

PR #53696 完整报告

vllm-project/vllm

[Bugfix][Models] Fix OpenPangu sleep mode with static sinks

合并时间: 2026-09-14 21:05

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/53696>

执行摘要

- 一句话: OpenPangu 静态 sink 睡眠模式崩溃修复
- 推荐动作: 建议精读, 尤其是关注两类可复用的 bug 模式: 一是 level-2 sleep 下 " 普通张量属性不会恢复、注册 buffer 才会 " 的 ownership 边界, 在排查其他模型的 sleep 回归时可直接复用; 二是 Attention.__init__ 与 CustomOp.__init__ 的协作式 MRO —— 显式重复初始化父类会重跑 nn.Module.__init__ 并静默清空已注册 buffer, 这类问题在编译 /CUDA graph 路径上极难定位。阅读时请以最终 diff 为准: PR 描述中关于 KV cache 管理器、num_reserved_blocks、reset_prefix_cache 和单测的部分已不再适用; 若关心 OpenPangu 在抢占 + prefix-cache reset 组合下的正确性, 需要跟踪被推迟的后续 PR。

功能与动机

动机来自 RFC #53343 (Sleep Mode Tensor Ownership and Recovery) 指出的语义缺口: sleep mode 的内存池 tag 是分配级的, level-2 sleep 会丢弃 weights 分配并重载参数, 但既不是 checkpoint 参数、也不在 named_buffers() 里的普通张量属性既不会被重载也不会被恢复。PR body 明确本变更是 "Follow-up to #53507 and part of #53343", 把 OpenPangu 与其 level-2 sleep 覆盖暴露出来的静态 sink 问题从通用 static-model-buffer PR 中单独隔离出来。具体故障有两条: 其一, param_sink_with_value=False 时 param_sink_value 是普通 torch.zeros 属性, sleep/wake 后会被破坏; 其二, StaticSinkAttention.__init__ 里重复调用 CustomOp.__init__(self) 会重跑 nn.Module.__init__, 清空 impl 及 _k_scale、_v_scale、_q_scale、_prob_scale, PR body 的 baseline 直接复现为 AttributeError: 'StaticSinkAttention' object has no attribute '_k_scale'。

实现拆解

1. 模型侧: 把静态 sink 常量提升为注册 buffer (vllm/model_executor/models/openpangu.py)。原实现中 param_sink_with_value=False 分支执行 self.param_sink_value = torch.zeros((param_sink_number, num_kv_heads, v_channels), ...), 它是普通张量属性; level-2 sleep 只对 checkpoint 参数和 named_buffers() 做保存 / 恢复, 这类属性既不被重载也不被恢复。改为 self.register_buffer("param_sink_value", torch.zeros(...), persistent=False) 后, 它进入既有的 level-2 buffer save/restore 路径, 同时 persistent=False 保证不进入 state_dict, checkpoint 契约与 load_state_dict 行为不变。param_sink_with_value=True 分支仍是带 output_dim=1 与 weight_loader 的 nn.Parameter, 权重加载路径未受影响。

2. 注意力层侧：移除重复的 CustomOp 初始化 (vllm/model_executor/layers/attention/static_sink_attention.py)。StaticSinkAttention.__init__ 在调用 Attention.__init__ 之后又显式调用 CustomOp.__init__(self)；由于 Attention 的协作式 MRO 已经覆盖 CustomOp，第二次调用等价于重跑 nn.Module.__init__，会把 Attention 刚注册的 impl 与量化 scale buffer 全部清空。删除该行后，sink 层在静态 sink + 量化 /level-2 sleep 组合下不再丢失 scale buffer。按 review 要求，作者最初为这行删除补写的大段解释性 docstring 也一并去掉，只保留最小改动。
3. 被 review 挡回并从 diff 移除的通用改动。PR 初版还包含：SinkFullAttentionManager 接受并转发 needs_kv_cache_zeroing、SingleTypeKVCacheManager 新增 num_reserved_blocks、BlockPool.reset_prefix_cache(num_reserved_blocks=0)，以及 tests/model_executor/layers/test_static_sink_attention.py 单测。aoshen02 与 ivanium 先后指出这些 scheduler / KV 管理器改动对单一模型过于 pervasive，且新测试文件不会被 CI 触发；作者逐一回复 "The fix of kv cache manager is removed." 与 "the test is removed"。
4. 测试与验证配套。合并后的 diff 不含任何测试文件；OpenPangu 的 generate -> sleep -> wake -> reload -> generate 回归由依赖 PR #53507 引入的 tests/model_executor/test_sleep_mode_tensor_ownership.py 参数化框架承接，PR body 给出 -k openpangu 在修复后 1 passed、在基线 commit 上因 _k_scale 缺失而失败的对照证据。作者同时说明本地因缺少 tblib 依赖、远程 CUDA 主机不可用，最终验证依赖 CI。
5. 遗留语义。因为 KV 管理器部分是移除而非修复，静态 sink 块在请求被抢占后仍保持 reserved，reset_prefix_cache 依旧不会排除这些已知保留块，这一正确性缺口被显式推迟到后续工作。

