PR #15852 完整报告

sgl-project/sglang

[Bugfix] fix npu get kv_item_lens in PD separation when use ASCEND_US···

合并时间：2026-03-23 15:56

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/15852

执行摘要

该 PR 修复了 NPU 硬件后端在启用 ASCEND_USE_FIA 标志时，内存池中 kv_item_lens 的计算错误。通过调整 get_contiguous_buf_infos 方法中的条件分支，确保在 PD 分离传输中数据大小评估正确。影响仅限于 NPU 用户，变更简单且 CI 通过，风险较低。

功能与动机

原始动机源于 ASCEND_USE_FIA 启用后 buffer 形状变化：k_buffer 形状从 (layer_num, size // page_size + 1, page_size, head_num, head_dim) 变为 (layer_num, (size // page_size + 1) * page_size, 1, head_num, head_dim)，导致 self.get_key_buffer(i)[0].nbytes 的元素大小不同。因此，计算 kv_item_lens 时需要乘以 page_size 以适应 FIA 模式，否则 PD 传输可能出错。PR body 明确指出此问题，目标为“Fix kv_item_lens for FIA, in order to compute data size in PD transfer cases”。

实现拆解

改动集中在文件 python/sglang/srt/hardware_backend/npu/memory_pool_npu.py 的 get_contiguous_buf_infos 方法。关键逻辑如下：

```
if self.use_fia:
    kv_item_lens = [
        self.get_key_buffer(i)[0].nbytes * self.page_size
        for i in range(self.start_layer, self.start_layer + self.layer_num)
    ] + [
        self.get_value_buffer(i)[0].nbytes * self.page_size
        for i in range(self.start_layer, self.start_layer + self.layer_num)
    ]
else:
    kv_item_lens = [
        self.get_key_buffer(i)[0].nbytes
        for i in range(self.start_layer, self.start_layer + self.layer_num)
    ] + [
        self.get_value_buffer(i)[0].nbytes
        for i in range(self.start_layer, self.start_layer + self.layer_num)
    ]
```

此修改引入了条件分支，根据 self.use_fia 标志调整计算，无需改动其他模块或函数。

评论区精华

Review 中无深入讨论，但维护者 iforgetmyname 在 Issue 评论中强调：

> “this pr only affects npu and npu ci have all passed” 这表明变更影响范围明确，且测试验证充分，无设计争议或额外疑虑。

风险与影响

风险：

- 条件判断 self.use_fia 若设置错误，可能导致 kv_item_lens 计算错误，影响 PD 数据传输和模型推理准确性。
- 变更涉及内存管理核心路径，错误可能引发性能问题或难以调试的 bug。 影响：
- 仅影响使用 NPU 并启用 ASCEND_USE_FIA 的用户，修复后确保数据大小计算正确，避免潜在推理错误。
- 对系统其他部分（如 CPU、GPU 后端）无影响，影响程度中等但对受影响的用户关键。

关联脉络

从历史 PR 看，PR 17695（“[NPU] enhance accuracy for model minimaxm2”）同样针对 NPU 准确性修复，显示 NPU 模块在持续优化 bugfix。本 PR 是这一趋势的延续，专注于内存计算细节，而非新功能引入。结合近期 PR 如 20697（修复 VRAM 泄漏）和 20978（性能优化），可见仓库在平衡功能开发与稳定性维护。