PR #34341 完整报告

sgl-project/sglang

[npu] [bugfix] Fix HiCache MHA backup for NPU

合并时间：2026-08-11 21:09

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34341

执行摘要

- 一句话：修复 NPU 上 HiCache MHA backup 回归
- 推荐动作：该 PR 值得快速浏览，改动虽小但揭示了一个重要的设计差异：NPU 池与 CUDA 池在内存布局上的根本不同（连续多层张量 vs. 指针数组）。关注点在于后端差异化的处理方式，以及 _validate_envelope_kv_layout 的校验逻辑。建议阅读 mha.py 中 backup_from_device_all_layer 的分支处理，理解 HiCache 在不同硬件上的适配模式。

功能与动机

PR body 明确说明：PR #30393 generalized the MHA HiCache backup path to support packed target and draft KV buffers. However, NPUMHATokenToKVPool uses contiguous multi-layer K/V tensors for the NPU transfer kernel and intentionally does not create k_data_ptrs/v_data_ptrs。因此泛化后的代码在 NPU 上会因访问不存在的指针数组而失败，本 PR 旨在修复这一回归。

实现拆解

1. 修改 python/sglang/srt/mem_cache/pool_host/mha.py 的 backup_from_device_all_layer：当 io_backend == "kernel_ascend" 时，不再调用 _resolve_device_transfer_buffers，直接将 device_kv_buffers 置为 None，避免访问 NPU 池不存在的 k_data_ptrs/v_data_ptrs。其他后端（如 kernel）保持原有逻辑不变，从 _resolve_device_transfer_buffers 获取指针数组和缓冲区。
2. 修改 python/sglang/srt/disaggregation/ascend/conn.py 的 send_kvcache：新增 dst_kv_item_len 和 dst_attn_tp_size 两个可选参数，并在函数开头调用 _validate_envelope_kv_layout 校验目标端 KV 布局，确保 PP 切片或 TP 分片时指针列表与布局参数一致。
3. 测试与配置：本 PR 未新增或修改测试文件，依赖现有 CI（run-ci 标签）覆盖 NPU 回归场景。

关键文件：

- python/sglang/srt/mem_cache/pool_host/mha.py（模块 缓存池；类别 source；类型 core-logic；符号 backup_from_device_all_layer, _resolve_device_transfer_buffers）：核心修复文件：在 backup_from_device_all_layer 中针对 kernel_ascend 后端跳过指针数组解析，避免 NPU 池访问不存在的 k_data_ptrs/v_data_ptrs。

- python/sglang/srt/disaggregation/ascend/conn.py（模块 PD 传输；类别 source；类型 core-logic；符号 send_kvcache, _validate_envelope_kv_layout）：为 Ascend PD 传输增加布局校验：新增参数并调用 _validate_envelope_kv_layout，确保 KV 传输时层数与 TP 大小匹配。

关键符号：backup_from_device_all_layer, _resolve_device_transfer_buffers, send_kvcache, _validate_envelope_kv_layout

