PR #21448 完整报告

sgl-project/sglang

[Fix] Fix Qwen3.5 MoE model loading and Mamba cache sharding in PP mode

合并时间：2026-03-30 11:57

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/21448

执行摘要

- 一句话：修复 Qwen3.5 MoE 模型在 PP 模式下的 Mamba 缓存分片和权重加载问题
- 推荐动作：此 PR 值得精读，特别是对于涉及 PP 模式或缓存管理的开发者。关注点包括：1) PP 感知的缓存分片设计在 model_runner_kv_cache_mixin.py 中的实现； 2) 权重加载的层跳过逻辑如何优雅处理 MoE 专家权重； 3) Review 讨论中关于 start_layer 的权衡，揭示了 HiCache 兼容性的考虑。

功能与动机

PR body 指出，尽管 PR #19670 和 #21070 部分解决了 Qwen3.5 在 PP 模式下的问题，但仍有缺陷：

1) Mamba 缓存未按 PP 分片，导致 GPU 内存浪费； 2) 加载 MoE 模型权重时，仍尝试加载当前 PP rank 范围外的专家权重，引发 KeyError（错误日志见 PR body）。这影响了使用 Qwen3.5 MoE 模型时的内存效率和可用性。

实现拆解

实现分为两部分：

1. Mamba 缓存 PP 分片：在 model_runner_kv_cache_mixin.py 的 _init_pools 方法中，根据 PP 的 start_layer 和 end_layer 过滤 mamba_layer_ids 和 full_attention_layer_ids，确保 HybridReqToTokenPool 和 HybridLinearKVPool 等缓存池仅分配当前 PP 阶段所需的层。相关改动涉及 mem_cache/memory_pool.py 和 disaggregation/decode.py 的构造函数，添加 mamba_layer_ids 和 start_layer 参数。
2. MOE 权重加载修复：在 models/qwen3_5.py 的 load_weights 和 load_fused_expert_weights 方法中，引入层 ID 检查逻辑，使用 get_layer_id 函数和模型的 start_layer、end_layer 属性跳过不属于当前 PP 阶段的权重加载，防止 KeyError。这部分参考了 PR #19254 的代码。

关键文件：

- python/sglang/srt/mem_cache/memory_pool.py（模块 mem_cache）：核心缓存管理模块，修改了 MambaPool 和 HybridMambaDecodeReqToTokenPool 的构造函数，添加 mamba_layer_ids 和 start_layer 参数以支持 PP 分片，直接决定缓存内存分配。

- python/sglang/srt/model_executor/model_runner_kv_cache_mixin.py（模块 model_executor）：初始化缓存池的关键文件，在 _init_pools 方法中添加 PP 过滤逻辑，确保 mamba_layer_ids 和 full_attention_layer_ids 仅包含当前 PP 阶段的层，影响整体缓存策略。
- python/sglang/srt/models/qwen3_5.py（模块 models）：处理 Qwen3.5 模型权重加载，在 load_weights 和 load_fused_expert_weights 方法中引入层跳过逻辑，修复 MoE 模型在 PP 模式下的加载失败问题。
- python/sglang/srt/disaggregation/decode.py（模块 disaggregation）：支持 decode PP 的缓存池，修改 HybridMambaDecodeReqToTokenPool 以传递 start_layer 和 mamba_layer_ids，确保 Mamba 缓存正确分片。
- test/registered/unit/mem_cache/test_mamba_unittest.py（模块 test）：单元测试文件，更新以反映缓存池构造函数的变化，验证 PP 分片逻辑的正确性，保障代码质量。

关键符号：__init__ in HybridMambaDecodeReqToTokenPool (decode.py), __init__ in MambaPool (memory_pool.py), _init_pools in ModelRunner (model_runner_kv_cache_mixin.py), load_weights in Qwen3VLForConditionalGeneration (qwen3_5.py), load_fused_expert_weights in Qwen3VLForConditionalGeneration (qwen3_5.py)

评论区精华

Review 评论中，核心讨论围绕 memory_pool.py 中 start_layer 的处理。hzh0425 建议改进 start_layer（原为硬编码 0），指出其为 HiCache 的层加载所需，sufeng-buaa 初始解释其非功能必需，后同意修复并更新代码。结论是添加了 PP 感知的 start_layer 设置，以支持 HiCache 兼容性，确保缓存池正确初始化。

- start_layer 在缓存池中的处理 (design): 更新代码以支持 PP 感知的 start_layer 设置，确保缓存池正确初始化和 HiCache 兼容性。

风险与影响

- 风险：技术风险包括： 1) 缓存分片逻辑风险：在 model_runner_kv_cache_mixin.py 和 memory_pool.py 中，层过滤逻辑如果基于错误的 PP 配置（如 start_layer 或 end_layer 计算错误），可能导致内存分配不足或溢出，影响模型推理稳定性。 2) 权重加载跳过风险：qwen3_5.py 中的层跳过逻辑依赖 get_layer_id 函数准确提取层 ID，若解析失败或 PP 边界设置不当，可能遗漏必要权重或导致模型输出错误。 3) 回归风险：改动涉及核心缓存和模型加载路径，需确保在非 PP 模式下功能不受影响，但测试覆盖（如 test_mamba_unittest.py 更新）部分缓解此风险。
  - 影响：影响范围：主要影响使用 Qwen3.5 MoE 模型并启用 Pipeline Parallelism 的用户，解决了 GPU 内存浪费和模型加载失败问题，提升资源利用率和系统可靠性。影响程度：中度到高度，因为修复了关键缺陷，可能避免生产环境中的内存不足错误和启动失败。对团队影响：完善了 PP 支持，减少了维护负担，后续相关开发需注意 PP 配置与缓存管理的协同。
  - 风险标记：缓存分片逻辑风险，权重加载跳过风险

关联脉络

- PR #19670 未提供，但 PR body 提及：PR body 提到此 PR 部分解决了 Qwen3.5 PP 问题，与本 PR 相关，同属 PP 支持改进线。
- PR #21070 未提供，但 PR body 提及：类似 PR #19670，涉及 Qwen3.5 PP 缺陷修复，是本 PR 的前序工作。
- PR #19254 未提供，但 PR body 提及：PR body 指出 MOE 权重加载修复部分复制自此 PR，显示跨 PR 的代码复用和问题关联。