PR #34331 完整报告

sgl-project/sglang

[quantization] Add tuned Triton tile configs for channelwise FP8 GEMM…

合并时间：2026-08-14 11:11

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34331

执行摘要

- 一句话：L40S 上调优 FP8 GEMM Triton tile 并默认启用
- 推荐动作：值得精读。看点有三：(1) apply_fp8_linear 的分支编排——tuned_config 查表夹在 CUTLASS 判定与执行之间，用 None 表示"保持默认"，侵入面极小；(2) 配置文件以 N/K/device_name/dtype 命名、最近 M 查找、null 占位防吸附的设计，与既有 get_w8a8_block_fp8_configs 惯例统一，可复用到其他量化路径；(3) torch.compile 下主动放弃该优化并明确注释的取舍。若团队在 L40S 上维护 FP8 服务的基线评测，需留意此 PR 会静默改变数值结果。

功能与动机

PR body 指出现状缺陷：apply_fp8_linear 的 per-token/per-channel FP8 路径在 weight shape 满足 CUTLASS 条件时固定走 CUTLASS fp8_scaled_mm，否则走 Triton scaled_mm_kernel，且 Triton 侧只用粗糙的 M 分档 tile 启发式，从不调 num_warps/num_stages。L40S/SM89 上对 Qwen3-8B 四个 FP8 linear 形状的离线扫描发现 tuned Triton tile 在几乎每个 token 数上都胜过 CUTLASS（小 M 快 3-10%，M=512 最高 1.8x），因此需要一张按设备 + 形状调优的配置表来释放收益。

实现拆解

实现按五步展开：

1. 新增配置查表（fp8_kernel.py）：新增 get_w8a8_channelwise_fp8_configs(N, K)，用 functools.lru_cache 缓存，按 device_name（空格转下划线）+ N/K 拼 JSON 文件名并读入 M→tile 映射；torch._dynamo.is_compiling() 时直接返回 None（host 侧设备名与文件 I/O 不可 trace，回退 CUTLASS，与既有 get_w8a8_block_fp8_configs 行为一致）。再新增 get_w8a8_channelwise_fp8_config(N, K, M) 按最近 M 返回配置或 None。
2. 扩展 triton_scaled_mm（fp8_kernel.py）：新增可选参数 num_warps/num_stages；由于 triton.jit 不接受显式 None，只在非 None 时装入 launch_kwargs，保证默认路径使用 triton 自身默认值。
3. 接入 apply_fp8_linear（fp8_utils.py）：新增 use_tuned_triton_channelwise 条件（use_cutlass_channelwise_gemm 且 SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE 开启），在 channelwise 分支先查 tuned_config；命中则走 triton_scaled_mm 并传入 BLOCK_SIZE_{M,N,K}、use_heuristic=False、num_warps、num_stages，未命中则保持

原 CUTLASS fp8_scaled_mm 路径。

4. 注册环境变量（environ.py）：在 Quantization 区块新增 SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE = EnvBool(True)，默认开启、仅作为 kill switch；注释明确说明其他 GPU/ 未调优 shape 下为 no-op。
5. 新增 4 份 L40S 配置 JSON（kernels/ops/quantization/configs/）：对应 Qwen3-8B 的 qkv（N=6144,K=4096）、o（N=4096,K=4096）、down（N=4096,K=12288）、gate_up（N=24576,K=4096）四个 FP8 linear，覆盖 M=1 到 8192 的不规则网格；gate_up 的 M=512/1024/2048 为 null（CUTLASS 胜出点，保留占位防止邻近 M 吸附）。

配套说明：本次提交没有新增 / 修改测试文件，仅靠 PR 描述中的内核级与端到端 benchmark 验证；提交演进上，第 2 个提交将功能从 opt-in 翻转为默认开启，第 3 个提交按 review 补充了 torch.compile 回退的注释说明。

