PR #50593 完整报告

vllm-project/vllm

[Kimi-K3][AMD] Fuse AttnRes state updates and norms

合并时间：2026-08-05 00:32

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50593

执行摘要

- 一句话：融合 AttnRes 前缀更新与归一化，AMD 解码吞吐 +14.2%
- 推荐动作：值得精读。两个核心看点：其一，如何用编译期常量开关（HAS_DELTA/WRITE_BLOCK/APPLY_OUTPUT_NORM）在一个 Triton 内核中组合 7 个算子的同时保持数值语义逐位一致，特别是 BF16 存储精度往返的设计；其二，性能 PR 的完整验证链条（契约测试矩阵、HIP graph 重放、GSM8K 双评估器、独立 cherry-pick 复测）可作团队模板。建议关注后续 #50682 跨层 delta 交接是否落地，以及融合内核在 Triton 升级后的稳定性。

功能与动机

PR body 明确指出：AMD Kimi-K3 解码器此前把 AttnRes 周边的操作拆成多个独立内核（前缀残差更新、块残差存储、AttnRes 加权聚合、输入 RMSNorm、自注意力输出加前缀、MLP AttnRes 聚合、post-attention RMSNorm），这些操作反复读写完整 hidden-state 张量并引入额外内核启动开销，而"解码阶段每个请求只处理少量 token，却要为每个生成 token 执行 69 层 KDA 层"，使该开销被显著放大。目标是把前缀更新、块写入、delta 累加与两层 RMSNorm 折叠进现有 AttnRes Triton 内核，让更新后的前缀值停留在寄存器中复用于块写入、聚合与归一化，避免中间张量物化。

实现拆解

1. Triton 内核融合（vllm/models/kimi_k3/amd/ops/attn_res.py，+101/-40）：_attn_res_kernel 新增 delta_ptr、output_norm_weight_ptr、block_write_idx 参数与 HAS_DELTA、WRITE_BLOCK、APPLY_OUTPUT_NORM 三个编译期常量开关，attn_res() 调用签名同步扩展。内核内前缀与 delta 均以 FP32 加载累加，随后经 BF16 存储精度往返再作为残差源，逐位匹配原始实现；块写入直接落在 block-residual 槽位，输出 RMSNorm 的方差归约留在 FP32。新增零块快速路径（num_blocks == 0 时 softmax 只有一个源，输出恒等于前缀，直接跳过 streaming softmax 循环）。启动配置采用启发式：num_tokens >= 256 或 num_blocks <= 1 时用单源 tile（BLOCK_L=1）服务 prefill 大批量，否则用多源 tile（BLOCK_L=4）摊薄 decode 的源循环开销。
2. 调用契约与层逻辑重构（vllm/models/kimi_k3/amd/linear.py，+16/-9）：_apply_attn_res 新增 delta、output_norm、block_write_idx 三个关键字参数并透传给内核；forward_attn_residual 把输入 RMSNorm（self.input_layernorm）和块写入下沉到第一段 AttnRes，把自注意力输出改为 prefix_delta 传给第二段 AttnRes，并把

post-attention RMSNorm（self.post_attention_layernorm）折叠进第二段。原 block_residual[:, self.block_write_idx, :].copy_(prefix_sum) 独立拷贝被删除，块写入层的前缀置空逻辑保持不变，返回契约（prefix_sum, block_residual）不变。

3. 测试配套（tests/models/kimi_k3/test_amd_attn_res.py，+91）：新增参数化测试 test_amd_attn_res_fused_contract，5 组用例覆盖 1/7/17/3/320 token、0-8 块、hidden size 128/1024/7168（生产规模）与三种开关组合，同时校验返回值、前缀原地更新、块存储的不变性与输出连续性；对 FP32 PyTorch 参考实现以 atol=8e-2、rtol=3e-2 对比（容差吸收 BF16 舍入）。原 reference 测试适配新签名（delta=None、block_write_idx=-1、output_norm_eps=0.0）。
4. 正确性与性能验证：PR body 附 HIP graph 捕获 / 重放验证（TP8、9K 长上下文、不同 shape 重放、无 GPU 内存访问错误）；gfx950 算子基准显示 1-128 token 区间延迟改善 36%-44%、512 token 改善 5.8%；100K prompt + 1K generation 端到端吞吐提升 14.2%、延迟下降 12.4%；50K-100K 长上下文 prefill 基本持平。Issue 评论补充了容器化复现脚本（podman + 源码 overlay）、完整环境变量与 GSM8K 双评估器 A/B。

