

PR #38402 完整报告

sgl-project/sglang

fix(npu): fix hybrid KV transfer with PP prefill in PD disaggregation

合并时间: 2026-09-16 09:10

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/38402>

执行摘要

- 一句话: 修复 NPU 混合线性 KV 在 PD + PP 下的传输出错
- 推荐动作: 值得精读, 尤其对从事 NPU/PD/PP 或 hybrid-linear 模型支持的同学。核心看点: `send_kvcache` 中按后端与布局切换“位置切分 vs layer-id 配对”的分发设计; `get_kv_layer_ids` 如何把稀疏层 id 与多 buffer 组对齐; `_send_mamba_state` 在传输前校验槽位大小的防御思路。合入前建议单独确认 `kimi_k3.py` 附带改动是否需要在 CUDA 侧补回归测试。

功能与动机

PR body 明确指出: Ascend 已支持 dense MLA/MHA 布局的 PP + PD, 但 Kimi-K3 使用 HybridLinearKVPool, 只有稀疏的 full-attention 层拥有 KV buffer, 且 Ascend 为每个这样的层暴露多个 buffer 组。因此“现有的按位置 PP 切分逻辑会在后续 PP stage 选出错误的 decode 条目” (existing positional PP slicing therefore selects the wrong decode entries for later PP stages)。修复目标是让 hybrid MLA + PP 场景改用 layer-id 配对; 同时 PR body 明确排除 DSpark prompt context collection、draft-KV 构建 / 传输与部分 draft 权重加载, 属于有意识的范围裁剪。

实现拆解

1. Ascend KV 传输分发入口改造 (`python/sglang/srt/disaggregation/ascend/conn.py`): 在 `send_kvcache` 中, `_validate_envelope_kv_layout` 之后、`group_concurrent_contiguous` 之前新增分支——当 `self.is_hybrid_mla_backend` and `self.pp_size > 1` 时直接调用 `_send_kvcache_generic`, 把 `self.kv_args.kv_data_ptrs / kv_item_lens / kv_layer_ids` 与 decode 端 `dst_layer_ids` 一并交给通用 layer-id 配对逻辑。这样 prefill 端 PP-local 条目与 decode 端全模型层条目按全局层 id 对齐, 绕过按位置切分。
2. KV layer-id 发布契约调整 (`python/sglang/srt/mem_cache/memory_pool.py`): `HybridLinearKVPool.get_kv_layer_ids` 在 `use_mla` and `_is_npu` and `layer_ids` 时, 先取 `get_contiguous_buf_infos()` 得到 buffer 指针数, 再返回 `layer_ids * (len(data_ptrs) // len(layer_ids))`, 即每个全注意力层按 buffer 组数重复发布全局层 id, 保证 layer 列表与 KV buffer 一一对应; 非 NPU 路径保持 `layer_ids` (MLA) 或 `layer_ids * 2` (MHA) 原行为。
3. Mamba 状态槽位校验 (`python/sglang/srt/disaggregation/mooncake/conn.py`): `_send_mamba_state` 新增 `dst_state_item_lens` 参数, 在 `build_transfer_entry_pairs` 配

对后逐对校验 src/dst 槽位大小，不一致时抛出 `RuntimeError`（提示两端 persistent state 布局需一致），替代原先可能发生的静默错误拷贝；`maybe_send_extra` 与 `_send_mamba_state_slice` 的回退路径同步透传 `dst_item_lens / dst_state_item_lens`。

4. 附带变更（`python/sglang/srt/models/kimi_k3.py`）：`_shared_experts_attn_tp_comm` 的计算条件从 `enable_shared_experts_attn_tp and self._ep_a2a and self._dp_attention and attn_tp_size > 1` 改为去掉 `self._dp_attention`。PR body 未解释该改动，提交历史上存在“change dp attn / revert dp attn change”的反复，最终保留放宽状态，属于需单独确认的跨后端行为变化。
5. 测试配套：`test/registered/unit/disaggregation/test_pp_hybrid_kv_transfer.py` 新增 `_RecordingAscendManager` 记录型 mock，验证 hybrid MLA + PP 走 `_send_kvcache_generic` 且 layer-id 正确透传；新增 `TestMambaSlotTransfer` 验证配对后槽位偏移计算与大小不匹配提前报错。`test/registered/unit/mem_cache/test_mamba_unit test.py` 新增 `test_hybrid_linear_kv_pool_npu_layer_ids_match_buffer_groups`，用 `patch` 切换 `_is_npu`，验证 device 分支与非 NPU 分支的 layer-id 输出。

