PR #40763 完整报告

vllm-project/vllm

[Bug] Fix GLM-5.1 running error on ROCm platform

合并时间：2026-04-25 03:54

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40763

执行摘要

- 一句话：修复 GLM-5.1 在 ROCm 上的 MLA 头部填充问题
- 推荐动作：该 PR 值得仔细阅读，特别是 AiterMLAHelper 类的设计——将特定后端的特殊需求集中管理，避免散落在各个 forward 方法中。建议未来在 AITER 上游修复后及时移除 workaround（参见代码中的 TODO）。

功能与动机

GLM-5.1 模型在 ROCm 平台上运行失败，原因是 AITER MLA 实现要求 num_heads>=16，而 GLM-5.1 在使用特定 tensor_parallel_size 时头部数量可能小于 16。同时 AITER 存在一个除零错误，需要规避。

实现拆解

1. 创建 AiterMLAHelper 类(vllm/v1/attention/backends/mla/rocm_aiter_mla.py): 封装了头部有效性检查、实际头部数量获取、Q 填充和 O 解填充等静态方法。
2. 常规 MLA 后端适配: 在 AiterMLAImpl 中，将原有的内联头部重复逻辑替换为 AiterMLAHelper 的方法，包括检查头部有效性、填充 Q、创建输出张量、最终解填充 O。
3. 稀疏 MLA 后端适配(vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py): 导入 AiterMLAHelper，在构造函数中调用检查方法，在前向传播中使用填充后的 Q 和实际头部数量，并解填充输出。
4. 修复 AITER 除零错误(vllm/v1/attention/ops/rocm_aiter_mla_sparse.py): 将 ChunkQ 参数设置为 heads 以规避除零问题，并添加 TODO 注释标记待后续移除。

关键文件：

- vllm/v1/attention/backends/mla/rocm_aiter_mla.py（模块 注意力后端；类别 source；类型 core-logic；符号 AiterMLAHelper, check_num_heads_validity, is_valid_num_heads, get_actual_mla_num_heads）：核心变更文件，新增 AiterMLAHelper 类封装头部填充逻辑，并重构 AiterMLAImpl 使用该辅助类。
- vllm/v1/attention/backends/mla/rocm_aiter_mla_sparse.py（模块 注意力后端；类别 source；类型 dependency-wiring）：稀疏 MLA 后端适配，导入并使用 AiterMLAHelper 进行头部检查、Q 填充和 O 解填充。
- vllm/v1/attention/ops/rocm_aiter_mla_sparse.py（模块 注意力操作；类别 infra；类型 infrastructure）：修复 AITER 除零错误，将 ChunkQ 参数设置为 heads 以规避问题。

关键符号：AiterMLAHelper.check_num_heads_validity, AiterMLAHelper.is_valid_num_heads, AiterMLAHelper.get_actual_mla_num_heads, AiterMLAHelper.get_mla_padded_q, AiterMLAHelper.get_mla_unpadded_o, AiterMLAImpl.forward_mqa, ROCMAiterMLASparseImpl._forward_bf16_kv, ROCMAiterMLASparseImpl.forward_mqa

关键源码片段

vllm/v1/attention/backends/mla/rocm_aiter_mla.py

核心变更文件，新增 AiterMLAHelper 类封装头部填充逻辑，并重构 AiterMLAImpl 使用该辅助类。

```
# class AiterMLAHelper: 封装 AITER MLA 头部填充逻辑
# AITER 要求 num_heads >= 16，若小于 16 则通过 repeat_interleave 填充
# 待下游计算完毕后，再通过切片还原

class AiterMLAHelper:
    _AITER_MIN_MLA_HEADS: Final = 16

    @staticmethod
    def check_num_heads_validity(num_heads: int):
        # 校验头部数量是否有效（必须是 16 的倍数或约数）
        assert AiterMLAHelper.is_valid_num_heads(num_heads), (
            f"Aiter MLA requires that num_heads be multiples or divisors of 16, "
            f"but provided {num_heads} number of heads.\n"
            f"Try adjusting tensor_parallel_size value."
        )

    @staticmethod
    def is_valid_num_heads(num_heads: int) -> bool:
        # 当 num_heads >= 16 时必须是 16 的倍数；否则 16 必须能被 num_heads 整除
        return (
            num_heads % AiterMLAHelper._AITER_MIN_MLA_HEADS == 0
            if num_heads >= AiterMLAHelper._AITER_MIN_MLA_HEADS
            else AiterMLAHelper._AITER_MIN_MLA_HEADS % num_heads == 0
        )

    @staticmethod
    def get_actual_mla_num_heads(num_heads: int) -> int:
        # 实际用于内核的头部数：至少为 16
        return max(num_heads, AiterMLAHelper._AITER_MIN_MLA_HEADS)

    @staticmethod
    def get_mla_padded_q(num_heads: int, q: torch.Tensor) -> torch.Tensor:
        # 若 num_heads < 16，沿头部维度 repeat_interleave 到 16
        return (
            q
            if num_heads >= AiterMLAHelper._AITER_MIN_MLA_HEADS
```

```
            else q.repeat_interleave(
                AiterMLAHelper._AITER_MIN_MLA_HEADS // num_heads, dim=1
            )
        )

    @staticmethod
    def get_mla_unpadded_o(num_heads: int, o: torch.Tensor) -> torch.Tensor:
        # 将填充后的输出切回原始头部数（每隔 factor 取一个）
        return (
            o
            if num_heads >= AiterMLAHelper._AITER_MIN_MLA_HEADS
            else o[:, :: AiterMLAHelper._AITER_MIN_MLA_HEADS // num_heads, :]
        )

# 在 AiterMLAImpl 的 forward_mqa 中的使用示例
mla_padded_q = AiterMLAHelper.get_mla_padded_q(self.num_heads, q)
mla_num_heads = AiterMLAHelper.get_actual_mla_num_heads(self.num_heads)
o = torch.empty(B, mla_num_heads, self.kv_lora_rank, ...)
# ... 调用 AITER 内核计算 ...
return AiterMLAHelper.get_mla_unpadded_o(self.num_heads, o)
```

