PR #25645 完整报告

sgl-project/sglang

[Diffusion] Support parallelism for GLM-Image

合并时间：2026-05-19 22:27

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25645

执行摘要

- 一句话：支持 GLM-Image 多设备并行生成
- 推荐动作：值得精读。尤其关注新增的 MAIN_RANK_ONLY_AND_SEND_TO_OTHERS 并行模式设计，它解决了自回归生成阶段在多卡环境中必须保持 token 一致性的问题。这种“单卡执行后广播”的范式对于混合不同并行策略的流水线很有参考价值。同时，AR 阶段与扩散阶段的拆分也体现了模块化思想。

功能与动机

目前 GLM-Image 只能在一个设备上生成图像，sp>1 出现形状错误，tp>1 存在准确性问题。本 PR 旨在通过合理拆分阶段并引入新的并行策略，使 GLM-Image 支持多卡加速并保持精度。

实现拆解

1. 拆分 AR 阶段：在 python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/glm_image.py 中，将原本混在一起的 GlmImageBeforeDenoisingStage 拆分为两个独立阶段：GlmImageAR（仅负责自回归 token 生成）和简化的 GlmImageBeforeDenoisingStage（仅负责扩散前处理）。GlmImageAR 通过 parallelism_type 声明为 MAIN_RANK_ONLY_AND_SEND_TO_OTHERS。
2. 新增并行模式枚举：在 python/sglang/multimodal_gen/runtime/pipelines_core/stages/base.py 的 StageParallelismType 中添加 MAIN_RANK_ONLY_AND_SEND_TO_OTHERS。
3. 实现新并行分支：在 python/sglang/multimodal_gen/runtime/pipelines_core/executors/parallel_executor.py 的 _execute_stages 中增加对应分支：仅 rank 0 执行阶段，然后通过 broadcast_pyobj 将结果广播给所有其他 rank，最后 barrier 同步。
4. 调整管道注册顺序：在 python/sglang/multimodal_gen/runtime/pipelines/glm_image.py 的 create_pipeline_stages 中，先添加 GlmImageAR，再添加 GlmImageBeforeDenoisingStage 和 DenoisingStage，移除 GlmImageBeforeDenoisingStage 对 processor 和 vision_language_encoder 的依赖。
5. 修复旋转嵌入 SP 分片：在 python/sglang/multimodal_gen/configs/pipeline_configs/glm_image.py 的 get_freqs_cis 中，将 rotary_emb 返回的 cos/sin 通过 shard_rotary_emb_for_sp 分片，并返回元组而非单个张量。

6. 升级依赖：在 python/pyproject_npu.toml 中将 cache-dit 从 1.2.1 升级到 1.3.5，以兼容 SP 相关修复。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/glm_image.py（模块 扩散模型；类别 source；类型 core-logic；符号 GlmImageAR, GlmImageBeforeDenoisingStage, parallelism_type）：核心变更文件，将 AR 阶段拆分为独立 PipelineStage，定义新并行模式。
- python/sglang/multimodal_gen/runtime/pipelines_core/executors/parallel_executor.py（模块 执行器；类别 source；类型 core-logic；符号 _execute_stages）：实现新并行模式的分支逻辑，确保 AR 阶段结果正确分发。
- python/sglang/multimodal_gen/runtime/pipelines/glm_image.py（模块 管道注册；类别 source；类型 core-logic；符号 create_pipeline_stages）：调整管道阶段注册顺序，匹配拆分后的阶段。
- python/sglang/multimodal_gen/configs/pipeline_configs/glm_image.py（模块 配置计算；类别 source；类型 core-logic；符号 get_freqs_cis, shard_rotary_emb_for_sp）：修复旋转嵌入的 SP 分片，确保多卡推理位置编码正确。
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/base.py（模块 框架基础；类别 source；类型 core-logic；符号 StageParallelismType）：新增并行模式枚举值，定义新模式的语义。
- python/pyproject_npu.toml（模块 依赖管理；类别 config；类型 configuration）：升级 cache-dit 依赖版本，兼容 SP 相关修复。

关键符号：GlmImageAR.init, GlmImageAR.parallelism_type, GlmImageAR._compute_generation_params, GlmImageBeforeDenoisingStage.init, GlmImagePipelineConfig.get_freqs_cis, ParallelExecutor._execute_stages, shard_rotary_emb_for_sp

