# PR #25284 完整报告

sgl-project/sglang

Support Gemma4 Pipeline Parallelism

合并时间：2026-05-19 22:40

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25284

## 执行摘要

- 一句话：支持 Gemma4 流水线并行
- 推荐动作：建议精读此 PR，特别是 pp_filter_load_weight 的设计和 forward 中 PP proxy 的处理，这是 SGLang 中标准 PP 适配模式。同时关注 PLE 兼容性讨论，体现了在边界情况下的设计权衡。对于需要部署 Gemma4 在单机多 GPU 环境的团队，应立即采用此变更。

## 功能与动机

根据 PR 描述，Gemma4 shipped without pipeline-parallel support: launching Gemma4ForCausalLM 或 Gemma4ForConditionalGeneration with --pp-size 2 immediately crashes with AssertionError: Pipeline Parallel is not compatible with this model。此外，26B BF16 检查点在每个 TP rank 上占用约 48 GB 权重，导致在 80 GB H100 上留给 KV cache 的空间有限。PP 分片可将每 rank 权重占用减半，KV 预算近似翻倍。

## 实现拆解

1. 修改 Gemma4TextModel 和 Gemma4ForCausalLM(gemma4_causal.py)：添加 pp_group 属性，使用 make_layers 根据 pp_rank 和 pp_size 切分层栈，非所属 rank 的层用 PPMissingLayer 占位。forward 接受 pp_proxy_tensors 参数，首 rank 正常计算嵌入和 per_layer_inputs，非首 rank 从 proxy 读取 hidden_states；非末 rank 返回 PPProxyTensors 包含 hidden_states 和残差占位。
2. 权重加载过滤：新增 pp_filter_load_weight 函数，根据 layer_id 跳过不在当前分片范围内的层权重；处理 tied embeddings 路由（末 rank 将 embed_tokens.weight 加载到 lm_head）；按 rank 跳过专属模块（首 rank 的 embed_tokens、末 rank 的 norm/lm_head 等）。
3. 多模态入口适配(gemma4_mm.py)：将 vision_tower、embed_vision、audio_tower、embed_audio 的实例化限制在首 rank（非首 rank 用 PPMissingLayer 占位）；forward 仅在首 rank 计算 per_layer_inputs，其余通过 proxy 传递；load_weights 添加相同过滤逻辑。
4. 注意力后端自动选择(server_args.py)：将 Gemma4ForCausalLM 加入自动选择 triton/trtllm_mha 后端的模型架构列表（原仅 Gemma4ForConditionalGeneration），解决 head_dim=512 超过 FlashAttention 上限的问题。

5. 冲突处理：当 pp_size>1 且 num_kv_shared_layers>0 时抛出 ValueError；当 pp_size>1 且使用 PLE（hidden_size_per_layer_input>0）且未禁用 CUDA graph 时抛出错误，提示 --disable-cuda-graph。
6. 测试(test_pp_single_node.py)：新增 TestGemma4PPAccuracy（PP=2 下 GSM8K 和 MMMU）和 TestGemma4PLEPPAccuracy（PP=2 下 GSM8K），覆盖 LM 和多模态路径。配套 test_utils.py 添加模型常量。

关键文件：

- python/sglang/srt/models/gemma4_causal.py（模块 LM 模型；类别 source；类型 data-contract；符号 pp_filter_load_weight, tie_weights, Gemma4TextModel, Gemma4ForCausalLM）：核心变更文件，实现了 Gemma4 LM 的 PP 支持，包括层切分、代理前向传播、权重过滤函数 pp_filter_load_weight 和 tied embeddings 处理。
- python/sglang/srt/models/gemma4_mm.py（模块 多模态模型；类别 source；类型 data-contract；符号 Gemma4ForConditionalGeneration）：多模态入口的 PP 适配，将 vision/audio 组件限制在首 rank，forward 处理代理张量。
- test/registered/distributed/test_pp_single_node.py（模块 PP 测试；类别 test；类型 test-coverage；符号 TestGemma4PPAccuracy, test_gsm8k, test_mmmu, TestGemma4PLEPPAccuracy）：添加了 Gemma4 PP 的端到端测试，覆盖 LM 和多模态，以及 PLE 变体。
- python/sglang/srt/server_args.py（模块 服务器配置；类别 source；类型 core-logic）：将 Gemma4ForCausalLM 加入自动选择注意力后端的模型列表，避免用户手动指定。
- python/sglang/test/test_utils.py（模块 测试工具；类别 test；类型 test-coverage）：添加了 Gemma4 测试模型的常量定义。

