PR #28514 完整报告

sgl-project/sglang

Fix ScheduleBatch req pool CPU metadata

合并时间：2026-06-18 10:25

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28514

执行摘要

- 一句话：修复 ScheduleBatch 请求池 CPU 元数据不同步问题
- 推荐动作：值得精读，尤其是对 SGLang 调度器内部请求池管理感兴趣的人。设计权衡：应在源头初始化 CPU 镜像，还是在所有消费点添加防御？PR 采用混合方案，但评审者认为仅源修复即可，此分歧值得关注。

功能与动机

ScheduleBatch keeps both device-side req_pool_indices and a CPU mirror req_pool_indices_cpu. ... This can lead to stale request-pool metadata being observed after filtering, or crashes when merging batches with a missing CPU mirror.

实现拆解

1. 初始化补全：在 scheduler.py 的 _build_hisparse_decode_batch 中，构建 HiSparse 解码批次时显式设置 batch.req_pool_indices_cpu（从 reqs 提取后构造张量）。
2. 解码前恢复：在 schedule_batch.py 的 prepare_for_decode 开头，若 req_pool_indices_cpu 缺失但设备张量存在，则通过 detach().cpu() 复原。
3. 空过滤清空：在 filter_batch 中，当所有请求被过滤掉时，将 req_pool_indices、req_pool_indices_cpu、seq_lens、seq_lens_cpu、orig_seq_lens 置为空张量，并清零 seq_lens_sum 和 out_cache_loc。
4. 合并前恢复：在 merge_batch 中，对自身和对方 batch 均检查并恢复缺失的 CPU 镜像。
5. 回归测试：新增 test_schedule_batch_req_pool_indices.py，涵盖解码前恢复、空过滤清空、合并前恢复三个场景，使用 mock 隔离外部依赖。

关键文件：

- test/registered/unit/managers/test_schedule_batch_req_pool_indices.py（模块 请求池同步；类别 test；类型 test-coverage；符号 TestScheduleBatchReqPoolIndices, test_prepare_for_decode_restores_missing_req_pool_indices_cpu, test_filter_batch_to_empty_clears_req_pool_metadata, test_merge_batch_restores_missing_req_pool_indices_cpu）：新增专用测试文件，覆盖三个回归场景，确保修复的可靠性。
- python/sglang/srt/managers/schedule_batch.py（模块 批次管理；类别 source；类型 core-logic；符号 prepare_for_decode, filter_batch, merge_batch）：核心类修改，在

prepare_for_decode、filter_batch、merge_batch 中添加防御性恢复 / 清空逻辑。

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic ；符号 _build_hisparse_decode_batch）：根源修复点，在 _build_hisparse_decode_batch 中显式初始化 req_pool_indices_cpu。

关键符号：_build_hisparse_decode_batch, prepare_for_decode, filter_batch, merge_batch, test_prepare_for_decode_restores_missing_req_pool_indices_cpu, test_filter_batch_to_empty_clears_req_pool_metadata, test_merge_batch_restores_missing_req_pool_indices_cpu

关键源码片段

python/sglang/srt/managers/schedule_batch.py

核心类修改，在 prepare_for_decode、filter_batch、merge_batch 中添加防御性恢复 / 清空逻辑。

```
# ---------- prepare_for_decode 中的防御性恢复 ----------
def prepare_for_decode(self):
    self.forward_mode = ForwardMode.DECODE
    bs = len(self.reqs)
    # 如果 CPU 镜像缺失但设备张量存在，则通过 detach().cpu() 恢复
    if self.req_pool_indices_cpu is None and self.req_pool_indices is not None:
        self.req_pool_indices_cpu = (
            self.req_pool_indices.detach().cpu().to(dtype=torch.int64)
        )
    self.input_embeds = None
    # ... 其余解码准备逻辑

# ---------- filter_batch 中空过滤清空元数据 ----------
def filter_batch(self, chunked_req_to_exclude=None, keep_indices=None):
    if keep_indices is None:
        # ... 计算 keep_indices
    if keep_indices is None or len(keep_indices) == 0:
        # Filter out all requests: 清空所有请求池和序列长度元数据
        self.reqs = []
        self.req_pool_indices = torch.empty(0, dtype=torch.int64, device=self.device)
        self.req_pool_indices_cpu = torch.empty(0, dtype=torch.int64)
        self.seq_lens = torch.empty(0, dtype=torch.int64, device=self.device)
        self.seq_lens_cpu = torch.empty(0, dtype=torch.int64)
        self.orig_seq_lens = torch.empty(0, dtype=torch.int32, device=self.device)
        self.out_cache_loc = None
        self.seq_lens_sum = 0
        return
    # ... 非空过滤逻辑

# ---------- merge_batch 中的防御性恢复 ----------
def merge_batch(self, other: ScheduleBatch):
    # 合并前确保双方 CPU 镜像都存在
```

```
    if self.req_pool_indices_cpu is None and self.req_pool_indices is not None:
        self.req_pool_indices_cpu = (
            self.req_pool_indices.detach().cpu().to(dtype=torch.int64)
        )
    if other.req_pool_indices_cpu is None and other.req_pool_indices is not None:
        other.req_pool_indices_cpu = (
            other.req_pool_indices.detach().cpu().to(dtype=torch.int64)
        )
    # ... 实际合并操作
```

