PR #40996 完整报告

vllm-project/vllm

DCP supports hybrid attention

合并时间：2026-07-10 12:34

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40996

执行摘要

- 一句话：DCP 支持混合注意力模型，非 DCP 层走本地缓存
- 推荐动作：值得精读。该 PR 是 DCP 支持混合注意力的关键里程碑，其中三处设计最值得关注：①把“分片 vs 复制”语义下沉到 KVCacheSpec 子类（max_num_blocks_per_req），解决 CP 缩放公式散落各处的问题；②rank-invariant 的 collective 门控原则（任何决定是否发起 DCP 通信的判定都必须基于全局、rank 无关的输入），这是分布式正确性的通用教训；③FA2 混合批次拆分的兼容性 workaround 思路。建议结合后续 #50823 一起阅读，观察该契约的演进。

功能与动机

PR body 明确说明：Hybrid-attention models such as Qwen/Qwen3.5-0.8B contain both DCP-capable full-attention layers and non-DCP layers. This PR enables DCP for supported attention groups without globally blocking hybrid-attention DCP, while keeping non-DCP groups on local cache/state handling. 此前 HybridKVCacheCoordinator 直接断言 dcp_world_size == 1，导致混合注意力模型无法享受 DCP 的长上下文 decode 加速。cjackal 在评论中提醒此前 #36480 的尝试在 397B MoE 上存在精度问题，作者补充了 Qwen3.5-397B-A17B-FP8 的 GSM8K 结果（TP4DCP1 0.9140625 / TP4DCP2 0.91015625）证明精度已无问题。

实现拆解

1. KV cache 契约：按组区分“分片”与“复制”— vllm/v1/kv_cache_interface.py 给 KVCacheSpec 新增 max_num_blocks_per_req(vllm_config, max_len) 方法；AttentionSpec 按 dcp * pcp 缩放（每 rank 只存 max_len // (dcp * pcp) 个 token），MambaSpec 不缩放，并在 align 模式按整序列行宽 + num_speculative_blocks 计算。vllm/v1/core/kv_cache_utils.py 的 resolve_kv_cache_block_sizes 从“多 block size + CP 直接报错”改为 attention 组 block_size * dcp * pcp、Mamba 组保持原值再求 LCM。vllm/v1/core/kv_cache_coordinator.py 移除 dcp_world_size == 1 断言，改为显式只允许 FullAttentionSpec 与 MambaSpec 两类组参与 DCP，并让 find_longest_cache_hit 基于每个 manager 的真实 block_size 对齐、对 full-attention 组传入 dcp_world_size。
2. BlockTable 与 slot mapping 分层— vllm/v1/worker/block_table.py 新增 SlotMappingMode（TOKEN_TO_KV_SLOT / NONE）。Mamba/GDN 组只把 block table 当作循环状态索引，不需要 token 级 slot mapping，NONE 模式下

compute_slot_mapping 直接返回，避免按 full-attention 的 DCP 布局算出错误 slot。Triton kernel 同时引入 KV_CACHE_BLOCK_SIZE 与 BLOCKS_PER_KV_BLOCK 常量，支持分配块大于 kernel 块的混合布局；max_num_blocks 改为由调用方显式传入，不再隐式用 get_total_cp_world_size 推导。

