PR #28624 完整报告

sgl-project/sglang

[diffusion] optimize LTX2.3 CFG/SP paths

合并时间：2026-06-27 19:23

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28624

执行摘要

- 一句话：优化 LTX2.3 CFG/SP 路径与自动并行策略
- 推荐动作：值得精读，尤其是广播优化和自动并行策略的设计决策。注意 review 中未解决的 list batch 问题，若使用 grouped 执行需关注。

功能与动机

PR body 指出：LTX2.3 在 8xB200 上 sp8 相比 cfg2+sp4 稳态步骤加速 1.23x；packed QKV input A2A 微基准有 1.06-1.27x 加速但端到端无可靠收益，故默认禁用。目的是提升 LTX2.3 推理性能，并让自动并行策略更合理。

实现拆解

1. 引入按 GPU 数量的 CFG 并行度配置：在 model_deployment_config.py 新增 auto_cfg_parallel_degree_by_num_gpus 元组字段和 get_auto_cfg_parallel_degree 方法，允许模型按实际 GPU 数指定 CFG 并行度（如 4 和 8 GPU 设度 1 表示禁用 CFG 并行）。
2. 修改自动并行策略：在 server_args.py 的 _adjust_parallelism 中，当 CFG 并行未指定时调用 get_auto_cfg_parallel_degree，根据推荐度决定是否启用 CFG 并行。这使 LTX2.3 在 4/8 GPU 自动选择纯 SP。
3. 添加 CFG 并行局部字段钩子：在 base.py 的 PipelineStage 基类增加 cfg_parallel_local_batch_fields 方法（默认返回空）；LTX2.3 的 DenoisingStage 覆盖该方法返回 ("latents", "audio_latents")，标记这些已在本地 GPU 的张量无需广播。
4. 改造 CFG 并行广播流程：在 parallel_executor.py 的 _execute_stages 中，CFG_PARALLEL 分支先调用钩子将指定字段置 None 再广播，非 rank0 从本地恢复。避免 Python 对象广播。
5. 增加可选 packed QKV input A2A：在 layer.py 的 USPAttention 中新增 enable_packed_qkv_input_a2a 参数，控制是否使用异步 A2A 合并 QKV 输入，默认关闭。
6. 更新配置与测试：LTX2.3 部署配置设置 auto_cfg_parallel_degree_by_num_gpus；添加单元测试验证自动并行选择。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/executors/parallel_executor.py（模块 执行器；类别 source；类型 core-logic）：核心执行器，修改 CFG 并行广播逻辑以

支持局部字段保留，减少通信开销

- python/sglang/multimodal_gen/runtime/server_args.py（模块 服务器参数；类别 source；类型 core-logic）：自动并行策略调整入口，整合按 GPU 数量的 CFG 并行度配置
- python/sglang/multimodal_gen/configs/pipeline_configs/model_deployment_config.py（模块 部署配置；类别 source；类型 data-contract；符号 get_auto_cfg_parallel_degree）：新增按 GPU 数量的 CFG 并行度配置接口，是扩展性的关键
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/ltx_2/denoising.py（模块 LTX2.3 去噪；类别 source；类型 data-contract；符号 cfg_parallel_local_batch_fields）：LTX2.3 特定阶段，覆盖局部字段钩子，返回 latents 和 audio_latents
- python/sglang/multimodal_gen/runtime/layers/attention/layer.py（模块 注意力层；类别 source；类型 core-logic）：注意力层核心，新增 packed QKV input A2A 可选路径
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/base.py（模块 流水线阶段；类别 source；类型 core-logic；符号 cfg_parallel_local_batch_fields）：PipelineStage 基类，新增 cfg_parallel_local_batch_fields 钩子定义
- python/sglang/multimodal_gen/test/unit/test_server_args.py（模块 服务器参数测试；类别 test；类型 test-coverage；符号 test_auto_ltx23_large_gpu_counts_prefer_sp_over_cfg_parallel）：单元测试验证自动并行策略预期行为
- python/sglang/multimodal_gen/runtime/models/dits/ltx_2.py（模块 LTX2.3 模型；类别 source；类型 data-contract）：模型定义，配合部署配置更新
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/ltx_2/decoding_av.py（模块 LTX2.3 解码；类别 source；类型 data-contract）：解码阶段调整以配合局部字段保留
- python/sglang/multimodal_gen/configs/models/dits/ltx_2.py（模块 模型配置；类别 source；类型 data-contract）：模型配置，设置 auto_cfg_parallel_degree_by_num_gpus
- python/sglang/multimodal_gen/configs/pipeline_configs/ltx_2.py（模块 管道配置；类别 source；类型 core-logic）：管道配置，配合模型设置

关键符号：cfg_parallel_local_batch_fields, get_auto_cfg_parallel_degree, _adjust_parallelism, _execute_stages

