PR #37141 完整报告

sgl-project/sglang

[Diffusion] Fuse FLUX.2 token concatenation and NVFP4 quantization

合并时间：2026-08-31 21:33

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/37141

执行摘要

- 一句话：融合 FLUX.2 token 拼接与 NVFP4 量化，端到端提速 2.4%
- 推荐动作：值得精读。对做 diffusion/quantization 性能优化的工程师尤其有参考价值：一是 " 融合 kernel 如何做到 bit-exact"——通过复用与 FlashInfer 完全相同的 cvt_warp_fp16_to_fp4 原语和 get_sf_out_offset_128x4 swizzle 偏移，从根上保证布局一致，再用字节级 torch.equal 与端到端像素比对锁定；二是 fallback 门控的完备性设计——try_flux2_token_cat_nvfp4 把能力探测、环境变量、dtype/stride/alignment、行数上限、编译与 capture 状态全部聚拢在一个函数里，任何不确定场景都安全回退；三是与 #37096 的划界方式——本 PR 保持 bit-exact，非 bit-exact 融合单独走 --quality=high 门控，避免两条优化路线互相污染。

功能与动机

FLUX.2 的 single-stream block 输出由 attention（6144 维）与 MLP（18432 维）两段 BF16 拼接而成，原有实现依次执行 torch.cat([attention_bf16, swiglu_bf16]) -> FlashInfer NVFP4 quantize -> to_out GEMM，每个 block 都要物化一个 24576 宽的 BF16 拼接张量并整宽读取做量化。PR body 明确指出该改动 "removes one 24,576-wide BF16 materialization and one full-width quantization read in each of the 48 single-stream blocks"，这是端到端 2.4% 提速的直接来源。

实现拆解

实现按 5 步拆解：

1. 新增融合 JIT 内核与 Python 门控：python/sglang/kernels/ops/diffusion/layout/flux2_token_cat_nvfp4_jit.py 新增 try_flux2_token_cat_nvfp4，集中收拢所有 fallback 条件——torch.compiler.is_compiling()、CUDA graph capture、非 dense BF16、指针未按 32 字节对齐、设备不匹配、行数超过 _MAX_ROWS = 65408（CUDA grid.y 上限 65535 向下取 128 的整数倍）、三个量化模式环境变量（FLASHINFER_DISABLE_FP4_QUANT_FAST_MATH、TRTLLM_DISABLE_FP4_QUANT_FAST_MATH、FLASHINFER_NVFP4_4OVER6）、非 SM103 能力集。任一条件不满足即返回 None，由调用方走 eager 原路径。python/sglang/kernels/jit/csrc/diffusion/flux2_token_cat_nvfp4.cuh 实现内核：每个线程处理一个 16 元素 group，从 attention 或 MLP 分支读取 BF16，复用 TensorRT-LLM 的 cvt_warp_fp16_to_fp4 原语（与 FlashInfer 相同）写 packed 值与 swizzled scales，并用 get_sf_out_offset_128x4 保证 scale 布局与 FlashInfer 完全

