

PR #7805 完整报告

verl-project/verl

[megatron] fix: make R2 router replay model-scoped and opt-in

合并时间: 2026-09-14 10:23

原文链接: <http://prhub.com.cn/verl-project/verl/pull/7805>

执行摘要

- 一句话: R2 回放模型作用域化并显式启用, 修复 Megatron 回归
- 推荐动作: 值得精读。重点学习三点: (1) opt-in + model-scoped 的双重机制如何从根上避免 #7106 的回归——这是 "修复被回退后重做" 的教科书案例; (2) 分布式集体通信中空 stage 用 0-layer 占位参与的设计, pp_gather 的防御性校验也值得借鉴; (3) 多模态 HF config 树遍历中 "只走 text 分支、隔离兄弟配置" 的约束, 对 Qwen3-Omni 类模型接入有直接价值。此外, SamitHuang 的三条 review 展示了如何把 "静默错误" 审出来并推动作者用测试固化。

功能与动机

R2 router replay 用于在 fully-async RL 的 actor 更新阶段精确重放记录过的 MoE 路由, 保证 old_log_probs 与更新前向的路由一致。原始修复 #7106 在 THD-packed batch 下修复了序列并行对齐与 PP/VPP 本地 router 选择, 但其实现会无条件查询进程全局的 RouterReplay.router_instances registry, 导致所有未启用 replay 的普通 Megatron SFT/RL 任务在 registry 为空时直接抛错: `RuntimeError: cannot map router replay registry to the current PP/VPP stage: registry=0, ...`。该回归迫使 #7786 整体回退。本 PR 的核心命题正如 body 所述: "This version makes router replay explicitly opt-in and scopes RECORD, REPLAY, and action transitions to the routers owned by the currently forwarded model", 即在恢复正确性的同时, 把默认 disabled 路径还原为完全 no-op。

实现拆解

1. 变更入口: forward_backward_batch 中的显式 opt-in 判定

verl/workers/engine/megatron/transformer_impl.py 是唯一执行入口。原实现每个 micro-batch 的 forward 通过 RouterReplayHelper.is_r2_record_action(tf_config) 这类全局查询决定行为; 现在改为一次性计算批级标志 `record_r2_routes = enable_routing_replay and forward_only and self.engine_config.router_replay.mode == "R2" and data.get("routed_experts", None) is None`, 并通过 `tu.assign_non_tensor(micro_batch, record_r2_routes=...)` 下发给每个 micro-batch。forward_step 中 RECORD/REPLAY 动作切换也全部改为携带 `model=unwrapped_model` 的查询形式——action 状态直接读自当前 forward 的模型, 进程内残留的全局 router 状态不再影响判断, 从语义上杜绝 #7106 的回归。

2. 核心逻辑: router_replay_utils.py 的 model-scoped 收集、空 stage 占位与排序

merge_router_topk_indices 与 set_router_replay_data 新增 model 参数: 当传入 forwarded model 时, 通过 iter_model_routers(model) 收集本模型 chunk 的 router, 并按 layer_number 排序、检测重复层号, 保证 RECORD 堆叠顺序与 REPLAY 按层索引严格一致。关键设计是空 router 列表的分层处理: replay 禁用的默认路径下必须是 no-op; 若该 PP stage 本应构建 MoE 层 (get_moe_num_layers_to_build 非零) 却一个 router 都没有, 则硬失败; 对真正空的 dense stage, 构造 [bs, seq_len, 0, topk] 的 0-layer 占位张量加入 mini_layer_topk_idx_list, 使该 rank 仍能进入 pp_gather 这一 PP 全量集合通信。set_router_replay_data 侧同样要求 replay 数据覆盖每个 forwarded MoE 层, 缺失即报错, 防止路由错位到错误层。

3. VPP 重排与 PP gather 的防御性校验

reorder_and_merge_vpp_layers 增加捕获数量与 schedule 表长度的强校验 (要求 R2 RECORD 必须覆盖每个调度的 microbatch/chunk, 包括 0 层 chunk); pp_gather 增加 PP group 大小校验、BSHD 路径的逐层 all_gather 长度断言与空 local map 的 0-layer 传播。这些校验让失败模式从 "路由错位到错误的 token/层" 变为显式报错。