关键源码片段

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

核心执行器，修改 CFG 并行广播逻辑以支持局部字段保留，减少通信开销

```
# 在 _execute_stages 的 CFG_PARALLEL 分支中，新增局部字段保留逻辑
elif paradigm == StageParallelismType.CFG_PARALLEL:
    local_batch = batch
    # 查询该阶段声明的无需广播的局部字段
    local_batch_fields = stage.cfg_parallel_local_batch_fields(batch, server_args)
```

```
    # rank 0 将局部字段置为 None，准备广播其余字段
    if rank == 0 and local_batch_fields:
        local_field_values = {name: getattr(batch, name) for name in local_batch_fields}
        for name in local_batch_fields:
            setattr(batch, name, None)
    else:
        local_field_values = {}

    obj_list = [batch] if rank == 0 else []
    try:
        # 广播 batch 对象（不含局部字段）
        broadcasted_list = broadcast_pyobj(
            obj_list,
            rank=get_world_rank(),
            dist_group=cfg_group.cpu_group,
            src=cfg_group.ranks[0],
        )
    finally:
        # 广播后恢复 rank 0 的局部字段
        if rank == 0:
            for name, value in local_field_values.items():
                setattr(batch, name, value)
    if rank != 0:
        batch = broadcasted_list[0]
        # 非 rank 0 从本地恢复局部字段（这些字段在广播前已存在于 local_batch 中）
        for name in local_batch_fields:
            setattr(batch, name, getattr(local_batch, name))
    batch = self._run_stage_with_executor_hooks(stage, stage_index, batch, server_args, run_
    stage, use_nvtx)
    torch.distributed.barrier()
```

python/sglang/multimodal_gen/runtime/server_args.py

自动并行策略调整入口，整合按 GPU 数量的 CFG 并行度配置

```
  # 在 _adjust_parallelism 中修改 auto-enable CFG parallel 部分
  if cfg_unspecified:
      deployment_config = self.pipeline_config.get_model_deployment_config()
      # 从模型配置中获取推荐的 cfg_parallel_degree
      auto_cfg_parallel_degree = deployment_config.get_auto_cfg_parallel_degree(self.num_gpus)
      if auto_cfg_parallel_degree < 1:
          self.enable_cfg_parallel = False
      else:
          cfg_group_size = self.dp_size * self.tp_size * auto_cfg_parallel_degree
          if (
              self.performance_mode != "manual"
              and deployment_config.auto_enable_cfg_parallel
              and self.num_gpus >= 2
              and self.num_gpus % cfg_group_size == 0
              and sp_unspecified
```

```
            and ulysses_unspecified
            and ring_unspecified
            and self._model_default_uses_cfg()
        ):
            self.cfg_parallel_degree = auto_cfg_parallel_degree
            self.enable_cfg_parallel = auto_cfg_parallel_degree > 1
            if self.enable_cfg_parallel:
                logger.info("自动启用 CFG parallel，degree %d，用于 %d GPUs",
                            self.cfg_parallel_degree, self.num_gpus)
            else:
                logger.info("自动禁用 CFG parallel for %d GPUs", self.num_gpus)
        else:
            self.enable_cfg_parallel = False
```

python/sglang/multimodal_gen/configs/pipeline_configs/model_deployment_config.py

新增按 GPU 数量的 CFG 并行度配置接口，是扩展性的关键

```
@dataclass(frozen=True)
class ModelDeploymentConfig:
    # ... 其他字段省略
    # 新增：按 GPU 数量指定 cfg_parallel_degree 的优先级列表
    auto_cfg_parallel_degree_by_num_gpus: tuple[tuple[int, int], ...] = ()

    def get_auto_cfg_parallel_degree(self, num_gpus: int) -> int:
        """
        根据 GPU 数量查找推荐的 cfg_parallel_degree，
        如果未配置则返回 2（兼容旧行为）。
        """
        for candidate_num_gpus, cfg_degree in self.auto_cfg_parallel_degree_by_num_gpus:
            if candidate_num_gpus == num_gpus:
                return cfg_degree
        return 2
```

评论区精华

gemini-code-assist[bot] 指出，在 grouped 执行时 batch 可能是 list，直接 getattr/setattr 可能引发 AttributeError。该评论未得到解决即合并，存在潜在的兼容性问题，建议使用者注意。

- Potential AttributeError for list batch in CFG parallel broadcast (correctness): 未修复，PR 已合并。可能当前 grouped 执行不使用 CFG 并行路径，但仍是潜在隐患。

风险与影响

- 风险：新引入的局部字段钩子如果实现错误可能导致数据不一致；packed QKV A2A 默认关闭但他人可能误开启导致性能下降；自动并行策略变化可能影响其他模型的默认行为，需确保模型配置正确；review 中未解决的 list batch 问题可能在使用 grouped 执行时引发崩溃。

- 影响：对 LTX2.3 用户：8 GPU 场景稳态步骤加速 23%，4 GPU 同样受益。对开发者：新增可扩展的并行策略配置接口和阶段钩子，未来模型可复用。对系统：无破坏性变更，但默认并行策略改变，用户可能需调整显式配置。
- 风险标记：核心路径变更：CFG 广播逻辑重写，默认并行策略变化：LTX2.3 自动模式切换，未处理的 list batch 兼容性风险，packed QKV A2A 默认关闭但可能误导

关联脉络

- 暂无明显关联 PR