一致；pad 行写 0 scale。

2. 新增预量化 GEMM 数据契约：python/sglang/multimodal_gen/runtime/layers/quantization/modelopt_quant.py 新增 apply_nvfp4_gemm_prequantized(layer, x_fp4, x_scale_interleaved, output_dtype, bias)，复用 ModelOptFp4LinearMethod.apply 后半段的 padding、dtype 转换、_get_fp4_gemm_op() 分发与 slice_nvfp4_output 逻辑，但跳过 fp4_quantize 与 pad_nvfp4_activation_for_cutlass 之外的量化步骤，使本 PR 不依赖 #37096 的 tuple 接口。
3. 模型层接入：python/sglang/multimodal_gen/runtime/models/dits/flux_2.py 在 Flux2ParallelSelfAttention.__init__ 中根据 tp_size == 1、设备能力集为 (10, 3)、to_out.quant_method 为 ModelOptFp4LinearMethod 三个条件设置 _enable_nvfp4_token_cat；forward 中先调用 try_flux2_token_cat_nvfp4(hidden_states, mlp_hidden_states, self.to_out.input_scale_inv)，返回 None 则保持 torch.cat + self.to_out 原路径，否则走 apply_nvfp4_gemm_prequantized 后 view(*output_shape)。
4. 依赖解析与算子注册：python/sglang/kernels/jit/utils/deps.py 新增 get_flashinfer_nv_internal_include_paths，定位 flashinfer/data/csrc/nv_internal 头文件路径，缺失时抛清晰错误；python/sglang/kernels/ops/diffusion/__init__.py 把 try_flux2_token_cat_nvfp4 注册进 _SPECS 并映射到 layout.flux2_token_cat_nvfp4_jit。
5. 测试与基准配套：test/registered/kernels/ops/diffusion/test_model_fast_paths.py 新增 SM103 限定的 test_token_cat_nvfp4_matches_flashinfer（字节级比对 packed 值与 scales）和 test_token_cat_nvfp4_falls_back_while_compiling；test/registered/kernels/ops/quantization/test_diffusion_nvfp4_scaled_mm.py 新增参数化 test_prequantized_input_matches_regular_apply（default 与 flashinfer_trtllm 两个后端下 torch.equal 断言）；test/manual/kernels/bench_flux2_token_cat_nvfp4.py 新增手工基准脚本，同时用 CUDA event 与墙钟计时并校验字节一致。

关键文件：

- python/sglang/kernels/ops/diffusion/layout/flux2_token_cat_nvfp4_jit.py（模块 JIT 内核；类别 source；类型 core-logic；符号 try_flux2_token_cat_nvfp4, _env_enabled, _is_dense_bf16, _module）：融合优化的 Python 入口，集中实现全部 fallback 门控、输出张量分配与 JIT module 调用，是理解整个 PR 安全边界的关键文件。
- python/sglang/kernels/jit/csrc/diffusion/flux2_token_cat_nvfp4.cuh（模块 CUDA 内核；类别 other；类型 core-logic；符号 sglang::flux2_token_cat_nvfp4::kernel, sglang::flux2_token_cat_nvfp4::Kernel::run）：实际执行融合的 CUDA kernel：逐 group 读取 attention/MLP 分支、复用 TensorRT-LLM 量化原语、按 FlashInfer 128x4 swizzle 写 scale，是 bit-exact 保证的落点。
- python/sglang/multimodal_gen/runtime/models/dits/flux_2.py（模块 模型层；类别 source；类型 core-logic；符号 Flux2ParallelSelfAttention.init, Flux2ParallelSelfAttention.forward）：模型层接入点：在 Flux2ParallelSelfAttention 中按能力集与量化方法条件启用融合路径，forward 中保持 eager 回退，是行为契约的核心。
- python/sglang/multimodal_gen/runtime/layers/quantization/modelopt_quant.py（模块 量化层；类别 source；类型 data-contract；符号 apply_nvfp4_gemm_prequantized）：新增 apply_nvfp4_gemm_prequantized 预量化 GEMM 数据契约，使融合 kernel 的

