PR #32708 完整报告

sgl-project/sglang

Split #32584 into 2/2: [LoRA] Shard attention LoRA by attn-TP and allow dynamic LoRA with dp attention

合并时间：2026-08-01 06:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32708

执行摘要

- 一句话：LoRA 按 attn-TP 切分并支持 dp attention 动态加载
- 推荐动作：值得精读。重点看三处设计：_effective_tp_size 如何把 moe_tp / attn_tp / tp 三种切分维度收敛到一个查询入口；RowParallel 归约组跟随 base layer 的 use_dp_attention_reduce 处理；_merge_lora_update_results 的失败优先与去重语义（以及全成功时返回首对象以保证 LRU 原地修改安全）。对要扩展 LoRA 到新并行组合的工程师，这是很好的参照。

功能与动机

PR body 明确说明：Attention layers shard on attn-TP (tp_size / dp_size), not the global rank, so adapter loads hit shape-mismatch asserts；动态 LoRA 端点此前 assert dp_size == 1，直接拒绝 dp attention 下的加载请求。此外 dp attention 会把 load/unload 扇出到每个 DP group 的一个 scheduler，原有 result[0] 读取会让非 0 号 rank 的失败被误报为成功，导致 tokenizer 侧注册表漂移。

实现拆解

1. 新增 attn-TP 模块分类（python/sglang/srt/lora/utils.py）：定义 ATTN_TP_LORA_MODULE_NAMES，收录 qkv_proj、o_proj、in_proj、in_proj_qkvz 等在 DP attention 下构建于 attn-TP 组的投影模块；注释说明 mamba.py 与 qwen3_5.py 在 dp attention 下用 attn_tp 参数构建这些层。
2. 内存池感知 attn_tp_size（python/sglang/srt/lora/mem_pool.py + python/sglang/srt/lora/lora_manager.py）：LoRAMemoryPool 构造函数新增必填 attn_tp_size；新增 _effective_tp_size(module_name) 统一按模块类型返回路由 MoE 的 moe_tp_size、注意力模块的 attn_tp_size 或默认外层 tp_size，并替换 get_lora_A_shape / get_lora_B_shape 中各自维护的内联分支。LoRAManager 在 init 时从 get_parallel() 提取 attn_tp_size（并行组在 init_torch_distributed 后冻结）并传入内存池；load_lora_weight_tensor 不再把 self.tp_rank 传给 slice 方法，切分 rank 由模块自身决定。
3. 权重切片与归约组跟随 base layer（python/sglang/srt/lora/layers.py）：所有 slice_lora_a_weights / slice_lora_b_weights 签名删除 tp_rank 参数，内部改取 self.base_layer.tp_rank（如 ColumnParallel / MergedColumnParallel / MLA / Inkling

系列）；RowParallelLinearWithLoRA.forward 依据 base_layer.use_dp_attention_reduce 选择 attn_tp_group.all_reduce 或全局 tensor_model_parallel_all_reduce，避免跨 DP 组混和。

4. 控制面合并与断言放宽（python/sglang/srt/managers/tokenizer_control_mixin.py）：新增 _merge_lora_update_results，全部成功时返回首 rank 原对象（调用方 LRU 驱逐会原地修改 loaded_adapters），任一失败则失败优先、错误消息按出现顺序去重后以 | 连接；load/unload/load_from_tensors 三个端点统一走合并逻辑，并放宽动态 LoRA 断言为 dp_size == 1 or enable_dp_attention（纯 dp_size > 1 仍禁止）。
5. 测试配套：新增 test/registered/unit/managers/test_lora_update_result_merge.py 覆盖合并语义（全成功返回首对象、部分失败优先、错误去重、无消息失败不崩溃）；扩展 test/registered/unit/lora/test_mem_pool_ep_unit.py 增加 TestAttnModulesShardByAttnTp，覆盖 attn_tp=1 全宽、attn_tp=2 二分片、in_proj_qkvz 按 attn-TP 分类三组形状断言，并同步改造 _FakeDenseLayer 等桩签名。