关键文件：

- python/sglang/srt/layers/quantization/fp8_utils.py（模块 量化层；类别 source；类型 core-logic；符号 apply_fp8_linear）：核心 dispatch 变更点：apply_fp8_linear 在 channelwise CUTLASS 分支前新增 tuned_config 查表，命中时改走 tuned Triton 路径并透传 tile 参数，未命中保持原 CUTLASS 路径；默认开启使 L40S 上四个 shape 的行为发生变化。
- python/sglang/kernels/ops/quantization/fp8_kernel.py（模块 内核层；类别 infra；类型 infrastructure；符号 get_w8a8_channelwise_fp8_configs, get_w8a8_channelwise_fp8_config, triton_scaled_mm）：新增按 (N, K) 与最近 M 查表的 get_w8a8_channelwise_fp8_configs / get_w8a8_channelwise_fp8_config，并为 triton_scaled_mm 增加 num_warps/num_stages 透传；torch.compile 下返回 None 回退 CUTLASS 的设计决策在此落地。
- python/sglang/srt/environ.py（模块 环境变量；类别 source；类型 configuration）：在 Quantization 区块注册 SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE（默认 True）作为 kill switch，并附说明：仅在存在匹配 JSON 的 GPU/shape（当前为 L40S）上生效。
- python/sglang/kernels/ops/quantization/configs/N=6144,K=4096,device_name=NVIDIA_L40S,dtype=fp8_w8a8_channelwise.json（模块 配置表；类别 infra；类型 configuration）：qkv_proj 的调优表：16 个 M 点的 tile 配置，覆盖 M=1 到 8192，是默认切换后 L40S 上最常被命中的权重形状之一。
- python/sglang/kernels/ops/quantization/configs/N=4096,K=4096,device_name=NVIDIA_L40S,dtype=fp8_w8a8_channelwise.json（模块 配置表；类别 infra；类型 configuration）：o_proj 的调优表：M=512 处取得全 PR 最大 1.8× 加速的配置（BM128 BN128 BK128 w8 s4）在此文件中。
- python/sglang/kernels/ops/quantization/configs/N=4096,K=12288,device_name=NVIDIA_L40S,dtype=fp8_w8a8_channelwise.json（模块 配置表；类别 infra；类型 configuration）：down_proj 的调优表：小 M 下普遍选择 BM16 BN32 BK1024，大 M 下选择 BM128 BN256 BK128，体现核宽 / 核深的 M 相关变化。
- python/sglang/kernels/ops/quantization/configs/N=24576,K=4096,device_name=NVIDIA_L40S,dtype=fp8_w8a8_channelwise.json（模块 配置表；类别 infra；类型 configuration）：gate_up_proj 的调优表：M=512/1024/2048 为 null（CUTLASS 胜出）

，直接体现了“null 占位 + 回退 CUTLASS”的设计，是理解配置语义的关键样例。

关键符号：apply_fp8_linear, get_w8a8_channelwise_fp8_configs, get_w8a8_channelwise_fp8_config, triton_scaled_mm

关键源码片段

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

核心 dispatch 变更点：apply_fp8_linear 在 channelwise CUTLASS 分支前新增 tuned_config 查表，命中时改走 tuned Triton 路径并透传 tile 参数，未命中保持原 CUTLASS 路径；默认开启使 L40S 上四个 shape 的行为发生变化。

```
# python/sglang/srt/layers/quantization/fp8_utils.py 的 apply_fp8_linear 内部
# 只有形状原本会走 CUTLASS 时才考虑 tuned Triton tile（离线扫描就是针对该路径）
# 默认开启，SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE=0 可关闭
use_tuned_triton_channelwise = (
    use_cutlass_channelwise_gemm and envs.SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE.get()
)

if channelwise_cutlass:
    # weight 是 [K, N]：N 取 weight.shape[1]，K 取 weight.shape[0]
    # 查不到 tuned 配置时返回 None，后端选择保持原样（与未开启完全一致）
    tuned_config = (
        get_w8a8_channelwise_fp8_config(
            N=weight.shape[1], K=weight.shape[0], M=qinput.shape[0]
        )
        if use_tuned_triton_channelwise
        else None
    )
    if not use_cutlass_channelwise_gemm:
        # 原 Triton 启发式路径（非 CUTLASS 兼容 shape），行为不变
        qinput = qinput.view(-1, qinput.shape[-1])
        output = triton_scaled_mm(qinput, weight, x_scale, weight_scale, output_dtype, bias)
    elif tuned_config is not None:
        # 新路径：使用离线调优的 tile 与 launch 参数，关闭启发式 tile
        qinput = qinput.view(-1, qinput.shape[-1])
        output = triton_scaled_mm(
            qinput, weight, x_scale, weight_scale, output_dtype, bias,
            block_size_m=tuned_config["BLOCK_SIZE_M"],
            block_size_n=tuned_config["BLOCK_SIZE_N"],
            block_size_k=tuned_config["BLOCK_SIZE_K"],
            use_heuristic=False,
            num_warps=tuned_config["num_warps"],
            num_stages=tuned_config["num_stages"],
        )
    else:
        # 原 CUTLASS 路径：fp8_scaled_mm 与基线逐字节一致
        output = fp8_scaled_mm(qinput, weight, x_scale, weight_scale,
                               out_dtype=output_dtype, bias=bias)
```

```
    return output.view(*output_shape)
```

