PR #34967 完整报告

sgl-project/sglang

[MoE] Add FlashInfer SM90 MXFP4 W4A8 CUTLASS MoE

合并时间：2026-09-01 11:04

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34967

执行摘要

- 一句话：SM90 新增 MXFP4 W4A8 MoE 路径，kernel 吞吐提升约 2 倍
- 推荐动作：值得精读。重点学习三点： 1) 在全局 pin 旧版 FlashInfer 时如何以 opt-in 方式安全接入新内核并即时失败提示； 2) 用 preserve_expert_range + last_real / k_real 处理对齐 padding 与预取整尾列，保证 Humming residual 数值稳定； 3) 对上游 ABI（routed-row vs per-local-expert）变更的取舍与版本门控设计。

功能与动机

SGLang 已有 FlashInfer SM90 MXFP4 W4A16 CUTLASS MoE 路径（由 #24816 基于 FlashInfer #3084 引入）。FlashInfer #3738 增加了另一条 Hopper 路径：动态把激活量化到 FP8，并运行 MXFP4 权重 x FP8 激活的 MoE GEMM，采用 Humming 风格 pre-MMA E8M0 scale fusion。本 PR 的目标是在不改默认行为的前提下，把该路径作为显式 opt-in 接入，并且只使用 FlashInfer #4431 修正后的 per-local-expert residual ABI，避免依赖被 #4411 回退的 routed-row 旧 ABI。

实现拆解

1. 运行时选择与兼容层：python/sglang/srt/server_args.py 为 flashinfer_mxfp4_moe_precision 增加 fp8 枚举；python/sglang/srt/layers/quantization/mxfp4.py 在 SM90 分支按 precision == "fp8" 设置 _use_sm90_humming，SM100/SM120 分支不报错（fp8 在 Blackwell 上惰性）；依赖版本用 check_pkg_version_at_least("flashinfer_python", "0.6.18") 门控，use_wfp4afp8_humming 关键字只在启用时传入，保持 0.6.17 兼容。
2. GPT-OSS 权重预处理：_process_weights_for_sm90_cutlass 中 _stack_up_gate_w13 / _pad_w2_3d 新增 preserve_expert_range 参数，用同 expert 已有 scale 填充对齐 padding，并按 last_real / k_real 截断未写入的预取整尾列，避免 _UE8M0_ONE 填充污染 Humming 的 min/max exponent 范围；python/sglang/srt/layers/moe/fused_moe_triton/layer.py 新增 hidden_size_unpadded 记录预取整值。
3. DeepSeek-V4 权重预处理：python/sglang/srt/layers/quantization/mxfp4_flashinfer_cutlass_moe.py 的 process_weights_after_loading 增加 Humming 分支，对 w13/w2 分别调用 preprocess_moe_weights_for_sm90_mixed_gemm_humming，注册 w13_humming_residual_scale、w2_humming_residual_scale（均乘 64 补偿 2^6）与 humming_fc2_act_scale。

4. 内核调用装配：python/sglang/srt/layers/moe/moe_runner/flashinfer_cutlass.py 的 FlashInferCutlassMxfp4MoeQuantInfo 增加三件套字段；fused_experts_none_to_flashinfer_mxfp4 构造五个 quant-scale 槽位（FC1 折叠权重块 scale、FC1 residual、FC2 激活 scale、FC2 折叠权重块 scale、FC2 residual），校验完整性、与 MXFP8 互斥，并仅在启用时传 use_wfp4afp8_humming=True。
5. 测试、基准与文档：test/registered/unit/layers/quantization/test_mxfp4_sm90_cutlass.py 新增预处理对比、padding 不变性、prerounded tail 排除、DSV4 预处理、fail-fast 等用例（18 passed + 1 skipped）；test/manual/layers/moe/bench_mxfp4_sm90_kernels.py 增加 W4A8 对比；server_arguments.mdx、DeepSeek-V4 cookbook 与 H200 配置片段更新并标记 in-progress。

