PR #28231 完整报告

sgl-project/sglang

Use Marlin for SM120 MXFP4 MoE

合并时间：2026-06-19 10:19

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28231

执行摘要

- 一句话：SM120 MXFP4 MoE 默认使用 Marlin 内核，删除 Triton 回退路径
- 推荐动作：值得精读。该 PR 展示了如何将 MXFP4 MoE 从手动编写 Triton 内核迁移到通用的 Marlin 后端，涉及数据格式转换、瓦片对齐和融合激活函数。关键设计决策包括：通过 allow_tile_padding 放松 Marlin 的形状要求，以及删除 SM120 专用的 warp 补丁。对 GPU kernel 优化和量化部署有借鉴意义。

功能与动机

SM120（桌面架构）上的 GPT-OSS MXFP4 MoE 之前使用 Triton 内核，但 Triton 内核主要面向数据中心 GPU，且社区在 Issue #19637 中提出了 SM120 性能优化需求。Marlin 后端为 MXFP4 提供了更高效的 GEMM 实现，因此决定将 Marlin 作为 SM120 MXFP4 MoE 的默认后端，并移除重复的 Triton 路径。

实现拆解

1. 数据格式转换：在 marlin_utils_fp4.py 中添加 deinterleave_moe_mxfp4_w13_for_marlin 函数，将 GPT-OSS 的交错 gate/up 行转换为 Marlin 期望的连续半部；并添加 _pad_w13 和 _pad_w2 函数，用于在 Marlin 重打包前对中间维度进行零填充以满足瓦片对齐要求。
2. 形状约束放宽：在 marlin_utils.py 中修改 check_moe_marlin_supports_layer，添加 allow_tile_padding 参数，当启用时仅要求 hidden_size % 64 == 0 和 intermediate_size % group_size == 0，不再强制 128/64 对齐。
3. 融合激活函数：在 fused_marlin_moe.py 中新增 swiglu_gpt_oss_sigmoid_alpha_contiguous 函数，实现 GPT-OSS 的 sigmoid 门控 (gate * sigmoid(gate * alpha) * (up + 1))；同时修改 fused_marlin_moe 以接收 w1_bias、w2_bias 和 gemm1_alpha 参数，并传递到底层内核。
4. 删除旧内核：移除 mxfp4_moe_sm120_triton.py 文件（约 450 行），包括其 FP4 LUT 反量化、逐槽 GEMV 和 GEMM 内核。
5. MoE 方法清理：在 mxfp4_marlin_moe.py 中移除 process_weights_after_loading 内部的 SM120 回退路径（之前跳过了 Marlin 重打包），现在始终执行 Marlin 检查并准备；并在 create_weights 中向上舍入尺寸。

6. 删除 SM120 补丁：在 mxfp4.py 中移除 _patch_sm120_mxfp4_min_warps 及相关全局变量，简化 _swizzle_mxfp4 为对所有架构使用通用布局路径。
7. 接口适配：在 moe_runner/marlin.py 中传递新的偏置参数，在 server_args.py 中调整默认后端名称。

关键文件：

- python/sglang/srt/layers/moe/fused_moe_triton/mxfp4_moe_sm120_triton.py（模块 MoE 内核；类别 source；类型 deletion；符号 _dequant_fp4_lut, _mxfp4_slot_gemv_kernel, _mxfp4_gemm_kernel, mxfp4_gemm_triton）：整个 SM120 专用 Triton MXFP4 MoE 内核被删除，因为 Marlin 后端已覆盖其功能，且该文件不再被使用。
- python/sglang/srt/layers/quantization/marlin_utils_fp4.py（模块 量化工具；类别 source；类型 core-logic；符号 deinterleave_moe_mxfp4_w13_for_marlin, _pad_w13, _pad_w2）：添加了 deinterleave_moe_mxfp4_w13_for_marlin 函数将 GPT-OSS 交错权重转换为 Marlin 格式，以及 _pad_w13 和 _pad_w2 函数处理 Marlin 张量流对齐。
- python/sglang/srt/layers/moe/fused_moe_triton/fused_marlin_moe.py（模块 融合 MoE；类别 source；类型 core-logic；符号 swiglu_gpt_oss_sigmoid_alpha_contiguous）：添加了 swiglu_gpt_oss_sigmoid_alpha_contiguous 融合激活函数以支持 GPT-OSS 的 sigmoid 门控，并修改 fused_marlin_moe 函数接收权重偏差和 gemm1_alpha 参数。
- python/sglang/srt/layers/quantization/mxfp4.py（模块 量化模块；类别 source；类型 dependency-wiring；符号 _patch_sm120_mxfp4_min_warps, _compute_num_warps_sm120_mxfp4）：移除了 _patch_sm120_mxfp4_min_warps 及相关全局变量，简化了 _swizzle_mxfp4 为公共布局路径，不再需要 SM120 特殊处理。
- python/sglang/srt/layers/quantization/mxfp4_marlin_moe.py（模块 MoE 方法；类别 source；类型 dependency-wiring）：移除了 SM120 回退路径，现在始终执行 Marlin 重打包；在 create_weights 中对尺寸进行 round_up 对齐。
- python/sglang/srt/layers/quantization/marlin_utils.py（模块 量化工具；类别 source；类型 core-logic；符号 check_moe_marlin_supports_layer）：修改 check_moe_marlin_supports_layer，添加 allow_tile_padding 参数以允许更宽松的形状检查支持 SM120 的瓦片对齐。
- python/sglang/srt/layers/moe/moe_runner/marlin.py（模块 Marlin 运行器；类别 source；类型 core-logic）：传递新的偏差参数 (w1_bias, w2_bias) 到 fused_marlin_moe 函数。
- python/sglang/srt/server_args.py（模块 服务配置；类别 source；类型 configuration）：调整默认后端名称相关配置。

