PR #33436 完整报告

sgl-project/sglang

fix: support FA4 backend for GLM4.7-flash

合并时间：2026-08-09 20:05

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33436

执行摘要

- 一句话：修复 GLM4.7-Flash 在 FA4 后端下 MLA head 比例不兼容问题
- 推荐动作：值得精读。该 PR 展示了如何在不改内核的前提下，通过包装器层做 head 比例 padding 来适配固定 tile 尺寸的 MLA 内核，设计简洁且伴随充分回归测试。关注点：_pad_mla_q_heads 中 2 的幂取整策略、_unpad_mla_result 对 LSE 的同步裁剪、以及强制 num_splits=1 的语义影响。建议后续跟踪 flash-attn 上游 PR 2670 合并后移除 hd256 workaround，并持续收集 FA4 在真实负载下的性能数据。

功能与动机

PR body 明确指出目标是让 GLM-4.7-Flash 在 Blackwell 上使用 --attention-backend fa4 且默认非确定性 *decode* 与 CUDA Graph 时能正确服务。SGLang 的 absorbed MLA 在 TP1 下将 20 个 Q/QV head 映射到 1 个 latent KV head，但 FA4 的 packed kernel 要求 head 比例与其 128 行 tile 兼容；当前 pack_gqa=False 的 QV varlen *decode* 路径在 bs>1 时存在查询读取跨序列 paged KV 的上游 bug，即使修复上游 bug，unpacking 也会把 Q head 单独调度，重复遍历共享 latent KV，严重利用不足 128 行 *decode* tile。

实现拆解

变更集中在 FA4 包装器 python/sglang/kernels/ops/attention/flash_attention_v4.py 与对应测试 test/registered/kernels/ops/attention/test_flash_attention_4.py，分四步落地：

1. 新增 head 比例 padding 逻辑：在 flash_attention_v4.py 中新增 _pad_mla_q_heads，当 qv 存在且 pack_gqa=True 时，计算每个 KV head 对应的 Q head 数 qhead_per_kvhead，若其与 128 行 tile 不兼容（128 % qhead_per_kvhead != 0 且 qhead_per_kvhead % 128 != 0），则将该比例向上取整到 2 的幂（GLM TP1 下 20→32），并按 KV group 在 head 维填充 Q/QV。
2. 新增输出裁剪逻辑：新增 _unpad_mla_result，在 kernel 返回后按保存的 padding 元组（num_kv_heads, qhead_per_kvhead, qhead_per_kvhead_padded）裁剪输出与 LSE 到原始 head 数，并做 contiguous() 保证后续算子安全。
3. 接入 flash_attn_varlen_func 并处理 split 与 hd256 特例：在函数入口对 q/k/v/qv 做 _maybe_contiguous 后，若是 hd256 非连续输入（qv 为 None 且 head_dim 均为 256），统一 contiguous() 以规避上游 vendored kernel 的 stride 假设（TODO 指向 flash-attn PR 2670）；随后调用 _pad_mla_q_heads 获取 padding 元组；若 qv 非

None 且 num_splits < 1（即自动选择），强制 num_splits = 1，因为 FA4 MLA 不实现 split-KV；最后在 kernel 调用后套一层 _unpad_mla_result。

4. 测试配套：将 test_flash_attn_varlen_qv_deepseek_absorbed 参数化扩展为 (seqlen_q, seqlen_k, nheads, nheads_k, num_splits)，覆盖 DeepSeek 风格 MQA、(8,1,1)、GQA、(8,4,1)、GLM TP1 (20,1,0) 及 TP2 (10,1,0)、TP4 (5,1,0) 特例，并增加 return_softmax_lse=True 的 output 与 LSE 双重校验；新增 test_flash_attn_qv_paged_decode_cuda_graph 用 CUDA Graph 捕获 / 回放 GLM TP1 decode 形状，逐位对比 eager；新增 test_flash_attn_hd256_noncontiguous_inputs 验证非连续 hd256 输入与连续输入结果一致。

关键文件：

- python/sglang/kernels/ops/attention/flash_attention_v4.py（模块 注意力后端；类别 source；类型 core-logic；符号 _pad_mla_q_heads, _unpad_mla_result, flash_attn_varlen_func）：FA4 包装器核心修复：新增 MLA head 比例 padding/unpadding，强制 num_splits=1，并处理 hd256 非连续输入；直接决定 GLM-4.7-Flash 能否在 FA4 + CUDA Graph 下正确服务。
- test/registered/kernels/ops/attention/test_flash_attention_4.py（模块 注意力测试；类别 test；类型 test-coverage；符号 test_flash_attn_varlen_qv_deepseek_absorbed, test_flash_attn_qv_paged_decode_cuda_graph, test_flash_attn_hd256_noncontiguous_inputs）：测试配套：扩展 absorbed-MLA QV 参数化到 GLM TP1/TP2/TP4 形状，新增 CUDA Graph 回放与 hd256 非连续输入回归测试，验证 padding/unpadding 在 varlen、decode、图形捕获下 output 与 LSE 均正确。