关键文件：

- vllm/models/kimi_k3/amd/ops/attn_res.py（模块 残差算子；类别 source；类型 core-logic；符号 _attn_res_kernel, attn_res）：融合内核的载体：_attn_res_kernel 新增 delta 累加、块写入与输出 RMSNorm，是本次性能收益的核心来源，也是数值语义保持的关键所在。
- vllm/models/kimi_k3/amd/linear.py（模块 解码层；类别 source；类型 data-contract；符号 _apply_attn_res, forward_attn_residual）：数据契约与层逻辑重构：_apply_attn_res 与 forward_attn_residual 的改动定义了 delta= 契约并内化块写入与两层 RMSNorm，是后续跨层融合的基础。
- tests/models/kimi_k3/test_amd_attn_res.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_amd_attn_res_fused_contract, test_amd_attn_res_matches_reference）：新增融合契约测试矩阵，是数值语义与原地更新语义的守护者，验证了内核重写的正确性。

关键符号：_attn_res_kernel, attn_res, _apply_attn_res, forward_attn_residual, test_amd_attn_res_fused_contract

关键源码片段

vllm/models/kimi_k3/amd/linear.py

数据契约与层逻辑重构：_apply_attn_res 与 forward_attn_residual 的改动定义了 delta= 契约并内化块写入与两层 RMSNorm，是后续跨层融合的基础。

```
# KimiDecoderLayer：把 AttnRes 周边的 7 个小内核折叠进 Triton 融合内核后，
# 层推理控制流显著简化；_apply_attn_res 的签名扩展为
# （delta, output_norm, block_write_idx）三个可选参数，默认值保持向后兼容。
def forward_attn_residual(
    self,
    positions: torch.Tensor,
    hidden_states: torch.Tensor,
```

```
    block_residual: torch.Tensor,
) -> tuple[torch.Tensor, torch.Tensor]:
    prefix_sum = hidden_states
    # 第一段 AttnRes：输入 RMSNorm 作为 output_norm 传入，块写入索引仅对
    # 块写入层生效（-1 表示不写），前缀更新与写块都在内核内完成
    hidden_states = _apply_attn_res(
        prefix_sum,
        block_residual,
        self.self_attention_res_proj,
        self.self_attention_res_norm,
        self.prev_valid_blocks,
        output_norm=self.input_layernorm,
        block_write_idx=(self.block_write_idx if self.is_block_write_layer else -1),
    )
    if self.is_block_write_layer:
        # 块写入已由内核完成，前缀残差不再跨内核传递
        prefix_sum = None

    hidden_states = self._run_self_attn(positions, hidden_states)

    if prefix_sum is None:
        prefix_sum = hidden_states
        prefix_delta = None
    else:
        # 自注意力输出不再物化“prefix_sum + hidden_states”中间张量，
        # 而是通过 delta= 契约交给第二段 AttnRes 在内核内做 FP32 累加
        prefix_delta = hidden_states

    mlp_valid_blocks = self.prev_valid_blocks + (
        1 if self.is_block_write_layer else 0
    )
    # 第二段 AttnRes：MLP 前的加权聚合与 post-attention RMSNorm 融合，
    # delta 语义与原实现“prefix_sum + hidden_states”完全一致
    hidden_states = _apply_attn_res(
        prefix_sum,
        block_residual,
        self.mlp_res_proj,
        self.mlp_res_norm,
        mlp_valid_blocks,
        delta=prefix_delta,
        output_norm=self.post_attention_layernorm,
    )

    hidden_states = self.mlp(hidden_states)
    prefix_sum = prefix_sum + hidden_states
    return prefix_sum, block_residual
```

评论区精华

1. 跨层 delta 交接提案（Fangzhou-Ai，关联 #50682）：本 PR 融合后层末尾仍剩 prefix_sum = prefix_sum + hidden_states，若把 MLP 结果经 delta= 契约直接传给下一层首个 AttnRes，可消除最后的 post-MoE 交接启动；合并流只在 aux 采样点、流水线边界与最终输出处物化。最终确定作为稳定后的后续工作，作者 LiuYinfeng01 已本地实现 PoC（仅 linear.py +25/-6）并通过 TP8 验证，未提交。
2. 复现与 A/B 要求（hongxiayang）：要求性能 PR 提供 vllm serve 命令、环境变量、bench 命令与前后对比结果；LiuYinfeng01 补齐完整环境（VLLM_ROCM_USE_AITER=1、AITER_ROCM_ARCH=gfx950 等）与 GSM8K A/B，随后获批。
3. GSM8K 双评估器结果（LiuYinfeng01）：vLLM 自研评估 93.8590% -> 95.1478%（+1.29pp），lm_eval 0.4.12 由 96.7400% 降至 95.9818%（-0.76pp），配对 McNemar p = 0.0755 无统计显著性；baseline 无乱码。数值语义由此过关，但评估器方向不一致被记录待观察。
4. 独立 current-stack 复测（Fangzhou-Ai）：将 PR cherry-pick 到固定基线 0601850（结果提交 d6d6bf26），同一容器镜像摘要与 mla_gluon.py SHA256，避免部分源码 overlay 的 ABI 不匹配，复测结论与原作者一致。