关键符号：pp_filter_load_weight, Gemma4TextModel.init, Gemma4TextModel.forward, Gemma4ForCausalLM.forward, Gemma4ForCausalLM.load_weights, Gemma4ForCausalLM.tie_weights, Gemma4ForConditionalGeneration.init, Gemma4ForConditionalGeneration.forward, Gemma4ForConditionalGeneration.load_weights

## 关键源码片段

### python/sglang/srt/models/gemma4_causal.py

核心变更文件，实现了 Gemma4 LM 的 PP 支持，包括层切分、代理前向传播、权重过滤函数 pp_filter_load_weight 和 tied embeddings 处理。

```
def pp_filter_load_weight(
    name,
    loaded_weight,
    *,
    pp_group,
    start_layer,
    end_layer,
    params_dict,
    loaded_params,
```

```
    tie_word_embeddings,
    embed_weight_name,
    first_rank_only_patterns=(),
    last_rank_only_prefixes=(),
    head_param_name='lm_head.weight',
):
    # Shared PP filter for Gemma4 load_weights paths.
    # 当 pp_size == 1 时直接放行，不做任何过滤。
    if pp_group.world_size <= 1:
        return False

    # 通过 get_layer_id 获取权重的层 ID，若不在当前切片范围内则跳过。
    layer_id = get_layer_id(name)
    if layer_id is not None and (layer_id < start_layer or layer_id >= end_layer):
        return True

    # 在最后一个 rank 上，将 tied embed_tokens 权重路由到 lm_head。
    # 在 PP 下 embed_tokens 和 lm_head 位于不同 rank，无法通过模块别名绑定。
    if tie_word_embeddings and pp_group.is_last_rank and name == embed_weight_name:
        head_param = params_dict.get(head_param_name)
        if head_param is not None:
            wl = getattr(head_param, 'weight_loader', default_weight_loader)
            wl(head_param, loaded_weight)
            loaded_params.add(head_param_name)
        return True

    # 跳过不属于首 rank 的模块（例如 embed_tokens、vision_tower 等）。
    if not pp_group.is_first_rank and any(p in name for p in first_rank_only_patterns):
        return True

    # 跳过不属于末 rank 的模块（例如 norm、lm_head）。
    if not pp_group.is_last_rank and any(
        name.startswith(p) for p in last_rank_only_prefixes
    ):
        return True

    return False
```

## python/sglang/srt/models/gemma4_mm.py

多模态入口的 PP 适配，将 vision/audio 组件限制在首 rank，forward 处理代理张量。

```
class Gemma4ForConditionalGeneration(PreTrainedModel):
    def __init__(self, config, quant_config=None, prefix=''):
        super().__init__(config=config)
        self.pp_group = get_pp_group() # 获取当前 rank 的 PP 组
        self.config = config
        self.quant_config = quant_config
        text_config = config.text_config
        prefix = add_prefix('model', prefix)
```

```
# 视觉 / 音频编码器仅在首 rank 实例化（输入嵌入阶段）
if self.pp_group.is_first_rank:
    self.vision_tower = Gemma4VisionEncoder(
        config=config.vision_config,
        quant_config=quant_config,
        prefix=add_prefix('vision_tower', prefix),
    )
    self.embed_vision = Gemma4MultimodalEmbedder(
        config.vision_config, text_config,
        quant_config=quant_config,
        prefix=add_prefix('embed_vision', prefix),
    )
    if getattr(config, 'audio_config', None) is not None:
        self.audio_tower = Gemma4AudioEncoder(
            config=config.audio_config,
            quant_config=quant_config,
            prefix=add_prefix('audio_tower', prefix),
        )
        self.embed_audio = Gemma4MultimodalEmbedder(
            config.audio_config, text_config,
            quant_config=quant_config,
            prefix=add_prefix('embed_audio', prefix),
        )
else:
    # 非首 rank 使用 PPMissingLayer 占位，确保模块存在但不占用权重
    self.vision_tower = PPMissingLayer()
    self.embed_vision = PPMissingLayer()
    self.audio_tower = PPMissingLayer() if getattr(config, 'audio_config', None) is not
    None else None
    self.embed_audio = PPMissingLayer() if getattr(config, 'audio_config', None) is not
    None else None
# 后续 lm_head 和 language_model 初始化类似处理 ...
```