关键文件：

- python/sglang/srt/layers/quantization/mxfp4.py（模块 量化方法；类别 source；类型 core-logic；符号 _stack_up_gate_w13, _pad_w2_3d）：核心量化方法：SM90 路径按 precision 选择 Humming，_stack_up_gate_w13 / _pad_w2_3d 增加 preserve_expert_range 与预取整尾列排除逻辑，是正确性关键。
- python/sglang/srt/layers/quantization/mxfp4_flashinfer_cutlass_moe.py（模块 后端适配；类别 source；类型 dependency-wiring）：DeepSeek-V4 的 FlashInfer CUTLASS 后端：在 post-load 阶段接入 Humming 预处理，并注册 residual 与 FC2 激活 scale。
- python/sglang/srt/layers/moe/moe_runner/flashinfer_cutlass.py（模块 MoE 运行器；类别 source；类型 core-logic）：内核调用装配点：构造 Humming 五个 quant-scale 槽位，校验完整性，并按需传 use_wfp4afp8_humming，是 0.6.17 兼容性的关键。
- test/registered/unit/layers/quantization/test_mxfp4_sm90_cutlass.py（模块 单元测试；类别 test；类型 test-coverage；符号 _build_method, test_process_weights_humming_matches_flashinfer_direct, test_humming_padding_preserves_per_expert_residual, _build_prerounded_case）：SM90 MXFP4 注册测试套件：新增 Humming 预处理对比、padding 不变性、prerounded tail 排除、DSV4 预处理、fail-fast 等 6 个核心场景。
- test/manual/layers/moe/bench_mxfp4_sm90_kernels.py（模块 基准脚本；类别 test；类型 test-coverage；符号 build_flashinfer_humming_inputs）：手动三路基准：SGLang Marlin W4A16、FlashInfer CUTLASS W4A16、FlashInfer CUTLASS W4A8，量化性能收益的数据来源。
- python/sglang/srt/server_args.py（模块 服务配置；类别 source；类型 configuration）：暴露用户入口：为 flashinfer_mxfp4_moe_precision 增加 fp8 枚举与说明，是 opt-in 运行时开关。
- python/sglang/srt/layers/moe/fused_moe_triton/layer.py（模块 模型层；类别 source；类型 core-logic）：记录 hidden_size_unpadded，为 SM90 后处理排除预取整尾列提供原始 K，是正确性修复的必要配套。
- docs/src/snippets/configs/deepseek-ai/deepseek-v4.jsx（模块 部署文档；类别 other；类型 configuration）：DeepSeek-V4 H200 低延迟配置由 Marlin W4A16 切换到 FlashInfer W4A8，标记 in-progress 并记录回退方式。

关键符号：_process_weights_for_sm90_cutlass, _stack_up_gate_w13, _pad_w2_3d, process_weights_after_loading, fused_experts_none_to_flashinfer_mxfp4, build_flashinfer_humming_inputs

关键源码片段

python/sglang/srt/layers/quantization/mxfp4.py

核心量化方法：SM90 路径按 precision 选择 Humming，_stack_up_gate_w13 / _pad_w2_3d 增加 preserve_expert_range 与预取整尾列排除逻辑，是正确性关键。

```
# 关键：FusedMoE 可能在 create_weights 之前就把 hidden 向上取整（GPT-OSS 2880 -> 3072），
# 因此 K_un 是取整后的值而非 checkpoint 原始 K。loader 从不写尾部列，
# 尾部 scale 列保留 _UE8M0_ONE（2^0）填充，会抬高 Humming 的 per-expert 取值范围。
# 这里只拷贝 checkpoint 真实 K 对应的列，剩余部分交给 preserve_expert_range 填充。
K_real = min(getattr(self, "_unpadded_hidden", None) or K_un, K_un)
w13_scale_real = -(-K_real // sf_block_size) # 向上取整：不满 32 的尾组仍是真实数据


def _stack_up_gate_w13(
    unpadded_w13, last_pad, last_un, preserve_expert_range=False, last_real=None
):
    """将 GPT-OSS 交错布局 [g0, u0, g1, u1, ...] 解交错并 padding 为 [up; gate]。

    last_real 用于截断拷贝，只复制 checkpoint 真实列数。
    """
    if _interleaved:
        gate_rows = unpadded_w13[:, 0::2, :]
        up_rows = unpadded_w13[:, 1::2, :]
    else:
        gate_rows = unpadded_w13[:, :N_un, :]
        up_rows = unpadded_w13[:, N_un:, :]
    out = torch.zeros(E, 2 * N_pad, last_pad, dtype=unpadded_w13.dtype, device=device)
    if preserve_expert_range:
        # Humming 从一个 expert 的 E8M0 min/max 推导 residual；
        # 用已有真实 scale 填充 padding，避免 padding 值改变取值范围。
        out.copy_(unpadded_w13[:, :1, :1])
    copy_un = (
        min(last_real, last_un)
        if (preserve_expert_range and last_real is not None)
        else last_un
    )
    out[:, :N_un, :copy_un] = up_rows[:, :, :copy_un]
    out[:, N_pad : N_pad + N_un, :copy_un] = gate_rows[:, :, :copy_un]
    return out


def _pad_w2_3d(unpadded, last_pad, last_un, preserve_expert_range=False, k_real=None):
```

```
    """w2 无需解交错，仅在 K 维从 K_un 对齐到 K_pad。"""
    out = torch.zeros(E, K_pad, last_pad, dtype=unpadded.dtype, device=device)
    if preserve_expert_range:
        out.copy_(unpadded[:, :1, :1])
    k_copy = (
        min(k_real, K_un) if (preserve_expert_range and k_real is not None) else K_un
    )
    out[:, :k_copy, :last_un] = unpadded[:, :k_copy, :]
    return out
```