关键源码片段

python/sglang/srt/mem_cache/pool_host/mha.py

核心修复文件：在 backup_from_device_all_layer 中针对 kernel_ascend 后端跳过指针数组解析，避免 NPU 池访问不存在的 k_data_ptrs/v_data_ptrs。

```
def backup_from_device_all_layer(
    self, device_pool, host_indices, device_indices, io_backend
):
    # NPU 传输使用连续多层张量，不构建 CUDA 风格的 ptr 数组，
    # 因此 kernel_ascend 后端直接跳过指针解析，避免访问缺失属性。
    if io_backend == "kernel_ascend":
        device_kv_buffers = None
    else:
        (
            device_k_data_ptrs,
            device_v_data_ptrs,
            device_k_buffers,
            device_v_buffers,
        ) = self._resolve_device_transfer_buffers(device_pool)
        device_kv_buffers = device_k_buffers + device_v_buffers

    if io_backend == "kernel":
        if self.layout == "layer_first":
            if self.can_use_jit:
                # JIT 路径：直接按 ptr 数组搬运所有层
                jit_transfer_hicache_all_layer(
                    k_ptr_dst=self.k_data_ptrs,
                    v_ptr_dst=self.v_data_ptrs,
                    indices_dst=host_indices,
                    k_ptr_src=device_k_data_ptrs,
                    v_ptr_src=device_v_data_ptrs,
                    indices_src=device_indices,
                    kv_cache_dst_stride_bytes=self.token_stride_size,
                    kv_cache_src_stride_bytes=self.token_stride_size,
                    element_size=self.element_dim * self.dtype.itemsize,
                )
            else:
                transfer_kv_all_layer(
                    src_k_layers=device_k_data_ptrs,
                    dst_k_layers=self.k_data_ptrs,
```

```
                src_v_layers=device_v_data_ptrs,
                dst_v_layers=self.v_data_ptrs,
                src_indices=device_indices,
                dst_indices=host_indices,
                item_size=self.token_stride_size,
                num_layers=self.layer_num,
            )
        elif self.layout == "page_first":
            # ... 其他 layout 的处理（省略）
            pass
```

python/sglang/srt/disaggregation/ascend/conn.py

为 Ascend PD 传输增加布局校验：新增参数并调用 _validate_envelope_kv_layout，确保 KV 传输时层数与 TP 大小匹配。

```
def send_kvcache(
    self,
    mooncake_session_id: str,
    prefill_kv_indices: npt.NDArray[np.int32],
    dst_kv_ptrs: list[int],
    dst_kv_indices: npt.NDArray[np.int32],
    executor: concurrent.futures.ThreadPoolExecutor,
    dst_layer_ids: Optional[List[int]] = None,
    dst_device_kv_indices: Optional[npt.NDArray[np.int32]] = None,
    dst_kv_item_len: Optional[int] = None,
    dst_attn_tp_size: Optional[int] = None,
):
    if dst_device_kv_indices is not None:
        raise NotImplementedError(
            "Ascend PD transfer does not support HiSparse "
            "destination device KV indices"
        )

    # 校验目标端 KV 布局：层数与 attention TP 大小必须与指针列表一致，
    # 避免 PP 切片或 TP 分片时错位。
    self._validate_envelope_kv_layout(
        dst_kv_ptrs, dst_kv_item_len, dst_attn_tp_size
    )
    # ... 后续传输逻辑省略
```

评论区精华

PR 无实质 review 讨论，仅由 sglang-npu-bot 自动审批通过（两次 APPROVED）。唯一 Issue 评论是 Bot 触发的 /tag-and-rerun-ci 命令，用于重跑 CI，属于流程自动化而非技术讨论。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险：mha.py 的改动只对 kernel_ascend 后端跳过指针解析，其他后端行为不变，但若 NPU 池未来开始构建 k_data_ptrs，该分支可能隐藏问题；建议后续补充 NPU 单测。
  2. 接口兼容风险：conn.py 的 send_kvcache 新增两个可选参数，若调用方未传递，_validate_envelope_kv_layout 收到 None 时的行为未在片段中展示，需确认其容错逻辑。
  3. 覆盖不足：无直接针对 kernel_ascend 分支的单元测试，回归依赖 CI 硬件环境。 - 影响：影响范围限定于 NPU（Ascend）平台：修复了 HiCache MHA backup 路径在 NPU 上的潜在崩溃，使启用 HiCache 的 NPU 用户能正常进行 K/V 备份。同时 Ascend PD 传输增加了布局校验，减少 PP/TP 分片错位风险。CUDA 等其他后端不受到影响，改动量小，风险可控。 - 风险标记：缺少测试覆盖，核心路径变更

关联脉络

- PR #30393 Generalize MHA HiCache backup path to support packed target and draft KV buffers: 本 PR 的动机来源：30393 泛化了 backup 路径，但未考虑 NPU 池不构建 k_data_ptrs 的情况，导致本 PR 需要修复该回归。