关键符号：_pad_mla_q_heads, _unpad_mla_result, flash_attn_varlen_func, test_flash_attn_varlen_qv_deepseek_absorbed, test_flash_attn_qv_paged_decode_cuda_graph, test_flash_attn_hd256_noncontiguous_inputs

关键源码片段

python/sglang/kernels/ops/attention/flash_attention_v4.py

FA4 包装器核心修复：新增 MLA head 比例 padding/unpadding，强制 num_splits=1，并处理 hd256 非连续输入；直接决定 GLM-4.7-Flash 能否在 FA4 + CUDA Graph 下正确服务。

```
# 将 MLA Q/QV 的 head 数按 KV group padding 到与 128 行 tile 兼容的 2 的幂
def _pad_mla_q_heads(q, qv, v, pack_gqa):
    # 仅对 MLA 吸收式（qv 存在）且启用 GQA 打包的场景处理
    if qv is None or pack_gqa is False:
        return q, qv, None

    num_heads = qv.shape[-2] # 总 Q/QV head 数
    num_kv_heads = v.shape[-2] # 总 latent KV head 数
    qhead_per_kvhead = num_heads // num_kv_heads

    # 若 head 比例已经是 128 行 tile 的约数 / 倍数，无需 padding
    if 128 % qhead_per_kvhead == 0 or qhead_per_kvhead % 128 == 0:
```

```
        return q, qv, None

    # 将每个 KV group 的 Q head 数向上取到 2 的幂（如 GLM TP1 下 20 -> 32）
    qhead_per_kvhead_padded = 1 << (qhead_per_kvhead - 1).bit_length()

    def pad(x):
        if x is None:
            return None
        prefix = x.shape[:-2]
        # 按 KV group 切分，在 head 维填充到 padded 数量
        x = x.reshape(*prefix, num_kv_heads, qhead_per_kvhead, x.shape[-1])
        x = F.pad(x, (0, 0, 0, qhead_per_kvhead_padded - qhead_per_kvhead))
        return x.reshape(*prefix, num_kv_heads * qhead_per_kvhead_padded, x.shape[-1])

    # 返回 padding 元组，供 _unpad_mla_result 恢复原始 head 布局
    return (pad(q), pad(qv), (num_kv_heads, qhead_per_kvhead, qhead_per_kvhead_padded))


# 将内核输出与 LSE 裁剪回原始 head 数
def _unpad_mla_result(result, head_padding):
    if head_padding is None:
        return result

    num_kv_heads, qhead_per_kvhead, qhead_per_kvhead_padded = head_padding
    out, lse = result
    prefix = out.shape[:-2]

    # 裁剪掉填充的 head，并重新整理为原始布局；contiguous() 保证后续算子安全
    out = out.reshape(*prefix, num_kv_heads, qhead_per_kvhead_padded, out.shape[-1])[
        ..., :qhead_per_kvhead, :
    ]
    out = out.reshape(*prefix, num_kv_heads * qhead_per_kvhead, out.shape[-1]).contiguous()

    # LSE 同样需要裁剪，否则采样阶段会读取到越界 head
    if lse is not None:
        prefix = lse.shape[:-1]
        lse = lse.reshape(*prefix, num_kv_heads, qhead_per_kvhead_padded)[
            ..., :qhead_per_kvhead
        ]
        lse = lse.reshape(*prefix, num_kv_heads * qhead_per_kvhead).contiguous()

    return out, lse

# flash_attn_varlen_func 中接入 padding 与 split 强制的关键片段
q, k, v, qv = [_maybe_contiguous(t) for t in (q, k, v, qv)]

if qv is None and q.shape[-1] == 256 and k.shape[-1] == 256 and v.shape[-1] == 256:
    # vendored hd256 内核假设 dense Q/K/V 布局
    # TODO: 上游 flash-attn PR 2670 合并后可移除该 workaround
```

```
    q, k, v = [t.contiguous() for t in (q, k, v)]

q, qv, mla_head_padding = _pad_mla_q_heads(q, qv, v, pack_gqa)

if qv is not None and num_splits < 1:
    # FA4 MLA 不实现 split-KV，自动选择必须收敛到单 split
    num_splits = 1

# ... 内核调用 ...
result = _unpad_mla_result(result, mla_head_padding)
```