关键文件：

- `python/sglang/srt/disaggregation/ascend/conn.py`（模块 KV 传输；类别 source；类型 core-logic；符号 `AscendKVManager.send_kvcache`）：核心分发逻辑：`send_kvcache` 为 hybrid MLA + PP>1 新增 layer-id pairing 分支，是本次修复的主入口。
- `python/sglang/srt/disaggregation/mooncake/conn.py`（模块 KV 传输；类别 source；类型 core-logic；符号 `MooncakeKVManager._send_mamba_state`，`MooncakeKVManager._send_mamba_state_slice`，`MooncakeKVManager.maybe_send_extra`）：Mamba 状态传输新增槽位大小校验与参数透传，防止 PP 配对后静默拷贝错误数据。
- `python/sglang/srt/mem_cache/memory_pool.py`（模块 内存池；类别 source；类型 core-logic；符号 `HybridLinearKVPool.get_kv_layer_ids`）：`get_kv_layer_ids` 新增 NPU 分支，按 buffer 组数重复发布层 id，是 layer-id 配对路径的数据契约来源。
- `test/registered/unit/disaggregation/test_pp_hybrid_kv_transfer.py`（模块 PD 测试；类别 test；类型 test-coverage；符号 `_RecordingAscendManager`，`TestAscendHybridPpDispatch.test_hybrid_pp_uses_layer_id_pairing_path`，`TestMambaSlotTransfer.test_stage1_uses_paired_slot_sizes_and_offsets`，`TestMambaSlotTransfer.test_slot_size_mismatch_rejected_before_transfer`）：新增 `_RecordingAscendManager` 与两组测试，锁定 hybrid MLA 走 layer-id 配对路径及 Mamba 槽位校验行为。
- `test/registered/unit/mem_cache/test_mamba_unittest.py`（模块 内存测试；类别 test；类型 test-coverage；符号 `test_hybrid_linear_kv_pool_npu_layer_ids_match_buffer_groups`）：验证 `get_kv_layer_ids` 在 NPU 上按 buffer 组数重复层 id，覆盖非 NPU 与多组数的设备分支。
- `python/sglang/srt/models/kimi_k3.py`（模块 模型层；类别 source；类型 data-contract；符号 `_shared_experts_attn_tp_comm`）：附带放宽共享专家 attn-tp 通信的 `_dp_attention` 条件，跨后端行为变化，需单独确认。

关键符号：AscendKVManager.send_kvcache, HybridLinearKVPool.get_kv_layer_ids, MooncakeKVManager._send_mamba_state, MooncakeKVManager._send_mamba_state_slice, MooncakeKVManager.maybe_send_extra, TestAscendHybridPpDispatch.test_hybrid_pp_uses_layer_id_pairing_path, TestMambaSlotTransfer.test_stage1_uses_paired_slot_sizes_and_offsets, TestMambaSlotTransfer.test_slot_size_mismatch_rejected_before_transfer, test_hybrid_linear_kv_pool_npu_layer_ids_match_buffer_groups

关键源码片段

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

核心分发逻辑：send_kvcache 为 hybrid MLA + PP>1 新增 layer-id pairing 分支，是本次修复的主入口。

```
# python/sglang/srt/disaggregation/ascend/conn.py
# send_kvcache 的混合线性分发头部：hybrid MLA + PP 走全局 layer-id 配对,
# 其余 Ascend 布局保持原有的按位置切分路径。

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"
        )
    self._validate_envelope_kv_layout(dst_kv_ptrs, dst_kv_item_len, dst_attn_tp_size)

    # Hybrid MLA 的 prefill 各 PP stage 只持有本 stage 的层条目,
    # 而 PP=1 的 decode 端注册了全部模型层：只有这种布局按全局
    # layer id 配对；其他 Ascend 布局继续走旧路径。
    if self.is_hybrid_mla_backend and self.pp_size > 1:
        return self._send_kvcache_generic(
            mooncake_session_id=mooncake_session_id,
            src_data_ptrs=self.kv_args.kv_data_ptrs,
            dst_data_ptrs=dst_kv_ptrs,
            item_lens=self.kv_args.kv_item_lens,
            prefill_data_indices=prefill_kv_indices,
            dst_data_indices=dst_kv_indices,
            executor=executor,
```

```

        src_layer_ids=self.kv_args.kv_layer_ids,
        dst_layer_ids=dst_layer_ids,
    )

    # 其余布局：先按索引分组，再走 MLA / MHA 的位置切分逻辑
    prefill_kv_blocks, dst_kv_blocks = group_concurrent_contiguous(
        prefill_kv_indices, dst_kv_indices
    )
    # ... 后续为原有的 get_mla_kv_ptrs_with_pp / get_mha_kv_ptrs_with_pp 分支

