PR #33148 完整报告

sgl-project/sglang

[Quantization] Route per-tensor FP8 checkpoints to FlashInfer on SM90

合并时间：2026-08-06 08:54

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33148

执行摘要

- 一句话：SM90 per-tensor FP8 改走 FlashInfer，prefill 提速 15.6%
- 推荐动作：值得精读，且改动集中（约 40 行、2 个文件），阅读成本很低。三个值得学习的点：(1) 用数据而不是架构承诺做路由决策——SM90 收益、SM89 回退，作者测量后把 Ada 排除出谓词；(2) 加载期与运行期条件统一的思路，消除静默降级路径；(3) 基准方法论：CUPTI 计时、冷 L2（L40 的 96MB L2 能装下 31MB 权重）、双机复测、区分 CUTLASS 3.x 与 TRT-LLM 手调 tile 表来解释架构差异。阅读时注意 PR body 中的基准数据比合并后的代码更能说明设计动机。

功能与动机

Issue #32993 指出：per-tensor FP8 检查点只带一个标量 weight scale 和一个标量 activation scale，但在 SM89/SM90 上被送进只接受 per-token 激活 scale 与 per-channel 权重 scale 的 AOT CUTLASS rowwise GEMM，标量被广播后还要付出 rowwise epilogue 的代价。FlashInfer 的 per-tensor 路径 apply_fp8_linear_bmm_flashinfer 已存在，但 flashinfer_bmm_fp8 custom op 定义在 is_blackwell_supported() 块内，Hopper 上符号不存在，翻转调用方 flag 只会 NameError。Issue 要求自动路由、无需用户传 flag，动机是性能（cuBLAS tile 配置比现有 GEMM 多）与一致性（让 SM89/SM90 与 SM100/SM120 走同一路径）。作者实测后仅保留 SM90 与 Blackwell：SM89（L40）端到端反而回退，因此 Ada 被排除。

实现拆解

1. fp8_utils.py：新增架构谓词并拆出 custom op。新增 flashinfer_per_tensor_fp8_supported()（@lru_cache(maxsize=1)），返回 is_flashinfer_available() and (is_sm90_supported() or is_sm100_supported() or is_sm120_supported())；把 flashinfer_bmm_fp8 custom op 及其 from flashinfer import bmm_fp8 导入从 if is_blackwell_supported() and is_flashinfer_available(): 块移到新谓词保护的独立块。Blackwell 专属的 SfLayout、mm_mxfp8、mxfp8_quantize、gemm_fp8_nt_groupwise 保持原门控不变。同时把 apply_fp8_linear_bmm_flashinfer 的 docstring 从“(SM100/SM120 Blackwell)”改为“(SM90 and newer)”，让 Hopper 上符号不再缺失且不新增导入面。
2. modelopt_quant.py：启用新谓词并新增 per-layer 判定。ModelOptFp8LinearMethod.__init__ 中 enable_flashinfer_bmm 从

(is_sm100_supported() or is_sm120_supported()) and is_flashinfer_available() 改为 flashinfer_per_tensor_fp8_supported()；新增 _can_use_flashinfer_bmm(layer)：先要求 enable 且 layer.input_scale is not None，再检查 k % 16 == 0 and n % 16 == 0。16 元素对齐下限直接借用 apply_fp8_linear 对 CUTLASS 内核的既有约束——实测 SM90 上 cuBLAS FP8 只需 K 被 4 整除、N 为偶数，因此该界限既覆盖更严的 CUTLASS 约束又对 cuBLAS 留有余量，避免 bmm_fp8 遇未对齐形状直接抛 CUBLAS_STATUS（它没有 fallback）。

3. modelopt_quant.py：加载期与运行期 gate 统一。process_weights_after_loading 新增 layer.use_flashinfer_bmm = self._can_use_flashinfer_bmm(layer)，把 convert_to_channelwise（标量 weight scale 广播成 per-channel）的门控从 not self.enable_flashinfer_bmm 改为 not layer.use_flashinfer_bmm；apply 运行期分支也从 self.enable_flashinfer_bmm and layer.input_scale is not None 改为读 layer.use_flashinfer_bmm。改动前加载期与运行期是两个不同条件（运行期多一个 input_scale is not None 项），一旦该项为假，层会保留标量 scale、在 apply_fp8_linear 的 weight_scale.numel() == weight.shape[1] 检查失败后无声落入未融合 dequant 路径；现在 scale 布局与分发目标是同一个决策。同时清理了不再使用的 is_flashinfer_available、is_sm100_supported 导入。
4. 验证与裁剪。作者在 H100（SM90）与 L40（SM89）分别做 CUPTI 计时（冷 L2、每 arm 双跑确认噪声），SM90 prefill 密集端到端 input tok/s +15.6%、TTFT -15.2%，decode m=1 合计约慢 2%（112.26us vs 114.31us）；SM89 端到端 -5.4%，故谓词排除 Ada；GSM8K 精度无实质变化。review 中维护者要求删除随 PR 新增的单元测试 test_fp8_per_tensor_flashinfer.py 与 benchmark 文件，最终合并仅含 2 个源码文件；既有 test/registered/quant/test_modelopt_fp8.py（1-gpu-large，H100）在 CI 中覆盖该路径，review 期间另 rerun 了 4 个相关 FP8 量化 / 扩散测试均通过。