- 跨层 MLP delta 交接提案（关联 #50682）(design): 确定为稳定后的协调后续（#50682）；LiuYinfeng01 已本地实现 PoC（仅 linear.py +25/-6）并完成 TP8 验证，未提交。
- 性能 PR 的复现步骤与 A/B 要求 (question): LiuYinfeng01 补交完整环境变量（VLLM_ROCM_USE_AITER 等）、serve 命令、容器化复现脚本（podman + overlay 镜像）与 GSM8K A/B，随后获批。
- GSM8K 数值一致性验证（双评估器方向不一致）(testing): 判断差异无统计显著性，正确性验收通过；评估器方向的差异被记录待观察。
- 独立 current-stack A/B 复测 (testing): 独立复测通过，验证结论与原作者一致。

风险与影响

- 风险：
  - 精度语义敏感：_attn_res_kernel 依赖“FP32 累加 -> BF16 存储精度往返 -> 再作残差源”的顺序才能逐位匹配原始实现，任何编译器重排或后续 Triton 行为变化都可能引入数值漂移；GSM8K 双评估器方向不一致（vLLM eval +1.29pp、lm_eval -0.76pp）提示需要保留观察窗口。
  - 收益场景局限：36%-44% 的算子级改善只在 1-128 token 解码区间，512 token 仅 5.8%，长 prefill 中性；若 --max-num-seqs 或批量策略变化，收益曲线会明显移动。
  - 调度启发式为经验值：num_tokens >= 256 或 num_blocks <= 1 切换 BLOCK_L 的阈值未做全批量网格搜索，边缘批量下可能不是最优配置。
  - 内核复杂度上升：新增 debug_barrier 控制寄存器活性、evict_first 缓存策略，较依赖编译器行为，Triton 版本升级存在回归风险。
  - 影响面隔离：改动仅在 vllm/models/kimi_k3/amd/ 路径内，NVIDIA 与通用路径零接触；但 _apply_attn_res 的新参数以默认值（None/-1）保持向后兼容，其他调用方无需改动。
- 影响：

- 用户侧：AMD MI355X（gfx950）上 Kimi-K3 长文本生成吞吐中位数从 29.86 提升到 34.09 output tokens/s（+14.2%），端到端延迟从 33.50 s 降至 29.34 s（-12.4%）；operator 级解码延迟降低 36%-44%。
- 系统侧：每个 token 生成流程减少约 6 次内核启动与数轮 full hidden-state 内存读写，69 层 KDA 逐层放大收益；对长上下文、低并发解码负载尤其有利。
- 团队侧：确立 AttnRes delta= 数据契约，为 #50682 的跨层 MLP-delta 交接打下基础；同时形成“单元测试 + HIP graph 重放 + 双评估器 GSM8K + 独立复测”的性能 PR 验证模板。
- 风险标记：核心解码路径变更，精度语义敏感，单平台验证（AMD gfx950），数据契约扩展，收益依赖小批量场景

关联脉络

- PR #50654 [ROCm][Perf] Kimi-K3 Fused kernel for KDA decode: 同一 Kimi-K3 AMD 性能融合主线的前序工作（KDA 解码融合内核，e2e +7%），本 PR 是该系列在 AttnRes 路径上的第二块拼图。
- PR #48223 [Perf][ROCm] Dual-stream decode with hipgraphs: 同为 ROCm 解码性能优化思路（双流 shared experts 解码），与本 PR 的“减少小内核启动与内存流量”目标一致。
- PR #50268 [Hardware][AMD] Enable fused bf16→fp32 router GEMM on ROCm: 同为 AMD 融合内核系列（消除拷贝内核），体现了 vLLM ROCm 路径持续把内存密集小算子折进内核的理念。