 MoE 对齐函数的控制流，但添加的是新分支，不影响现有后端，风险较低。

    - 测试覆盖局限：新测试仅限 ROCm 运行，若未来 *emulation* 后端在其他平台启用，可能遗漏。
    - 依赖风险：修复依赖 OCP_MX_BLOCK_SIZE 常量的正确性。
- 影响：
    - 用户影响：修复了 ROCm 用户使用 *--moe-backend emulation* 加载 Quark MXFP4 量化 GPT-OSS 模型时的阻断性错误。
    - 系统影响：改动局限在维度对齐路径，不涉及 kernel 或运行时。
    - 团队影响：代码结构改进有助于后续维护。
    - 风险标记：权重加载路径变更，测试限 ROCm，回归风险

# 关联脉络

- 暂无明显关联 PR