关键文件：

- vllm/model_executor/models/openpangu.py (模块 模型定义；类别 source；类型 data-contract；符号 param_sink_value)：本 PR 的主修复点：把 param_sink_with_value=False 时全零的 param_sink_value 从普通张量属性改为 register_buffer(..., persistent=False)，使其进入 level-2 sleep 的 buffer 保存 / 恢复路径，同时不污染 state_dict。
- vllm/model_executor/layers/attention/static_sink_attention.py (模块 注意力层；类别 source；类型 core-logic；符号 StaticSinkAttention, StaticSinkAttention.init)：删除 StaticSinkAttention.__init__ 中重复的 CustomOp.__init__(self)。该调用会经协作式 MRO 二次执行 nn.Module.__init__，清空 impl 以及 _k_scale、_v_scale、_q_scale、_prob_scale buffer，是 OpenPangu level-2 sleep 回归的直接崩溃原因。

关键符号：StaticSinkAttention.init, StaticSinkAttention.update_sink_kv, StaticSinkAttention.forward_native, OpenPangu 注意力层 __init__ 中的 param_sink_value 注册逻辑

关键源码片段

vllm/model_executor/models/openpangu.py

本 PR 的主修复点：把 `param_sink_with_value=False` 时全零的 `param_sink_value` 从普通张量属性改为 `register_buffer(..., persistent=False)`，使其进入 level-2 sleep 的 buffer 保存 / 恢复路径，同时不污染 `state_dict`。

```
# vllm/model_executor/models/openpangu.py
# OpenPangu 注意力层构造函数中的 sink 参数分支：本次修复只动 else 分支
if self.param_sink_with_value:
    # 有可学习 value sink：仍是普通 Parameter，由 weight_loader 从 checkpoint 填充，
    # output_dim=1 表示沿 KV head 维度做张量并行切分（未被本次改动触及）
    self.param_sink_value = torch.nn.Parameter(
        torch.empty(
            (
                self.param_sink_number,
                self.num_kv_heads,
                self.v_channels,
            ),
            device=current_platform.current_device(),
            dtype=config.torch_dtype,
        )
    )
    set_weight_attrs(
        self.param_sink_value,
        {
            "output_dim": 1,
            "weight_loader": self.weight_loader,
        },
    )
else:
    # 核心修复：无 value sink 时用全零常量。
    # 旧写法 self.param_sink_value = torch.zeros(...) 是普通张量属性，
    # 既不是 checkpoint 参数，也不会出现在 named_buffers() 中，
    # 于是 level-2 sleep 丢弃 weights 分配并恢复 buffer 之后，它不会被重建 / 恢复。
    # 注册为 buffer 后：
    # 1) 走既有的 level-2 buffer save/restore 路径，wake 后内容与地址都保持稳定；
    # 2) persistent=False 让它不进 state_dict，checkpoint 契约与 load_state_dict 行为不变；
    # 3) 张量仍常驻 device，CUDA graph 捕获到的指针保持有效。
    self.register_buffer(
        "param_sink_value",
        torch.zeros(
            (
                self.param_sink_number,
                self.num_kv_heads,
                self.v_channels,
            ),
            device=current_platform.current_device(),
            dtype=config.torch_dtype,
        ),
        persistent=False,
    )
```

```
# To enable dummy run with out weight
self.post_weight_load()
```