4. 嵌套多模态 HF config 属性解析

verl/utils/megatron_utils.py 新增 _get_hf_text_config、_iter_hf_config_tree、get_hf_config_attr, 并重写 get_hf_rope_theta。遍历只走 text_config、language_config、thinker_config、model_config 四类 text 分支, 绝不进入任意 sub_configs——因为 audio/code2wav 等兄弟配置可能暴露冲突的 hidden_size/rope_theta (Qwen3-Omni 是典型场景)。make_megatron_module 中 hidden_size 的获取也切换为 get_hf_config_attr。

5. 配置互斥校验

verl/utils/config.py 新增 _validate_router_replay_config: R2 依赖 actor old_log_probs 前向来记录路由, 与 rollout_correction.bypass_mode=True (跳过该前向) 语义冲突, 因此在 worker 启动前直接拒绝该组合, 报错信息明确说明原因。

6. 测试与验证配套

本 PR 新增 / 扩展 8 个测试文件, 形成三层防护: CPU 单测 (test_router_replay_utils_on_cpu.py 覆盖 registry 布局、空 registry no-op、forwarded-model 优先、排序与重复层号; test_megatron_utils_nested_config.py 覆盖嵌套 config 的 text 优先与兄弟隔离; test_config_router_replay_on_cpu.py 覆盖 R2 与 bypass_mode 互斥)、真实分布式测试 (test_router_replay_empty_pp.py 用 4-GPU PP 验证空 / 不均匀 stage 的 gather)、以及 THD padding 与 DCP 边界测试 (test_megatron_bshd_preprocess.py、test_megatron_router_replay_dcp.py)。验证结果: 39 项聚焦测试与 87 项扩展测试全部通过, 8 个真实 4-GPU PP/VPP 用例通过, 32-GPU Qwen3-Omni R2 PP2/VPP2 fully-async E2E 完成 1 次 actor 更新与权重同步, rollout 参数版本从 0 推进到 1。

关键文件:

- verl/utils/megatron/router_replay_utils.py (模块 路由重放; 类别 source; 类型 entrypoint; 符号 merge_router_topk_indices, set_router_replay_data, reorder_and_merge_vpp_layers, pp_gather) : R2 router replay 的核心工具模块: merge_router_topk_indices 与 set_router_replay_data 改为 model-scoped 收集, 新增空 stage 0-layer 占位、按 layer_number 排序、重复层号与缺失路由硬校验, 是本次修复的主战场。
- verl/utils/megatron_utils.py (模块 配置解析; 类别 source; 类型 core-logic; 符号 get_hf_config_attr, get_hf_rope_theta, _iter_hf_config_tree, _get_hf_text_config) : 新增嵌套多模态 HF config 属性解析工具 (_get_hf_text_config/_iter_hf_config_tree/get_hf_config_attr), 重写 get_hf_rope_theta, 并让 make_megatron_module 的 hidden_size 获取走 get_hf_config_attr, 支撑 Qwen3-Omni 类模型的 R2 训练。
- verl/workers/engine/megatron/transformer_impl.py (模块 引擎层; 类别 source; 类型 core-logic; 符号 forward_backward_batch, forward_step) : MegatronEngine 核心热路径: forward_backward_batch 新增 record_r2_routes 批级 opt-in 判定, forward_step 的 RECORD/REPLAY 切换改为 model-scoped 查询, 是根治 #7106 回归的入口层改动。
- tests/workers/engine/megatron/test_router_replay_utils_on_cpu.py (模块 路由重放; 类别 test; 类型 test-coverage; 符号 _FakeRouter, test_empty_registry_is_a_noop_for_replay_disabled_engine, test_action_queries_use_forwarded_model_instead_of_global_registry, test_registry_layouts_select_exact_moe_layers) : 新增 429 行的核心单测: 覆盖空 registry no-op、forwarded-model 优先于全局 registry、local/global registry 布局、VPP 层号选择、重复层号与排序、空 MoE stage 的 0-layer map, 是本 PR 正确性论证的基石。
- tests/utils/test_megatron_utils_nested_config.py (模块 配置解析; 类别 test; 类型 test-coverage; 符号 test_get_hf_rope_theta_from_nested_omni_text_config, test_get_hf_config_attr_handles_nested_configs_and_cycles, test_nested_config_lookup_does_not_fall_back_to_multimodal_sibling, test_get_hf_config_attr_preserves_falsey_text_values) : 新增嵌套多模态 config 解析的专项测试: text 优先、兄弟隔离、防环、falsey 值保留, 验证 get_hf_config_attr/get_hf_rope_theta 在多模态与 transformers>=5 配置下的行为。
- verl/utils/config.py (模块 配置校验; 类别 source; 类型 core-logic; 符号 _validate_router_replay_config) : 新增 _validate_router_replay_config: 在 worker 启动前拒绝 R2 与 rollout_correction.bypass_mode=True 的组合, 因为 R2 依赖 actor old_log_probs 前向记录路由, bypass 会跳过该前向。
- tests/special_distributed/test_router_replay_empty_pp.py (模块 流水线测试; 类别 test; 类型 test-coverage; 符号 test_empty_and_uneven_pp_collectives) : 新增 4-GPU 真实 PP collective 测试: 验证空 / 不均匀 MoE 层分布的 PP stage 在 pp_gather 与 VPP 重排下的路由完整性, 直接回应 SamitHuang 关于 collective 一致性的 review。
- tests/utils/test_config_router_replay_on_cpu.py (模块 配置校验; 类别 test; 类型 test-coverage; 符号 test_r2_rejects_bypass_mode_before_worker_startup, test_non_r2_modes_allow_bypass) : 覆盖 R2 + bypass_mode 配置互斥的新增校验, 以及 non-R2 模式允许 bypass 的兼容行为。

