

PR #51705 完整报告

vllm-project/vllm

[ROCm][MLA][DCP] Support causal multi-token verification

合并时间: 2026-09-01 01:42

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/51705>

执行摘要

- 一句话: ROCm AITER MLA 支持 DCP 下因果多 token 投机验证
- 推荐动作: 值得精读, 尤其关注三点设计决策: 一是 metadata builder 作为 decode 路由唯一所有者的模式, AiterMLADCPVerifyMetadata 的存在即路由结果, 避免 builder 与 impl 对同一条件推导两次产生分叉; 二是 DCP 下因果窗口的推导——"每个行 drop 的 token 落在不同的 rank 上, 不能从本地长度减偏移", 这是教科书式容易写错的地方, 配套的负向对照实验非常有说服力; 三是 segmented MLA 的 split-K 部分和如何以 natural-log LSE 桥接进现有 DCP 跨 rank 合并。建议结合 #53598 (前缀缓存 CoW 修复) 一起阅读, 理解 DCP 在 KV 缓存管理层的完整影响面。

功能与动机

在 DCP 多卡推理下, 投机解码的多 token 验证阶段需要为每个验证 token 构造正确的因果注意窗口, 而 DCP 将 KV 按 round-robin 切分到多个 rank, rank-local 长度无法直接表达全局因果尾。PR body 明确说明目标是: "Each verification token gets its correct per-rank causal KV window, and rank-local partial outputs are merged through the existing DCP attention merge"。review 中 GirasoleY 还指出早期实现 "omits the current verification block, resulting in incorrect attention output", 推动作者将行视图计算完全收敛到 rocm_aiter_mla.py 中。

实现拆解

1. 新增 segmented DCP 验证元数据与路由判定: 在 `vllm/v1/attention/backends/mla/rocm_aiter_mla.py` 中新增 `AiterMLADCPVerifyMetadata` dataclass (`row_lens`、`block_table`、`qo_indptr`、`page_size`、`max_kv_seq_len`), 并把它挂在 `AiterMLADecodeMetadata.dcp_verify` 上。其存在本身就是路由决策, impl 不再重复推导。新增 `_segmented_dcp_verify_supported()` 判定: `dcp_world_size > 1` 且 `cp_interleave == 1` 且 AITER 导出 segmented MLA decode。
2. 构造每行因果窗口的 rank-local 视图: `_build_dcp_verify_row_view()` 对每个验证行 `t` 计算全局因果上界 `[0, seq_len - qlen + t]` (含自身 token), 再按 round-robin 求该窗口在本 rank 上的本地 KV 长度。测试 `test_dcp_verify_row_view_is_causal_per_row` 和 `test_dcp_verify_rows_sum_to_the_global_window_across_ranks` 分别守护单行因果性和跨 rank 总窗口覆盖。旧的 "只给 committed prefix" 计算会确定性失败 (2,671/147,456 元素超容差, 最大差 0.1311), 负向对照已写入验证说明。