vllm/model_executor/layers/attention/static_sink_attention.py

删除 `StaticSinkAttention.__init__` 中重复的 `CustomOp.__init__(self)`。该调用会经协作式 MRO 二次执行 `nn.Module.__init__`，清空 `impl` 以及 `_k_scale`、`_v_scale`、`_q_scale`、`_prob_scale` buffer，是 OpenPangu level-2 sleep 回归的直接崩溃原因。

```
# vllm/model_executor/layers/attention/static_sink_attention.py
# StaticSinkAttention 派生自 Attention，而 Attention 的协作式 MRO 中已包含 CustomOp，
# 因此构造函数里只能初始化一次，不能再显式调用 CustomOp.__init__。
```

```
class StaticSinkAttention(Attention):
```

```
    def __init__(
        self,
        num_heads: int,
        head_size: int,
        scale: float,
        sink_len: int,
        attn_backend: type[AttentionBackend] | None = None,
        cache_config: CacheConfig | None = None,
        **kwargs,
```

```
):
```

```
    dtype = torch.get_default_dtype()
```

```
    # 没有 cache_config 时回退到 "auto"，避免后面取 cache_dtype 出错
```

```
    if cache_config is not None:
```

```
        kv_cache_dtype = cache_config.cache_dtype
```

```
    else:
```

```
        kv_cache_dtype = "auto"
```

```
    # 允许外部注入底层后端，否则按 head_size / dtype / kv_cache_dtype 自动选择
```

```
    if attn_backend is not None:
```

```
        underlying_attn_backend = attn_backend
```

```
    else:
```

```
        underlying_attn_backend = get_attn_backend(head_size, dtype, kv_cache_dtype)
```

```
    # 用 sink_len 包装底层后端，得到静态 sink 专用后端
```

```
    attn_backend = create_static_sink_attention_backend(
```

```
        underlying_attn_backend, # type: ignore[arg-type]
```

```
        sink_len=sink_len,
```

```
)
```

```
    # 只保留这一次初始化：它会沿协作式 MRO 依次执行 nn.Module.__init__ 与 CustomOp.__init__
```

```
    -
```

```
    Attention.__init__(
```

```
        self=self,
```

```
        num_heads=num_heads,
```

```
        head_size=head_size,
```

```
        scale=scale,
```

```
        cache_config=cache_config,
```

```

        attn_backend=attn_backend,
        **kwargs,
    )
    # 本 PR 删除的正是紧随其后的一行 CustomOp.__init__(self):
    # 重复执行会再次触发 nn.Module.__init__, 把刚注册的 impl 与
    # _k_scale / _v_scale / _q_scale / _prob_scale buffer 全部清空,
    # 表现为 AttributeError: 'StaticSinkAttention' object has no attribute '_k_scale'。
    # 作者最初补写的解释性 docstring 也已按 review 意见删除, 保持最小改动。

    # sink 状态: sink_len 固定, sink_key/sink_value 由模型层通过 update_sink_kv 注入
    self.sink_len = sink_len
    self.sink_populated = False
    self.sink_key = None
    self.sink_value = None

def update_sink_kv(self, sink_key, sink_value) -> None:
    # 权重加载或状态恢复后重新绑定静态 sink 的 K/V
    self.sink_key = sink_key
    self.sink_value = sink_value

def forward_native(
    self,
    query: torch.Tensor,
    key: torch.Tensor,
    value: torch.Tensor,
    output_shape: torch.Size | None = None,
) -> torch.Tensor:
    assert self.sink_key is not None and self.sink_value is not None, (
        "sink_key and sink_value have not been prepared"
    )
    # 尚未填充时, 首次前向把静态 sink 写进本层的 KV cache, 之后走父类实现
    if not self.sink_populated:
        self_kv_cache = self.kv_cache
        torch.ops.vllm.maybe_populate_sink(
            self_kv_cache, _encode_layer_name(self.layer_name)
        )

    return super().forward(query, key, value, output_shape)

```

评论区精华

review 的主线是 " 作用域收敛 ", 而不是代码细节:

ivanium (vllm/v1/core/block_pool.py) : "I feel this is a bit pervasive and too specific for this model. I'd suggest we fix the model local issues first, and leave full sleep mode support later."

aoshen02 (vllm/v1/core/single_type_kv_cache_manager.py, 随后 @ivanium) : "I think only this model need self.num_reserved_blocks." → ivanium 回复: "Yeah I

think this is kinda unique."

aoshen02 (tests/model_executor/layers/test_static_sink_attention.py) : "We can add the test to CI. I don't think simply adding a file will be triggered during CI."

作者 Ronald1995 的回应都是直接回退: "The fix of kv cache manager is removed."、"the test is removed"、"the docstring is removed"。最终 aoshen02 与 ivanium 均 APPROVED (ivanium: "LGTM!")。claude[bot] 因 PR 来自 fork 未执行自动审查, 其评论不构成技术意见。