关键文件：

- python/sglang/srt/lora/layers.py（模块 LoRA 层；类别 source；类型 core-logic；符号 slice_lora_a_weights, slice_lora_b_weights, RowParallelLinearWithLoRA.forward）：核心切片与归约逻辑：所有 slice_lora_a_weights / slice_lora_b_weights 删除 tp_rank 参数，改由 base_layer.tp_rank 决定切分位置；RowParallelLinearWithLoRA.forward 按 use_dp_attention_reduce 选择 attn-TP 或全局 TP 归约组，是修复 shape-mismatch 与跨 DP 混和的关键。
- python/sglang/srt/lora/mem_pool.py（模块 内存池；类别 source；类型 core-logic；符号 _effective_tp_size, get_lora_A_shape, get_lora_B_shape, load_lora_weight_tensor）：LoRAMemoryPool 新增 attn_tp_size 构造参数与 _effective_tp_size 分类方法，统一决定 LoRA 缓冲区沿哪种 TP 维度切分；get_lora_A_shape / get_lora_B_shape 与 load_lora_weight_tensor 全部接入新逻辑。
- python/sglang/srt/managers/tokenizer_control_mixin.py（模块 控制通道；类别 source；类型 core-logic；符号 _merge_lora_update_results, load_lora_adapter, unload_lora_adapter, load_lora_adapter_from_tensors）：新增 _merge_lora_update_results 合并 DP fan-out 的各 rank 回复，修正 result[0] 误读导致的注册表漂移；同时放宽 load/unload 的 dp_size 断言并接入三个端点。
- python/sglang/srt/lora/utils.py（模块 工具库；类别 source；类型 configuration；符号 ATTN_TP_LORA_MODULE_NAMES）：新增 ATTN_TP_LORA_MODULE_NAMES 特例集合，标注哪些 LoRA 模块在 DP attention 下按 attn-TP 构建，是缓冲区分类的配置源头。
- python/sglang/srt/lora/lora_manager.py（模块 LoRA 管理；类别 source；类型 dependency-wiring；符号 init_memory_pool, attn_tp_size）：从 get_parallel() 提取 attn_tp_size 并在 init_memory_pool 时传入 LoRAMemoryPool，完成构造链路的依赖接线。
- test/registered/unit/managers/test_lora_update_result_merge.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestMergeLoRAUpdateResults, test_all_success_returns_first_rank_result, test_any_rank_failure_wins, test_duplicate_error_messages_deduplicated）：新增测试覆盖 _merge_lora_update_results 的全部语义分支：全成功返回首对象、部分失败优先、错误

去重、无消息失败不崩溃。

- test/registered/unit/lora/test_mem_pool_ep_unit.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestAttnModulesShardByAttnTp, test_attn_tp_1_keeps_attention_buffers_full_width, test_attn_tp_gt1_still_shards_attention_buffers, test_linear_attention_in_proj_shards_by_attn_tp）：新增 TestAttnModulesShardByAttnTp 回归测试，验证 attn_tp=1 全宽、attn_tp=2 二分片与 in_proj_qkvz 按 attn-TP 分类，并同步改造测试桩签名。

关键符号：slice_lora_a_weights, slice_lora_b_weights, _effective_tp_size, get_lora_A_shape, get_lora_B_shape, _merge_lora_update_results, load_lora_adapter, unload_lora_adapter, load_lora_adapter_from_tensors, init_memory_pool