3. 物理块到子页的展开与静态图边界: `_segmented_mla_page_size()` 取物理块的最大 2 的幂因子、上限 128, 作为 segmented kernel 的 TILE_SIZE; `_fill_dcp_verify_page_table()` 将物理块表展开为子页级 block table。FULL graph capture 期间使用配置最大 DCP-local KV bound、分配稳定地址的 `_dcp_verify_buffers`, replay 时按实际行长度屏蔽 inactive segments。
4. 分段部分和合并为 BF16 + natural-log LSE: 新增 `vllm/v1/attention/ops/rocm_aiter_mla_merge.py`, 用 Triton kernel `_merge_mla_segments_kernel` 复刻 AITER 的 split-K 段规约 (含 `tiles_per_segment = cdiv(seq_len, NUM_SEGMENTS * TILE_SIZE)` 划分), 将每段 `max/expsum/output` 合并为 BF16 输出和 float32 自然对数 LSE, 供跨 rank DCP 合并使用; 单 token DCP decode 也改为直接调用 AITER 并保留未填充 LSE (`test_single_token_dcp_decode_returns_unpadded_lse` 守护)。
5. 路由归属与图模式调整: Gluon 仍是小头验证首选, DCP 多 token 验证在 Gluon 可用时走 Gluon flatten, 否则回退 segmented MLA; 非 DCP 路径保持原有 Gluon/ASM 选择 (`use_gluon_verify` 增加 `dcp_world_size` 参数)。`vllm/platforms/rocm.py` 的 `check_and_update_config()` 移除 "DCP 强制 PIECEWISE" 的降级, 仅保留 PCP 的 PIECEWISE 限制, 使 ROCm DCP 可配合 FULL CUDA graphs。
6. 测试配套: `tests/v1/attention/test_rocm_aiter_mla_mtp_split.py` 新增 DCP 行视图、静态图边界、FP8 segmented 构建、单 token 未填充 LSE、部分注意合并等用例; `test_rocm_aiter_mla_fp8_decode_routing.py` 新增 DCP 多 token 验证永不走 Gluon/ 对 Gluon 无依赖的断言; `test_rocm_aiter_mla_causal_verify_mask.py` 与 `test_rocm_aiter_mla_head_padding.py` 适配了 builder 路由时代后的测试夹具改变。

关键文件:

- `vllm/v1/attention/backends/mla/rocm_aiter_mla.py` (模块 MLA 后端; 类别 source; 类型 core-logic; 符号 `_segmented_mla_page_size`, `_segmented_dcp_verify_supported`, `AiterMLADCPVerifyMetadata`, `_build_dcp_verify_row_view`): 核心实现文件: 新增 segmented DCP 验证元数据与路由判定、每行因果窗口视图构建、子页块表展开、FULL graphs 静态边界缓冲, 是全部正确性逻辑的载体。
- `vllm/v1/attention/ops/rocm_aiter_mla_merge.py` (模块 合并内核; 类别 source; 类型 core-logic; 符号 `_merge_mla_segments_kernel`, `merge_mla_segments_triton`): 新增文件: 提供 Triton 段合并内核, 把 AITER segmented MLA 的 split-K 部分和 (base-2 max/expsum) 合并为 BF16 输出和 natural-log LSE, 是跨 rank DCP 合并的前置条件。
- `vllm/platforms/rocm.py` (模块 平台配置; 类别 source; 类型 configuration; 符号 `check_and_update_config`): 行为变更: 移除 "DCP 与 FULL CUDA graphs 不兼容" 的强制 PIECEWISE 降级, 仅保留 PCP 限制, 是本 PR 能跑通 FULL graph capture 的关键配置改动。
- `tests/v1/attention/test_rocm_aiter_mla_mtp_split.py` (模块 验证测试; 类别 test; 类型 test-coverage; 符号 `_lse_combine_natural`, `test_dcp_verify_row_view_is_causal_per_row`, `test_dcp_verify_rows_sum_to_the_global_window_across_ranks`, `test_dcp_verify_row_view_uses_static_graph_bound`): 最重要的测试文件: 覆盖 DCP 行视图因果性、跨 rank 总窗口、静态图边界、FP8 segmented 构建、单 token 未填充 LSE 与部分注意合并, 还包含负向对照语义。

- tests/v1/attention/test_rocm_aiter_mla_fp8_decode_routing.py (模块 路由测试; 类别 test; 类型 test-coverage; 符号 test_dcp_multitoken_verify_never_uses_gluon, test_segmented_dcp_verify_does_not_depend_on_gluon) : 守护路由正确性: DCP 多 token 验证永不进入 Gluon、segmented 路由不依赖 Gluon, 且 round-robin interleave 非 1 被拒绝, 防止测试期已修掉的路由漂移回归。
- tests/kernels/attention/test_rocm_aiter_mla_causal_verify_mask.py (模块 验证掩码; 类别 test; 类型 test-coverage) : 适配 builder 时代: 路由决策移到 build() 后, gfx950 特征探测需要在 build 与 forward 两处同时被打桩, 并直接断言 metadata 上的 use_gluon_verify 标志。
- tests/kernels/attention/test_rocm_aiter_mla_head_padding.py (模块 头填充; 类别 test; 类型 test-coverage) : 小改动但必要: 为 impl stub 补上 dcp_world_size 与 qk_rope_head_dim 字段, 保持 12-head ASM decode 对照测试在新 impl 契约下可用。

