

PR #55239 完整报告

vllm-project/vllm

[ROCm][Bugfix] Route GLM-5.3-Flash MTP through ragged sparse MLA

合并时间: 2026-09-11 09:13

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55239>

执行摘要

- 一句话: ROCm 上放宽 sparse-MLA 分派谓词, 修复 GLM-5.3 MTP 静默错算
- 推荐动作: 值得精读, 但只需要读一个函数。建议重点关注三点: (1) `_use_rocm_sparse_triton()` 从“形状判定”转为“几何能力判定”的依据是否成立——即 ragged Triton 内核是否真的对多 token 验证行与普通 decode 行有相同前提, 这是整个修复的立论基础; (2) 把“静默回退到不兼容内核”视为可复现的失败模式, 评估是否需要在谓词为 False 且形状不被支持时加 `assert/` 告警, 而不是继续依赖人工 GSM8K 验证; (3) 若你是 ROCm 后端或投机解码的维护者, 建议补一条能进 CI 的端到端精度用例 (或至少在 AMD CI 中挂上该测试文件), 否则这次修复在回归保护上是空白的。对于只想了解结论的读者, 记住一句话即可: 这是一次“放宽分派谓词 + 补形状用例”的小改动, 修的是静默错算, 不是新功能。

功能与动机

PR body 明确说明: 现有 selector 只把普通的 rope-free BF16 decode 路由到 Triton ragged 内核, 当 `max_query_len > 1` 或 `num_decode_tokens != num_decodes` (即投机解码的验证行) 时会被排除, 这些行“fall through to an incompatible AITER kernel and produce corrupted output leading to 0% correct on GSM8k”。实测数据也印证了危害: 修复前 MTP5 运行输出重复 token 且有效准确率 0%, 修复后同一硬件 (4x MI355X、TP=4) 上 GSM8K 达到 91.74% (eager 基线 91.13%, 无 MTP 全图 90.67%), MTP5 耗时 153.1s 对比无 MTP 的 173.0s。该改动实现了 issue #53943 讨论中提出的“基于几何 (geometry-based) 分派”方案; issue #54369 跟踪的 draft step 之间重建 sparse-MLA/kpool metadata 的性能开销不在本 PR 范围内。

实现拆解

1. 变更入口: `vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py` 的模块级谓词 `_use_rocm_sparse_triton()`, 它是 `ROCMAiterMLASparseImpl` 选择 Triton ragged sparse-MLA 还是 AITER sparse MLA 的唯一开关。
2. 核心逻辑改造: 删除局部变量 `plain_decode = num_decode_tokens == num_decodes`, 并把返回条件从 `not fp8 and head_size == kv_lora_rank and plain_decode and (num_prefills > 0 or (num_decodes > 0 and max_query_len == 1))` 收敛为 `not fp8 and head_size == kv_lora_rank and (num_prefills > 0 or num_decodes > 0)`。语义上不再按“批次形状是否规整”排除, 而是按“内核能力几何”判定: ragged Triton 内核是逐 query

token 索引 metadata 的，所以多 token 投机验证行与普通 decode 行具备相同的能力前提。FP8 KV cache 与 `head_size != kv_lora_rank` (带 rope 的几何) 仍保持原有 AITER 分派，未受影响。

3. 可读性配套: docstring 由单行说明扩写为带原理的说明 (“The ragged Triton kernel indexes metadata per query token, so multi-token speculative verification rows have the same capability requirements as plain decode rows.”)，把这次放宽的依据固化在代码里，避免以后被当成过度宽松的谓词再收回去。
4. 测试配套: `tests/v1/attention/test_rocm_glm5next_sparse.py` 的 `test_rocm_sparse_triton_route` 参数化表更新——原用例 (“auto”, 512, 0, 2, 4, 2) 由 `False` 翻转为 `True` (正是 MTP 验证形状)，新增 (“auto”, 512, 0, 2, 12, 6, `True`) (更深的投机形状，覆盖更深 MTP) 与 (“auto”, 512, 0, 0, 0, 0, `False`) (整批无活跃行时不选 Triton，钉住 `num_prefills/num_decodes` 全为 0 的边界)，并补上 docstring “Validate Triton routing for prefill, decode, and MTP verification.”。
5. 未改动的部分: `fit_kpool_indices_to_aiter()`、FP8 分派路径、kpool 索引裁剪逻辑均未触碰；“draft step 之间重建 metadata”的性能问题按 PR body 归到 issue #54369。

