PR #44599 完整报告

vllm-project/vllm

[Bugfix] Mamba CPU Offloading

合并时间：2026-06-12 12:07

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44599

执行摘要

- 一句话：修复 Mamba CPU Offloading 在 block 边界命中错误
- 推荐动作：建议阅读以了解 Mamba 模型与 offloading 交互的特殊处理，以及设计决策中 round_down 位置、配置来源选择的权衡。单元测试（ test_mamba_align_cpu_offload ）的设计值得参考。

功能与动机

CPUOffloading + Mamba models + cache-mode "align" 组合在 prompt 恰好结束于 offloaded block 边界时产生错误输出。Offloading 代码将 MambaSpec 当作 sliding-window attention 处理，返回 hit-tokens 不是 mamba cache 大小的倍数，这不符合 Mamba 单状态的设计。必须修复以确保 Mamba 模型在该模式下结果与冷启动一致。

实现拆解

1. 在 vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py 中新增 resolve_mamba_align_size 函数，遍历 KV cache group 中的每项，识别 MambaSpec 且 mamba_cache_mode == "align" 的组，计算其 offloaded block size 作为对齐大小，并断言所有此类组大小一致。
2. 在 OffloadingConnectorScheduler.__init__ 中调用该函数，结果保存至实例属性 _mamba_align_size。
3. 在 _lookup 方法中，若 _mamba_align_size 不为 None，则将 max_hit_size_tokens 通过 round_down 向下舍入至对齐大小的倍数，从而确保 hit window 不跨越 Mamba block 边界，避免加载应被忽略的缓存块。
4. 在 tests/v1/kv_connector/unit/test_offloading_connector.py 中新增 test_mamba_align_cpu_offload 测试用例，使用 state-spaces/mamba-1.4b-hf 模型，分别测试 prompt 位于 block 边界和 mid-block 两种场景，比较冷启动与 CPU 命中输出是否一致；同时添加辅助函数 _get_output_str 和 _verify。

关键文件：

- vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py（模块 卸载调度；类别 source；类型 core-logic；符号 resolve_mamba_align_size）：核心修复文件，新增 resolve_mamba_align_size 函数并在 _lookup 中增加对齐约束

- tests/v1/kv_connector/unit/test_offloading_connector.py（模块 卸载测试；类别 test；类型 test-coverage；符号 test_mamba_align_cpu_offload, _get_output_str, _verify）：新增 Mamba CPU offloading 完整单元测试，覆盖边界场景

关键符号：resolve_mamba_align_size, _lookup, test_mamba_align_cpu_offload

关键源码片段

vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py

核心修复文件，新增 resolve_mamba_align_size 函数并在 _lookup 中增加对齐约束

```
# 新增函数：扫描 KV cache groups 获取 mamba 对齐大小
def resolve_mamba_align_size(spec: OffloadingSpec) -> int | None:
    """Return the mamba alignment size in tokens, or None if no group requires it."""
    mamba_align_size: int | None = None
    for idx, gpu_block_size in enumerate(spec.gpu_block_size):
        kv_spec = spec.kv_cache_config.kv_cache_groups[idx].kv_cache_spec
        if isinstance(kv_spec, MambaSpec) and kv_spec.mamba_cache_mode == "align":
            offload_block_size = gpu_block_size * spec.block_size_factor
            # 断言所有 Mamba group 具有相同的对齐大小
            assert (mamba_align_size is None or
                    mamba_align_size == offload_block_size)
            mamba_align_size = offload_block_size
    return mamba_align_size

# 在 OffloadingConnectorScheduler 初始化中调用：
self._mamba_align_size = resolve_mamba_align_size(spec)

# 在 _lookup 方法中应用（仅展示关键行）：
if self._mamba_align_size is not None:
    # 将 max_hit_size_tokens 向下对齐到 mamba block 边界，
    # 避免返回跨越 block 边界的命中。
    max_hit_size_tokens = round_down(
        max_hit_size_tokens, self._mamba_align_size
    )
```

