PR #36025 完整报告

sgl-project/sglang

[AMD][MORI] Deduplicate CP-replicated state transfers

合并时间：2026-08-25 05:43

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36025

执行摘要

- 一句话：MORI 下仅 CP rank 0 发送复制状态，传输流量降 85%
- 推荐动作：值得快速浏览：整体改动仅 4 个文件、71 行，核心是 _should_skip_cp_replicated_state_transfer() 的策略上移与 MORI send() 的去重接入。对从事 PD 传输或 AMD 后端的工程师，公共策略 + layer split 例外的设计值得借鉴；普通使用者无需深入。若后续要在 MORI/Mooncake 之外的新后端复用，注意其前提是「CP all-gather 后 state 全等」。

功能与动机

PR body 指出：开启 SGLANG_DISAGGREGATION_ALL_CP_RANKS_TRANSFER=1 的 Prefill CP 场景下，MORI 将 KV pages 按 CP rank 切分，但 state_indices 原样传给每个 rank，导致同一份 state 被重复传输 cp_size 次；而 Prefill CP 阶段会先在全局 token 序上做 all-gather，每个 rank 本就持有完整 state。本 PR 是 Mooncake 侧同类修复（PR#32620）的 MORI 版本，目标是把 state 复制流量降为单份。

实现拆解

1. 策略上移：在 python/sglang/srt/disaggregation/common/conn.py 的 CommonKVManager 新增 _should_skip_cp_replicated_state_transfer()，用 attn_cp_size > 1、attn_cp_rank != 0、enable_dsa_cache_layer_split 三个条件判定是否跳过 state 传输。选择放在公共基类是为了让 Mooncake 与 MORI 共用同一语义，避免两份内联判断漂移。
2. Mooncake 复用：mooncake/conn.py 的 _get_dsa_cache_transfer_skip_flags() 删除原来的内联条件，改为调用公共方法，同时移除不再使用的 get_parallel 导入，行为等价，消除重复实现。
3. MORI 去重：mori/conn.py 的 MoriKVManager.send() 在 _prepare_send_indices 之后、_normalize_state_indices_per_component 与 _record_transfer_indices 之前，把 state_indices 替换为 None（当公共策略判定需要跳过时）；KV 分片、aux 数据、完成通知路径完全不变，并保留 cache layer-split 下每 rank 发送自己 state 层的行为。
4. 测试与验证：test_disaggregation_wire.py 新增 TestCPReplicatedStateTransfer，用 5 组参数化用例覆盖单 rank、rank 0、非零 rank、layer split 四种情形，并断言 Mooncake 的 _get_dsa_cache_transfer_skip_flags 返回 (False, True)，保证重构后行为等价。PR body 另附 MI355X 上 CP8 的准确率与流量实测。

关键文件：

- python/sglang/srt/disaggregation/common/conn.py（模块 公共传输层；类别 source；类型 core-logic；符号 _should_skip_cp_replicated_state_transfer）：新增 _should_skip_cp_replicated_state_transfer() 公共策略，是本 PR 的核心设计：把 CP 复制状态去重逻辑从 Mooncake 私有实现提升为所有传输后端可复用的基类方法，并文档化 layer split 例外。
- python/sglang/srt/disaggregation/mori/conn.py（模块 MORI 后端；类别 source；类型 core-logic；符号 send）：MORI 发送路径 send() 是去重的实际落地位置：在 enqueue 与记账前把非零 CP rank 的 state_indices 置空，同时保住 KV 分片、aux 与 layer split 语义。
- python/sglang/srt/disaggregation/mooncake/conn.py（模块 Mooncake 后端；类别 source；类型 dependency-wiring；符号 _get_dsa_cache_transfer_skip_flags）：重构对象：删除 _get_dsa_cache_transfer_skip_flags() 里的内联 CP 跳过条件与 get_parallel 导入，改为调用公共策略，验证行为等价并消除实现漂移。
- test/registered/unit/disaggregation/test_disaggregation_wire.py（模块 传输测试；类别 test；类型 test-coverage；符号 TestCPReplicatedStateTransfer, test_only_nonzero_cp_ranks_without_layer_split_skip_state, test_mooncake_uses_common_cp_state_policy）：新增 TestCPReplicatedStateTransfer 单测，用 5 组参数化用例覆盖单 rank、rank 0、非零 rank 与 layer split 4 类场景，并验证 Mooncake 复用公共策略后 skip 行为不变，是本次改动的回归保障。

关键符号：_should_skip_cp_replicated_state_transfer, send, _get_dsa_cache_transfer_skip_flags

关键源码片段

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

新增 _should_skip_cp_replicated_state_transfer() 公共策略，是本 PR 的核心设计：把 CP 复制状态去重逻辑从 Mooncake 私有实现提升为所有传输后端可复用的基类方法，并文档化 layer split 例外。

```
# 公共策略：判断当前 prefill rank 是否需要跳过 CP 复制状态的传输。
# 背景：Prefill CP 阶段先做 all-gather 再写入 state 池，
# 故每个 CP rank 持有的 state 完全相同；当各 rank 各自发送 KV 分片时，
# 只需 rank 0 发送全量 state，非零 rank 跳过即可避免流量放大 cp_size 倍。
# 例外：DSA cache layer split 开启时每个 rank 持有不同的 state 层，
# 必须由各 rank 分别发送自己拥有的部分，因此不能跳过。
def _should_skip_cp_replicated_state_transfer(self) -> bool:
    """Whether this prefill rank should omit CP-replicated state.

    Prefill CP materializes global token order before writing state pools, so
    every CP rank holds the same state. When all CP ranks transfer their KV
    shards, only rank 0 needs to send that state. Cache layer split is the
    exception because each CP rank owns different state layers.
```

```
    """
    return (
        self.attn_cp_size > 1 # CP 规模大于 1 才存在复制语义
        and self.attn_cp_rank != 0 # rank 0 负责发送全量 state
        and not get_parallel().enable_dsa_cache_layer_split # layer split 例外
    )
```

