PR #34421 完整报告

sgl-project/sglang

[AMD] [Perf] Fuse GatedDeltaNet QKVZBA split/reshape/cat into a single Triton kernel for Qwen3.5-architecture MoE on HIP

合并时间：2026-08-13 15:18

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34421

执行摘要

- 一句话：GDN 融合 kernel 扩展到头组比例 8，AMD 解码提速 73.5%
- 推荐动作：值得精读，是一个小而精准的性能优化样板。两个设计决策值得借鉴：一是用模块级白名单常量 + 后端守卫把「启用范围」与「实现」分离，做到对未验证后端零影响；二是用 profiling 数据（每 kernel us、每迭代总耗时）驱动优化，并用 Kernel-to-E2E 一致性分析（预测 ITL -5.19% vs 实测 -3.61%）解释收益边界。缺点是无测试配套，后续若该 kernel 在更多 AMD 卡或新比例上启用，应补充 kernel 级数值等价测试。

功能与动机

PR body 给出的动机非常量化：Qwen3_5GatedDeltaNet.forward 只在 num_v_heads // num_k_heads 为 1、2、4 时使用融合 Triton split/reshape/cat 路径，比例 8 的布局回退到 *eager* 序列（fix_query_key_value_ordering -> .contiguous() -> torch.cat），每 GDN 层付出 4 次独立数据搬运（17.0 us、4 次 launch）；全模型 GDN 层合计约占每次 *decode* 迭代 1.17 ms，且 whole-trace 的 *cat*/contiguous/reshape 桶是 GPU 时间第 6 大 kernel 类别（4.9%）。融合 kernel 本就覆盖全部工作，只是从未对比例 8 启用，且硬编码 num_warps=1 对每次要搬运 8 个 v-head 数据的程序过窄。

实现拆解

1. 数据驱动的瓶颈定位：作者在 MI355X（TP=8、aiter 后端）上用 CUDA graph trace 定位到 GDN 层 *eager* 回退的 4 个数据搬运 kernel（b.contiguous() 3.8 us、a.contiguous() 4.0 us、CatArrayBatchedCopy 4.8 us、为 z.reshape(...) 物化非连续 z 的 4.4 us，合计 17.0 us），并得出「融合 kernel 已具备全部能力，只缺启用以调参」的结论。此步无代码改动。
2. kernel 启动配置自适应（python/sglang/kernels/ops/attention/triton_gdn_fused_proj.py）：在 fused_qkvzba_split_reshape_cat_contiguous 中，num_warps 由硬编码 1 改为在 _use_aiter（SGLANG_USE_AITER 且 is_hip()）下按 v_elems_per_program = (num_heads_v // num_heads_qk) * head_v 计算启发式（<= 512 元素用 1 warp，否则 4 warps）。对已启用的比例 1/2/4 在当前 head dim 下仍解析为 1，launch 配置与旧行为完全一致；kernel body 未改动。这一步是性能收益的实际来源。
3. dispatch 白名单模块化（python/sglang/srt/models/qwen3_5.py）：新增模块级常量 _GDN_FUSED_QKVZBA_RATIOS = (1, 2, 4, 8) if _use_aiter else (1, 2, 4),

Qwen3_5GatedDeltaNet.forward 的分发条件从内联 in [1, 2, 4] 改为 in_GDN_FUSED_QKVZBA_RATIOS。非 aiter 后端元组与旧列表逐元素相同，CUDA/CPU/NPU 控制流完全不变；eager 回退完整保留，供白名单之外的所有布局使用。

4. 验证与配套：GSM8K（1319 题、并行 512、max-tokens 2048，TP=8 MI355X）得分 0.9795 vs 基线 0.9787；trace 确认 GDN 模块内 direct_copy_kernel_cuda 与 CatArrayBatchedCopy 归零、融合 kernel 每 GDN 调用恰好 1 launch；conc4/conc64 E2E 吞吐 +3.79%/+1.23%，ITL -3.61%/-2.46%。PR checklist 未勾选单元测试，未新增测试文件，依赖单模型精度与 profiling 验证。