python/sglang/srt/managers/scheduler.py

根源修复点，在 _build_hisparse_decode_batch 中显式初始化 req_pool_indices_cpu。

```
def _build_hisparse_decode_batch(self, reqs):
    """Build a ScheduleBatch for hisparse requests transitioning from staging to decode."""
    device = self.device

    batch = ScheduleBatch.init_new(
        reqs=reqs,
        req_to_token_pool=self.req_to_token_pool,
        token_to_kv_pool_allocator=self.token_to_kv_pool_allocator,
        tree_cache=self.tree_cache,
        model_config=self.model_config,
        enable_overlap=self.enable_overlap,
        spec_algorithm=self.spec_algorithm,
    )

    # 先提取 req_pool_indices 列表，避免重复遍历
    req_pool_indices = [r.req_pool_idx for r in reqs]
    batch.req_pool_indices = torch.tensor(
        req_pool_indices, dtype=torch.int64, device=device
    )
    # 原来缺失的 CPU 镜像同步初始化
    batch.req_pool_indices_cpu = torch.tensor(req_pool_indices, dtype=torch.int64)

    seq_lens = [len(r.origin_input_ids) + len(r.output_ids) - 1 for r in reqs]
    batch.seq_lens = torch.tensor(seq_lens, dtype=torch.int64, device=device)
    batch.seq_lens_cpu = torch.tensor(seq_lens, dtype=torch.int64)
    batch.orig_seq_lens = torch.tensor(seq_lens, dtype=torch.int32, device=device)
    batch.seq_lens_sum = sum(seq_lens)
    # ... 其余初始化
    return batch
```

评论区精华

评审者 gemini-code-assist 指出 filter_batch 在非空过滤时若 req_pool_indices_cpu 为 None 会导致切片崩溃，建议在方法开头添加类似恢复逻辑，并增加对应测试。评审者 hnyls2002 认为 schedule_batch.py 中的多处防御性恢复是冗余的，强调只有 scheduler.py 的初始化修复是必要的，并标记了“冗余保护”请移除，但最终合并时仍保留了这些逻辑。

- filter_batch 中缺失 CPU 镜像时非空过滤可能崩溃 (correctness): PR 未采纳此建议，仅修复了空过滤场景。后续可能仍需处理非空过滤时的防御恢复。
- schedule_batch.py 中防御性恢复是否冗余 (design): PR 作者保留了这些 guard，hnyls2002 仍批准了 PR，可能认为虽冗余但无害。

风险与影响

- 风险：添加的防御性恢复可能掩盖源头未同步的 bug，但当前修改已补全已知路径，不会引入新问题。性能上，增加少量 CPU 拷贝（detach().cpu()），仅在缺失时触发，对吞吐影响可忽略。兼容性良好，未改动数据模型，纯新增保护代码。
- 影响：修复了可能导致推理崩溃的隐含 bug，尤其影响使用 HiSparse 解码、批次过滤后合并等路径。用户受影响概率低，但崩溃影响大。新增测试确保回归覆盖，提升系统稳定性。
- 风险标记：核心路径变更，讨论中存在设计分歧，防御性恢复可能掩盖源头问题

关联脉络

- 暂无明显关联 PR