关键源码片段

python/sglang/srt/lora/layers.py

核心切片与归约逻辑：所有 slice_lora_a_weights / slice_lora_b_weights 删除 tp_rank 参数，改由 base_layer.tp_rank 决定切分位置；RowParallelLinearWithLoRA.forward 按 use_dp_attention_reduce 选择 attn-TP 或全局 TP 归约组，是修复 shape-mismatch 与跨 DP 混和的关键。

```
# 片段一：RowParallelLinearWithLoRA.forward 中的归约组选择。
# use_dp_attention_reduce 表示 base layer 在 attn-TP 组内行并行归约，
# LoRA 增量必须归约到同一组，避免跨 DP 组混合中间结果。
if self.base_layer.use_dp_attention_reduce:
    all_reduce = get_parallel().attn_tp_group.all_reduce
else:
    all_reduce = tensor_model_parallel_all_reduce

# 片段二：MergedColumnParallel 的 LoRA-B 切片。
# 切分 rank 直接取 base_layer.tp_rank（attn-TP 局部 rank）；外层全局
# TP rank 在 DP attention 下会越出本地权重宽度，导致 shape-mismatch。
def slice_lora_b_weights(self, B: torch.Tensor):
    local_tp_rank = self.base_layer.tp_rank
    partition_sizes = self.base_layer.output_partition_sizes
    output_sizes = self.base_layer.output_sizes
    slices = []
    offset = 0
    for full_size, part_size in zip(output_sizes, partition_sizes):
        start_idx = local_tp_rank * part_size
        end_idx = start_idx + part_size
        slices.append(B[offset + start_idx : offset + end_idx, :])
        offset += full_size
    return torch.concat(slices, dim=0)
```

python/sglang/srt/lora/mem_pool.py

LoRAMemoryPool 新增 attn_tp_size 构造参数与 _effective_tp_size 分类方法，统一决定 LoRA 缓冲区沿哪种 TP 维度切分；get_lora_A_shape / get_lora_B_shape 与 load_lora_weight_tensor 全部接入新逻辑。

```
# 判断某个模块的 LoRA 缓冲区应沿哪个 TP 维度切分。
# 路由 MoE 专家按 moe_tp_size；注意力投影按 attn_tp_size（DP attention
# 下小于外层 tp_size）；其余模块按外层 tp_size。
def _effective_tp_size(self, module_name: str) -> int:
    if self.is_moe_module(module_name) and not self.is_shared_moe_module(
        module_name
    ):
        return self.moe_tp_size
    if module_name in ATTN_TP_LORA_MODULE_NAMES:
        return self.attn_tp_size
    return self.tp_size

# get_lora_A_shape 中的用法：行并行模块的输入维按 effective_tp_size
# 均分，保证缓冲区宽度与 base layer 的实际权重切分一致。
effective_tp_size = self._effective_tp_size(module_name)
if (
    effective_tp_size > 1
    and module_name in ROW_PARALLELISM_LINEAR_LORA_NAMES
    and module_name not in REPLICATED_LINEAR_LORA_NAMES
):
    input_dim = divide(input_dim, effective_tp_size)
```

python/sglang/srt/managers/tokenizer_control_mixin.py

新增 _merge_lora_update_results 合并 DP fan-out 的各 rank 回复，修正 result[0] 误读导致的注册表漂移；同时放宽 load/unload 的 dp_size 断言并接入三个端点。

```
def _merge_lora_update_results(results: List[LoRAUpdateOutput]) -> LoRAUpdateOutput:
    """合并 DP fan-out 返回的各 rank LoRA 加载/卸载结果。

    只有全部 rank 成功才算成功：把部分失败上报为成功会让 tokenizer 侧
    LoRA 注册表与失败 rank 的实际状态脱节。失败优先：去重后的错误信息
    以 | 连接，loaded_adapters 取第一个失败 rank 的现场。
    """
    failed = [r for r in results if not r.success]
    if not failed:
        # 全成功时直接返回首 rank 原对象：调用方在 LRU 驱逐时会
        # 原地修改 result.loaded_adapters，合成新对象会静默破坏该逻辑。
        return results[0]
    error_messages = list(
        dict.fromkeys(r.error_message for r in failed if r.error_message)
    )
    return LoRAUpdateOutput(
        success=False,
        error_message=" | ".join(error_messages),
        loaded_adapters=failed[0].loaded_adapters,
```

