PR #23550 完整报告

sgl-project/sglang

[Bug Fix] GLM-5.1: drop constexpr on page_indice_batch_offset, skip offloader post_init on draft worker, support N=32 in copy_to_gpu_no_ce

合并时间：2026-05-09 15:43

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23550

执行摘要

- 一句话：修复 GLM-5.1 专家卸载路径的三个阻塞问题
- 推荐动作：值得精读，特别是 Triton constexpr 取舍、offloader 在多 worker 下的状态隔离设计。可作为类似性能问题的参考。

功能与动机

多个问题阻塞了 GLM-5.1 expert-offload 路径，并在长序列 / per-batch 运行时引发过度的 Triton 重编译。具体包括：(1) NSA kernel 中 page_indice_batch_offset 随 batch 变化，保持 constexpr 导致每次新值都触发 JIT 重编译；(2) MTP draft worker 中误执行 offloader post_init，与目标模型状态冲突；(3) copy_to_gpu_no_ce 缺少 N=32 分支，导致 expert-offload 路径报 unexpected N。

实现拆解

1. NSA Triton kernel 移除 constexpr：在 python/sglang/srt/layers/attention/nsa/index_buf_accessor.py 中，将 _get_k_and_s_triton_kernel 函数签名中的 page_indice_batch_offset: tl.constexpr 改为运行时参数，避免因 batch 尺寸变化引发的 Triton JIT 重编译，同时保持功能不变。
2. Model runner 跳过 draft worker offloader：在 python/sglang/srt/model_executor/model_runner.py 的 load_model 方法中，将无条件调用 get_offloader().post_init() 改为仅当 self.is_draft_worker 为 False 时执行，防止 draft worker 与 target model 争夺 offloader 状态。
3. CUDA copy kernel 增加 N=32 分支：在 sgl-kernel/csrc/elementwise/copy.cu 的 copy_to_gpu_no_ce 函数中添加 else if (N == 32) copy_to_gpu_no_ce_impl<32>(...)，满足 GLM-5.1 expert-offload 路径的特定块大小请求。
4. 新增 MoE 调优配置：添加 python/sglang/srt/layers/moe/moe_runner/triton_utils/configs/triton_3_5_1/E=257,N=256,device_name=NVIDIA_B200,dtype=fp8_w8a8,block_shape=[128, 128].json，为 GLM-5.1 on B200 TP=8 场景提供预调优的 fused-MoE kernel 配置。

关键文件：

- python/sglang/srt/layers/attention/nsa/index_buf_accessor.py（模块 NSA 注意；类别 source；类型 core-logic；符号 _get_k_and_s_triton_kernel）：核心修复：移除 Triton kernel 中 page_indice_batch_offset 的 constexpr，避免 JIT 重编译。
- python/sglang/srt/model_executor/model_runner.py（模块 模型运行；类别 source；类型 data-contract；符号 load_model）：避免 draft worker 与 target model 的 offloader 状态冲突。
- sgl-kernel/csrc/elementwise/copy.cu（模块 CUDA 核；类别 other；类型 core-logic；符号 copy_to_gpu_no_ce）：为 GLM-5.1 expert-offload 路径添加缺少的 N=32 分支。
- python/sglang/srt/layers/moe/moe_runner/triton_utils/configs/triton_3_5_1/E=257,N=256,device_name=NVIDIA_B200,dtype=fp8_w8a8,block_shape=[128, 128].json（模块 MoE 配置；类别 config；类型 configuration）：新增 GLM-5.1 on B200 TP=8 的 fused-MoE 调优配置。

关键符号：_get_k_and_s_triton_kernel, load_model, copy_to_gpu_no_ce

关键源码片段

python/sglang/srt/layers/attention/nsa/index_buf_accessor.py

核心修复：移除 Triton kernel 中 page_indice_batch_offset 的 constexpr，避免 JIT 重编译。

```
# 参数 page_indice_batch_offset 从 constexpr 改为运行时参数
# 原因：该值随 batch 变化，保持 constexpr 导致 Triton 为每个新值重编译
@triton.jit
def _get_k_and_s_triton_kernel(
    buf_ptr,
    page_indices_ptr,
    k_out_ptr,
    s_out_ptr,
    seq_len_ptr,
    seq_len_num_pow: tl.constexpr,
    page_size: tl.constexpr,
    buf_numel_per_page: tl.constexpr,
    index_head_dim: tl.constexpr,
    s_offset_in_page: tl.constexpr,
    page_indice_batch_offset, # 原为 page_indice_batch_offset: tl.constexpr
    BLOCK_SIZE: tl.constexpr,
    BLOCK_SIZE_K: tl.constexpr,
):
    # 函数体内仅做标量比较和乘法，无编译期常量优化收益
    ...
    page_indices_base = batch_id * page_indice_batch_offset
    page_idx_valid_mask = page_idx < page_indice_batch_offset
```

评论区精华

作者在 issue 评论中解释了移除 constexpr 的原因：page_indice_batch_offset 取值来自 page_indices.shape[1]，在 serving 中跨 batch 变化，保持 constexpr 强制 Triton 为每个新

值 JIT 重编译并膨胀 kernel cache，而该参数仅用于标量乘法和比较，constexpr 折叠无收益，作为运行时 int 可生成单一 kernel 变体且无性能开销。

- 移除 page_indice_batch_offset 的 constexpr 原因 (performance): 接受修改，确认无 measurable overhead。

风险与影响

- 风险：
  - 性能风险：移除 constexpr 后 Triton 内部可能因缺少编译期常量而略有性能损失，但作者已确认无 measurable overhead。
  - 状态一致性风险：跳过 draft worker 的 post_init 后，若 draft worker 实际需要 offloader（如某些共享权重场景），可能导致 offloader 状态未初始化。需确认 is_draft_worker 判断覆盖所有 draft 类型。
  - 兼容性风险：新增 N=32 分支在 CUDA 核函数中属于窄场景，但若未来其他模型也使用该函数但 N=32 行为不同，可能引入错误。当前仅 GLM-5.1 使用。
  - 测试覆盖不足：本次改动无直接测试文件，核心路径变更缺少回归验证。
  - 影响：直接影响 GLM-5.1 模型在 long-serving 场景下的专家卸载路径，消除 Triton 重编译，提升首次推理延迟稳定性。间接影响其他使用 NSA 注意力和 copy_to_gpu_no_ce 的模型（概率低）。对系统和团队无负面影响。
  - 风险标记：Triton 重编译避免需验证性能无退化，Draft worker offloader 状态隔离需全面验证，缺少回归测试覆盖

关联脉络

- 暂无明显关联 PR