关键源码片段

python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/glm_image.py

核心变更文件，将 AR 阶段拆分为独立 PipelineStage，定义新并行模式。

```
class GlmImageAR(PipelineStage):
    """自回归生成阶段：仅在主 rank 执行，结果广播给其他 rank。"""

    def __init__(self, processor, vision_language_encoder):
        super().__init__()
        self.processor = processor
        self.vision_language_encoder = vision_language_encoder

    @property
    def parallelism_type(self) -> StageParallelismType:
        # 声明并行策略：主 rank 执行后广播
```

```
        return StageParallelismType.MAIN_RANK_ONLY_AND_SEND_TO_OTHERS

    @staticmethod
    def _compute_generation_params(image_grid_thw, is_text_to_image):
        # 根据图像网格 shape 计算生成 tokens 数、偏移和目标分辨率
        grid_sizes = []
        grid_hw = []
        for i in range(image_grid_thw.shape[0]):
            t, h, w = image_grid_thw[i].tolist()
            grid_sizes.append(int(h * w))
            grid_hw.append((int(h), int(w)))
        if not is_text_to_image:
            max_new_tokens = grid_sizes[-1] + 1
            large_image_start_offset = 0
            target_grid_h, target_grid_w = grid_hw[-1]
        else:
            total_tokens = sum(grid_sizes)
            max_new_tokens = total_tokens + 1
            large_image_start_offset = sum(grid_sizes[1:])
            target_grid_h, target_grid_w = grid_hw[0]
        return max_new_tokens, large_image_start_offset, target_grid_h, target_grid_w
```

python/sglang/multimodal_gen/runtime/pipelines_core/executors/parallel_executor.py

实现新并行模式的分支逻辑，确保 AR 阶段结果正确分发。

```
    elif paradigm == StageParallelismType.MAIN_RANK_ONLY_AND_SEND_TO_OTHERS:
        # 仅主 rank（rank 0）执行阶段，其他 rank 等待
        if rank == 0:
            self.before_stage(stage, stage_index, batch, server_args)
            batch = stage(batch, server_args)
            self.after_stage(stage_index)
        torch.distributed.barrier()

        # 将 batch 从主 rank 广播到所有 rank
        obj_list = [batch] if rank == 0 else []
        broadcasted_list = broadcast_pyobj(
            obj_list, rank=rank, dist_group=group.cpu_group, src=0
        )
        if rank != 0:
            batch = broadcasted_list[0]
        torch.distributed.barrier()
```

评论区精华

本 PR 无公开 review 讨论，审核者 ping1jing2 直接批准。从提交历史看，经过两次合并主分支后最终提交稳定。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 新并行模式验证不足：MAIN_RANK_ONLY_AND_SEND_TO_OTHERS 仅在 GLM-Image 上测试，其他扩散模型未使用，但模式本身是通用的，若被复用需确保语义正确。
  - AR 阶段单点瓶颈：AR 生成仅在 rank 0 执行，如果 AR 生成时间远大于扩散阶段，可能成为瓶颈；但当前数据下 AR 阶段耗时较少。
  - 旋转嵌入分片影响：shard_rotary_emb_for_sp 仅在 GLM-Image 配置中调用，不影响其他模型，但若未来其他模型复用需检查兼容性。
  - 依赖升级风险：cache-dit 从 1.2.1 跳至 1.3.5，可能引入新行为，需关注 CI 测试结果。
- 影响：
  - 用户：GLM-Image 用户现在可以指定 --num-gpus N --sp-degree N 实现多卡加速，实测 SP=2 时速度提升约 2 倍（从 31.66s 降至 16.86s）。
  - 系统：扩展了扩散模型的并行策略选项，为其他模型提供可参考的“单卡执行后广播”范式。
  - 团队：需维护新增的并行枚举值和对应逻辑，但设计上与现有模式保持一致，维护成本可控。
  - 风险标记：新并行模式仅在 GLM-Image 验证，AR 阶段单点瓶颈，旋转嵌入分片影响范围有限，cache-dit 版本升级

关联脉络

- 暂无明显关联 PR