关键文件：

- python/sglang/srt/layers/quantization/modelopt_quant.py（模块 量化分发；类别 source；类型 data-contract；符号 _can_use_flashinfer_bmm, ModelOptFp8LinearMethod, flashinfer_per_tensor_fp8_supported）：变更的核心消费端：新增 _can_use_flashinfer_bmm，并在 process_weights_after_loading 中把 scale 布局（是否 convert_to_channelwise）与运行期分发（是否走 apply_fp8_linear_bmm_flashinfer）统一为每层一次的 use_flashinfer_bmm 决策，消除加载 / 运行 gate 分歧导致的静默降级风险；同时把 enable 判断切换到新的 SM90 谓词。
- python/sglang/srt/layers/quantization/fp8_utils.py（模块 量化工具；类别 source；类型 core-logic；符号 flashinfer_per_tensor_fp8_supported, flashinfer_bmm_fp8）：变更入口：新增 flashinfer_per_tensor_fp8_supported() 谓词（lru_cache），把 flashinfer_bmm_fp8 custom op 从 Blackwell-only 块拆出到 SM90+ 守卫下，使 Hopper 上该符号不再缺失；同时明确 SM89 不在谓词内。

关键符号：flashinfer_per_tensor_fp8_supported, flashinfer_bmm_fp8, _can_use_flashinfer_bmm, process_weights_after_loading, apply

关键源码片段

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

变更的核心消费端：新增 _can_use_flashinfer_bmm，并在 process_weights_after_loading 中把 scale 布局（是否 convert_to_channelwise）与运行期分发（是否走 apply_fp8_linear_bmm_flashinfer）统一为每层一次的 use_flashinfer_bmm 决策，消除加载 / 运行 gate 分歧导致的静默降级风险；同时把 enable 判断切换到新的 SM90 谓词。

```
class ModelOptFp8LinearMethod(LinearMethodBase):
    """ModelOpt 静态 FP8 的线性层实现：per-tensor 检查点只带标量 scale。"""

    def __init__(self, quant_config: ModelOptFp8Config):
        super().__init__()
        self.quant_config = quant_config
        self.cutlass_fp8_supported = cutlass_fp8_supported()
        # 原实现只对 SM100/SM120 生效；现在 SM90（H100/H200）也进入
        # FlashInfer cuBLAS per-tensor 路径，SM89（Ada）实测回退所以不在谓词里
        self.enable_flashinfer_bmm = flashinfer_per_tensor_fp8_supported()
        self.use_marlin = False
        if is_cuda():
            self.use_marlin = (
                envs.SGLANG_FORCE_FP8_MARLIN.get() or can_auto_enable_marlin_fp8()
            )

    def _can_use_flashinfer_bmm(self, layer: torch.nn.Module) -> bool:
        # 需要 checkpoint 的静态激活 scale；K/N 对齐要求沿用 apply_fp8_linear
        # 对 CUTLASS 内核的 16 元素下限。SM90 上 cuBLAS FP8 实测只要
        # K % 4、N % 2，这个界限更保守并给各架构留出余量，因为 bmm_fp8
        # 对未对齐形状没有 fallback，会直接抛 CUBLAS_STATUS
        if not self.enable_flashinfer_bmm or layer.input_scale is None:
            return False
        k, n = layer.weight.shape
        return k % 16 == 0 and n % 16 == 0

    def process_weights_after_loading(self, layer: torch.nn.Module) -> None:
        max_w_scale, quantized_weight = requantize_with_max_scale(
            layer.weight, layer.weight_scale, layer.logical_widths
        )
        layer.weight = Parameter(quantized_weight.t(), requires_grad=False)
        # 每层在加载期只决策一次：走 FlashInfer 就保留标量 weight scale，
        # 否则马上转成 channelwise 布局，保证运行期 apply 的判断与这里的
        # scale 布局永远一致（旧实现两处条件不同，可能静默降级）
        layer.use_flashinfer_bmm = self._can_use_flashinfer_bmm(layer)
        if self.cutlass_fp8_supported and not layer.use_flashinfer_bmm:
            max_w_scale = convert_to_channelwise(max_w_scale, layer.logical_widths)
        layer.weight_scale = Parameter(max_w_scale, requires_grad=False)
        layer.input_scale = Parameter(layer.input_scale.max(), requires_grad=False)
        if self.use_marlin:
            prepare_fp8_layer_for_marlin(layer)
            del layer.input_scale
```

```
    def apply(
        self,
        layer: torch.nn.Module,
        x: torch.Tensor,
        bias: Optional[torch.Tensor] = None,
    ) -> torch.Tensor:
        """运行期分发与加载期布局决策共用同一个 per-layer 标志。"""
        if self.use_marlin:
            return torch.ops.sglang.apply_fp8_marlin_linear(
                input=x,
                weight=layer.weight,
                weight_scale=layer.weight_scale,
                workspace=layer.workspace,
                size_n=layer.output_size_per_partition,
                size_k=layer.input_size_per_partition,
                bias=bias,
            )
        if layer.use_flashinfer_bmm:
            # 标量 scale 原样传给 per-tensor cuBLAS bmm，不再做广播
            return apply_fp8_linear_bmm_flashinfer(
                input=x,
                weight=layer.weight,
                weight_scale=layer.weight_scale,
                input_scale=layer.input_scale,
                bias=bias,
            )
        # 回退到 AOT CUTLASS rowwise（SM89、未对齐形状、Marlin 之外的默认路径）
        return apply_fp8_linear(
            input=x,
            weight=layer.weight,
            weight_scale=layer.weight_scale,
            input_scale=layer.input_scale,
            bias=bias,
            cutlass_fp8_supported=self.cutlass_fp8_supported,
        )
```