packed 输出可以直接进入既有 FP4 GEMM，是本 PR 不做成 #37096 依赖的关键。

- python/sglang/kernels/jit/utils/deps.py（模块 依赖解析；类别 source；类型 core-logic；符号 get_flashinfer_nv_internal_include_paths）：新增依赖注册 get_flashinfer_nv_internal_include_paths，定位 FlashInfer 的 nv_internal 头文件路径，缺头时给出明确错误而非静默失败。
- test/registered/kernels/ops/diffusion/test_model_fast_paths.py（模块 快速路径；类别 test；类型 test-coverage；符号 test_token_cat_nvfp4_matches_flashinfer, test_token_cat_nvfp4_falls_back_while_compiling）：核心 bit-exact 验证：test_token_cat_nvfp4_matches_flashinfer 在 SM103 上逐字节比对 fused 与 flashinfer.fp4_quantize(torch.cat(...)) 的 packed 值和 scales；同时验证编译期回退。
- test/registered/kernels/ops/quantization/test_diffusion_nvfp4_scaled_mm.py（模块 量化测试；类别 test；类型 test-coverage；符号 test_prequantized_input_matches_regular_apply）：验证预量化 GEMM helper 在 default 与 flashinfer_trtllm 两个后端下与常规 quantize+GEMM 路径 torch.equal，锁定数据契约。
- test/manual/kernels/bench_flux2_token_cat_nvfp4.py（模块 基准脚本；类别 test；类型 test-coverage；符号 _benchmark, _benchmark_wall, _run_case）：手工基准脚本：CUDA event 与墙钟双计时，覆盖 17/512/4096/4608 四档 token 数，并校验 fused 与 baseline 字节精确，对应 PR body 的 microbenchmark 数据。
- python/sglang/kernels/ops/diffusion/__init__.py（模块 算子注册；类别 infra；类型 infrastructure）：把 try_flux2_token_cat_nvfp4 注册进 diffusion 算子规格表并映射到 layout 模块，决定算子的后端选择与分发。

关键符号：try_flux2_token_cat_nvfp4, apply_nvfp4_gemm_prequantized, get_flashinfer_nv_internal_include_paths, Flux2ParallelSelfAttention.init, Flux2ParallelSelfAttention.forward

关键源码片段

python/sglang/multimodal_gen/runtime/models/dits/flux_2.py

模型层接入点：在 Flux2ParallelSelfAttention 中按能力集与量化方法条件启用融合路径，forward 中保持 eager 回退，是行为契约的核心。

```
# 单流块输出 = attention 分支 + MLP 分支，原本先 cat 成
# [.., 6144+18432] 的 BF16，再做 FP4 量化 + to_out GEMM；
# SM103 + NVFP4 下改为让 JIT kernel 直接产出 packed 激活和
# swizzled scales，再喂给预量化 GEMM，省掉一次全宽物化与全宽读取
output_shape = (*hidden_states.shape[:-1], self.out_dim)
packed = None
if self._enable_nvfp4_token_cat:
    packed = try_flux2_token_cat_nvfp4(
        hidden_states, mlp_hidden_states, self.to_out.input_scale_inv
    )
if packed is None:
    # 门控未通过：保持 eager 原路径，确保 torch.compile、
    # CUDA graph、非 SM103、TP>1 等场景行为完全不变
```

```
            hidden_states = torch.cat([hidden_states, mlp_hidden_states], dim=-1)
            hidden_states, _ = self.to_out(hidden_states)
        else:
            hidden_states = apply_nvfp4_gemm_prequantized(
                self.to_out,
                *packed,
                output_dtype=hidden_states.dtype,
                bias=self.to_out.bias,
            ).view(*output_shape)

        return hidden_states
```

评论区精华

核心交锋发生在 PR 评论区：

> sushildubey171：对比图中看到肉眼可见差异，怀疑现有指标没覆盖到，建议用 PR#25871 的 T2I accuracy metric 做 pre-PR vs post-PR 对比。
>
> BBuf：你看到的差异来自旧对比中叠加上去的父 PR #37096，不是本 token-cat 融合；已让 #37141 完全独立于 #37096 并重新基于 main。重新验证三件事：融合内核与 flashinfer.fp4_quantize(torch.cat(...)) 的 packed 值与 scale 字节完全一致；预量化 GEMM 在两个 FlashInfer 后端下与常规路径完全一致；1024x1024、50 步端到端生成 PNG 字节相同（SHA256 一致、像素差 0）。对 bit-exact 优化做 FID 对比没有额外信号。
>
> BBuf：补充澄清 #37096 的非 bit-exact FLUX.2 NVFP4 FC1+SwiGLU+FC2-input 融合现在是 request-scoped，仅 --quality=high 启用，默认 --quality=lossless 保留参考实现。