python/sglang/srt/layers/quantization/mxfp4_flashinfer_cutlass_moe.py

DeepSeek-V4 的 FlashInfer CUTLASS 后端：在 post-load 阶段接入 Humming 预处理，并注册 residual 与 FC2 激活 scale。

```
if self._use_sm90_humming:
    # DSv4 权重保持 native E8M0，直接调用 Humming 预处理；
    # 它同时完成权重 / scale 折叠，并返回 per-local-expert residual。
    w13_il, w13_s_il, w13_residual = preprocess_moe_weights_for_sm90_mixed_gemm_humming(
        layer.w13_weight.data.view(torch.uint8).contiguous(), w13_scale_u8
    )
    w2_il, w2_s_il, w2_residual = preprocess_moe_weights_for_sm90_mixed_gemm_humming(
        layer.w2_weight.data.view(torch.uint8).contiguous(), w2_scale_u8
    )
    # 乘以 64（2^6）补偿 FP4 -> FP8 的指数偏移，以 FP32 存每个本地 expert 一份；
    # FlashInfer 内部会基于 EP 拓扑把全局 expert 映射到对应 residual。
    layer.w13_humming_residual_scale = Parameter(
        (w13_residual * 64.0).contiguous(), requires_grad=False
    )
    layer.w2_humming_residual_scale = Parameter(
        (w2_residual * 64.0).contiguous(), requires_grad=False
    )
    layer.humming_fc2_act_scale = Parameter(
        torch.ones((), dtype=torch.float32, device=w13_scale_u8.device),
        requires_grad=False,
    )
```

python/sglang/srt/layers/moe/moe_runner/flashinfer_cutlass.py

内核调用装配点：构造 Humming 五个 quant-scale 槽位，校验完整性，并按需传 use_wfp4afp8_humming，是 0.6.17 兼容性的关键。

```
# 五个 Humming quant-scale 槽位（FlashInfer #4431 修正契约）：
# [FC1 折叠权重块 scale, FC1 每本地 expert residual*64, FC2 激活保留 scale,
# FC2 折叠权重块 scale, FC2 每本地 expert residual*64]
humming_scales = (
    w13_humming_residual_scale,
    w2_humming_residual_scale,
    humming_fc2_act_scale,
)
use_wfp4afp8_humming = any(scale is not None for scale in humming_scales)
```

```
# 三件套必须齐全，否则静默走错路径会得到错误数值。
if use_wfp4afp8_humming and not all(scale is not None for scale in humming_scales):
    raise ValueError(
        "SM90 Humming MXFP4 MoE requires both expert residual scales "
        "and the FC2 activation scale."
    )
if use_wfp4afp8_humming and use_mxfp8_act_scaling:
    raise ValueError("SM90 Humming and SM120 MXFP8 scaling are mutually exclusive.")

# ... 在 SM120 MXFP8 分支之后、普通 W4A16 分支之前 ...
elif use_wfp4afp8_humming:
    quant_scales = [
        quant_info.w13_weight_scale.view(torch.int32),
        w13_humming_residual_scale,
        humming_fc2_act_scale,
        quant_info.w2_weight_scale.view(torch.int32),
        w2_humming_residual_scale,
    ]

# FlashInfer 0.6.17 曾回退掉 Humming API，旧路径不能传新关键字，
# 否则会破坏当前 pin 版本下 W4A16 / MXFP8 路径的兼容性。
humming_kwargs = {"use_wfp4afp8_humming": True} if use_wfp4afp8_humming else {}
# ... flashinfer_cutlass_fused_moe(..., **humming_kwargs)
```

评论区精华

kaixih：K_un 真的是 unpadded 吗？FusedMoE 已经提前把 GPT-OSS hidden 从 2880 round 到 3072，尾部 E8M0 scale 保持 127（_UE8M0_ONE）填充。把那些列纳入 Humming 的 per-expert min/max 会改变真实权重。

yuan-luo：同意。create_weights 运行时 hidden 总是 3072，preserve_expert_range 当时保护的是空区域；已通过记录 hidden_size_unpadded 修复，并补充生产 shape 的专项测试。

kaixih：版本比较建议直接用 Version.parse(flashinfer_version) >= Version.parse("0.6.18")，release[:3] 会放行 0.6.18rc*。