python/sglang/srt/layers/quantization/fp8_utils.py

变更入口：新增 flashinfer_per_tensor_fp8_supported() 谓词（lru_cache），把 flashinfer_bmm_fp8 custom op 从 Blackwell-only 块拆出到 SM90+ 守卫下，使 Hopper 上该符号不再缺失；同时明确 SM89 不在谓词内。

```
@lru_cache(maxsize=1)
def flashinfer_per_tensor_fp8_supported() -> bool:
    # SM90（H100/H200）与 Blackwell SM100/SM120 都走 FlashInfer 的
    # cuBLAS per-tensor FP8 路径；SM89（Ada/L40）实测端到端回退 5.4%，
    # 所以继续留在 AOT CUTLASS rowwise，不放进这个谓词
    return is_flashinfer_available() and (
        is_sm90_supported() or is_sm100_supported() or is_sm120_supported()
    )
```

```
if flashinfer_per_tensor_fp8_supported():
    from flashinfer import bmm_fp8 as _raw_flashinfer_bmm_fp8

    # 用 custom op 包一层，让 torch.compile 不会 trace 进 flashinfer 的
    # JIT 编译代码（pathlib/cubin_loader），fake_impl 只给出输出形状
    @register_custom_op(
        op_name="flashinfer_bmm_fp8",
        mutates_args=[],
        fake_impl=lambda q_input, weight, x_scale, weight_scale, out_dtype: (
            q_input.new_empty((q_input.shape[0], weight.shape[1]), dtype=out_dtype)
        ),
    )
    def flashinfer_bmm_fp8(
        q_input: torch.Tensor, # [M, K] fp8 e4m3
        weight: torch.Tensor, # [K, N] fp8 e4m3, column-major
        x_scale: torch.Tensor, # per-tensor 标量
        weight_scale: torch.Tensor, # per-tensor 标量
        out_dtype: torch.dtype,
    ) -> torch.Tensor:
        m, n = q_input.shape[0], weight.shape[1]
        # 把 per-tensor 问题喂给 bmm，batch 维取 1，靠 cuBLAS 后端拿更密的
        # tile 配置；标量原样传入，不做任何广播
        return _raw_flashinfer_bmm_fp8(
            q_input.unsqueeze(0),
            weight.unsqueeze(0),
            x_scale.reshape(1),
            weight_scale.reshape(1),
            out_dtype,
            backend="cublas",
        ).view(m, n)
```

评论区精华

b8zhong 在 modelopt_quant.py:565 追问 CUTLASS 路径的 N/K 限制（“I feel it might be even more restrictive”），adityakamat24 回答：CUTLASS 要求 128-bit 对齐（fp8 的 A/B 需 K % 16、bf16 输出需 N % 8），既有 gate 两者都查 % 16、不满足才退回 triton；实测 cuBLAS 在 SM90 更宽松（K % 4、N % 2），所以 % 16 只是借用既有下限，并反问是否要放宽。b8zhong 要求 “Delete all AI comments that only explain the code”，最终长 docstring 被移除；对新单测文件回复 “Delete this”、对 benchmark 文件回复 “We can delete it. The benchmark looks fine”，最终 PR 只剩 2 个源码文件。批准时 b8zhong 评价：“This makes sense. Thanks for the perf measuring! The gap at small M is not a big deal.”——接受 decode 小 M 的少量回退换取 prefill 的大幅提升。