关键文件：

- python/sglang/srt/models/qwen3_5.py（模块 模型层；类别 source；类型 core-logic；符号 _GDN_FUSED_QKVZBA_RATIOS, Qwen3_5GatedDeltaNet.forward）：Qwen3_5GatedDeltaNet.forward 的 dispatch 白名单从内联 [1, 2, 4] 提升为模块级 _GDN_FUSED_QKVZBA_RATIOS，aiter 下加入比例 8，是决定哪些后端走融合路径的主路径改动；非 aiter 元组与旧列表完全一致，保证 CUDA/CPU/NPU 行为不变。
- python/sglang/kernels/ops/attention/triton_gdn_fused_proj.py（模块 算子层；类别 source；类型 core-logic；符号 fused_qkvzba_split_reshape_cat_contiguous）：融合 kernel wrapper 的 num_warps 从硬编码 1 改为按每 program 搬运元素数自适应（最多 4 warps），解决了宽 head-group（比例 8、head_v=128 时每 program 搬 1024 元素）在单 warp 上串行化的问题，是性能收益的实际来源；启发式被 _use_aiter 守卫，其他后端 launch 配置不变。

关键符号：fused_qkvzba_split_reshape_cat_contiguous, Qwen3_5GatedDeltaNet.forward

关键源码片段

python/sglang/srt/models/qwen3_5.py

Qwen3_5GatedDeltaNet.forward 的 dispatch 白名单从内联 [1, 2, 4] 提升为模块级 _GDN_FUSED_QKVZBA_RATIOS，aiter 下加入比例 8，是决定哪些后端走融合路径的主路径改动；非 aiter 元组与旧列表完全一致，保证 CUDA/CPU/NPU 行为不变。

```
# python/sglang/srt/models/qwen3_5.py
# 多头组比例（num_v_heads // num_k_heads）白名单：这些布局走融合的
# split/reshape/cat Triton 路径。AMD/aiter 后端额外启用比例 8，可省掉 eager
# 回退中的两次 .contiguous() 拷贝与 torch.cat；其余后端保持原元组，控制流不变。
_GDN_FUSED_QKVZBA_RATIOS = (1, 2, 4, 8) if _use_aiter else (1, 2, 4)

# Qwen3_5GatedDeltaNet.forward：输入投影之后的 dispatch
if (
    self.num_v_heads // self.num_k_heads in _GDN_FUSED_QKVZBA_RATIOS
    and not _is_npu
):
    # 融合路径：单个 Triton kernel 完成 QKVZBA 的 split/reshape/cat，
    # 且把 z 写成连续内存，令后续 z.reshape(...) 变为零开销。
    if _is_cpu:
        num_k_heads_tp = self.num_k_heads // self.attn_tp_size
```

```
            num_v_heads_tp = self.num_v_heads // self.attn_tp_size
        else:
            num_k_heads_tp = triton.cdiv(self.num_k_heads, self.attn_tp_size)
            num_v_heads_tp = triton.cdiv(self.num_v_heads, self.attn_tp_size)
        mixed_qkv, z, b, a = fused_qkvzba_split_reshape_cat_contiguous(
            projected_states_qkvz,
            projected_states_ba,
            num_k_heads_tp,
            num_v_heads_tp,
            self.head_k_dim,
            self.head_v_dim,
        )
    else:
        # eager 回退：仅服务白名单之外的布局（当前即非 aiter 下的比例 8+）。
        query, key, value, z, b, a = self.fix_query_key_value_ordering(
            projected_states_qkvz, projected_states_ba
        )
        b = b.contiguous()
        a = a.contiguous()
        query, key, value = map(lambda x: x.reshape(x.shape[0], -1), (query, key, value))
        mixed_qkv = torch.cat((query, key, value), dim=-1)
```