Fridge003：这里只需查属性 preprocess_moe_weights_for_sm90_mixed_gemm_humming，不需要版本条件；升级后（#36954）可移除防御性检查。

yuan-luo：已改用现有 check_pkg_version_at_least 统一门控；#36954 落地后移除防御性检查与相关 import。

Fridge003：SM120 的 fp8 guard（line 42-46）应移除——Blackwell 本来就跑 MXFP8 activation，报错会破坏配置可移植性。

guzekai01：基准表里 Humming W4A8 可能被误读为 upstream inclusionAI/Humming 后端，实际是 FlashInfer 的 use_wfp4afp8_humming 参数。

yuan-luo：命名已更新为 Cutlass。

- FusedMoE 预取整 hidden 导致 Humming residual 范围被尾部填充污染 (correctness): 新增 hidden_size_unpadded 记录，post-load 处理器按 last_real / k_real 截断拷贝并补测试 test_humming_range_ignores_prerounded_hidden_tail。
- FlashInfer 版本门控需识别 prerelease (correctness): 已解决：替换手写门控，删除 try/except 与 hasattr 脚手架。
- SM100/SM120 上 fp8 选项应 inert 而非报错 (design): 已解决：四个 guard 全部移除，fp8 在 Blackwell 上被文档化为 inert。
- 防御性版本检查是否应在 FlashInfer 升级后移除 (design): 已解决：FlashInfer 0.6.18 升级后移除防御性检查，保留属性探测兜底。
- 基准表命名歧义（Humming vs inclusionAI backend） (style): 已解决：更新为 Cutlass。

风险与影响

- 风险:
  1. 依赖与发布风险：SGLang 全局仍 pin FlashInfer 0.6.17，选择 fp8 会立即失败；性能收益需要 #36954 协调升级 FlashInfer Python 包、cubin 与 JIT-cache 后才能生产启用。
  2. 正确性风险：pre-round hidden tail 曾导致 Humming residual 被 _UE8M0_ONE 填充污染（kaixih 发现，已修复并加测试）；padding 填充依赖"同 expert 已有 scale"，若某 expert 的 scale 全 0 或特殊值，preserve_expert_range 填充行为可能偏离预期。
  3. 兼容性风险：use_wfp4afp8_humming 只在 Humming 路径传入，0.6.17 下现有 W4A16/MXFP8 路径不受影响；FlashInferCutlassMxfp4MoeQuantInfo 的三件套完整性校验可防止半套字段静默进入错误路径。
  4. 回归风险：W4A16 路径行为不变，但 padding 辅助函数签名与调用点均有调整；SM120 测试被同步修改以覆盖新的 server args 上下文。
  5. 运维风险：DeepSeek-V4 文档配置标记 in-progress，在重新 benchmark 完成前可能给用户带来性能预期偏差。 - 影响：用户侧：H100/H200 上 GPT-OSS 风格模型与 DeepSeek-V4 FP4 部署可获得 W4A8 选项；GSM8K 200 例精度（0.985）与 W4A16 持平，端到端吞吐从 611.8 提升到 683.6 token/s（+11.7%），kernel 级在 token >= 2048 时提速约 2 倍。系统侧：TP/EP 下 residual 按本地 expert 索引存储，FlashInfer 内部完成全局到本地 expert 映射，避免每 forward 构造 routed-row 张量；SM100/SM120 上 fp8 标志被文档化为惰性，一份配置可跨硬件迁移。团队侧：后续需要维护 FlashInfer 版本契约，并跟进 #3738 -> #4411 -> #4431 的 ABI 演进。 - 风险标记：依赖上游未升级，核心推理路径变更，版本门控陷阱，精度依赖数值填充策略

关联脉络

- PR #24816 Add FlashInfer SM90 MXFP4 W4A16 CUTLASS MoE: 本 PR 的直接前身：引入 SM90 MXFP4 W4A16 路径，本 PR 在其基础上扩展 W4A8 Humming 路径，并复用其 GSM8K 评测口径。
- PR #36954 Upgrade FlashInfer to 0.6.18: Fridge003 在评论中确认该 PR 是 FlashInfer 0.6.18 升级点；本 PR 的 fp8 路径依赖该升级才能真正启用，mmangkad 也借此要求移除

防御性版本检查。