关键符号: _segmented_mla_page_size, _segmented_dcp_verify_supported, _build_dcp_verify_row_view, _fill_dcp_verify_page_table, merge_mla_segments_triton, _merge_mla_segments_kernel, use_gluon_verify, forward_mqa, check_and_update_config

关键源码片段

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

核心实现文件: 新增 segmented DCP 验证元数据与路由判定、每行因果窗口视图构建、子页块表展开、FULL graphs 静态边界缓冲, 是全部正确性逻辑的载体。

vllm/v1/attention/backends/mla/rocm_aiter_mla.py (整理后的核心实现片段)

```
def _segmented_mla_page_size(block_size: int) -> int:
```

```
    """取物理 KV block 的最大 2 的幂因子作为子页大小, 上限 128。
```

```
    子页大小会成为 segmented 内核的 TILE_SIZE: 调用前缓存被重解释为
    (-1, page_size, 1, head_dim)。内核用 tl.arange(0, TILE_SIZE) 遍历 tile,
    因此必须能被物理块整除且为 2 的幂; 128 是 tiling 能支持的""最大""tile,
    更大的管理块会被拆成多个子页而不是加宽 tile。
```

```
    """
```

```
    assert block_size > 0
```

```
    return min(128, largest_power_of_2_divisor(block_size))
```

```
def _segmented_dcp_verify_supported(dcp_world_size: int, cp_interleave: int) -> bool:
```

```
    """配置层面判定该组合能否走 segmented DCP 验证。
```

```
    只判配置: 某个 batch 是否真的切入该路由还取决于 query 长度。
```

```
    round-robin interleave 非 1 被排除, 因为该布局下的逐行因果窗口尚未验证。
```

```
    """
```

```
    return (
```

```
        dcp_world_size > 1
```

```
        and cp_interleave == 1
```

```

        and _segmented_mla_decode_supported()
    )

@dataclass
class AiterMLADCPVerifyMetadata:
    """每个验证 token 一行 paged-KV，供 segmented DCP 验证使用。

    这些张量要么一起产生、要么都不产生，所以作为一个整体值传递：
    它出现在 decode metadata 上这一事实本身就等于 builder 的路由决策，
    impl 不再重新推导。
    """
    # 每个验证行的本地 KV 长度，已经过因果边界裁剪。
    row_lens: torch.Tensor
    # 每行读取的子页 ID，形状 [num_rows, max_local_pages]。
    block_table: torch.Tensor
    # 每行恰好一个 query，因此这里是 arange。
    qo_indptr: torch.Tensor
    # block table 使用的子页大小，也是 segmented 内核的 TILE_SIZE；
    # 由 builder 显式携带，避免 impl 从缓存形状重新推导产生不一致。
    page_size: int
    # 内核可见的 KV 上界。启用 FULL graphs 时对每个 batch 都是配置最大值，
    # 保证 page table 在多次 replay 间保持单一形状。
    max_kv_seq_len: int

def _build_dcp_verify_row_view(self, qlen, block_table, dcp_tot_seq_lens):
    """为每个验证 token 构造一条因果的 rank-local 行视图。

    行 t 的全局可见范围是 [0, seq_len - qlen + t]，包含自身 token。
    直接从请求的本地长度减去因果偏移是错的：某一行丢弃的 token 分散在
    qlen - 1 - t 个不同的 rank 上，而减法会从每个 rank 各扣一个 token，
    只有按全局边界重新数 round-robin 命中数才是正确长度。
    """
    assert self.dcp_world_size > 1
    num_reqs = dcp_tot_seq_lens.numel()
    # 每个验证行的全局窗口上界（含），对应 [0, seq_len - qlen + t]。
    global_window_ends = (
        dcp_tot_seq_lens.unsqueeze(1) - (qlen - 1) + torch.arange(1, qlen + 1)
    )
    # 本地长度 = 该全局窗口内本 rank (round-robin interleave=1) 持有的位置数。
    # 对窗口长度 w 与 rank r：命中数为 floor((w - 1 - r) / c) + 1（当非负）。
    local_lens = ... # 经由 get_dcp_local_seq_lens 等工具按 rank 计算
    return _fill_dcp_verify_page_table(local_lens, block_table, qlen)