- CUTLASS 路径的 N/K 对齐限制对比 (design): 维持 % 16 界限：既能覆盖更严的 CUTLASS 约束，又对 cuBLAS 的架构差异留有余量；真实 transformer 形状均满足。
- 删除新增单测与 benchmark 文件 (testing): 最终合并仅包含 2 个源码文件；PR body 中描述的单测（H100 上 12 passed）与 benchmark 未进入 main，路径回归保障依赖既有

test_modelopt_fp8.py。

- 删除 AI 生成的解释性注释 (style): 最终代码中 _can_use_flashinfer_bmm 不再保留长 docstring，仅保留必要判断逻辑。
- decode 小 M 性能回退是否可接受 (performance): 接受 decode 小 M 回退，换取 prefill 体量的大幅收益；SM90 路由保留。

风险与影响

- 风险:
  - 后端对齐与版本依赖：_can_use_flashinfer_bmm 用 K % 16、N % 16 作界限，实测 SM90 cuBLAS 下限为 K % 4、N % 2，留有余量；但 bmm_fp8 对未对齐形状没有 fallback，直接抛 CUBLAS_STATUS，若未来某架构或 flashinfer 新版本收紧对齐要求会出现新报错。路由可用性还依赖 is_flashinfer_available() 与 flashinfer 版本（PR 确认 0.6.14/0.6.15.post1 的 cuBLAS 后端覆盖 SM89~SM121），老版本若 SM90 的 bmm_fp8 缺失会在 import 阶段失败。
  - 数值路径切换：换 GEMM 后端改变浮点累加顺序，GSM8K 显示无实质退化（bf16）；fp16 数值一致性在 PR body 中声称由单测覆盖，但该单测文件已在 review 中删除，合并后无对应自动化保障。
  - 性能回退：decode m=1 总耗时约慢 2%（TP4 四投影合计 112.26us vs 114.31us），对纯 decode 服务有小幅负面影响；SM89 被排除正是因为端到端回退 5.4%，未来若有人把 SM89 加回谓词且未复测会产生回归。
  - 回归覆盖依赖既有测试：合并不含新增测试，回归保障主要靠 test/registered/quant/test_modelopt_fp8.py（H100）；review 期间 4 个 rerun 任务全部通过。
  - 提交历史噪音：6 个 commit 中 5 个是 main 合并与冲突解决（fp8_utils.py 两次冲突），说明该文件在合并窗口期被并行改动，最终稳定性依赖反复同步 main。
- 影响:
  - 用户 / 服务侧：所有使用 per-tensor FP8 检查点（如 nvidia/Llama-3.1-8B-Instruct-FP8）且运行在 SM90（H100/H200）的部署，加载与运行行为自动改变且无需任何参数：prefill 密集型服务 input tok/s +15.6%、median TTFT -15.2%；m >= 128 的批处理全面提速（1.02x~1.79x）；m=1 纯 decode 约慢 2%。SM89/Ada 与 K 或 N 非 16 倍数的形状保持原 CUTLASS rowwise 路径，行为不变。
  - 一致性：SM90 的 per-tensor checkpoint 现在与 SM100/SM120 走同一条 FlashInfer per-tensor 路径，回应 issue 的一致性诉求。
  - 工程侧：量化层的 GEMM 选择与 scale 布局成为单点决策（layer.use_flashinfer_bmm），消除加载 / 运行 gate 不一致造成的潜在静默降级；fp8_utils.py 的 Blackwell 专属块变薄，未来扩展架构只需改谓词。
  - 团队 /CI：合并后无新增测试文件，需依赖既有 H100 覆盖；维护者与作者在 CI 失败归属上达成一致（非本 PR 引起）。

- 风险标记：核心 GEMM 分发路径变更，合并后无新增测试覆盖，依赖 flashinfer 对齐限制与版本，decode 小 M 约 2% 回退

关联脉络

- PR #33469 kernels: scalar scale A support for fp8_gemm: 同为 per-tensor FP8 场景：给 AOT CUTLASS rowwise fp8_gemm 内核补标量 scale A 支持；本 PR 在 SM90 上让 per-tensor checkpoint 绕开该内核走 FlashInfer，两者共同完善 per-tensor FP8 的 GEMM 后端能力。
- PR #33474 Select DeepGEMM standard layouts by memory budget: 同在量化 GEMM 后端选择线上：按内存预算自动选择 DeepGEMM 布局，与本 PR 的按架构 / 形状自动路由 FP8 GEMM 同属 quant 层后端决策自动化演进。
- PR #33621 Pin online NVFP4 4over6 quantization settings: 同在量化加载路径上加固格式 / 后端约束，与本 PR 的加载期对齐约束（K/N 被 16 整除）动机一致。