3. FlashAttention DCP forward 路径重构— vllm/v1/attention/backends/flash_attn.py 的 metadata 扩展 num_decode_reqs / num_prefill_reqs / num_decode_tokens / num_prefill_tokens 拆分计数。builder 用全局（rank 无关的）seq_lens_cpu_upper_bound 计算 context_kv_lens_cpu：全 0 则走 skip_dcp_context_attention fast path（max_dcp_context_kv_len = 0，forward 直接退化为普通 varlen attention，不发任何 DCP collectives）；否则用 split_dcp_context_queries 把重排后的 query 拆成 decode 与 extend 区段。对 FA2 混合批次（decode + 带 context 的 extend + 纯 prefill 同时出现），should_split_fa2_dcp_context_attention 返回 True，forward 改走 run_split_fa2_dcp_context_attention 分段计算并拼接 LSE 与输出。
4. CUDA graph 捕获的 dummy 输入— vllm/v1/worker/gpu_model_runner.py 的 _dummy_run 调用 get_dcp_dummy_context_len 决定是否给 seq_lens 增加 dcp_world_size * cp_kv_cache_interleave_size 的假 context，随后 prepare_dcp_dummy_context_metadata 为每个 BlockTable 行填充合法 block id（取模分配已分配块）、修正 positions 与 slot mapping，保证 DCP context 路径能被图正常捕获。
5. 通信与测试配套— vllm/distributed/device_communicators/cuda_communicator.py 的 all_gather 支持非 0 维（DCP 沿 seq 维 gather query），通过 pynccl all-gather 后 reshape/movedim 实现；测试新增 tests/v1/worker/test_cp_utils.py（rank-invariant skip gate）、tests/distributed/test_pynccl.py（非零维 all-gather），并改 tests/models/language/generation/test_hybrid.py 覆盖 DCP 配置。

关键文件：

- vllm/v1/worker/cp_utils.py（模块 上下文并行；类别 source；类型 core-logic；符号 get_dcp_dummy_context_len, prepare_dcp_dummy_context_metadata, should_skip_dcp_context_attention, split_dcp_context_queries）：DCP 混合注意力的核心新逻辑所在：dummy context 长度计算、dummy metadata 填充、rank-invariant skip gate、query 拆分以及 FA2 混合批次的分段 attention 执行，全部集中于此。
- vllm/v1/attention/backends/flash_attn.py（模块 注意力后端；类别 source；类型 core-logic；符号 FlashAttentionMetadata, build, _forward_with_dcp）：DCP forward 路径核心重构：skip gate 接入 metadata builder、FA2 混合批次拆分、零上下文 fast path、workspace 尺寸调整，直接影响 decode 热路径。
- vllm/v1/core/kv_cache_coordinator.py（模块 缓存协调；类别 source；类型 core-logic；符号 HybridKVCacheCoordinator.init, _get_block_hashes, find_longest_cache_hit）：移除 DCP 混合注意力的全局禁止断言，改为按 spec 类型校验，并让 cache-hit 对齐使用每个 manager 的真实 block_size 与 dcp_world_size。
- vllm/v1/worker/block_table.py（模块 块表；类别 source；类型 core-logic；符号 SlotMappingMode, BlockTable.compute_slot_mapping, MultiGroupBlockTable）：引入

SlotMappingMode 区分 token-KV 缓存组与 Mamba/GDN 状态缓存组；slot mapping 内核支持分配块大于 kernel 块的混合布局。

- vllm/v1/worker/gpu_model_runner.py（模块 模型运行器；类别 source；类型 data-contract；符号 _dummy_run, may_reinitialize_input_batch）：CUDA graph dummy run 接入 DCP 假 context metadata；InputBatch 重初始化改为基于 spec 的 max_num_blocks_per_req 计算并携带 slot_mapping_modes。
- vllm/v1/kv_cache_interface.py（模块 缓存契约；类别 source；类型 data-contract；符号 KVCacheSpec.max_num_blocks_per_req, AttentionSpec.max_num_blocks_per_req, MambaSpec.max_num_blocks_per_req）：定义 spec 级 max_num_blocks_per_req 契约：AttentionSpec 按 DCP/PCP 分片缩放，MambaSpec 保持复制语义，是本次改动的基础抽象。
- vllm/v1/core/kv_cache_utils.py（模块 缓存工具；类别 source；类型 core-logic；符号 resolve_kv_cache_block_sizes）：resolve_kv_cache_block_sizes 解除混合缓存组 + CP 的硬限制，attention 组缩放、Mamba 组不缩放，是调度端正确配置的基础。
- vllm/distributed/device_communicators/cuda_communicator.py（模块 通信器；类别 source；类型 core-logic；符号 CudaCommunicator.all_gather）：all_gather 支持非 0 维，DCP 按 seq 维 all-gather query 所必需；绕过 symbol memory 路径，用 pynccl + reshape/movedim 实现。
- tests/v1/worker/test_cp_utils.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_skip_gate_only_for_zero_context, test_skip_gate_rank_invariant_with_divergent_local_context）：新增 skip gate 的 rank-invariant 单元测试，专门覆盖 context 短于 interleave 轮次时各 rank 本地 context 发散的死锁场景。