```

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

Mamba 状态传输新增槽位大小校验与参数透传，防止 PP 配对后静默拷贝错误数据。

```

# python/sglang/srt/disaggregation/mooncake/conn.py
# _send_mamba_state: 按 layer-id 配对后先校验两侧槽位大小,
# 再构造传输块，避免 PP 错层导致的内存越界或静默错数据。

def _send_mamba_state(
    self,
    req: TransferInfo,
    prefill_mamba_index: list,
    src_state_data_ptrs: list[int],
    src_state_item_lens: list[int],
    dst_state_data_ptrs: list[int],
    dst_mamba_index: list,
    src_layer_ids: Optional[List[int]] = None,
    dst_layer_ids: Optional[List[int]] = None,
    dst_state_item_lens: Optional[list[int]] = None,
):
    # Mamba 状态每个请求只有一个整体 index
    assert len(prefill_mamba_index) == 1, "Mamba should have single state index"

    transfer_blocks = []
    pairs = build_transfer_entry_pairs(
        src_layer_ids or [],
        dst_layer_ids or [],
        len(src_state_data_ptrs),
        len(dst_state_data_ptrs),
        allow_positional_fallback=self.pp_size == 1,
    )
    for i, j in pairs:
        dst_state_ptr = dst_state_data_ptrs[j]
        length = src_state_item_lens[i]
        # PP 配对后校验 src / dst 槽位大小一致；不一致直接报错，
        # 而不是把错误长度的数据拷到 decode 端
        if dst_state_item_lens and length != dst_state_item_lens[j]:
            raise RuntimeError(
                "Prefill/Decode Mamba slot size mismatch "
                f"(src={length}, dst={dst_state_item_lens[j]})."
            )

```

```

        "Configure matching persistent state layouts on both peers."
    )
    src_addr = src_state_data_ptrs[i] + length * int(prefill_mamba_index[0])
    dst_addr = dst_state_ptr + length * int(dst_mamba_index[0])
    transfer_blocks.append((src_addr, dst_addr, length))

    return self._transfer_data(req.mooncake_session_id, transfer_blocks)

```

python/sglang/srt/mem_cache/memory_pool.py

get_kv_layer_ids 新增 NPU 分支，按 buffer 组数重复发布层 id，是 layer-id 配对路径的数据契约来源。

```

# python/sglang/srt/mem_cache/memory_pool.py
# HybridLinearKVPool.get_kv_layer_ids:
# 全局层 id 必须与 full-attention KV buffer 组一一对应。

def get_kv_layer_ids(self):
    """Global layer ids aligned with the full-attention KV buffers."""
    layer_ids = list(self.full_attention_layer_id_mapping)
    # NPU 上每个 full-attention 层会暴露多个连续 buffer 组，
    # 层 id 需要按 buffer 组数重复，才能与 kv_data_ptrs 等长对齐；
    # 非 NPU 或非 MLA 布局保持原有行为。
    if self.use_mla and _is_npu and layer_ids:
        data_ptrs, _, _ = self.get_contiguous_buf_infos()
        return layer_ids * (len(data_ptrs) // len(layer_ids))
    # MLA 每层一组 KV；MHA 每层 K / V 两组，因此重复两遍
    return layer_ids if self.use_mla else layer_ids * 2

```

评论区精华

该 PR 没有任何人工 review 评论与 review thread: `review_comments_count = 0`，唯一的 Issue 评论是 `sglang-npu-bot` 的 `/tag-and-rerun-ci`（仅为触发 CI 重跑）。因此评审信号主要来自 PR body 的说明与 15 个提交的演进痕迹：提交中包含多次 `Merge branch 'main'`、一次 `dp attention` 相关改动及其 `revert (change dp attn / revert dp attn change)`、一次 `--unified-memory` 相关改动的 `revert (revert change related to --unified-memory)`，说明实现在 DP attention 条件与 unified memory 配置上经过反复取舍，最终落定的范围比最初尝试更窄。