# 评论区精华

## 全局索引争论

- gemini-code-assist 提出在非首 rank 使用全局 layer_idx 索引 self.layers 会导致 IndexError，建议减去 start_layer。
- 作者 Yuan-Luo 指出 make_layers 返回的 ModuleList 包含 PPMissingLayer 占位，因此全局索引正确，且与其他模型（mimo_v2.py、llama4.py）一致，拒绝修改。

## PLE+CUDA graph 兼容性

- kpham-sgl 指出 cuda_graph_runner 的 proxy schema 仅包含 hidden_states/residual，缺少 per_layer_inputs，导致 PLE 模型在 PP+CUDA graph 下产生垃圾输出。
- 作者添加 guard：当 pp_size>1 且 hidden_size_per_layer_input>0 且未 disable_cuda_graph 时抛出异常，并新增 TestGemma4PLEPPAccuracy 测试。

### 权重过滤函数复用

- BBuf 建议将 gemma4_causal.py 和 gemma4_mm.py 中重复的 PP 权重过滤逻辑提取为公共函数。
- 作者采纳，创建 pp_filter_load_weight 函数，在两个文件中复用。
- 全局层索引 vs 本地层索引的正确性 (correctness): 作者 Yuan-Luo 指出 make_layers 返回的 ModuleList 包含 PPMissingLayer 占位，因此全局索引正确且与其他模型一致，拒绝该建议。
- PLE 模型与 CUDA graph 的 proxy schema 不兼容 (correctness): 作者添加了 guard：当 pp_size>1 且 hidden_size_per_layer_input>0 且未 disable_cuda_graph 时抛出异常，并新增 TestGemma4PLEPPAccuracy 测试。
- 权重过滤逻辑的公共函数提取 (design): 作者采纳，将 pp_filter_load_weight 作为独立函数定义在 gemma4_causal.py 中，并在 MM 文件中导入复用。

## 风险与影响

- 风险：
  1. PLE+CUDA graph 冲突：虽然添加了 guard，但用户需手动禁用 CUDA graph 才能为 PLE 模型启用 PP，可能损失部分性能。
  2. 权重加载过滤正确性：pp_filter_load_weight 依赖 get_layer_id 和命名模式，若模型配置有特殊结构可能导致权重误加载。
  3. 多模态组件分布：vision_tower 等仅在首 rank，若未来有不同分配策略需重新适配。
  4. 注意力后端自动选择：将 Gemma4ForCausalLM 纳入自动选择，但该模型可能未在 triton/trtllm_mha 后端充分测试。
  5. 测试覆盖局限：仅测试 26B-A4B 和 E4B 变体，其他规模（如 8B、27B）未覆盖。 - 影响：用户影响：首次支持 Gemma4 模型在单机多 GPU 场景下通过 PP 拆分，权重占用降低 48%，KV cache 容量提升 130%，用户无需额外参数即可使用。系统影响：运行时保持一致，PP 代理 schema 扩展支持 per_layer_inputs（非 PLE 模型）。团队影响：提供了可复用的 PP 适配模式（pp_filter_load_weight 函数），后续模型添加 PP 时可参考。 - 风险标记：PLE+CUDA graph guard，权重过滤依赖命名模式，多模态组件 rank 分布，注意力后端自动选择覆盖不足，测试覆盖局限

## 关联脉络

- 暂无明显关联 PR