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

MORI 发送路径 send() 是去重的实际落地位置：在 enqueue 与记账前把非零 CP rank 的 state_indices 置空，同时保住 KV 分片、aux 与 layer split 语义。

```
def send(
    self,
    kv_indices: npt.NDArray[np.int32],
    state_indices: Optional[List] = None,
    num_kv_tokens: Optional[int] = None,
):
    # 统一的发送前准备：切 KV 分片、定位索引切片、标记最后一块并计算整体跳过
    kv_indices, index_slice, is_last_chunk, should_skip = (
        self._prepare_send_indices(kv_indices, state_indices)
    )
    if should_skip:
        return

    # 核心去重点：Prefill CP 下所有 rank 持有相同 state，仅 rank 0 需要发送。
    # 非零 CP rank 在 enqueue 与记账前把 state_indices 置空，KV 分片不受影响。
    transfer_state_indices = (
        None
        if self.kv_mgr._should_skip_cp_replicated_state_transfer()
        else state_indices
    )

    # 只在最后一块按组件归一化 state 索引，供对端正确还原状态布局
    normalized_state = (
        _normalize_state_indices_per_component(transfer_state_indices)
        if is_last_chunk
        else None
    )
    # 记账与入队统一使用裁剪后的 state，保证统计与传输一致
    self._record_transfer_indices(kv_indices, transfer_state_indices)
    wait_event = getattr(self, "_early_send_wait_event", None)
    self._early_send_wait_event = None
    self.kv_mgr.enqueue_transfer(
        _TransferChunk(
            sender=self,
            kv_indices=kv_indices,
            index_slice=index_slice,
            is_last_chunk=is_last_chunk,
            aux_index=self.aux_index if is_last_chunk else None,
```

```
                normalized_state=normalized_state,
                wait_event=wait_event,
            )
        )
        self._maybe_finalize_if_room_failed()
```

评论区精华

该 PR 没有实质 review 评论：HaiShaw 直接 APPROVED，唯一的 issue 评论是 /tag-and-rerun-ci 触发 CI 重跑。PR body 中的设计决策（只让 rank 0 发送全量 state、layer split 例外、KV 分片与完成通知路径不变）由作者说明并附 MI355X 实测数据背书，未产生争议。CI 状态显示 Base 测试通过，Extra 与 AMD ROCm 7.2 两个 job 失败，失败原因未在材料中说明。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 正确性依赖「CP all-gather 后各 rank state 全等」这一前提：一旦未来某种 state 类型不再被 all-gather 复制（例如硬件私有状态），_should_skip_cp_replicated_state_transfer() 会让非零 rank 漏发数据，导致 decode 侧状态缺失；该函数位于公共基类，任何子类扩展 state 类型时都要重新审视该前提。
  2. MORI 在 send() 中把 state_indices 置 None 后再记账，若未来同一条路径另有消费 state_indices 的旁路（如重试、统计），可能拿到裁剪后的数据。
  3. Mooncake 重构移除了 get_parallel 导入，若存在从 mooncake/conn.py 间接导入 get_parallel 的外部代码会报 ImportError，仓库内未见此类调用。
  4. 测试只覆盖 CPU 单测与策略分支，未把 CP8 端到端传输测试入库，回归保障依赖 PR body 的人工准确率验证；Extra 与 AMD ROCm CI 失败原因未说明。 - 影响：该 PR 只影响 PD 场景下启用 Prefill CP 的 MORI 与 Mooncake 传输路径。实测（MI355X、DeepSeek-V4-Pro、Prefill CP8 / Decode TP8）：kv_transfer_total_mb 降低 84.97%，可显著缓解跨节点与互联带宽压力，为更大 CP 规模铺路；TTFT 与输入吞吐几乎不变，说明该链路当前不是延迟瓶颈。对代码结构而言，CP 复制状态策略从 Mooncake 私有逻辑提升为 CommonKVManager 公共策略，后续新传输后端可直接复用，减少实现漂移。无模型 forward、无 kernel、无 API 变更，对普通推理用户透明。 - 风险标记：PD 传输核心路径，依赖 rank 0 全量状态，端到端传输测试未入库，公共策略影响多后端

关联脉络

- PR #32620 Deduplicate CP-replicated state transfers (Mooncake): 本 PR body 明确说明其是 #32620 的 MORI counterpart，将 Mooncake 侧已有的 CP 复制状态去重策略推广到 MORI；该标题为按 body 描述推断，仓库提供的历史 PR 列表中未包含该条目。
- PR #35957 Fix recurrent state loss on decode retraction: 同属 PD state 生命周期与传输正确性修复线，改动 schedule_batch 的 retraction 状态备份与传输，与本 PR 的 state 传输去重互为补充。

- PR #35840 Add PD test for inkling with mxfp8 KV: 同为 disaggregation 传输链路新增测试覆盖，扩展 PD state/KV 传输的回归保障。