PR #51108 完整报告

vllm-project/vllm

[BugFix][KV Cache] Fix hybrid prefix caching with hidden-state extraction

合并时间：2026-08-06 14:42

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/51108

执行摘要

- 一句话：修复 hidden-state 缓存层破坏混合前缀缓存分块导致的启动崩溃
- 推荐动作：值得精读。此 PR 展示了典型的 " 隐式全局状态耦合导致启发式判断失效 " 的回归修复：EngineCore 全局重置 cache_config.block_size 的时序行为与 kv_cache_utils 推断逻辑之间的隐性依赖。修复思路（用模式判断替代数值比较、用 GCD 因子约束保证整除性）对维护 V1 hybrid KV cache 的工程师有直接参考价值；同时可关注 ivanium 提出的 follow-up：将 GCD 逻辑统一收敛进 resolve_kv_cache_block_sizes。

功能与动机

PR body 复现了 main 分支的启动崩溃：AssertionError: Each KV cache group's real block_size must be divisible by hash_block_size. block_sizes=[544, 544, 544, 544, 181], hash_block_size=98464，并明确指出该错误由 #50991（Mamba 默认启用 prefix cache）引入。ZJY0516 在 Issue 评论中提出核心意见：I feel like we should consider hidden spec when we caculate block size for mamba and full attention，即 hidden 层 block_size 的计算必须顾及与其他组尺寸的整除关系，避免浪费字节。

实现拆解

1. 根因定位：resolve_kv_cache_block_sizes（vllm/v1/core/kv_cache_utils.py）中判断 Mamba 组是否破坏整除性的条件是 g.kv_cache_spec.block_size != cache_config.block_size；而 EngineCore 内 kv_cache_manager.py 会把 cache_config.block_size 重置为所有组 block_size 的最小值，引入 HiddenStateCacheSpec（181）后，align 模式 Mamba（544）被误判为非 align，函数回退返回 (scheduler_block_size, scheduler_block_size)，其中 scheduler_block_size = lcm(544, 544, 544, 544, 181) = 98464，最终导致 coordinators 断言 544 % 98464 != 0 崩溃。
2. 修复判断逻辑：将条件改为 g.kv_cache_spec.mamba_cache_mode != "align"，直接表达设计意图，不再依赖易被全局最小值重置污染的 cache_config.block_size。
3. 修复 hidden 层 block_size 选取：get_kv_cache_groups 中新增 group_block_size = math.gcd(*(g.kv_cache_spec.block_size for g in groups))，并用新 helper _largest_divisor_at_most(value, limit) 取不超过每页 token 上限的最大整除因子，保证 hidden 组尺寸能整除其他组；同时新增 logger.info 输出所选 block size 与每页浪费字节 / 百分比（如 block size 136 时浪费 278528 字节即 25%）。

4. 测试与工程配套：新增 CPU 单测 test_hidden_state_group_preserves_hybrid_prefix_cache_granularity（tests/v1/core/test_kv_cache_utils.py），构造 FullAttentionSpec + MambaSpec(align) + HiddenStateCacheSpec 三层规格，断言 hidden 组 block_size==136 且 resolve_kv_cache_block_sizes 返回 (544, 136)；提交历史包含 lint 修复、评论意见处理、UT 修复与 main 合并，PR body 附有 e2e 通过日志。

关键文件：

- vllm/v1/core/kv_cache_utils.py（模块 KV 缓存；类别 source；类型 core-logic；符号 _largest_divisor_at_most, get_kv_cache_groups, resolve_kv_cache_block_sizes）：核心修复文件：修改 resolve_kv_cache_block_sizes 中 Mamba 非 align 判定条件，并在 get_kv_cache_groups 中为 HiddenStateCacheSpec 选取能整除其他组的 block_size，新增 _largest_divisor_at_most 与浪费字节日志。
- tests/v1/core/test_kv_cache_utils.py（模块 缓存测试；类别 test；类型 test-coverage；符号 test_hidden_state_group_preserves_hybrid_prefix_cache_granularity）：新增回归单测 test_hidden_state_group_preserves_hybrid_prefix_cache_granularity，精确复现 544/181 不可整除场景并验证修复后的分组与 block size 解析结果。

关键符号：_largest_divisor_at_most, get_kv_cache_groups, resolve_kv_cache_block_sizes, test_hidden_state_group_preserves_hybrid_prefix_cache_granularity