关键符号：get_dcp_dummy_context_len, prepare_dcp_dummy_context_metadata, should_skip_dcp_context_attention, split_dcp_context_queries, should_split_fa2_dcp_context_attention, run_split_fa2_dcp_context_attention, KVCacheSpec.max_num_blocks_per_req, AttentionSpec.max_num_blocks_per_req, MambaSpec.max_num_blocks_per_req, BlockTable.compute_slot_mapping, CudaCommunicator.all_gather, HybridKVCacheCoordinator.find_longest_cache_hit

关键源码片段

vllm/v1/worker/cp_utils.py

DCP 混合注意力的核心新逻辑所在：dummy context 长度计算、dummy metadata 填充、rank-invariant skip gate、query 拆分以及 FA2 混合批次的分段 attention 执行，全部集中于此。

```
def should_skip_dcp_context_attention(context_kv_lens_cpu: torch.Tensor) -> bool:
    """是否可跳过 DCP context attention。

    必须只基于 rank 无关的全局 context 长度计算（而不是本 rank 由
    get_dcp_local_seq_lens 得到的本地份额）：非跳过路径会发起 DCP
    集体通信（query all-gather + LSE combine），因此所有 DCP rank
    必须走同一分支。某个 rank 可能持有 0 个本地 context token，
    而其它 rank 仍持有同一批请求的 context。
```

```
    """
    return int(context_kv_lens_cpu.max().item()) == 0


def split_dcp_context_queries(
    query_start_loc: torch.Tensor,
    seq_lens_cpu_upper_bound: torch.Tensor | None,
    max_query_len: int,
    num_actual_tokens: int,
) -> tuple[int, int, int, int]:
    """把重排后的 DCP context query 拆成 decode 与 extend 两个区段。"""
    num_reqs = query_start_loc.shape[0] - 1
    if max_query_len <= 1:
        # 纯 decode 批次：所有请求都是 decode，无需拆分
        return num_reqs, 0, num_actual_tokens, 0
    if seq_lens_cpu_upper_bound is None:
        # 没有 CPU 上界信息时保守处理：全部按 extend 处理
        return 0, num_reqs, 0, num_actual_tokens
    # 复用通用拆分工具，区分 decode 区段与 extend 区段的请求数和 token 数
    common_attn_metadata = cast(
        CommonAttentionMetadata,
        SimpleNamespace(
            max_query_len=max_query_len,
            num_reqs=num_reqs,
            num_actual_tokens=num_actual_tokens,
            query_start_loc_cpu=query_start_loc,
            seq_lens_cpu_upper_bound=seq_lens_cpu_upper_bound,
            is_prefilling=None,
        ),
    )
    (
        num_decodes,
        num_extends,
        _num_prefills,
        num_decode_tokens,
        num_extend_tokens,
        _num_prefill_tokens,
    ) = split_decodes_prefills_and_extends(common_attn_metadata)
    return num_decodes, num_extends, num_decode_tokens, num_extend_tokens
```