)

评论区精华

GitHub 界面上 Fridge003 直接 approve，copilot 因请求者配额耗尽未能审查；PR Test (Extra) 首轮失败后作者执行 /rerun-failed-ci。提交历史保留了 #32584 的两轮 review 反馈及其落地：

1) idle-forward 的 LoRA batch 重置应成为显式方法而非 prepare_lora_batch 的特殊分支（commit 70d8512 实现 reset_lora_batch）； 2) 初版用 LORA_LINEAR_SHARD_SPECS 表统一分类并行度被建议收窄，最终改为 ATTN_TP_LORA_MODULE_NAMES 特例集合 + _effective_tp_size 查询，把 LoRA 编排重构留到单独计划（commit 1316bfc）。

- Idle-forward 的 LoRA batch 重置是否应为显式方法 (design): 已采纳并实现为显式 reset_lora_batch 方法，无行为变化。
- 用 shard-spec 表还是特例集合处理 attn-TP (design): 采用特例集合方案，恢复原有 ROW_PARALLELISM / REPLICATED 列表，无功能变化；shard-spec 化重构留待后续。

风险与影响

- 风险：
  1. 核心路径变更：LoRAMemoryPool 构造函数新增必填 attn_tp_size，所有直接构造点必须同步，仓库内仅 lora_manager 一处（已更新），外部自定义引擎会 TypeError。
  2. 破坏性签名变更：layers.py 的 slice_lora_a_weights / slice_lora_b_weights 删除 tp_rank 参数，仓库外 LoRA 子类若仍按旧签名实现会运行时报错；仓库内测试桩已同步。
  3. 属性访问无防护：RowParallelLinearWithLoRA.forward 直接读取 base_layer.use_dp_attention_reduce 而没有 hasattr 保护，若某 base layer 缺该属性将 AttributeError；当前 SGLang 内置层均有该属性，风险中等偏低。
  4. 新能力组合首次启用：dp attention + 动态 LoRA 的部分失败语义为 "adapter 保持未注册但服务继续运行 "，用户侧需要感知该不一致窗口；纯 dp_size > 1 无 dp attention 仍被拒绝。
  5. 端到端 CI 覆盖偏薄：形状逻辑有 CPU 单元测试，真实多卡验证为手动 8x H200；PR Test (Extra) 首轮失败后 rerun，长期自动覆盖仍不足。 - 影响：用户影响：启用 --enable-dp-attention + --enable-lora 的部署可动态 load/unload LoRA，此前直接断言失败或在加载期崩溃；部分失败时 adapter 保持未注册、不可路由，错误信息去重后返回。 系统影响：LoRAMemoryPool 与各 LoRA 包装层对并行度的认知从单一外层 TP 演进为按模块分类（moe_tp / attn_tp / tp），缓冲区宽度更贴近实际权重切分，attn_tp 小于 tp 时注意力 LoRA 内存分配更紧凑。 团队影响：layers.py 的 slice 签名变更（删除 tp_rank）是破坏性 API，依赖旧签名的外部 LoRA 子类需同步；LoRAMemoryPool 构造新增必填参数，所有构造点需传入 attn_tp_size。该改动为后续 LoRA 编排重构（shard-spec 化）打下基础。
- 风险标记：核心路径变更，破坏性签名变更，端到端 CI 覆盖偏薄，新能力组合首启，属性访问无防护

关联脉络

- PR #32707 Split #32584 into 1/2: [LoRA] Guard DP-attention idle forwards against stale LoRA batch state: 本 PR 的前置、同源拆分，body 明确 stacked on #32707；其 idle-forward batch 状态守卫解决了 dp attention 下 LoRA 的第一处崩溃，本 PR 修复其余 shape-mismatch 与控制面问题。
- PR #32584 [title unavailable]: 被拆分的原始 PR，本 PR 为 2/2 的一半；提交历史中多处引用其 review feedback（显式 reset 方法、shard-spec 表收窄为特例集合）。