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

融合 kernel wrapper 的 num_warps 从硬编码 1 改为按每 program 搬运元素数自适应（最多 4 warps），解决了宽 head-group（比例 8、head_v=128 时每 program 搬 1024 元素）在单 warp 上串行化的问题，是性能收益的实际来源；启发式被 _use_aiter 守卫，其他后端 launch 配置不变。

```
# python/sglang/kernels/ops/attention/triton_gdn_fused_proj.py
# 每个 program 要搬运的 v/z 元素数 = 每个 k-head 对应的 v-head 数 × head_v。
# 小多头组（<= 512 元素）单 warp 最优；宽布局（如比例 8、head_v=128 时每个
# program 要搬 1024 个元素）需要更多 warp lane，否则向量 load/store 会在
# 单个 warp 上串行化。阈值基于 MI355X 调优，因此只作用于 HIP/aiter 路径，
# 其余后端保持原 num_warps=1，launch 配置不变。
a = torch.empty_like(b)
grid = (batch * seq_len, num_heads_qk)

num_warps = 1
if _use_aiter:
    v_elems_per_program = (num_heads_v // num_heads_qk) * head_v
    num_warps = 1 if v_elems_per_program <= 512 else 4

fused_qkvzba_split_reshape_cat_contiguous_kernel[grid](
    mixed_qkv,
    z,
    b,
    a,
    # ... 中间若干步长 / 指针参数省略 ...
    num_heads_v,
```

```
        head_qk,
        head_v,
        num_warps=num_warps, # 原为硬编码 1，现按搬运量自适应
        num_stages=3,
    )
    return mixed_qkv, z, b, a
```

评论区精华

review 只有一条实质线程，围绕「作用域控制」展开：HaiShaw 在 triton_gdn_fused_proj.py 的 num_warps 启发式 diff 上要求 "let's confine this changes to hip/aiter only."；作者 yichiche 回复 "Add if_hip guard, kick-off ci again." 并新增 _use_aiter 守卫后重新触发 CI。结论：启发式仅在 HIP/aiter 路径生效，CUDA/CPU/NPU 后端保持 num_warps=1，HaiShaw 随后批准合并。这条讨论的价值在于：性能优化默认保守，未经测量验证的改动不进入其他后端。

- num_warps 启发式是否应限定在 HIP/aiter 路径 (design): 接受评审意见：num_warps 启发式仅当 _use_aiter（SGLANG_USE_AITER 且 is_hip()）时生效，其余后端保持 num_warps=1；HaiShaw 随后批准合并。

风险与影响

- 风险：
  1. 缺少测试覆盖：ratio=8 融合路径首次启用，无 kernel 级数值等价测试，仅靠 GSM8K 统计等价（0.9795 vs 0.9787，落在采样噪声内），未做逐位对比。
  2. 静态阈值风险：512 元素阈值只在 MI355X 上调优，其他 AMD GPU 上 4 warps 可能过分配或欠分配，虽不产生错误但可能丢性能；作者刻意将改动锁在 _use_aiter 路径以控制风险范围。
  3. 对非 HIP 后端影响为零：qwen3_5.py 的元组在非 aiter 下与旧列表完全相同，triton_gdn_fused_proj.py 的 num_warps 启发式被 _use_aiter 守卫，CUDA/CPU/NPU 的 branch 与 launch 配置均不变。
  4. 热路径变更：qwen3_5.py 的 dispatch 位于 decode/prefill 每一轮的 GDN 前向热路径，但改动仅是元组查表，无额外开销。 - 影响：对 AMD/aiter + Qwen3.5 架构大 MoE（含 GatedDeltaNet 线性注意力层）模型：decode 每迭代省约 0.86-0.93 ms（每个 GDN 层 12.5-13.5 us，-73.5%）；conc4 吞吐 +3.79%、ITL -3.61%，conc64 吞吐 +1.23%、ITL -2.46%；prefill 同样受益（conc4 每层区域 60.4 -> 41.9 us）。decode GDN 层 kernel 数从 13 降到 10。对仓库整体影响面可控：分发只在 Qwen3_5GatedDeltaNet.forward 单一调用点，无新配置项、无部署改动；CUDA/CPU/NPU 用户无感知。团队层面，该 PR 提供了「融合 kernel 已存在但 dispatch 白名单过窄」的修复范式，可复用于其他 head-group 布局或设备族。 - 风险标记：缺少测试覆盖，AMD/aiter 专属变更，decode 热路径变更，静态调优阈值

关联脉络

- PR #33623 [Kimi K3] Fuse MLA gate projection into QKV-A GEMM: 同为前向投影 GEMM/ 数据搬运融合优化线，目标都是减少每层 kernel 数；可对照其后续回滚评估本 PR 的稳健性设计。
- PR #34642 Revert "[Kimi K3] Fuse MLA gate projection into QKV-A GEMM": 该融合优化因长序列回归被回滚，提示融合优化必须搭配范围守卫与数值验证；本 PR 的 aiter-only 守卫 + GSM8K 等价验证正是对这一教训的回应。