```

vllm/v1/attention/ops/rocm_aiter_mla_merge.py

新增文件：提供 Triton 段合并内核，把 AITER segmented MLA 的 split-K 部分和（base-2 max/expsum）合并为 BF16 输出和 natural-log LSE，是跨 rank DCP 合并的前置条件。

vllm/v1/attention/ops/rocm_aiter_mla_merge.py (新增文件, 整理后的核心实现)

```
def merge_mla_segments_triton(
    segm_output: torch.Tensor, # [num_tokens, num_heads, num_segments, kv_lora_rank]
    segm_max: torch.Tensor, # [num_tokens, num_heads, num_segments] 各段局部最大值
    segm_expsum: torch.Tensor, # 各段 exp2 和
    seq_lens: torch.Tensor, # 每行的真实长度, 用于计算 active segments
    tile_size: int, # 与 segmented 内核一致的 TILE_SIZE
    out_dtype: torch.dtype,
) -> tuple[torch.Tensor, torch.Tensor]:
    """把 AITER base-2 段部分和合并成输出和 natural-log LSE。

    这是 rank-local 的 split-K 归约, 与任何集合通信无关; 返回的自然对数 LSE
    正是跨 rank DCP 合并要消费的统计量。分区方式必须与 AITER 自身的
    tiles_per_segment = cdiv(seq_len, NUM_SEGMENTS * TILE_SIZE) 保持一致。
    """
    num_tokens, num_heads, num_segments, kv_lora_rank = segm_output.shape
    output = torch.empty((num_tokens, num_heads, kv_lora_rank), dtype=out_dtype)
    lse = torch.empty((num_tokens, num_heads), dtype=torch.float32)
    _merge_mla_segments_kernel[(num_tokens, num_heads)](
        output, lse, segm_output, segm_max, segm_expsum, seq_lens,
        num_query_heads=num_heads,
        out_stride0=output.stride(0), out_stride1=output.stride(1),
        lse_stride0=lse.stride(0),
        TILE_SIZE=tile_size, KV_LORA_RANK=kv_lora_rank,
        NUM_SEGMENTS_PER_SEQ=num_segments, LOGE2=LOGE2,
    )
    return output, lse
```

vllm/platforms/rocm.py

行为变更: 移除 "DCP 与 FULL CUDA graphs 不兼容" 的强制 PIECEWISE 降级, 仅保留 PCP 限制, 是本 PR 能跑通 FULL graph capture 的关键配置改动。

vllm/platforms/rocm.py 的 check_and_update_config() 片段 (整理后)

```
@classmethod
def check_and_update_config(cls, vllm_config: "VllmConfig") -> None:
    from vllm.config.compilation import CUDAGraphMode

    compilation_config = vllm_config.compilation_config
    parallel_config = vllm_config.parallel_config

    # PCP (prefill context parallel) 仍与 FULL graphs 不兼容, 保持强制降级。
    if (
        compilation_config.cudagraph_mode.has_full_cudagraphs()
        and parallel_config.prefill_context_parallel_size > 1
    ):
        logger.warning_once(
            "Prefill context parallel (PCP) is enabled, which is "
```

```

        "incompatible with full CUDA graphs. "
        "Overriding cudagraph_mode to PIECEWISE."
    )
    compilation_config.cudagraph_mode = CUDAGraphMode.PIECEWISE

# DCP 不再强制 PIECEWISE: 本 PR 的验证元数据在 capture 阶段使用
# 配置最大 DCP-local KV 上界, replay 时按实际行长度屏蔽,
# 因此 ROCm DCP 组合可以直接使用 FULL CUDA graphs。

if parallel_config.worker_cls == "auto":
    parallel_config.worker_cls = "vllm.v1.worker.gpu_worker.Worker"

```

评论区精华

review 中最有价值的几轮交锋:

GirasoleY (发现正确性缺口): "It looks like the ROCm K3 DCP verification path omits the current verification block... `_build_dcp_verify_row_view()` excludes the entire current verification block and relies on `forward_mqa_with_dcp_verify_window()` to add it back. The ROCm K3 target uses `amd/linear.py` and the generic `MLAAttention.forward_impl()`, which calls only `forward_mqa()`, so `k_window` is always `None` and the current keys are omitted from target attention."