关键源码片段

vllm/v1/core/kv_cache_utils.py

核心修复文件：修改 resolve_kv_cache_block_sizes 中 Mamba 非 align 判定条件，并在 get_kv_cache_groups 中为 HiddenStateCacheSpec 选取能整除其他组的 block_size，新增 _largest_divisor_at_most 与浪费字节日志。

```
def _largest_divisor_at_most(value: int, limit: int) -> int:
    # 从 limit 向下线性扫描，返回 value 在 [1, limit] 内的最大因子。
    # value、limit 都是 KV cache block 尺寸（四位数以内），且只会在启动期调用几次，
    # 线性扫描足够快，不需要更复杂的质因数分解。
    for candidate in range(min(value, limit), 0, -1):
        if value % candidate == 0:
            return candidate
    return 1


def get_kv_cache_groups(vllm_config, kv_cache_spec):
    # 先把 HiddenStateCacheSpec 层摘出来，避免它们参与 attention 层的页面统一。
    hidden_specs = {
        k: v for k, v in kv_cache_spec.items() if isinstance(v, HiddenStateCacheSpec)
    }
    filtered_spec = {
        k: v for k, v in kv_cache_spec.items() if not isinstance(v, HiddenStateCacheSpec)
    }
    # ... 其余 attention/mamba 分组逻辑保持不变 ...
```

```
    groups = _get_kv_cache_groups_uniform_page_size(filtered_spec)

    # hidden 层必须与已有组保持整除关系，否则 resolve_kv_cache_block_sizes 计算出的
    # hash_block_size（各组的 GCD）会无法整除 hidden 组尺寸，启动时直接断言崩溃。
    if hidden_specs:
        common_page = get_uniform_page_size([g.kv_cache_spec for g in groups])
        # 其他组的 block_size 的 GCD，hidden 层 block_size 必须是它的因子。
        group_block_size = math.gcd(*(g.kv_cache_spec.block_size for g in groups))
        for name, spec in hidden_specs.items():
            per_token = spec.num_kv_heads * spec.head_size * get_dtype_size(spec.dtype)
            max_block_size = max(common_page // per_token, 1)
            # 既不超过每页能容纳的 token 数，又保持整除约束。
            new_bs = _largest_divisor_at_most(group_block_size, max_block_size)
            wasted_bytes = common_page - new_bs * per_token
            logger.info(
                "Using block size %d for hidden-state cache layer %s; "
                "page alignment wastes %d bytes (%.2f%%) per block",
                new_bs, name, wasted_bytes, wasted_bytes / common_page * 100,
            )
            aligned = replace(spec, block_size=new_bs, page_size_padded=common_page)
            groups.append(KVCacheGroupSpec([name], aligned))
    return groups

# resolve_kv_cache_block_sizes 中关键的 Mamba 判断修正。
scheduler_block_size = math.lcm(*group_block_sizes)

# Mamba 组在非 align 模式下块大小与调度块不一致，会破坏整除性，回退到调度块大小。
# 不能再用 block_size != cache_config.block_size 判断：EngineCore 会把
# cache_config.block_size 重置为所有组的最小值，hidden 层会把这个值拉低，
# 导致 align 模式的 Mamba 被误判为非 align（这就是 block_sizes=[544, ..., 181] 时报错
# hash_block_size=98464 的原因）。直接看 group spec 里的 mode 更稳健。
if any(
    isinstance(g.kv_cache_spec, MambaSpec)
    and g.kv_cache_spec.mamba_cache_mode != "align"
    for g in groups
):
    return scheduler_block_size, scheduler_block_size
```

tests/v1/core/test_kv_cache_utils.py

新增回归单测 test_hidden_state_group_preserves_hybrid_prefix_cache_granularity，精确复现 544/181 不可整除场景并验证修复后的分组与 block size 解析结果。

```
def test_hidden_state_group_preserves_hybrid_prefix_cache_granularity():
    block_size = 544
    # FullAttentionSpec 同时存 K 和 V，页大小为 544 * 1 * (512 + 512) * 2 = 1,114,112 字节。
    full_spec = FullAttentionSpec(
        block_size=block_size, num_kv_heads=1, head_size=512, dtype=torch.bfloat16,
    )
    mamba_spec = MambaSpec(
```

```
    block_size=block_size, shapes=((557056,),), dtypes=(torch.bfloat16,),
    mamba_cache_mode="align",
)
# hidden 层每 token 需要 3 * 1024 * 2 = 6144 字节，544 * 6144 > 页大小，
# 朴素整除得到 181，但 181 无法整除其他组的 544，必须取 544 的因子。
hidden_spec = HiddenStateCacheSpec(
    block_size=block_size, num_kv_heads=3, head_size=1024, dtype=torch.bfloat16,
)

groups = get_kv_cache_groups(
    _grouping_config(),
    {"model.full_attn": full_spec, "model.mamba": mamba_spec, "cache_only_layers.0": hidden_
    spec},
)

hidden_group = next(
    g for g in groups if isinstance(g.kv_cache_spec, HiddenStateCacheSpec)
)
# 544 在 181 以内的最大因子是 136，修复后 block_size 应为 136。
assert hidden_group.kv_cache_spec.block_size == 136

kv_cache_config = KVCacheConfig(num_blocks=1, kv_cache_tensors=[], kv_cache_groups=
groups)
vllm_config = SimpleNamespace(
    cache_config=SimpleNamespace(block_size=16, enable_prefix_caching=True, prefix_match_
    unit=None),
    parallel_config=SimpleNamespace(decode_context_parallel_size=1),
    kv_transfer_config=object(),
)
# 调度块 544 与哈希块 136 均能被所有组整除，前缀缓存粒度得以保留。
assert kv_cache_utils.resolve_kv_cache_block_sizes(kv_cache_config, vllm_config) == (544,
136)
```