tests/v1/kv_connector/unit/test_offloading_connector.py

新增 Mamba CPU offloading 完整单元测试，覆盖边界场景

```
# 测试 Mamba align cache 模式下 CPU offload 的正确性
@pytest.mark.parametrize("model,block_size,tp_size",
    [("state-spaces/mamba-1.4b-hf", 16, 1)])
def test_mamba_align_cpu_offload(model, block_size, tp_size):
    # 配置 offloading connector
    kv_transfer_config = KVTransferConfig(
        kv_connector="OffloadingConnector", kv_role="kv_both",
        kv_connector_extra_config={"cpu_bytes_to_use": 4 << 30, "block_size": block_size})
    llm = LLM(model=model, max_model_len=block_size*10,
              gpu_memory_utilization=0.85, tensor_parallel_size=tp_size,
```

```
            kv_transfer_config=kv_transfer_config,
            language_model_only=True, enable_prefix_caching=True,
            mamba_cache_mode="align", disable_hybrid_kv_cache_manager=False)
    # 构造长度为 block_size * 2 的 prompt（恰好 block 边界）
    tokenizer = llm.get_tokenizer()
    raw_ids = tokenizer.encode("Hi. Give me a set of trivia questions and their answers ")
    while len(raw_ids) < block_size * 2:
        raw_ids = tokenizer.encode("....") + raw_ids
    initial_ids = raw_ids[:block_size * 2]
    sampling_params = SamplingParams(max_tokens=128, temperature=0, ignore_eos=True)
    failures = []

    def _verify(llm, prompt, label):
        cold_outputs = llm.generate([prompt], sampling_params, use_tqdm=False)
        _wait_for_prefix_cache_reset(llm)
        cpu_outputs = llm.generate([prompt], sampling_params, use_tqdm=False)
        cold_text = cold_outputs[0].outputs[0].text
        cpu_text = cpu_outputs[0].outputs[0].text
        if cold_text != cpu_text:
            failures.append(f"{label}: mismatch")

    # 测试 block 边界 prompt
    _verify(llm, TokensPrompt(prompt_token_ids=initial_ids), "block-boundary-prompt")
    # 测试 mid-block prompt（添加一个 token 破坏边界）
    _verify(llm, TokensPrompt(prompt_token_ids=[0] + initial_ids), "block-mid-prompt")
    assert not failures
```

评论区精华

- round_down 放置位置：作者 varun-sundar-rabindranath 最初询问是否应该对 num_hit_tokens 进行舍入，后自行纠正意识到 num_hit_tokens 语义不同。最终确定对 max_hit_size_tokens 进行 round_down。
- 信号复用 vs 新增：orozery 建议复用 alignment_block_count 字段，但作者认为需要独立的 Mamba 对齐信号；后 orozery 承认 OffloadingConnectorScheduler 已有 specs 信息，不必在 GroupOffloadConfig 中增加字段，作者移除该字段并改用 resolve_mamba_align_size。
- 配置来源：orozery 提议直接使用 spec.vllm_config.cache_config.mamba_block_size 简化代码，作者发现集群中 mamba_block_size 未被更新，改用 KV groups 扫描以保证准确性。
- 测试模型规模：orozery 建议 CI 中只测试 ~1B 模型，作者将参数化列表中的 7B 模型替换为 1.4B 模型 state-spaces/mamba-1.4b-hf。
- 额外检查移除：orozery 指出 round_down 后的 if 检查（max_hit_size_tokens < 1）是冗余的，因为已有 max_hit_size_tokens - num_computed_tokens < offloaded_block_size 检查，作者同意移除。
    - round_down 应用于 max_hit_size_tokens 而非 num_hit_tokens (correctness): 确定 round_down max_hit_size_tokens 是正确的做法。

- 用独立 mamba_align_size 信号还是复用 alignment_block_count (design): 移除 GroupOffloadConfig.mamba_align_size，通过 resolve_mamba_align_size 函数从 spec 计算。
- 使用 cache_config.mamba_block_size 还是扫描 KV groups (design): 保留基于 KV groups 的扫描方式。
- CI 测试模型规模选择 (testing): 最终仅测试 state-spaces/mamba-1.4b-hf。
- 移除 round_down 后的冗余检查 (style): 移除该 if 检查。

风险与影响

- 风险:
  - 回归风险：变更仅影响 OffloadingConnectorScheduler._lookup 中 _mamba_align_size 非 None 的路径，非 Mamba 模型不受影响；通过 assert 保证所有 Mamba align 组大小一致，配置错误时可早期暴露。
  - 测试覆盖：仅使用 state-spaces/mamba-1.4b-hf 模型测试，可能无法覆盖所有 Mamba 变体或混合架构；但核心逻辑独立于模型，风险可控。
  - 性能风险：round_down 操作为 O(1)，对性能无影响。
  - 兼容性：无破坏性变更，新增的 resolve_mamba_align_size 函数仅在 OffloadingConnectorScheduler 中使用，不改变公共接口。
- 影响:
  - 用户影响：修复了 Mamba 模型在 align cache 模式下 CPU offloading 的输出错误，受影响用户可从此 PR 受益。
  - 系统影响：仅修改 CPU offloading 调度器的 lookup 逻辑，不影响 GPU-only 或其他 offloading 模式。
  - 团队影响：无，维护成本低。
  - 影响程度：中低——特定模型、特定配置下的 bugfix，但输出错误对用户体验影响大。
  - 风险标记：Mamba 特定路径，测试模型单一，assert 早期失败

关联脉络

- PR #44592 [Bugfix] OffloadingConnector: respect skip_reading_prefix_cache flag: 修改相同文件 vllm/distributed/kv_transfer/kv_connector/v1/offloading/scheduler.py，同属 OffloadingConnector bugfix 系列