PR #29615 完整报告

sgl-project/sglang

Make mem_fraction_static reserve disaggregation-mode aware

合并时间：2026-07-05 09:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29615

执行摘要

- 一句话：让 mem_fraction_static 按分解模式智能预留
- 推荐动作：建议精读。该 PR 展示了一个小而精准的优化，在分解架构下通过条件化预留换取有效显存，设计清晰、改动集中。值得关注的是 DeepEP 预留的硬编码值，后续可能需要根据实际 Profiling 调整。由于无测试，建议在后续 PR 中添加分解模式的 mem_fraction 自动测试。

功能与动机

在 PD 分解部署中，每个节点只运行一个阶段，原先的预留估计会为另一阶段保留死空间（dead headroom），不必要地缩小 KV 缓存。作者在 PR body 中明确说明了 prefill 节点仍保留解码 CUDA 图 + DP attention 填充内存，decode 节点仍按 prefill token 尺寸计算激活松弛并保留预填充分段 CUDA 图内存，导致内存浪费。

实现拆解

1. 激活松弛条件化：在 _handle_gpu_memory_settings 中，当 disaggregation_mode == "decode" 时，激活松弛按 decode batch 大小（max_running_requests || decode_cuda_graph_config.max_bs || 1）乘以 speculative_num_draft_tokens || 1 计算，下限 2048，再乘以 1.5。其余情况下保持原有逻辑（chunked_prefill_size 或 max_prefill_tokens）。
2. 解码 CUDA 图预留条件化：将原有的 reserved_mem += decode_cuda_graph_config.max_bs * 2 改为仅在 disaggregation_mode != "prefill" 且 decode CUDA 图后端启用时执行。
3. DP attention 填充条件化：将 enable_dp_attention 的额外预留增加 disaggregation_mode != "prefill" 条件。
4. 预填充分段 CUDA 图预留条件化：将原有的 prefill_cuda_graph_config.backend != Backend.DISABLED 增加 disaggregation_mode != "decode" 条件。
5. DeepEP all-to-all 缓冲区额外预留：新增条件，当 disaggregation_mode != "prefill" 且 decode_cuda_graph_config.backend != Backend.DISABLED 且 moe_a2a_backend == "deepep" 时，额外预留 2 GiB。
6. 所有变更均位于单个文件 python/sglang/srt/server_args.py，无测试或配置配套改动。

关键文件：

- python/sglang/srt/server_args.py（模块 配置启动器；类别 source；类型 core-logic）：唯一的变更文件，包含所有逻辑：在 _handle_gpu_memory_settings 中根据 disaggregation_mode 条件化激活松弛、解码 CUDA 图、DP attention 填充、预填充分段 CUDA 图和 DeepEP all-to-all 预留。

关键符号：_handle_gpu_memory_settings

关键源码片段

python/sglang/srt/server_args.py

唯一的变更文件，包含所有逻辑：在 _handle_gpu_memory_settings 中根据 disaggregation_mode 条件化激活松弛、解码 CUDA 图、DP attention 填充、预填充分段 CUDA 图和 DeepEP all-to-all 预留。

```
# python/sglang/srt/server_args.py
# 在 _handle_gpu_memory_settings 中，当 mem_fraction_static 为 None 时自动推导。
# 根据 disaggregation_mode 调整保留内存，避免跨阶段死空间。

if self.mem_fraction_static is None:
    # Constant meta data (e.g., from attention backend)
    reserved_mem = 512 # MB

    # For activation slack: 根据分解模式调整
    if self.disaggregation_mode == "decode":
        # Decode 节点不做 prefill，激活松弛按 decode batch 大小计算
        running_requests = (
            self.max_running_requests or decode_cuda_graph_config.max_bs or 1
        )
        draft_tokens = self.speculative_num_draft_tokens or 1
        reserved_mem += max(running_requests * draft_tokens, 2048) * 1.5
    elif self.chunked_prefill_size > 0:
        reserved_mem += max(self.chunked_prefill_size, 2048) * 1.5
    else:
        reserved_mem += max(self.max_prefill_tokens, 2048) * 1.5

    # For decode cuda graphs (skip on prefill-only nodes)
    if (
        self.disaggregation_mode != "prefill"
        and decode_cuda_graph_config.backend != Backend.DISABLED
    ):
        reserved_mem += decode_cuda_graph_config.max_bs * 2

    # ... 其他调整和 DP attention 条件化 ...

    # For prefill piecewise cuda graphs (skip on decode-only nodes)
    if (
        self.disaggregation_mode != "decode"
        and prefill_cuda_graph_config.backend != Backend.DISABLED
    ):
```

```
        if not self.use_mla_backend():
            reserved_mem += len(prefill_cuda_graph_config.bs) * 8
        else:
            reserved_mem += 1.5 * 1024

    # DeepEP all-to-all 缓冲区解码图捕获会真实额外分配，在 floor 之上预留
    if (
        self.disaggregation_mode != "prefill"
        and decode_cuda_graph_config.backend != Backend.DISABLED
        and self.moe_a2a_backend == "deepep"
    ):
        reserved_mem += 2 * 1024 # 2 GiB
```

评论区精华

无 review 评论。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险：变更集中在 _handle_gpu_memory_settings 的 mem_fraction_static 自动推导路径，非分解模式行为不变（通过条件 self.disaggregation_mode 字符串值判断，默认 "null" 不会命中新分支），但若 disaggregation_mode 值意外被设置（如旧代码中可能为空字符串），可能导致行为不一致。
  2. 数值精度风险：新的激活松弛计算采用 max(running_requests * draft_tokens, 2048)，比原有预填充尺寸可能更小，若 running_requests 或 draft_tokens 取值异常，可能导致预留不足。
  3. DeepEP 预留：新增的 DeepEP all-to-all 预留（2 GiB）是硬编码，未考虑模型大小或实际需求，可能在非 EP 场景下浪费空间。
  4. 缺少测试覆盖：无直接测试验证各分解模式下的 mem_fraction_static 值，CI 中的浮点精度失败（已由作者 rerun 解决）表明测试环境存在稳定性问题。 - 影响：影响范围：仅影响启用 PD 分解部署的用户，且仅在 --mem-fraction-static 未设置时自动推导生效。对非分解部署无影响。影响程度：中等偏低。能显著提升分解场景下的 KV 缓存可用内存（PR body 中的数值示例显示 decode 节点从 0.83 提升到 0.875），但实际收益取决于模型和配置。用户：需要明确设置 --disaggregation-mode 才能受益；默认值不变。团队：无后续维护负担。 - 风险标记：缺少测试覆盖，数值精度风险

关联脉络

- 暂无明显关联 PR