关键符号：deinterleave_moe_mxfp4_w13_for_marlin, _pad_w13, _pad_w2, check_moe_marlin_supports_layer, swiglu_gpt_oss_sigmoid_alpha_contiguous, fused_marlin_moe, _swizzle_mxfp4, process_weights_after_loading

关键源码片段

python/sglang/srt/layers/quantization/marlin_utils_fp4.py

添加了 deinterleave_moe_mxfp4_w13_for_marlin 函数将 GPT-OSS 交错权重转换为 Marlin 格式，以及 _pad_w13 和 _pad_w2 函数处理 Marlin 张量流对齐。

```
def deinterleave_moe_mxfp4_w13_for_marlin(layer: torch.nn.Module) -> None:
    """Convert GPT-OSS interleaved w13 rows to Marlin's contiguous halves.

    GPT-OSS stores gate/up rows as [gate0, up0, gate1, up1, ...]. The Marlin
    fused activation consumes [all_gate_rows, all_up_rows].
    """
    w13 = layer.w13_weight.data
    w13_scale = _get_optional_param(layer, "w13_weight_scale", "w13_weight_scale_inv")
    w13_bias = _get_optional_param(layer, "w13_weight_bias", "w13_bias")

    if w13.shape[1] % 2 != 0:
        raise ValueError(f"Expected even w13 row dimension, got {w13.shape}.")

    e, n, k = w13.shape
    # Reshape to (experts, gate/up, n//2, k) then transpose gate/up dim to front
    layer.w13_weight.data = (
        w13.view(e, n // 2, 2, k).permute(0, 2, 1, 3).contiguous().view(e, n, k)
    )

    if w13_scale is not None:
        scale = w13_scale.data
        if scale.shape[1] != n:
            raise ValueError(
                f"Expected w13 scale row dimension {n}, got {scale.shape}."
            )
        w13_scale.data = (
            scale.view(e, n // 2, 2, scale.shape[-1])
            .permute(0, 2, 1, 3)
            .contiguous()
            .view(e, n, scale.shape[-1])
        )

    if w13_bias is not None:
        bias = w13_bias.data
        if bias.shape[1] != n:
            raise ValueError(f"Expected w13 bias row dimension {n}, got {bias.shape}.")
        w13_bias.data = bias.view(e, n // 2, 2).permute(0, 2, 1).contiguous().view(e, n)
```

python/sglang/srt/layers/moe/fused_moe_triton/fused_marlin_moe.py

添加了 swiglu_gpt_oss_sigmoid_alpha_contiguous 融合激活函数以支持 GPT-OSS 的 sigmoid 门控，并修改 fused_marlin_moe 函数接收权重偏差和 gemm1_alpha 参数。

```
def swiglu_gpt_oss_sigmoid_alpha_contiguous(
    output: torch.Tensor,
    input: torch.Tensor, # first half is gate, second half is up
    gemm1_alpha: float,
```

```
    gemm1_limit: float,
) -> None:
    d = input.shape[1] // 2
    gate = input[:, :d].clamp(max=gemm1_limit)
    up = input[:, d:].clamp(min=-gemm1_limit, max=gemm1_limit)
    # GPT-OSS activation: gate * sigmoid(gate * alpha) * (up + 1)
    output.copy_(gate * torch.sigmoid(gate * gemm1_alpha) * (up + 1))
```

评论区精华

审查者 b8zhong 和 Fridge003 均批准了 PR，其中 b8zhong 评论 'Nice cleanup. Thanks'，表明这是清理性变更。无实质性讨论或争议。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险在于 Marlin 内核在 SM120 上的覆盖完整性。尽管基准测试显示了性能提升，但特定模型可能触发未覆盖的路径；尤其是 allow_tile_padding 放宽了形状约束，可能导致中间维度填充后的数值精度问题。此外，删除 Triton 内核后，如果 Marlin 在某些边缘情况下失败，将没有自动回退机制。建议在更多模型和批量大小下测试。
- 影响：对用户：SM120 用户使用 GPT-OSS MXFP4 模型将自动获得 Marlin 后端的性能提升（输出吞吐量提升约 40%）。系统：减少约 450 行维护代码，简化了 MXFP4 MoE 后端选择逻辑。团队：需要确保 Marlin 后端在 SM120 上持续正确，并注意未来架构扩展时 shape 约束。
- 风险标记：核心路径变更，后端替换，缺少测试配套

关联脉络

- PR #19637 SM120 Performance Optimization Plan: 该 PR 是 SM120 性能优化计划的一部分，实现了 MXFP4 MoE 的 Marlin 后端，并删除了不再需要的 Triton 内核。