- 无实质 review 讨论（仅 bot 触发 CI 重跑）(other): 合入决策主要由维护侧 CI 状态与 PR body 的实机 benchmark (GSM8K 96.5%、GPQA 98.4%) 支撑；`kimi_k3.py` 附带改动的动机未在讨论中说明。

风险与影响

- 风险:
 1. `kimi_k3.py` 附带变更为跨后端行为变化：移除 `self._dp_attention` 条件后，任何后端只要满足 `enable_shared_experts_attn_tp + _ep_a2a + attn_tp_size > 1` 就会走共享专家 `attn-tp` 通信分支（gather 输入 + reduce-scatter 输出）。PR body 未说明该动机，

提交历史显示曾 revert 过，需确认 CUDA/ROCm 等后端在无 DP attention 时不会引入额外通信开销或行为回归。

1. 整除假设无防御：memory_pool.py 中 len(data_ptrs) // len(layer_ids) 假设 buffer 组数是层数整数倍；若非整除会截断，导致 kv_layer_ids 与 kv_data_ptrs 长度不一致，配对时静默丢条目。当前仅有 layer_ids 非空保护，没有整除断言或日志。
2. Ascend 新分支绕过索引分组：ascend/conn.py 的 hybrid 分支在 group_concurrent_contiguous 之前 return，直接按 layer-id 配对，隐式假设 hybrid MLA + PP 场景不需要并发连续索引分组。这是设计契约，但对后续演进（如 HiSparse 类设备 KV 索引）是隐藏约束。
3. 测试为 mock 级：新增单测全部基于 _RecordingAscendManager / SimpleNamespace，未覆盖真实 NPU 传输路径与 build_transfer_entry_pairs 的失败模式；真实硬件验证依赖 PR body 中手工跑的 GSM8K (96.5%) 与 GPQA (98.4%) 结果。
4. Mamba 校验新增失败路径：RuntimeError 会让配置不一致的 PD 请求显式失败而非静默错数据，需要部署侧保证 prefill/decode 两端 persistent state 布局一致，否则会出现新的可观测报错。
 - 影响：影响范围集中在 NPU (Ascend) + PD 拆分 + PP prefill (pp_size > 1) + hybrid-linear (Kimi-K3) 组合：修复了 decode 端 KV 错层与越界风险，并给 Mamba 状态传输加了槽位一致性护栏。GPU 等后端的 dense MLA/MHA 传输路径保持不变 (get_mla_kv_ptrs_with_pp / get_mha_kv_ptrs_with_pp 分支未动)。但 kimi_k3.py 的共享专家条件放宽是全后端生效的，把影响面从 NPU 扩大到所有 Kimi-K3 部署。团队层面，该 PR 确立了“每 NPU buffer 组对应一个全局层 id”的数据契约，为 PR body 中明确排除的 DSpark prompt context、draft-KV 等后续工作铺路。整体影响程度中等。
 - 风险标记：kimi_k3 附带跨后端行为变更，NPU 专用分支绕过索引分组，buffer 组数整除假设无防御，测试为 mock 级，缺真实 NPU 集成覆盖，Mamba 槽位校验新增显式失败路径

关联脉络

- PR #38984 [Intra-node PD][DSV4] Pack all layers into one batch for INTRA_NODE_NVLINK path: 同改 python/sglang/srt/disaggregation/mooncake/conn.py 的 KV 传输路径，属于 PD 传输批处理与配对逻辑的连续演进。
- PR #39584 [Fix][PD] Give prefill and decode their own RDMA NICs in disaggregation tests: 同为 PD 拆分场景的稳定性修复，涉及多个 test_disaggregation_* 测试的 fixture 调整，与本 PR 的 PD + PP 测试目标互补。
- PR #39544 [misc] Trim redundant variants from the 8-gpu-h20 disaggregation test suite: 对 disaggregation 测试套件做收敛与裁剪，与本 PR 扩展 PP hybrid KV 传输测试属于同一测试空间的管理脉络。