关键符号: merge_router_topk_indices, set_router_replay_data, reorder_and_merge_vpp_layers, pp_gather, get_hf_config_attr, get_hf_rope_theta, _iter_hf_config_tree, _get_hf_text_config, _validate_router_replay_config, forward_backward_batch, forward_step, is_r2_record_action, is_replay_forward_action, is_replay_backward_action, _get_action_router_list

关键源码片段

verl/utils/megatron/router_replay_utils.py

R2 router replay 的核心工具模块: merge_router_topk_indices 与 set_router_replay_data 改为 model-scoped 收集, 新增空 stage 0-layer 占位、按 layer_number 排序、重复层号与缺失路由硬校验, 是本次修复的主战场。

```
def merge_router_topk_indices(attention_mask, input_ids, mini_layer_topk_idx_list,
                              tf_config, vp_rank=None, local_cp_size=None, model=None):
    """把当前 micro-batch 记录到的 top-k 路由合并进 mini_layer_topk_idx_list"""
    with torch.no_grad():
        # BSHD 非嵌套输入必须带 attention_mask, 否则无法做 pack/unpack 对齐
        if not input_ids.is_nested and attention_mask is None:
            raise RuntimeError("router replay RECORD requires attention_mask for non-nested BSHD inputs.")
        vp_rank = 0 if vp_rank is None else vp_rank

        # 优先从当前 forward 的模型收集 router, 而不是扫描进程全局 registry。
        # 这是规避 #7786 回归的关键: replay 禁用时根本不会走到这里。
        if model is not None:
            # 按 layer_number 排序后再堆叠, 保证 RECORD 的层序与 REPLAY
            # 按 layer_number - 1 / MoE 序号索引时严格一致
            model_routers = sorted(iter_model_routers(model), key=lambda item: item[0])
            layer_numbers = [layer_number for layer_number, _ in model_routers]
            if len(layer_numbers) != len(set(layer_numbers)):
                raise RuntimeError("router replay RECORD found duplicate layer numbers in the forwarded model: "
                                   f"layer_numbers={layer_numbers}, vp_rank={vp_rank}")
            router_instances_list = [router for _, router in model_routers]
        else:
            # 兼容旧调用方: 仍支持按全局 registry 定位 router
            router_instances_list = RouterReplayHelper.get_micro_batch_router_list(tf_config, vp_rank)

        # 空 router 列表: replay 禁用语义下是 no-op, 但本应含 MoE 层的 stage 必须硬失败
        if not router_instances_list:
            expected = get_moe_num_layers_to_build(tf_config, vp_rank)
            if expected:
                raise RuntimeError("router replay RECORD found no routers in a stage that expects MoE layers: "
                                   f"expected={expected}, vp_rank={vp_rank}")
            topk = getattr(tf_config, "moe_router_topk", None)
```

```

if not isinstance(topk, int) or topk <= 0:
    raise RuntimeError("router replay RECORD cannot build an empty local route map
    without a positive "
        f"moe_router_topk, got {topk!r}.")
# 0-layer 占位 map: 让空 PP stage 也能参与 pp_gather 这一 PP 全量集体通信
if input_ids.is_nested:
    empty_routes = [torch.empty((tokens.shape[0], 0, topk), dtype=torch.int16)