评论区精华

核心讨论围绕性能验证与测试覆盖展开：

- padding 性能开销：BBuf 指出 20→32 padding 让 MLA 内核多处理 60% 的 query head 行，且每层有两次 pad/crop，处于 decode 热路径，要求提供 B200 端到端数据。作者回复没有 B200，改用 GB300 补充了吞吐与 TPOT 表，低并发下 FA4 低于 Triton，并发 64 时反超。
- CUDA Graph 回归测试：BBuf 要求验证真实配置（非确定性 decode + CUDA Graph）并加回归测试，作者新增 test_flash_attn_qv_paged_decode_cuda_graph，在 20 head、num_splits=0 下捕获图并回放，与 eager 逐位对比 output 与 LSE。
- TP2/TP4 与 LSE 覆盖：BBuf 建议补充 (10,1,0) 和 (5,1,0)，同时验证 LSE 裁剪路径，作者均采纳并在测试中断言裁剪后 LSE 与参考一致。
- hd256 workaround 影响范围：BBuf 认为 workaround 影响所有非 QV 的 hd256 调用者，建议单独 PR 或补测试，作者新增 test_flash_attn_hd256_noncontiguous_inputs 并保留 TODO。
- Codex bot P2 提示：指出某个 has_qv=True 用例构建了 qv 参考但 FA4 调用未传 qv 参数，导致测试用普通 attention 与 QV 参考对比；未看到作者明确回复，最终合并版本未保留该处参数化路径，疑已移除。
  - padding 带来的 decode 性能开销 (performance): 作者称没有 B200，改用 GB300 补充了吞吐与 TPOT 数据；低并发下 FA4 低于 Triton，并发 64 时反超。
  - CUDA Graph 捕获与回放回归测试 (testing): 作者新增 test_flash_attn_qv_paged_decode_cuda_graph，在 20 head、num_splits=0 下捕获图并回放，与 eager 逐位对比 output 与 LSE。
  - TP2/TP4 形状与 LSE 验证 (testing): 作者添加 glm-tp2 与 glm-tp4 参数，并让测试校验输出与 LSE 均与参考一致。
  - hd256 非连续输入 workaround 的影响范围 (design): 作者添加 test_flash_attn_hd256_noncontiguous_inputs 验证非连续输入与连续输入一致，并保留 TODO 指向 flash-attn PR 2670。
  - has_qv=True 用例未传 qv 参数 (correctness): 未看到作者明确回复；最终合并版本未保留该处 has_qv 参数化路径，疑已移除。
  - 真实数据集精度对比 (question): 作者在 PR 描述中添加 GSM8K 200 题精度表：FA4 0.830 vs Triton 0.820。

风险与影响

- 风险：
  1. 解码热路径性能风险：padding 使内核处理 60% 额外 query head 行，且每个 layer 两次 pad/crop。从 GB300 数据看，低并发下 FA4 output tok/s 明显低于 Triton（如 Chat 1K/1K 并发 1：90.86 vs 138.41），高并发才反超，需要关注真实场景的并发分布。
  2. 正确性 / 回归风险：_unpad_mla_result 依赖 head 维连续布局，contiguous() 调用保证安全，但新增的 padding 逻辑覆盖所有使用 FA4 MLA 的模型，虽然兼容比例直接返回，但 num_splits=1 的强制改写会影响后续 FA4 对 split-KV 的支持，属于潜在行为变更。
  3. 上游依赖风险：hd256 contiguous workaround 依赖 flash-attn 上游 PR 2670，若升级到的 flash-attn-4 版本未包含该修复，workaround 可能失效；反之合并后需及时清理 TODO。
  4. CI 状态：PR 描述中 CI 显示 （Run #31140424842 与 #31140424713），虽最终合并，但需要确认失败是否为已知 flaky 或遗留问题。 - 影响：直接影响：启用 GLM-4.7-Flash 在 Blackwell（SM100/SM110）上用 FA4 后端，默认非确定性 decode + CUDA Graph 正常工作；同时通过 padding 绕开了 bs>1 时 pack_gqa=False QV varlen decode 路径的跨序列 paged KV 读取问题。对 DeepSeek 等 head 比例天然兼容的 MLA 模型无影响（直接返回）。团队需持续维护 wrapper 中 padding 与上游 flash-attn 的版本同步，测试矩阵新增了 GLM TP1/TP2/TP4 与 CUDA Graph 场景，提高了 FA4 后端的回归保护。 - 风险标记：解码热路径变更，padding 引入额外开销，依赖上游 flash-attn 修复，CI 状态异常

关联脉络

- PR #34159 Fix deterministic inference all-reduce for tp>1: 该 PR 与本文同属提高多卡 / 图模式下推理一致性的系列修复；本文新增 CUDA Graph 回放一致性测试，与 deterministic all-reduce 的稳定性目标一致。
- PR #33423 Deterministic gumbel sampling: clamp u=1 so masked tokens can't be sampled: 同为确定性 / 一致性方向的修复，涉及采样路径的位级一致性，与本 PR 对 CUDA Graph 回放一致性的关注点相关。