关键文件:

- `vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py` (模块 稀疏注意力; 类别 source; 类型 core-logic; 符号 `_use_rocm_sparse_triton`) : 唯一源码改动文件, `_use_rocm_sparse_triton()` 谓词放宽直接决定 ROCm sparse-MLA 在 Triton ragged 与 AITER 两条内核路径之间的分派, 是本次正确性修复的全部逻辑。
- `tests/v1/attention/test_rocm_glm5next_sparse.py` (模块 单元测试; 类别 test; 类型 test-coverage; 符号 `test_rocm_sparse_triton_route`) : 配套回归测试, 更新 `test_rocm_sparse_triton_route` 的参数化表: 把 MTP 验证形状从 `False` 翻转为 `True`, 并补齐深投机与空批次两个边界用例; 但它只断言路由布尔值, 不验证内核输出。

关键符号: `_use_rocm_sparse_triton`, `test_rocm_sparse_triton_route`

关键源码片段

`vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py`

唯一源码改动文件, `_use_rocm_sparse_triton()` 谓词放宽直接决定 ROCm sparse-MLA 在 Triton ragged 与 AITER 两条内核路径之间的分派, 是本次正确性修复的全部逻辑。

```
# vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py
# 该谓词是 ROCMAiterMLASparseImpl 选择内核的唯一开关:
# True -> Triton ragged sparse-MLA (逐 query token 索引 metadata)
# False -> AITER sparse MLA
# 注意签名里的 num_decode_tokens / max_query_len 在本 PR 后已不再参与判断,
# 只是为了让调用方与测试不用改签名而保留。
def _use_rocm_sparse_triton(
    *,
    kv_cache_dtype: str,
    head_size: int,
    kv_lora_rank: int,
    num_prefills: int,
```

```

num_decodes: int,
num_decode_tokens: int,
max_query_len: int,
) -> bool:
    """Select the rope-free BF16 path not supported by AITER sparse MLA.

    The ragged Triton kernel indexes metadata per query token, so multi-token
    speculative verification rows have the same capability requirements as
    plain decode rows.
    """

    # 两条硬性能力条件，保持与本 PR 之前一致：
    # 1) FP8 KV cache 仍归 AITER 路径（本 PR 明确不动这条）；
    # 2) head_size == kv_lora_rank 表示“不带 rope 的 BF16 几何”，
    # 这正是 AITER sparse MLA 不支持的组合。
    return (
        not kv_cache_dtype.startswith("fp8")
        and head_size == kv_lora_rank
        # 只要这一批存在活跃行（有 prefill 行或有 decode 行）就交给 Triton。
        # 修复前这里还额外要求 plain_decode (num_decode_tokens == num_decodes)
        # 且 max_query_len == 1，于是 MTP 的多 token 验证行被排除，
        # 静默落到与 rope-free BF16 不兼容的 AITER 内核，产出损坏结果。
        and (num_prefills > 0 or num_decodes > 0)
    )