python/sglang/kernels/ops/quantization/fp8_kernel.py

新增按 (N, K) 与最近 M 查表的 get_w8a8_channelwise_fp8_configs / get_w8a8_channelwise_fp8_config，并为 triton_scaled_mm 增加 num_warps/num_stages 透传；torch.compile 下返回 None 回退 CUTLASS 的设计决策在此落地。

```
# python/sglang/kernels/ops/quantization/fp8_kernel.py

@functools.lru_cache
def get_w8a8_channelwise_fp8_configs(N: int, K: int) -> Optional[Dict[int, Any]]:
    """返回当前设备上某 weight shape 的 tuned Triton tile 表。

    null 项表示该 M 下 CUTLASS 更快，网格保留占位点防止邻近 M 吸附，
    但调用方必须回退；文件缺失 / shape 未调优同样表示保持默认。
    """
    # torch.compile 下返回 None 并回退 CUTLASS：host 侧设备名 + 文件 I/O
    # 不可 trace，这与主线 get_w8a8_block_fp8_configs 行为保持一致（有意为之）
    if torch._dynamo.is_compiling():
        return None

    device_name = get_device_name().replace(" ", "_")
    json_file_name = (
        f"N={N},K={K},device_name={device_name},dtype=fp8_w8a8_channelwise.json"
    )
    config_file_path = os.path.join(
        os.path.dirname(os.path.realpath(__file__)), "configs", json_file_name
    )
    if not os.path.exists(config_file_path):
        return None

    with open(config_file_path) as f:
        log_info_on_rank0(
            logger,
            f"Using configuration from {config_file_path} for W8A8 channelwise FP8 GEMM.",
        )
        return {int(key): val for key, val in json.load(f).items()}


def get_w8a8_channelwise_fp8_config(N: int, K: int, M: int) -> Optional[Dict[str, int]]:
    """返回距 M 最近的 tuned 配置；None 表示保持默认路径。"""
    configs = get_w8a8_channelwise_fp8_configs(N, K)
    if not configs:
        return None
    return configs[min(configs.keys(), key=lambda x: abs(x - M))]


# triton_scaled_mm 内新增的 launch 参数透传：
```

```
# num_warps / num_stages 是 triton.jit 的 launch kwarg，显式 None 会报错，
# 所以只在调用方显式传入时放进 launch_kwargs，默认路径保持 triton 自身默认值
launch_kwargs = {}
if num_warps is not None:
    launch_kwargs["num_warps"] = num_warps
if num_stages is not None:
    launch_kwargs["num_stages"] = num_stages
scaled_mm_kernel[grid](..., **launch_kwargs)
```

