PR #47881 完整报告

vllm-project/vllm

[Feature] Migrate moe sp support to non-torch compiled path for GLM5.2

合并时间：2026-07-16 07:33

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47881

执行摘要

- 一句话：迁移 GLM5.2 MoE 序列并行到非 torch 编译路径
- 推荐动作：值得精读，因为它展示了在非 torch compile 环境中如何手动管理序列并行的 all-gather/ reduce-scatter 以及 fused norms。关注点：设计者如何通过在 Decoder 层内检测序列并行状态来避免额外的通信，以及 MTP 层中显式属性与隐式形状推断之间的权衡。该 PR 的修改是干净的，但缺少测试是明显弱点。

功能与动机

PR body 明确说明“已为正常路径启用，但非 torch 编译路径也有此需求”，因此需要迁移序列并行支持。该 PR 为 GLM5.2 模型提供完整的序列并行能力，使其在不启用 torch compile 的情况下也能使用 MoE 序列并行。

实现拆解

1. 新增序列并行辅助函数：在 vllm/models/deepseek_v32/nvidia/model.py 中新增 _all_gather_sp_states 函数，将两个张量（hidden_states 和 residual）拼接后一次 all-gather 再切分，减少通信次数。
2. Decoder 层改造：在 DeepseekV32DecoderLayer.__init__ 中添加 use_sequence_parallel_moe 属性（仅在满足并行配置且 MLP 是 MoE 时启用），并记录 tp_size。在 forward 中，通过判断输入形状与 full_num_tokens 是否一致来识别序列并行模式，在序列并行时对输入先做 all-gather 再传给 attention，并在 MoE 输出处调用 _all_gather_sp_states 恢复完整 token 布局，最后根据配置决定是否做 RMSNorm 的 fused all-reduce。
3. MTP 层适配：修改 vllm/models/deepseek_v32/nvidia/mtp.py 中的 forward，将原来无条件执行的 tensor_model_parallel_all_reduce 改为条件执行：仅在非序列并行时才进行 all-reduce；同时向 _restore_full_token_layout_if_needed 传递 is_sequence_parallel 参数。
4. 共享工具函数更新：在 vllm/model_executor/models/deepseek_mtp.py 中修改 _restore_full_token_layout_if_needed，新增 is_sequence_parallel 参数，控制是否跳过已知非 SP 情况下的 check。两个调用点都传入 self.mtp_block.use_sequence_parallel_moe。

5. MoE 专家配置调整：在 vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py 中，修改 TrtLlmFp8ExpertsBase._supports_parallel_config 和 TrtLlmFp8ExpertsModular._supports_parallel_config，在排除条件中增加 moe_parallel_config.is_sequence_parallel，明确序列并行模式下不支持该 monolithic kernel。

关键文件：

- vllm/models/deepseek_v32/nvidia/model.py（模块 模型层；类别 source；类型 core-logic；符号 DeepseekV32DecoderLayer, _all_gather_sp_states）：核心文件，实现 Decoder 层的序列并行支持，新增 _all_gather_sp_states 函数和 use_sequence_parallel_moe 逻辑，改动量最大。
- vllm/models/deepseek_v32/nvidia/mtp.py（模块 模型层；类别 source；类型 core-logic；符号 DeepseekV32MultiTokenPredictorLayer.forward）：MTP 层适配序列并行，在 forward 中根据 use_sequence_parallel_moe 有条件执行 all-reduce，并将参数传递给辅助函数。
- vllm/model_executor/models/deepseek_mtp.py（模块 模型层；类别 source；类型 data-contract；符号 _restore_full_token_layout_if_needed）：通用 MTP 辅助函数 _restore_full_token_layout_if_needed 新增 is_sequence_parallel 参数，使调用者能显式控制是否跳过 all-gather。
- vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py（模块 模型层；类别 source；类型 data-contract；符号 TrtLlmFp8ExpertsBase._supports_parallel_config, TrtLlmFp8ExpertsModular._supports_parallel_config）：在 MoE 专家基类和模组类的 _supports_parallel_config 中排除序列并行模式，确保序列并行下不会使用不兼容的 monolithic kernel。