vllm/v1/kv_cache_interface.py

定义 spec 级 max_num_blocks_per_req 契约：AttentionSpec 按 DCP/PCP 分片缩放，MambaSpec 保持复制语义，是本次改动的基础抽象。

```
class KVCacheSpec:
    def max_num_blocks_per_req(self, vllm_config: VllmConfig, max_len: int) -> int:
        """每个请求需要的 block table 行宽（即 worker 侧每行列数）。"""
        return cdiv(max_len, self.block_size)
```

```
class AttentionSpec(KVCacheSpec):
    def max_num_blocks_per_req(self, vllm_config: VllmConfig, max_len: int) -> int:
        # Attention KV 按 token 在 DCP/PCP rank 间交错分片，每个 rank
        # 只保存 max_len // (dcp * pcp) 个 token，所以行宽要除以总 CP 大小。
        parallel_config = vllm_config.parallel_config
        total_cp_size = (
            parallel_config.decode_context_parallel_size
            * parallel_config.prefill_context_parallel_size
        )
        return cdiv(max_len, self.block_size * total_cp_size)


class MambaSpec(KVCacheSpec):
    def max_num_blocks_per_req(self, vllm_config: VllmConfig, max_len: int) -> int:
        # Mamba 状态在 DCP/PCP rank 间复制而非分片，因此不做 CP 缩放。
        if vllm_config.cache_config.mamba_cache_mode == "align":
            # align 模式下 block 行按整序列位置索引（旧状态会被
            # remove_skipped_blocks 置空），行宽仍需覆盖 max_len。
            return cdiv(max_len, self.block_size) + self.num_speculative_blocks
        return cdiv(self.max_memory_usage_bytes(vllm_config), self.page_size_bytes)
```

评论区精华

主要交锋集中在三块：

1. 死锁风险（阻塞项，已修复）— GirasoleY 指出 skip-context-attention 若按 rank 本地 context 长度判定，context 短于一个 interleave 轮次时会落在部分 rank 上，导致部分 rank 跳过 DCP collectives 而其它 rank 进入，形成死锁。最终由 GirasoleY 提交 e201de2 改为基于全局 context_kv_lens_cpu 判定，并新增 test_skip_gate_rank_invariant_with_divergent_local_context 回归测试。
2. 设计整合（已落实）— GirasoleY 的 claude review 指出 block-size / CP-sharding 逻辑在 resolve_kv_cache_block_sizes、SingleTypeKVCacheManager、KVCacheCoordinator、MambaManager、_get_max_num_blocks_per_req 多处重复，建议 consolidate the KV-cache block-size / CP-sharding logic onto KVCacheSpec。最终通过 max_num_blocks_per_req 方法收口到 spec 子类。
3. 正确性 bug（已修复）— gemini-code-assist[bot] 指出 q_descale/k_descale/v_descale 应按 token 数而非请求数切片，且非 FP8 模型下为 None 需判空；最终 run_split_fa2_dcp_context_attention 中已是 q_descale[:num_decode_reqs] if q_descale is not None else None 形式。
4. 行为回退（已解决）— GirasoleY 担心 check_attention_cp_compatibility 去掉 interleave_size > 1 检查会破坏 #25049 验证的 FAv3 + MTP 支持。作者回复 this pr does support MTP, reverted this change，并同步回退 vllm/config/vllm.py 中 DCP 不支持 speculative decoding 的断言。
5. 疑问与澄清— ZJY0516 问为什么要拆 prefill/decode，作者说明是为了分离纯 prefill 与混合批次，最后演化为 FA2 专用 workaround；noooop 问 hybrid 模型为何默认关闭 prefix caching，作者解释 hybrid 支持 prefix caching 但保持 opt-in 等待成熟。