评论区精华

核心讨论围绕"默认开启"的取舍展开：

1. 模型级精度验证（environ.py:709）：BBuf 担心默认开启改变累加顺序，单层 cosine 不能证明误差不跨层累积，建议保持默认关闭直到拿到模型级对比。RunkaiTao 回复已将 GSM8K 全量对比放入 PR 描述（tuned 0.9346 vs baseline 0.9193，+1.5pp，在 1σ≈±0.7pp 内），最终维持默认开启 + kill switch。
2. torch.compile 下查表失效（fp8_kernel.py）：BBuf 问 apply_fp8_linear 被 trace 时返回 None、inductor prefill 用不到 tuned 路径是否有意。RunkaiTao 确认为有意设计：这是 eager 路径优化，与主线 block-FP8 查表行为一致，并已扩充注释、写入 PR 风险说明。
3. 描述一致性 + CI：BBuf 综述评论指出 Risk 部分仍写"默认关闭"与代码不一致，且 draft 期间 CI 被阻塞；作者更新描述并重跑 CI 后，BBuf APPROVED。

- 默认开启带来的累加顺序与精度风险 (correctness): PR 描述补充 GSM8K 全量对比：tuned 0.9346 vs baseline 0.9193（+1.5pp，在 1σ≈±0.7pp 方差内），作者判断在方差内，维持默认开启 + kill switch。
- torch.compile 下 tuned 查表返回 None 是否有意 (question): 确认为有意设计：torch.compile（含 inductor prefill）下该优化不生效，回退 CUTLASS，行为与既有 block-FP8 查表一致。
- PR 描述与实现默认值不一致 + CI 重跑 (documentation): 作者将描述同步为默认开启并补充模型级精度表格，重跑 CI 后 BBuf APPROVED。

风险与影响

- 风险：
  1. 累加顺序变化（fp8_utils.py / L40S）：四个 shape 默认切换到 Triton 后累加顺序与 CUTLASS fp8_scaled_mm 不同，cos≈0.9999+，bit-exact 或精度敏感基线需重测；已提供 SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE=0 回退。
  2. torch.compile 下不生效（fp8_kernel.py）：inductor prefill 阶段查表返回 None 回退 CUTLASS，属预期性能损失，无正确性风险，但用户需知晓。
  3. 缺少自动化测试：本次无测试文件覆盖 get_w8a8_channelwise_fp8_config 查表与 apply_fp8_linear 新分支，回归只能靠手工 benchmark。
  4. 配置名耦合：JSON 文件名由 f-string 拼出（N/K/device_name/dtype），未来设备名或约定变化会静默查不到配置（安全回退但不生效）。

5. 调优面窄：仅覆盖 Qwen3-8B 四个 shape 与 L40S，其他模型 /GPU 不受益也不受损害。 - 影响：用户侧：L40S 上运行 Qwen3-8B FP8 的用户默认获得端到端吞吐 +3~6%、TPOT/ITL 下降 2~6%（c=1/8/32 实测），以及大 M 下最高 1.8× 的 GEMM 级加速；代价是数值基线微移，可通过 kill switch 恢复。系统侧：查表严格按 device_name+shape 门控，其他 GPU 与未调优形状走 byte-identical 原路径，torch.compile 路径不受影响，风险极低。团队侧：确立了一套“设备 + 形状 JSON 调优表 + 默认开启 + kill switch”的可复用机制，后续可按相同约定为其他 GPU/ 模型补配置即可，无需改动 dispatch 代码。 - 风险标记：核心路径默认行为变更，数值精度变化，缺少测试覆盖，torch.compile 下不生效，配置仅覆盖 L40S 特定 shape

关联脉络

- PR #34730 [Core] Organize environment variable registry: 重构 environ.py 为按主题区块注册环境变量；本 PR 新增的 SGLANG_ENABLE_FP8_GEMM_CONFIG_TUNE 与既有 Quantization 区块环境变量同处一个文件与体系，后续维护需保持一致。
- PR #28354 [FlashInfer v0.6.16] Support FlashInfer CuTe DSL NVFP4 MoE quantization: 同为量化路径的后端选择与配置工作（FP8/NVFP4 GEMM）；本次建立的 device+shape JSON 调优表模式有望扩展到 NVFP4 等其他量化方案。