作者回复: 将全部改动收敛到 `rocm_aiter_mla.py`, 用 "每个验证行构造从 committed prefix 到自身 token 的因果本地长度" 的方式让 segmented 内核直接读取当前 tokens, 避免改动 NVIDIA 模型路径。

jamesETsmith (分组语义): 合并 draft 与 target 的 KV cache group 会让 `non_causal_multi_token_decode=True` 导致整个 group 被复制、失去 DCP 优势; 作者确认 draft/target 因该 flag 不同而保持独立分组, 且 draft 也做 DCP 分片。

okorzh-amd (路由回归风险): `use_gluon_verify` 放宽到 96 head 会把非 DCP 的 16-head TP8 MTP 从 asm 验证改到 Gluon; 作者回应宽路由仅以 `dcp_world_size > 1` 生效, 非 DCP 保持原样。

billishyahao (工程约束): 反对用 regex/inspect 探测 AITER 支持矩阵 ("Please dont use regex here for detecting aiter gluon support matrix from source code"), 作者移除 regex, 改用显式 96-head DCP 能力 + Triton 版本检查; 同时要求恢复 Gluon 为首选路径, 作者最终以 "Gluon 首选、segmented MLA 兜底" 收敛。

GirasoleY (DCP+FULL graphs): 质疑 `VLLM_ALLOW_DCP_FULL_CUDAGRAPH` flag 必要性, 作者同意删除冗余 flag 直接放开; 对 `_DCPA2ABufferPool` 要求给出 RCCL 在 cudagraph 下的最小复现, 作者因无法复现而移除了该缓冲池。

- DCP 验证路径遗漏当前验证块 (correctness): 作者将全部逻辑收敛到 `rocm_aiter_mla.py`, 行 t 使用全局范围 $[0, \text{seq_len} - \text{qlen} + t]$ 的 rank-local round-robin 长度, segmented 内核直接从缓存读取当前 tokens。

- Draft/target 共享 KV 组导致 DCP 失效 (design): 作者确认 draft 与 target 因 flag 不同保持独立 KV 组, draft MLA 同样做 DCP 分片。
- 放宽 Gluon verify 到 96 head 是否影响非 DCP (correctness): 宽路由仅以 dcp_world_size > 1 生效, 非 DCP 的 16-head MTP 保持 ASM 路径。
- kv_cache_utils 中 draft-only bucket 谓词不一致 (correctness): 作者放弃 drafter-invariant primary_sizes 路径, group_size 恢复为对所有 bucket 的 min/max, PD 分组不变性留作后续工作。
- 禁止用 regex/ 源码探测 AITER 支持矩阵 (design): 作者移除 regex 与 inspect, 改用显式 96-head DCP 能力加 Triton 版本检查。
- 恢复 Gluon 作为 DCP verify 首选路径 (design): 最终 Gluon 为 DCP 多 token 验证首选 (K3 原生 96-head flatten), segmented MLA 作为 Gluon 不可用时的兜底。
- DCP8+DSpark+prefix caching 触发 CoW 断言崩溃 (correctness): 作者确认属既有 CoW 缺陷被新组合暴露, 建议 cherry-pick #53598 修复, 并认可补端到端 DCP8+DSpark+prefix-cache 回归测试的提议。