- DCP context-attention skip gate 的 rank-invariant 死锁风险 (correctness): 必须用全局 context_kv_lens_cpu 判定，而非 get_dcp_local_seq_lens 的本地结果。GirasoleY 提交 e201de2 修复，并新增测试 test_skip_gate_rank_invariant_with_divergent_local_context。
- KV-cache block-size / CP-sharding 逻辑应整合进 KVCacheSpec (design): 通过 KVCacheSpec.max_num_blocks_per_req 方法收口：AttentionSpec 缩放、MambaSpec 不缩放；resolve_kv_cache_block_sizes 与 kv_cache_coordinator 均改为读取 manager 实际 block_size。
- FA2 混合批次中 descale 切片与 None 判空 (correctness): 最终 run_split_fa2_dcp_context_attention 按 token 级切片，并在每个 descale 参数上使用 if q_descale is not None 条件。
- 保留 MTP / speculative decoding 与 DCP 的兼容性 (design): 作者回退相关改动，确认该 PR 支持 MTP；revert mtp 提交落地。
- 为什么需要拆分 prefill 与 decode 批次 (question): 拆分为 FA2 paged-varlen context attention 在 decode + extend + 纯 prefill 混合提交时的 workaround；代码用 should_split_fa2_dcp_context_attention 条件化，带 TODO 注释说明 FA4 就绪后可移除。
- hybrid 模型 prefix caching 默认策略 (question): 作者说明 hybrid 模型支持 prefix caching 但保持 opt-in 等待功能成熟，默认仍关闭。
- 合并前阻塞项与后续跟进 (other): hang 问题由后续 commit e201de2 修复，PR 通过并合并。

风险与影响

- 风险:
  1. 核心路径变更：KV cache 契约、BlockTable、注意力 forward 都是 decode 热路径，max_num_blocks_per_req 从隐式全局推导改为按 spec 计算，需警惕 block table 行宽不足或浪费。
  2. FA2 专用 workaround：run_split_fa2_dcp_context_attention 只覆盖 fa_version == 2 且混合批次的场景，代码留有 TODO 等待 FA4 支持 Qwen3.5 head_size = 256 后移除；期间 FA3/FA4 走旧路径，两条路径行为需保持一致。
  3. 死锁风险：skip gate 若误用 rank 本地 context 长度会引发 DCP 集体通信死锁；本 PR 已用全局判定并补测试，但仍属高敏感逻辑，后续改动需特别小心。
  4. 兼容性限制：DCP + hybrid 下仅接受 FullAttentionSpec 与 MambaSpec，sliding window 等其他 spec 的混合模型会被显式拒绝；PCP + hybrid 仍不支持。
  5. 性能：skip gate 需要一次 CPU 同步（context_kv_lens_cpu.max().item()），仅发生在存在 seq_lens_cpu_upper_bound 的批次；all_gather 非零维实现含 reshape/movedim，可能略逊于专用 kernel。 - 影响：对用户：Qwen3.5 系列等混合注意力模型首次可在 DCP 下运行，长上下文 decode 可获得跨 rank 并行收益；此前该组合被整体禁止。对系统：v1 引擎的 KV cache 契约（spec 级 max_num_blocks_per_req）和各层初始化逻辑发生结构性变化，影响 BlockTable 行宽、slot mapping kernel 与 CUDA graph 捕获路径，所有 DCP 用户都会受影响。对团队：需要长期维护 FA2 混合批次拆分这一兼容路径，并在 FA4 就绪后清理；prepare_dcp_dummy_context_metadata 等函数增加了

CUDA graph 捕获的复杂度。 - 风险标记：核心缓存契约变更，DCP 死锁风险，FA2 专用 workaround，CUDA graph 捕获路径变更，兼容性限制

关联脉络

- PR #50823 [Bugfix] Shard UniformTypeKVCacheSpecs block table width under DCP: 本 PR 引入按 spec 计算 max_num_blocks_per_req 的契约后，后续修复 DCP 下块表宽度不一致的问题，属于同一契约线的延续。
- PR #50432 [Bugfix][Hybrid] Fix cross-block race on num_accepted in MRv2 align prefix cache: 同为 hybrid（Mamba + attention）模型的 v1 路径修复，与本 PR 共享 Mamba 状态缓存与 hybrid 调度的上下文。