```

tests/v1/attention/test_rocm_glm5next_sparse.py

配套回归测试，更新 `test_rocm_sparse_triton_route` 的参数化表：把 MTP 验证形状从 False 翻转为 True，并补齐深投机与空批次两个边界用例；但它只断言路由布尔值，不验证内核输出。

```

# tests/v1/attention/test_rocm_glm5next_sparse.py
@pytest.mark.parametrize(
    (
        "kv_cache_dtype",
        "head_size",
        "num_prefills",
        "num_decodes",
        "num_decode_tokens",
        "max_query_len",
        "expected",
    ),
    [
        ("auto", 512, 1, 0, 0, 32, True), # 纯 prefill: 走 Triton
        ("auto", 512, 1, 2, 2, 32, True), # prefill + decode 混合, token 数与请求数对齐
        ("auto", 512, 0, 2, 2, 1, True), # 普通 decode: max_query_len == 1
        ("fp8", 512, 1, 0, 0, 32, False), # FP8 KV cache: 保持 AITER 路径
        ("auto", 576, 1, 0, 0, 32, False), # head_size != kv_lora_rank: 非 rope-free, 走 AITER
        # 本 PR 的核心翻转点: num_decode_tokens != num_decodes 且 max_query_len > 1,
        # 即投机解码的多 token 验证行, 修复前为 False (静默走 AITER 导致错算)。
        ("auto", 512, 0, 2, 4, 2, True),
    ]
)

```

```

# 新增：更深的投机验证形状（12 个 decode token / 6 个请求），
# 对应 PR 描述里 MTP5 的实际几何。
("auto", 512, 0, 2, 12, 6, True),
# 新增：整批无活跃行时必须返回 False，钉住 (num_prefills > 0 or num_decodes > 0)
的下界。
("auto", 512, 0, 0, 0, 0, False),
],
)
def test_rocm_sparse_triton_route(
    kv_cache_dtype,
    head_size,
    num_prefills,
    num_decodes,
    num_decode_tokens,
    max_query_len,
    expected,
):
    """Validate Triton routing for prefill, decode, and MTP verification."""
    assert (
        _use_rocm_sparse_triton(
            kv_cache_dtype=kv_cache_dtype,
            head_size=head_size,
            # kv_lora_rank 固定为 512，与上面 head_size == 512 的用例配对。
            kv_lora_rank=512,
            num_prefills=num_prefills,
            num_decodes=num_decodes,
            num_decode_tokens=num_decode_tokens,
            max_query_len=max_query_len,
        )
        is expected
    )

```

评论区精华

本 PR 的讨论几乎全部是流程性的，缺少行级技术交锋，这一点本身值得记录：

coderabbitai[bot]: “No actionable comments were generated in the recent review. 🤖” (本轮 review 未产出可执行意见)

claude[bot]: “This pull request is from a fork — automated review is disabled.” (fork PR 关闭自动评审，需维护者手动 @claude review，本次未触发)

dllehr-amd: “/ci retry”; jamesETsmith 连续三次 “/ci run” “/ci retry”，最终由 tjtnaa 给出 APPROVED。

也就是说，这个正确性修复的“评审证据”基本是作者自测的 GSM8K 数据（91.74% vs 修复前 0%）加上一次人工批准，review_comments_count 为 0，没有维护者对谓词放宽边界提出质疑或补充要求的记录。

- 评审以 CI 流程为主，缺少对谓词放宽的行级讨论 (other): 改动以“作者自测数据 + 单次人工批准”落地，没有维护者对“去掉 plain_decode 与 max_query_len 约束是否会误伤其他形状”提出过质疑。
- 回归测试只覆盖路由谓词，未覆盖内核数值正确性 (testing): 接受以路由单测 + 人工精度跑分作为验证方式，端到端精度回归未被纳入 CI。
- metadata 重建开销与 gate_score 缺口的后续归属 (performance): 本 PR 只解决 MTP 分派导致的错算，kpool 侧的参数缺口与逐 draft step 重建 metadata 的性能问题另线跟踪。