- 端到端图像精度质疑与 FID 度量建议 (correctness): BBuf 澄清该差异来自旧对比中叠加上去的父 PR #37096，而非本 token-cat 融合；本 PR 已独立并重新基于 main。重新验证量化字节一致、GEMM 一致、端到端 PNG 像素差为 0，因此 FID 对比对 bit-exact 优化无额外信号。
- 37096 非 bit-exact 融合的 quality gate 边界 (design): 明确 --quality=lossless 默认路径保持参考实现，bit-exact 与 fast-path 两条优化路线通过质量档位隔离。
- CI 失败重跑 (other): 未涉及技术争议，纯 CI 运维操作。

风险与影响

- 风险：
  1. FlashInfer 内部头文件依赖：try_flux2_token_cat_nvfp4 在门控通过后调用 _module()，若安装的 FlashInfer wheel 缺少 nv_internal 头文件，load_jit 会抛 RuntimeError 而不是回退 eager 路径。PR body 明确这是有意的 "fail clearly"，但意味着 SM103 环境暴露在 " 纯功能降级被挡死 " 的风险下，需要团队确认所有分发的 FlashInfer 构建都包含这些头。
  2. bit-exact 依赖 FlashInfer 原语版本：内核复用 cvt_warp_fp16_to_fp4，若未来 FlashInfer 改变舍入行为或布局（如引入 4-over-6 模式），字节精确性会被破坏；当前靠三个环境变量门控与两个 committed 测试兜底。

3. JIT 编译无失败降级：@cache_once 缓存的 module 在进程内复用，首次编译失败或首次调用后升级 FlashInfer 导致缓存失效，都会让 FLUX.2 前向直接报错，测试未覆盖缺头文件分支。
4. 影响面收敛在窄配置：仅 SM103 + TP1 + ModelOpt NVFP4；flux_2.py 是核心生成路径，try_* 返回非 None 后若 packed tensor 布局错误会直接进 GEMM，当前依赖测试覆盖与 CI 的 B200/SM103 验证。 - 影响：用户侧：FLUX.2-dev NVFP4 在 GB300 上 Denoise 每步 136.883 ms -> 133.580 ms（-2.413%），端到端 6980.339 ms -> 6813.233 ms（-2.394%），且输出与原实现逐字节一致，用户无感知风险。系统侧：新增一条 JIT kernel 汇编链（cuh + Python 门控 + 依赖解析 + 算子注册），并新增对 FlashInfer nv_internal 头文件的构建期依赖；非 SM103、非 NVFP4、TP>1、torch.compile、CUDA graph 场景完全不受影响。团队侧：确立了 bit-exact 融合的验证模板——量化原语一致 + 字节级 equality + 端到端 PNG SHA256/SSIM/PSNR/LPIPS 比对，后续 diffusion 快速路径可复用这套方法论。 - 风险标记：FlashInfer 头文件依赖，JIT 编译失败无降级，SM103 限定，bit-exact 依赖原语版本

关联脉络

- PR #37096 （未在历史列表，PR body 提及）FLUX.2 NVFP4 FC1+SwiGLU+FC2-input 量化融合：本 PR 最初基于 #37096，后解耦为独立分支；#37096 的非 bit-exact 融合现仅 --quality=high 启用，与本 PR 的 bit-exact 路线形成互补与边界划分。
- PR #37116 [diffusion] perf: absorb Qwen-Image output projection biases: 同一 diffusion 性能优化系列，同样是 JIT kernel 融合消除中间物化，且同样以 bit-exact 或近似等价为验收标准。
- PR #36916 [Diffusion] Detect quantized transformer replacements: 同属 diffusion 量化家族，处理预量化模型加载与量化声明检测，与本 PR 的 NVFP4 量化路径有上下文关联。
- PR #36991 [Diffusion] Add exact component precision overrides: 组件级精度覆盖与加载 / 驻留阶段打通，与本 PR 的量化精度保证（bit-exact）属于同一设计主线。
- PR #36907 [Diffusion] Enforce component attention backend application: 组件级后端选择强制机制，与本 PR 的 " 仅 SM103 + 特定量化方法启用 " 条件门控共享设计思路。