评论区精华

1. 判断条件争议：ZJY0516 质疑 mamba 判断条件改动必要性，gcanlin 解释根因（EngineCore 全局重置 cache_config.block_size 为各组最小值，hidden 层将值拉低到 181 导致 align 模式被误判），并认为 mamba_cache_mode != "align" 更稳健。
2. helper 简化：ZJY0516 建议用线性扫描的简单实现，gcanlin 采纳。
3. 可观测性：ZJY0516 要求增加 block size 与内存浪费日志，gcanlin 已补充。
4. 后续重构：ivanium 建议用 resolve_kv_cache_block_sizes() 统一 GCD 逻辑，留作 follow-up，不阻塞本 PR。

- Mamba 非 align 判断条件是否需要修改 (correctness): 保留修改，改用 mamba_cache_mode != "align" 直接表达意图，更稳健。
- _largest_divisor_at_most 实现简化 (style): gcanlin 采纳简化建议。

- hidden 层 block size 与浪费内存的可观测性 (design): gcanlin 已补充 logger.info，输出 block size、浪费字节与百分比。
- GCD 逻辑统一到 resolve_kv_cache_block_sizes (design): 本 PR 保持现状，留作 follow-up 统一重构。

风险与影响

- 风险：
  1. 核心路径变更：get_kv_cache_groups 与 resolve_kv_cache_block_sizes 是 V1 hybrid KV cache 分组核心逻辑，回归影响面覆盖 hybrid 模型启动与调度正确性。
  2. 隐式状态耦合：修复依赖 cache_config.block_size 被全局重置的时序问题，同类耦合可能存在于其他启发式判断中，需警惕后续回归。
  3. 空间浪费：hidden 层 block_size 从 544 降为 136（每页 4 个 token），浪费 25% 页容量，由整除约束与每 token 6144 字节的固有代价决定。
  4. 复杂度：_largest_divisor_at_most 为 O(min(value, limit)) 线性扫描，但仅启动期调用且数值小，可忽略。
  5. 测试覆盖：目前仅有 CPU 单测，未覆盖多卡或真实模型 e2e；mamba_cache_mode 字段语义若调整需同步维护。 - 影响：影响范围：修复启用 hybrid KV cache（含 Mamba 层）+ hidden-state extraction（如 cache_only_layers 或类似 speculative 缓存层）模型在 main 分支无法启动的问题，恢复 #50991 默认启用 Mamba prefix cache 的可用性，并保持前缀缓存分块粒度（136 token）。对无 hidden-state 层模型零影响（该分支仅在 hidden_specs 非空时进入）。对团队无 API/ 配置变更，为 KV cache 分组逻辑提供了整除约束设计参考，遗留 GCD 统一重构的 follow-up。 - 风险标记：核心路径变更，启动崩溃回归修复，隐式全局状态耦合，依赖模式字段语义，已补回归测试

关联脉络

- PR #50991 [Mamba] enable prefix cache by default: 本 PR 直接修复 #50991 引入的回归，PR body 明确点名此错误由 #50991 引入。
- PR #51113 [Bugfix] Keep mamba align prefill chunks block-aligned past last_cache_position: 同属 Mamba align 模式前缀缓存正确性修复脉络，均涉及 hybrid KV cache 的对齐与分块逻辑。
- PR #51100 [Bugfix] Fix Mamba all-mode CPU offload boundary alignment: 同为 Mamba cache 边界对齐 bugfix，与 block 尺寸 / 边界约束相关的系列修复。
- PR #50276 [Bugfix] Fix packed KV block zeroing stride: 同为 V1 KV cache 正确性 bugfix，涉及 block 管理与页对齐问题的同类修复。