# PR #47718 完整报告

vllm-project/vllm

[ROCm][Perf] DSv4 two-stage compressor kernel for HCA prefill

合并时间：2026-07-16 07:31

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47718

## 执行摘要

- 一句话：两阶段 split compressor 优化 DSv4 prefill TTFT
- 推荐动作：值得精读。此 PR 展示了如何通过 occupancy-targeted split 解决 kernel 级串行瓶颈，设计思路可推广到其他类似 pattern。关键设计决策：split 沿 head 维度无数据依赖、num_splits 动态计算、平台条件编译。

## 功能与动机

在 ROCm 上，单阶段 compressor-norm-rope-cache fusion 以每 token 一个 workgroup 调度。当 compress ratio=128 时，只有位于 CR 边界的 token（每 128 个 token 一个）执行计算，导致大量 CU 闲置（例如 MI355 有 256 个 CU 但仅 64 个活跃）。为提升 occupancy，需将单个 program 拆分为更多独立 tasks，使 CU 饱和。

## 实现拆解

1. 平台守卫与元数据扩展：在 compressor.py 中添加 _prefer_two_stage_compressor() 函数，仅在 ROCm 返回 True；在 CompressorMetadataBuilder.build() 中通过 split_decodes_and_prefills 获取 decode token 数，并新增 num_decode_tokens 字段。
2. Scratch buffer 分配与前向分发：在 CompressorStateCache.__init__ 中，当 head_dim=512 且 compress_ratio=128（无 overlap）时分配 fp32 格式的 _compress_scratch tensor [max_num_batched_tokens, head_dim]；在前向方法中根据 num_decode_tokens 选择单阶段或两阶段调用。
3. 两阶段 kernel 实现：在 fused_compress_quant_cache.py 中新增 _pick_compress_num_splits 根据 CU 数量和 boundary 数计算最优 splits（2 的幂，32 的倍数）；_compress_gather_split_sparse_attn 为每个 (token, split) 启动一个 program，沿 head 维度分 tile 计算加权求和并写入 fp32 scratch；_finalize_norm_rope_quant_store_sparse_attn 在原有单 program 策略下完成 norm、rope、量化、插入。compress_norm_rope_store_two_stage_triton 对外暴露的接口。
4. 测试覆盖：新增 test_fused_kv_insert_split 参数化 token 数（1/4/8/17）和 page 大小（16/64），验证两阶段 pipeline 输出与参考一致。

关键文件：

- vllm/models/deepseek_v4/common/ops/fused_compress_quant_cache.py（模块 算子层；类别 source；类型 core-logic；符号 _n_cu, _pick_compress_num_splits,

_compress_gather_split_sparse_attn, _finalize_norm_rope_quant_store_sparse_attn）：实现两阶段 split compressor kernel 的核心文件，包含 occupancy 驱动的 split 计数、head-split compress gather kernel 和 finalize kernel。

- vllm/models/deepseek_v4/compressor.py（模块 压缩器；类别 source；类型 data-contract；符号 _prefer_two_stage_compressor）：修改压缩器入口，添加平台守卫和 scratch buffer 分配，根据 decode 数量选择两阶段路径。
- tests/kernels/test_compressor_kv_cache.py（模块 测试；类别 test；类型 test-coverage；符号 test_fused_kv_insert_split）：新增 Test F 覆盖两阶段 split compressor 的正确性，参数化 token 数和 page 大小。

关键符号：_pick_compress_num_splits, _compress_gather_split_sparse_attn, _finalize_norm_rope_quant_store_sparse_attn, _launch_two_stage_sparse_attn_compressor, compress_norm_rope_store_two_stage_triton, _prefer_two_stage_compressor, test_fused_kv_insert_split

## 关键源码片段

### vllm/models/deepseek_v4/compressor.py

修改压缩器入口，添加平台守卫和 scratch buffer 分配，根据 decode 数量选择两阶段路径。

```
def _prefer_two_stage_compressor() -> bool:
    """
    当前平台是否偏好两阶段 split compressor。
    Triton 实现在 AMD 上验证，因此暂时仅返回 ROCm。
    """
    return current_platform.is_rocm()

# 在 CompressorStateCache.__init__ 中：
self._use_two_stage_fused_compressor = (
    _prefer_two_stage_compressor()
    and head_dim == 512
    and not self.overlap # cr>=128 才能开启
)
if self._use_two_stage_fused_compressor:
    self._compress_scratch = torch.empty(
        self.max_num_batched_tokens,
        self.head_dim,
        dtype=torch.float32,
        device=self.device,
    )

# 在 forward 中：
elif self._use_two_stage_fused_compressor:
    compress_norm_rope_store_fn = compress_norm_rope_store_two_stage_triton
    extra_kwargs = {
        "num_decode_tokens": state_metadata.num_decode_tokens,
        "compress_scratch": self._compress_scratch,
```

```
}
```

### tests/kernels/test_compressor_kv_cache.py

新增 Test F 覆盖两阶段 split compressor 的正确性，参数化 token 数和 page 大小。

```
@pytest.mark.skipif(
    not current_platform.is_rocm(),
    reason="two-stage split compressor is only enabled for ROCm at the moment",
)
@pytest.mark.parametrize("num_tokens", [1, 4, 8, 17])
@pytest.mark.parametrize("kv_block_size", [16, 64])
def test_fused_kv_insert_split(num_tokens: int, kv_block_size: int):
    """
    验证两阶段 split compressor 全流程正确性:
    state_cache gather → softmax 加权压缩 → RMSNorm → RoPE → FP8 量化 → paged insert。
    """
    HEAD_DIM = 512
    NOPE_DIM = 448
    ROPE_DIM = 64
    HEAD_BYTES = 584 # 448 fp8 + 128 bf16 + 8 uint8 scale
    RMS_EPS = 1e-6
    FP8_MAX = 448.0
    # ... 初始化数据，调用 _launch_two_stage_sparse_attn_compressor 后与参考比较
```

## 评论区精华

tjtanaa 询问 split_decodes_and_prefills 是否应平台无关，kliuae 回应其实为 Triton 实现，理论上可跨平台，但目前仅 ROCm 经测试，故暂时限定 ROCm。

- 平台限定 vs 通用 (design): 作者使用 _prefer_two_stage_compressor 包裹来限定 ROCm。

## 风险与影响

- 风险：新 kernel 仅作用于 ROCm 且 head_dim=512、cr>=128 场景，其他路径不受影响。主要风险：Triton 实现的跨平台兼容性未验证；scratch buffer 占用额外显存（max_batched_tokens * 512 * 4 bytes）；两阶段分别执行增加了 kernel 启动开销，但在 prefill 下收益显著；解码路径仍使用单阶段。测试覆盖了多种 token 数与 page 大小，但未覆盖边缘 case（如 token 数极少）。
- 影响：用户侧：ROCm 部署 DeepSeek-V4 模型时 prefill TTFT 降低 3-4%，吞吐小幅提升。解码路径无影响。系统侧：新增约 2MB/token（max_batched_tokens 1M+）scratch 空间。团队侧：需维护两套压缩路径，但差异化较小。对其他平台无影响。
- 风险标记：仅 ROCm 验证，Triton 跨平台未测试，scratch 显存开销

## 关联脉络

- PR #47463 [Perf] Optimize fused_topk_bias for DSv4, 1.5~2x kernel performance improvement: 同为 DeepSeek-V4 性能优化，聚焦 kernel occupancy