        for tokens in input_ids.unbind()]
    mini_layer_topk_idx_list.append(torch.nested.as_nested_tensor(empty_routes, layout=
    torch.jagged))
else:
    batch_size, seq_len = attention_mask.shape[:2]
    mini_layer_topk_idx_list.append(
        torch.empty((batch_size, seq_len, 0, topk), dtype=torch.int16, device="cpu"))
return

# 校验本 stage 所有 router 都完成了记录, 缺失即报错, 防止静默错位
missing = [idx for idx, router in enumerate(router_instances_list)
    if router.recorded_topk_idx is None]
if missing:
    local_rank_info = get_current_rank_layer_info(tf_config, vp_rank)
    raise RuntimeError("router replay RECORD did not capture all local routers: "
        f"missing_local_positions={missing}, local_layer_range={local_rank_info},
        "
        f"pp_rank={mpu.get_pipeline_model_parallel_rank()}, vp_rank={vp_rank}")
# 后续: 堆叠、uint8 view、SP gather、THD/BSHD pack 对齐……

```

verl/utils/megatron_utils.py

新增嵌套多模态 HF config 属性解析工具 (`_get_hf_text_config/_iter_hf_config_tree/get_hf_config_attr`) , 重写 `get_hf_rope_theta`, 并让 `make_megatron_module` 的 `hidden_size` 获取走 `get_hf_config_attr`, 支撑 Qwen3-Omni 类模型的 R2 训练。

```

# 只允许 text 类子配置参与遍历; 绝不进入任意 sub_configs,
# 避免 audio/code2wav 等兄弟分支暴露冲突的 hidden_size / rope_theta
_HF_CONFIG_CHILD_NAMES = ("text_config", "language_config", "thinker_config", "model_
config")

```

```

def _get_hf_text_config(hf_config):
    # 优先走 transformers 的规范化 get_text_config() 访问器
    get_text_config = getattr(hf_config, "get_text_config", None)
    if callable(get_text_config):
        text_config = get_text_config()
        if text_config is not None:
            return text_config
    return getattr(hf_config, "text_config", None)

```

```

def _iter_hf_config_tree(hf_config):

```

```

seen = set()

def _visit(config):
    if id(config) in seen:
        return # 防环: 嵌套 config 可能反向引用根对象
    seen.add(id(config))

    text_config = _get_hf_text_config(config)
    if text_config is not None and text_config is not config:
        yield from _visit(text_config) # 优先 text 分支
    for child_name in _HF_CONFIG_CHILD_NAMES:
        child_config = getattr(config, child_name, None)
        if child_config is not None and child_config is not text_config:
            yield from _visit(child_config)
    yield config

yield from _visit(hf_config)

```

```

def get_hf_config_attr(hf_config, attr_name):
    # 先从规范化 text config 直接取值, 避免被根对象或兄弟分支的属性抢先
    text_config = _get_hf_text_config(hf_config)
    if text_config is not None:
        value = getattr(text_config, attr_name, None)
        if value is not None:
            return value
    for config in _iter_hf_config_tree(hf_config):
        value = getattr(config, attr_name, None)
        if value is not None:
            return value
    raise AttributeError(f"{type(hf_config).__name__} has no nested {attr_name}.")