风险与影响

- 风险:
 1. 核心注意力路径变更: rocm_aiter_mla.py 改动 612 行, decode/verify 路由从 impl 移到 builder, 且 use_gluon_verify 签名增加 dcp_world_size, 任何遗漏调用点都可能让 DCP verify 走到错误内核; 测试虽覆盖路由谓词, 但仍缺少自动化端到端 DCP8 目标验证测试 (依赖人工 GSM8K 全模型回归)。
 2. CUDA graph 行为放开: vllm/platforms/rocm.py 允许 DCP 走 FULL graphs, capture 阶段的静态 max_kv_seq_len bound 与 replay 时的实际行长度屏蔽逻辑直接决定正确性; Kimi-K3 1M context 组合首次被支持, 若未来出现更长 context 或混合 interleave, 需重新验证。
 3. 强依赖 AITER 版本: segmented MLA decode (aiter.ops.triton.attention.mla.mla_decode_fwd) 依赖 ROCm/aiter#4412 的 96-head 网格分解; andyluo7 明确说明 stock Dockerfile 使用 AITER v0.1.19、不包含该能力, 需依赖 #52826 升级到 v0.1.20 才能开箱即用。
 4. DCP 与前缀缓存组合: jamesETsmith 报告 DCP8+DSpark+prefix caching 下 _apply_cow 断言崩溃 (single_type_kv_cache_manager.py:416-420), 本 PR 的 DCP 支持暴露了既有 CoW 缺陷, 作者建议 cherry-pick #53598 修复; 该组合在 CI 中仍缺乏端到端回归。
 5. 平台隔离风险: 本 PR 限定 ROCm AITER 后端, 但 AiterMLADCPVerifyMetadata 位于公共 MLA 目录, 后续演进若被其他平台误用, 可能引入共享代码回归。- 影响: 用户影响: ROCm (gfx942/gfx950) 上使用 MLA 架构 + DCP 的投机解码用户 (Kimi-K3 DSpark、DeepSeek MTP 等) 首次获得正确的多 token 因果验证, DCP8 + FP8 KV + FULL graphs 组合可用; GSM8K 5-shot exact_match 0.9613 (target-only) 与 0.9628 (stacked Triton draft) 均表明精度无退化。系统影响: 解码路径新增 segmented MLA + Triton merge 两条新执行路径, DCP 与 FULL cudagraph 的组合从 "强制降级" 变为 "允许", 对 ROCm 推理系统的内存布局与图捕获稳定性有直接影响。

团队影响：这是 AMD 与上游协作的大型跨模块 PR，经历了 6 轮 commit、49 条 review 评论和 AI 辅助修复（andyluo7 多轮 rebase/ 修复），沉淀了 "builder 拥有路由决策" "因果窗口必须按全局位置计算" 等可复用的设计原则，也为后续 Triton MLA 非因果 draft (#54546) 铺路。

- 风险标记：核心注意力路径变更，CUDA graph 行为放开，强依赖 AITER 版本，平台限定 ROCm，端到端回归覆盖不足

关联脉络

- PR #53598 [ROCm][DSpark][DCP] Serve prefix cache hits under DCP for Kimi-K3: 同属 DCP+DSpark 功能线：修复 DCP 下混合 KV 组缓存几何不一致。本 PR issue 评论中作者让 jamesETsmith 直接 cherry-pick 该 PR 以恢复前缀缓存命中并规避 CoW 断言崩溃。
- PR #54546 [Stacked] Enable Triton MLA for the non-causal draft group: PR body 明确说明该 PR 干净地叠放在本 PR head 上，为非因果 draft group 启用 Triton MLA；两者联合跑通 DCP8 GSM8K，合流后成为同一架构的两半。
- PR #52188 [Merged] DFlash DCP slot geometry: andyluo7 在修复说明中确认 current main 已通过合并 #52188 提供 DFlash DCP slot geometry（共享 cp_local_slot 辅助函数），本 PR 不再重复实现该部分，属于同 DCP 基础设施的既有上游工作。
- PR #52826 [Dependency] Pin AITER v0.1.20: segmented MLA（96-head 网格分解）依赖 ROCm/aiter#4412，首个包含该能力的发布是 v0.1.19.post1；#52826 将 AITER 固定到 v0.1.20，是本 PR 在 stock 镜像上开箱可用的前置依赖。