PR #22811 完整报告

sgl-project/sglang

Fix for the low-probability garbled output issue in the GLM-5 series models.

合并时间：2026-04-17 09:52

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22811

执行摘要

- 一句话：修复 GLM-5 系列模型在 HiCache 下低概率输出乱码问题，确保 NSA 索引读取与缓存加载同步。
- 推荐动作：该 PR 值得精读，因为它揭示了 HiCache 与 NSA 模型交互时一个微妙但关键的同步问题。关注点在于如何通过统一的 layer_transfer_counter 机制来协调异步缓存加载与数据访问，这是分布式缓存系统中常见的并发控制模式。

功能与动机

根据 PR body 描述，当使用 GLM-5 模型并启用 L2 HiCache 时，存在小概率（约万分之几）的输出乱码问题。根本原因是 NSA 模型的稀疏特性导致缓存损坏可能以多种方式影响请求，包括乱码、重复、罕见字符甚至模型精度下降。具体来说，异步层级缓存加载与 NSA 索引器读取之间存在竞争，某些索引读取方法未等待缓存加载完成，导致读取到陈旧或部分数据。

实现拆解

1. 同步 NSA 索引读取方法：在 memory_pool.py 中的三个 NSA 索引读取方法（get_index_k_continuous、get_index_k_scale_continuous、get_index_k_scale_buffer）开头添加 layer_transfer_counter.wait_until(layer_id - self.start_layer) 调用，确保在 HiCache 启用时等待对应层的缓存加载完成。
2. 统一同步契约：使这些方法与已有的 get_index_k_with_scale_buffer 和 get_kv_buffer 等方法保持一致，都遵循层传输同步机制。
3. 无测试或配置配套改动：本次变更仅涉及核心逻辑修复，未添加测试或修改配置文件。

关键文件：

- python/sglang/srt/mem_cache/memory_pool.py（模块 内存缓存；类别 source；类型 core-logic；符号 get_index_k_continuous, get_index_k_scale_continuous, get_index_k_scale_buffer）：这是唯一变更的文件，包含了修复 NSA 索引读取竞争的核心逻辑。

关键符号：get_index_k_continuous, get_index_k_scale_continuous, get_index_k_scale_buffer

关键源码片段

python/sglang/srt/mem_cache/memory_pool.py

这是唯一变更的文件，包含了修复 NSA 索引读取竞争的核心逻辑。

```
  def get_index_k_continuous(
      self,
      layer_id: int,
      seq_len: int,
      page_indices: torch.Tensor,
  ):
      # 添加同步等待：如果启用了 HiCache，确保对应层的缓存加载完成后再读取索引缓冲区
      if self.layer_transfer_counter is not None:
          self.layer_transfer_counter.wait_until(layer_id - self.start_layer)
      buf = self.index_k_with_scale_buffer[layer_id - self.start_layer]
      return index_buf_accessor.GetK.execute(
          self, buf, seq_len=seq_len, page_indices=page_indices
      )

  def get_index_k_scale_continuous(
      self,
      layer_id: int,
      seq_len: int,
      page_indices: torch.Tensor,
  ):
      # 同上，为 scale 连续读取添加同步，避免竞争条件
      if self.layer_transfer_counter is not None:
          self.layer_transfer_counter.wait_until(layer_id - self.start_layer)
      buf = self.index_k_with_scale_buffer[layer_id - self.start_layer]
      return index_buf_accessor.GetS.execute(
          self, buf, seq_len=seq_len, page_indices=page_indices
      )

  def get_index_k_scale_buffer(
      self,
      layer_id: int,
      seq_len_tensor: torch.Tensor,
      page_indices: torch.Tensor,
      seq_len_sum: int,
      max_seq_len: int,
  ):
      """
      融合方法，一次性获取索引 K 和 scale 数据。
      修复前：缺少同步，可能导致读取到未完全加载的缓存数据。
      修复后：添加同步等待，确保数据一致性。
      """
      if self.layer_transfer_counter is not None:
          self.layer_transfer_counter.wait_until(layer_id - self.start_layer)
      buf = self.index_k_with_scale_buffer[layer_id - self.start_layer]
      return index_buf_accessor.GetKAndS.execute(
          self,
```

```
        buf,
        page_indices=page_indices,
        seq_len_tensor=seq_len_tensor,
        seq_len_sum=seq_len_sum,
        max_seq_len=max_seq_len,
    )
```

评论区精华

Review 中仅有一条评论，由 hzh0425 在 get_index_k_continuous 方法处留下，内容为 “HiCache part lgtm”，表示对 HiCache 相关部分的认可。没有出现争议或深度讨论，表明修复方案直接且被快速接受。

- HiCache 同步修复认可 (correctness): 修复被接受，无进一步讨论。

风险与影响

- 风险：低风险：变更仅在现有同步机制上扩展，未引入新逻辑或复杂依赖。
- 回归风险：极小，因为只添加了条件等待，不影响正常流程下的性能或行为。
- 性能风险：在 HiCache 启用时可能轻微增加延迟，因为需要等待缓存加载，但这是修复数据竞争的必要代价，且原本就存在于其他类似方法中。
- 兼容性风险：无，不影响 API 或外部接口。
- 安全风险：无。
- 影响：对用户：修复了 GLM-5 系列模型在 HiCache 下的输出乱码问题，提升模型服务稳定性和输出质量。 对系统：确保 NSA 模型与 HiCache 的协同工作正确性，避免因数据竞争导致的非确定性错误。 对团队：提供了针对稀疏模型缓存同步问题的标准处理模式，可作为类似问题的参考。
- 风险标记：核心路径变更，并发竞争修复

关联脉络

- PR #22924 [UnifiedRadixTree]: Add HiCache hook interface for TreeComponent: 同属 HiCache 相关改进，涉及缓存组件接口扩展，可参考其设计模式。
- PR #22990 [Bug Fix] Ensure prefill_info_table is populated before honoring disagg_prefill_dp_rank: 同属缓存或调度相关的竞态条件修复，展示了类似的数据初始化同步问题。