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

稀疏 MLA 后端适配，导入并使用 AiterMLAHelper 进行头部检查、Q 填充和 O 解填充。

```
# 稀疏 MLA 后端的头部适配关键变更
from vllm.v1.attention.backends.mla.rocm_aiter_mla import (
    AiterMLAHelper,
)

class ROCMAiterMLASparseImpl(SparseMLAAttentionImpl[ROCMAiterMLASparseMetadata]):
    def __init__(self, ..., **mla_args):
        AiterMLAHelper.check_num_heads_validity(num_heads) # 新增头部校验
        # ...

    def _forward_bf16_kv(self, q, kv_c_and_k_pe_cache, topk_indices, attn_metadata):
        mla_num_heads = AiterMLAHelper.get_actual_mla_num_heads(self.num_heads)
        output = torch.empty([num_tokens, mla_num_heads, self.kv_lora_rank], ...) #
        使用实际头部数
        # ... 调用 AITER 内核 ...
        return AiterMLAHelper.get_mla_unpadded_o(self.num_heads, output) # 解填充

    def forward_mqa(self, q, kv_c_and_k_pe_cache, attn_metadata, layer):
        mla_padded_q = AiterMLAHelper.get_mla_padded_q(self.num_heads, q) # 对 Q 填充
        attn_out = self._forward_bf16_kv(mla_padded_q, ...)
        return attn_out, None
```

vllm/v1/attention/ops/rocm_aiter_mla_sparse.py

修复 AITER 除零错误，将 ChunkQ 参数设置为 heads 以规避问题。

```
# 修复 AITER 除零错误的 workaround
# 当 heads 较小时，AITER 内部计算 ChunkQ 可能为 0 导致除零错误
# 此处直接传递 heads 作为 ChunkQ，待 AITER PR 合并后移除

deepgemm_fp8_paged_mqa_logits_stage1(
    q_fp8, kv_cache_fp8,
    ...
    ChunkQ=heads, # 原为 ChunkQ=auto 或缺失，现显式传递
)
```

评论区精华

gemini-code-assist[bot] 提出了三点建议：

- 修正 docstring 中的逻辑错误：16 // num_heads == 0 应为 16 % num_heads == 0
- 更新错误消息以反映支持除 16 的倍数外的约数（如 4,8）
- 在 is_valid_num_heads 中处理 num_heads<=0 的情况，避免 ZeroDivisionError 这些建议均已被采纳。
- docstring 逻辑错误 (correctness): 作者接受建议，已修正。
- 错误消息未反映实际支持范围 (correctness): 作者接受建议，已更新。
- is_valid_num_heads 需要处理非正输入 (correctness): 作者接受建议，已添加。

风险与影响

- 风险：
  1. 性能影响: 头部填充 / 解填充会引入额外内存复制和计算开销，但仅影响头部数小于 16 的场景，影响可控。
  2. 正确性风险: padding 逻辑可能在非标准 MLA 实现中导致非对齐错误，但通过 lm_eval 验证了 GLM-5.1 和 DeepSeek-R1 的精度在可接受范围内。
  3. 兼容性: 该修改仅限于 ROCm 平台的 AITER 后端，不影响其他 GPU 后端。
- 影响：
  1. 用户：使得 GLM-5.1 在 ROCm 平台（TP=8 或 TP=4）上可以正常运行。
  2. 系统: 增加了 padding 和 unpadding 步骤，但仅在小头部数场景生效，整体开销低。
  3. 团队: 引入 AiterMLAHelper 类统一了头部处理逻辑，便于后续维护和扩展。 - 风险标记：依赖上游 AITER 修复，padding 引入性能开销，仅影响 ROCm 平台

关联脉络

- PR #37892 Support only half types for concat_mla_q kernel: 同样与 ROCm 和 MLA 内核相关，涉及 kernel 类型限制。