风险与影响

- 风险:
 1. 谓词放宽的影响面大于 MTP 本身: 新谓词在形状维度上不再做任何限制，任何 `num_decode_tokens != num_decodes` 或 `max_query_len > 1` 的 decode 批次（投机解码验证行、以及未来其他多 token decode 形态）都会进入 Triton 路径。当前等价于“投机验证行”，但谓词本身不再表达这一语义，后续若 AITER 内核补齐能力、或出现新的多 token decode 形态，容易出现误路由或误回退。
 1. 静默回退的失败模式未根治: 谓词返回 False 时依旧静默走 AITER sparse MLA，不会 assert 或告警。本次修复只是让更多形状命中 True，并未改变“不兼容形状静默产出损坏结果”的机制，同类问题仍可能在别的几何组合下重演。
 2. 测试是布尔断言，不是数值验证: `test_rocm_sparse_triton_route` 只断言 `_use_rocm_sparse_triton` 的返回值，没有任何端到端或数值精度用例（例如对比 Triton 与参考实现的输出一致性）。正确性证据完全依赖 PR body 中人工跑出的 GSM8K 数字，CI 无法守护回归。
 3. 性能与 CUDAGraph 交互未充分基准: PR 给出的 173.0s → 153.1s 是 MTP5 与“无 MTP”的对比，不是同一 MTP 配置下修复前后的吞吐对比，也没有普通 decode（无投机）路径的吞吐 / 显存数据。分支名 `rocm/glm53-cudagraph` 暗示与全图捕获相关，但 PR 未说明 Triton ragged 路径在 full graph 下的表现。
 4. 接口残留: `_use_rocm_sparse_triton` 签名中的 `num_decode_tokens`、`max_query_len` 已不再参与判断却仍保留，既可能触发未使用参数类静态检查，也会误导读者以为它们仍影响分派。
 5. 兼容性: FP8 (`kv_cache_dtype.startswith("fp8")`) 与 `head_size != kv_lora_rank` 路径保持原行为，是本次改动中最安全的边界；但由于该谓词被其他 rope-free BF16 sparse-MLA 模型共用，行为变化并不限于 GLM-5.3。- 影响: 对用户: 直接受益的是在 ROCm (gfx950/MI350X/MI355X) 上以投机解码方式部署 GLM-5.3-Flash 的用户——从“输出损坏、GSM8K 0%”恢复到 90%+ 可用水平，并且 MTP5 相比不开 MTP 反而更快 (153.1s vs 173.0s)。对其他用户: 不涉及 FP8 路径、不涉及非 rope-free 几何，理论上无行为变化，但同模块的 BF16 rope-free sparse-MLA 模型会走同一分派函数。对系统: 改动集中在单个模块级谓词，运行时代码路径切换粒度小，无数据结构或内存布局变化。对团队: 该 PR 是本条 ROCm sparse-MLA/kpool 技术线上“先修正确性、再谈性能”的中间落点，issue #53943 (gate_score 缺口) 与 #54369 (metadata 重建开销) 仍处于 open 状态，说明 AMD 侧 GLM-5.3-Flash 的完整可用性还有后续工作。- 风险标记: 分派谓词放宽影响面超出 MTP 场景，不兼容形状静默回退，无断言 / 告警，缺少端到端数值回归测

试，正确性证据仅来自人工 GSM8K 跑分，签名残留未使用参数 `num_decode_tokens / max_query_len`

关联脉络

- PR #53695 [ROCm][Feature] Support KV connectors with ROCm_AITER_UNIFIED_ATTN: 同属 ROCm attention backend 的支持面扩展，同样修改 `vllm/v1/attention/backends` 下的 ROCm 后端并配套 `tests/v1/attention` 用例，体现 ROCm 注意力后端“先打通后端、再补几何边界”的推进节奏。
- PR #54889 [DCP][Kernel][Perf] Fuse the empty-shard LSE mask into the A2A pack kernel: 同样是 vLLM attention 路径上的 ROCm/kernel 侧性能与正确性调整（`vllm/v1/attention/ops/dcp.py`），与本 PR 所处的 `sparse/MLA` 注意力算子地带相邻，可对照理解该模块的内核分派与融合策略。
- PR #56228 [Model] DeepSeek-V4.1-Flash Model Definitions: 新增模型定义包并新增 DSA 候选块判定内核（`vllm/model_executor/kernels/attention/dsa/candidate_blocks.py`），与 GLM-5.3-Flash 共享 DeepSeek-V3.2 式的 `sparse-MLA / indexer` 技术线，说明稀疏注意力在 ROCm/NVIDIA 两条线上都在持续加码。