```

verl/workers/engine/megatron/transformer_impl.py

MegatronEngine 核心热路径: `forward_backward_batch` 新增 `record_r2_routes` 批级 opt-in 判定, `forward_step` 的 RECORD/REPLAY 切换改为 model-scoped 查询, 是根治 #7106 回归的入口层改动。

```

enable_routing_replay = tu.get_non_tensor_data(data, key="enable_routing_replay", default=False)
# 批级一次性判定 R2 RECORD: 默认 disabled 路径完全不会触碰 replay 状态,
# 这是相对 #7106 回归 (无条件查询全局 registry) 的根治性改动
record_r2_routes = (
    enable_routing_replay
    and forward_only
    and self.engine_config.router_replay.mode == "R2"
    and data.get("routed_experts", None) is None
)
for micro_batch in micro_batches:
    tu.assign_non_tensor(micro_batch, num_micro_batch=n_micro_batch)

```

```

tu.assign_non_tensor(micro_batch, record_r2_routes=record_r2_routes)
# ... 其余 micro_batch 预处理

# R2 RECORD 完成后的路径: VPP 时按 schedule 重排各 chunk,
# 非 DCP 场景把 routed_experts 写回 model_output 供 actor 更新阶段 REPLAY
if record_r2_routes:
    if not self.mini_layer_topk_idx_list:
        raise RuntimeError("R2 RECORD completed without capturing any router routes.")
    if self.tf_config.virtual_pipeline_model_parallel_size is not None:
        # VPP 分支: 先按 schedule 重排再合并各 chunk 的 routes
        ...
    else:
        layers_topk_idx = torch.cat(self.mini_layer_topk_idx_list, dim=0)
    if record_r2_routes and dcp_group is None:
        output["model_output"]["routed_experts"] = layers_topk_idx

```

评论区精华

SamitHuang 的 review 是本 PR 质量的关键推手，两条 should-fix 直接塑造了最终实现：

should-fix 1 (空 PP stage 与 collective) : `pp_gather` 是 PP 全量集合通信，所有 rank 必须进入。若某个 PP rank 拥有 0 个 MoE 层，不应在 `pp_gather` 前抛错，而应以 0-layer 张量参与；只有 " 该 rank 有 router 却没捕获到数据 " 时才允许硬失败。

should-fix 2 (堆叠顺序) : `merge_router_topk_indices` 丢弃了 `layer_number`、按 `modules()` walk order 堆叠，而 `set_router_replay_data` 按 `layer_number - 1/` 全局 MoE 序号索引；若 walk order 不是递增 `layer_number`，RECORD 与 REPLAY 会静默不一致，应先按 `layer_number` 排序再堆叠。

nit 3 (多模态兄弟回退) : `get_text_config()` 未命中时，BFS 可能返回 `audio/code2wav` 等兄弟配置的冲突属性；应限制遍历到 `text` 分支，并注意 `getattr(..., None)` 会把合法的 `0/False` 当作缺失。

作者在 Issue 评论中逐条回应了落地方式: "Preserve empty PP/VPP chunks and handle uneven route counts during gathering; missing routes on expected MoE stages still fail. Sort RECORD routers by layer number to match REPLAY ordering. Restrict nested config lookup to text wrappers, preserving 0/False." 三个问题均在后两个 commit (0fdc1f8、f7b44d8) 中解决，测试同步覆盖；最终由 wuxibin89 给出 APPROVED。

- 空 MoE 的 PP 阶段与 `pp_gather` collective 一致性 (correctness): 作者在 0fdc1f8 中改为 : 空 stage 生成 `[bs, seq_len, 0, topk]` 占位 map 并 append，只有 " 本应含 MoE 层却无 router " 时才硬失败；新增 4-GPU 分布式测试 `test_empty_and_uneven_pp_collectives` 固化。
- RECORD 堆叠顺序与 REPLAY 层索引的一致性 (correctness): 作者改为 `sorted(iter_model_routers(model), key=lambda item: item[0])` 先按层号排序，并校验重复层号；测试 `test_merge_router_topk_indices_uses_forwarded_model_with_leftover_vpp_recordings` 用乱序 walk 验证。