关键符号：_all_gather_sp_states, DeepseekV32DecoderLayer.forward, DeepseekV32DecoderLayer.init, DeepseekV32MultiTokenPredictorLayer.forward, _restore_full_token_layout_if_needed, TrtLlmFp8ExpertsBase._supports_parallel_config, TrtLlmFp8ExpertsModular._supports_parallel_config

关键源码片段

vllm/models/deepseek_v32/nvidia/mtp.py

MTP 层适配序列并行，在 forward 中根据 use_sequence_parallel_moe 有条件执行 all-reduce，并将参数传递给辅助函数。

```
# Inside DeepseekV32MultiTokenPredictorLayer.forward:
hidden_states, residual = self.mtp_block(
    positions=positions, hidden_states=hidden_states, residual=None
)
hidden_states, residual = _restore_full_token_layout_if_needed(
    hidden_states,
    residual,
    positions.shape[0],
    is_sequence_parallel=self.mtp_block.use_sequence_parallel_moe,
```

```
)
if not self.mtp_block.use_sequence_parallel_moe:
    # Only reduce when not sequence parallel; otherwise tokens are already full.
    hidden_states = tensor_model_parallel_all_reduce(hidden_states)
# ... remaining recycle logic unchanged
```

评论区精华

- 关于 residual 需要 contiguous 的讨论：审查者 tlrmchlsmth 询问为什么对 residual 调用 .contiguous() 而对 hidden_states 不这样做。作者 yewentao256 回复指出底层的 fused_add_rms_norm 内核中有 STD_TORCH_CHECK(residual.is_contiguous()) 断言，因此 residual 必须连续，而 hidden_states 没有此要求。
- 关于序列并行探测方式的讨论：审查者 tlrmchlsmth 指出通过 `hidden_states.shape[0] != positions.shape[0]` 来推断序列并行是一种隐式约定，建议添加注释并希望有更显式的表达。作者 yewentao256 接受建议，将条件改为显式使用 `self.mtp_block.use_sequence_parallel_moe` 属性，并添加了注释说明。
  - residual 是否需要 contiguous (correctness): 接受当前设计，因底层内核要求 residual 连续。
  - 序列并行探测方式的隐式协议 (design): 已从隐式形状推断改为使用显式布尔属性，并添加注释。

风险与影响

- 风险：
  - 核心路径变更：对 DeepSeek V3.2 模型的 Decoder 层和 MTP 层的前向逻辑做了修改，可能影响非序列并行路径的稳定性（但改动中保留了旧分支，风险较低）。
  - 缺少测试覆盖：本次无直接对应的测试文件变更，可能存在回归风险，尤其是序列并行与 MTP 的组合场景。
  - 隐式协议：之前通过形状推断序列并行的方式已被改为显式属性，但在其他部分（如 _all_gather_sp_states 中通过 all-gather 截断来匹配 num_tokens）仍依赖调用者传递正确的 token 数，若调用不匹配可能导致静默错误。
  - 专家配置：trtllm_fp8_moe.py 中增加禁用序列并行的逻辑，若其他组件在序列并行下使用了该专家实现，会直接失败（但期望如此），需确保依赖方已适配。
  - 影响：该 PR 影响范围集中于 NVIDIA GPU 上使用 GLM5.2 模型的用户，特别是启用 Expert Parallel (EP) 和 Sequence Parallel (SP) 时的推理正确性和性能。MTP (Multi-Token Prediction) 流水线也随之适配。影响程度中等：新增功能不影响默认路径（非序列并行行为保持不变），但为特定配置解锁了重要能力。团队需留意后续对同一模型区域的重构可能会与此处隐式约定冲突。
  - 风险标记：核心路径变更，缺少测试覆盖，隐式序列并行协议

关联脉络

- PR #45895 [MTP] Fix post-norm recycle logic for DeepSeek MTP: 该 PR（#45895）启用了 post-norm recycle，当前 PR 的 mtp.py 中注释提到的 PR #45895 正是此 PR，说明当前递归逻辑依赖 #45895 的变更。
- PR #47070 [Perf] GLM5.2 MoE SP performance optimization: PR body 中提到了性能参考链接到 #47070，该 PR 与 GLM5.2 的 MoE SP 性能优化相关，当前 PR 迁移其非 torch 编译路径。