- BlockPool.reset_prefix_cache 增加 num_reserved_blocks 参数被认为过于 pervasive (design): KV 管理器侧的通用改动从本 PR 移除, 本 PR 只保留 OpenPangu 与 StaticSinkAttention 的模型局部修复; 静态 sink 保留块与 reset_prefix_cache 的交互被推迟。
- SingleTypeKVCacheManager 新增 num_reserved_blocks 仅为单模型服务 (design): 两位 reviewer 一致认为该字段是模型专属需求, 不适合放在通用的 single type KV cache manager 中, 对应的 SinkFullAttentionManager 转发 needs_kv_cache_zeroing 改动也一并回退。
- 新增单测文件不会被 CI 触发 (testing): 新增单测被删除, 合并后的 diff 不再包含任何测试改动; 回归验证依赖 #53507 的 test_sleep_mode_tensor_ownership.py 参数化框架。
- StaticSinkAttention 中解释性 docstring 是否保留 (style): 删除 CustomOp.__init__ 时补充的大段注释被移除, 最终只保留删除该行本身这一最小改动。

风险与影响

- 风险:
 1. 测试覆盖被削弱: 合并后的 diff 只有 2 个源文件、无测试改动。作者按 review 意见删除了新增的 tests/model_executor/layers/test_static_sink_attention.py, 理由是单文件不会被 CI 触发; 回归保护完全依赖 #53507 引入的 tests/model_executor/test_sleep_mode_tensor_ownership.py -k openpangu。从提供的材料无法确认该用例最终落在哪个 PR, 也无法确认它是否被 CI 实际执行, 因此 buffer 注册与 _k_scale 保留这两个修复点缺少本 PR 自带的锁定测试。
 2. 正确性缺口被推迟而非消除: vllm/v1/core/block_pool.py 的 reset_prefix_cache 未增加保留块参数, single_type_kv_cache_manager.py 未维护 num_reserved_blocks, SinkFullAttentionManager 也未转发 needs_kv_cache_zeroing。静态 sink 块在请求被抢占后仍处于 reserved 状态, prefix-cache reset 或 KV 块零化路径仍可能触碰这些块, OpenPangu 在 "抢占 + reset_prefix_cache" 组合下仍有潜在正确性风险。
 3. PR 描述与最终 diff 不一致: PR body 的 Changes 列表仍包含已删除的 KV 管理器改动、新增单测和 docstring 说明, Test Plan 也引用了已不存在的测试文件。后续读者若以描述为准会误判改动范围, 属于文档漂移风险。
 4. 语义变化面较小但需注意: param_sink_value 从普通属性变为 buffer 后, 会出现在 named_buffers() 中并参与 to()/设备迁移与 level-2 的 buffer 保存 / 恢复; persistent=False 已隔离 state_dict, 但如果仓库其他位置按属性存在性判断该张量是否为参数 (例如 dummy run 或权重加载逻辑), 需要确认判定条件仍成立。

5. 单 commit、单作者：仅 1 个 commit (43fef06)，没有返工历史，但也意味着两处修复未被其他实现路径交叉验证。 - 影响：用户侧：OpenPangu 使用者在 level-2 sleep + 静态 sink 场景下不再遇到 `AttributeError: 'StaticSinkAttention' object has no attribute '_k_scale'` 这类直接崩溃，也不再因 `param_sink_value` 未被恢复而读到脏数据；由于改动只落在 `openpangu.py` 的 `param_sink_with_value=False` 分支与 `StaticSinkAttention.__init__`，其他模型与默认路径（level-1 sleep、非 sink 注意力）行为不变。系统侧：sleep mode 的 tensor ownership 规则（RFC #53343）又落地一个模型实例，验证了 "注册 buffer 走既有 save/restore 路径" 这一模式的可复用性；同时本次把 KV 管理器的通用改动挡回，避免了为单模型在 scheduler 关键路径上引入参数污染。团队侧：明确了后续同类问题应先做模型局部修复、通用框架改动单独提案的协作惯例，`SingleTypeKVCacheManager / BlockPool` 的接口设计将成为下一轮讨论焦点。 - 风险标记：合并后无测试改动，单测文件被移除，PR 描述与最终 diff 不一致，静态 sink 保留块未处理，scheduler 与 KV 管理器缺口被推迟，回归保护依赖 #53507 框架

关联脉络

- PR #53507 [Bugfix][Models] Register sleep-managed runtime buffers: 直接依赖与同一功能线：本 PR 是其 follow-up，复用其参数化 sleep-mode 回归测试框架（`tests/model_executor/test_sleep_mode_tensor_ownership.py`）；#53507 的 body 明确把 OpenPangu 与 Static Sink Attention 问题移到本 PR。两者采用完全相同的修复模式——把运行时张量注册为非持久 buffer。
- PR #53344 （#53507 body 中标注为拆分来源的 sleep-mode 系列 PR，仅被引用）：
#53507 的 body 写明 "Split from #53344"，本 PR 与之共享 #53343 的 sleep mode tensor ownership 治理方向；材料中未提供其具体 diff，无法进一步确认改动范围。