- 嵌套多模态 config 属性查找的兄弟回退问题 (design): 作者将遍历限制到 `_HF_CONFIG_CHILD_NAMES` (text/language/thinker/model config), 绝不进入任意 `sub_configs`; text 直接路径保留 falsey 值, 测试 `test_nested_config_lookup_does_not_fall_back_to_multimodal_sibling` 与 `test_get_hf_config_attr_preserves_falsey_text_values` 覆盖。

风险与影响

- 风险:
 1. 分布式集体通信一致性: `pp_gather` 是 PP 全量 `all_gather`, 要求所有 rank 以相同分支进入。本 PR 用 0-layer 占位解决 "空 MoE stage", 但若出现 "config 判定为有 MoE 而模型实际没有 router" 的拓扑不一致, 仍会在 `collective` 前抛错——这是有意设计的 fail-fast, 但需要保证各 rank 的 `tf_config` 解析完全一致。
 2. registry 布局兼容假设: `get_micro_batch_router_list` 同时接受 local 与 global 两种布局并做完整性校验; 任何按旧布局构造 registry 的第三方路径 (如 `model=None` 的 positional helper) 在布局不匹配时会触发新的 `RuntimeError`。属于行为变化, 方向是更安全的显式错误, 但可能影响外部集成。
 3. 热路径开销: `is_r2_record_action/is_replay_forward_action` 现在接收 `model=unwrapped_model` 并遍历模型收集 router, 仅 R2/R3 启用时发生; disabled 路径新增开销为常数级。但 `sorted()` 与重复层号检查在每次 RECORD forward 都会执行, 超深 MoE 模型上的 CPU 开销需留意。
 4. 配置解析行为变化: `make_megatron_module` 的 `hidden_size` 获取从 "root 或 text_config 直接属性" 变为树遍历; 对不存在 text_config 的纯单模态 config 行为不变, 对多模态 config 优先 text 分支, 与旧行为可能不同, 测试覆盖了主要模式, 特殊 config 仍需观察。
 5. 新配置互斥: `R2 + bypass_mode=True` 会在启动时抛 `ValueError`, 影响下游 `verl-omni` 等依赖旧行为的配置, 但属于启动期显式报错而非运行期异常。
 6. 验证盲区: 32-GPU E2E 仅验证了 1 次 actor 更新, R3 路径保持原语义但未新增测试; 4-GPU 分布式测试需要 CUDA 环境, CI 中实际执行频率取决于调度。
 - 影响: 用户影响: 启用 R2 的 Megatron 用户 (如 `verl-omni #290`) 在 PP/VPP/THD 多模态模型上获得正确且可用的路由回放; 普通 Megatron SFT/RL 任务回到 #7106 之前的稳定行为。
 - 系统影响: MegatronEngine 是 verl 核心执行引擎, 改动集中在 replay 分支; 错误模式从 "路由错位到错误 token/层" 改进为 "显式报错", `pp_gather/reorder_and_merge_vpp_layers` 成为更安全的组件。
 - 团队影响: 新增的 4-GPU 分布式测试模板与 CPU 单测矩阵为后续 PP 相关改动提供基线; `get_hf_config_attr` 成为可复用的嵌套配置解析基础设施, 后续多模态模型接入可直接受益。
 - 风险标记: 分布式集体通信一致性, 核心热路径变更, 历史回归二次修复, registry 布局兼容假设, 配置契约变更

关联脉络

- PR #7106 [megatron] fix: preserve R2 router replay for THD-packed batches: 本 PR 取代的原始实现: 修复 THD-packed batch 下 R2 路由回放正确性, 但因 replay 禁用时仍

查询全局 registry 引入回归而被回退。

- PR #7786 Revert "[megatron] fix: preserve R2 router replay for THD-packed batches": 本 PR 修复的回归来源：回退了 #7106；#7805 以 opt-in + model-scoped 的方式重新引入其正确性修复。
- PR #7702 [megatron] fix: pad VLM THD input to TP/CP alignment before SP scatter: 同属 Megatron THD/ 序列并行正确性维护线，说明 verl 在 Megatron 深度模型的 THD packing 与分布式对齐上持续投入。