PR #6062 完整报告

verl-project/verl

[fully_async, rollout, trainer, tool, cfg] fix: ROCm async training compatibility for AMD MI300X

合并时间：2026-04-20 16:26

原文链接：http://prhub.com.cn/verl-project/verl/pull/6062

执行摘要

- 一句话：修复完全异步 FSDP2 训练在 AMD ROCm 平台（MI300X 系列）的兼容性问题。
- 推荐动作：该 PR 值得精读，特别是 ZMQ IPC 句柄重构的设计决策，它展示了如何通过平台无关的 rank 算术解决硬件特定的通信不匹配问题。建议关注 _get_zmq_handle 方法的实现和讨论中关于错误处理与 Actor 命名的权衡。

功能与动机

根据 PR body 描述，完全异步 FSDP2 训练在 AMD ROCm 平台（MI300X 系列）上无法正常工作，具体表现为 FSDP 初始化失败（ncclSystemError）、ZMQ 通信不匹配、JSON 序列化错误和 IPC 套接字残留导致重启失败。作者在 AMD Instinct MI3xx（8x GPU, 192 GB HBM *each*）和 NVIDIA H20 上进行了交叉验证，旨在使训练流程在 AMD 平台上稳定运行。

实现拆解

1. 添加 AMD RCCL 必需的环境变量：在 verl/trainer/constants_ppo.py 的 PPO_RAY_RUNTIME_ENV 中添加 "HSA_NO_SCRATCH_RECLAIM": "1"，以解决 FSDP 初始化时的 ncclSystemError。该变量在非 AMD 平台上会被忽略。
2. 修复 JSON 序列化错误：在 verl/trainer/ppo/ray_trainer.py 的 _dump_generations 方法中，将 json.dumps 调用改为 json.dumps(..., default=str)，以处理 NumPy 2.x 中 numpy.bool_ 不再是 Python bool 子类的问题。
3. 重构 ZMQ IPC 句柄生成逻辑：在 verl/workers/rollout/vllm_rollout/vllm_rollout.py、verl/workers/rollout/vllm_rollout/utils.py 和 verl/workers/rollout/vllm_rollout/vllm_async_server.py 中，将句柄从基于 GPU UUID 改为基于 (replica_rank, local_rank)。这解决了 ROCm 平台上因 CUDA_VISIBLE_DEVICES/HIP_VISIBLE_DEVICES 设置不同导致的 GPU UUID 不匹配问题，并避免了多副本共享节点时的套接字冲突。
4. 清理残留的 ZMQ IPC 套接字文件：在 verl/workers/rollout/vllm_rollout/bucketed_weight_transfer.py 的 _init_socket 和 _cleanup 方法中，添加逻辑在绑定前和清理后删除 IPC 套接字文件，使用 except OSError 捕获权限错误等异常，防止重启时出现 Address already in use 错误。
5. 修复 Hydra 配置路径：将 verl/experimental/fully_async_policy/config/fully_async_ppo_trainer.yaml 中的搜索路径从 file://verl/trainer/config 改为 pkg://verl.trainer.config，以支持可编辑安装。

6. 防止 Ray Actor 重复创建：在 verl/tools/sandbox_fusion_tools.py 的 init_execution_pool 函数中，为 ExecutionWorker 添加 name="sandbox-execution-pool" 和 get_if_exists=True 选项，确保在同一训练会话中重用 Actor。

关键文件：

- verl/workers/rollout/vllm_rollout/utils.py（模块 Rollout 通信；类别 source；类型 core-logic；符号 _get_zmq_handle）：修改了 vLLMColocateWorkerExtension 和 vLLMOmniColocateWorkerExtension 的 _get_zmq_handle 方法，将句柄从基于 GPU UUID 改为基于 (replica_rank, local_rank)，这是解决 ROCm 通信不匹配的核心改动之一。
- verl/workers/rollout/vllm_rollout/vllm_rollout.py（模块 Rollout 通信；类别 source；类型 core-logic；符号 init）：修改了 VLLMRollout 类的 __init__方法，重构了 ZMQ 句柄生成逻辑，使用 replica_rank 和 node-local rank 替代 GPU UUID，并添加了详细注释解释原因。
- verl/workers/rollout/vllm_rollout/bucketed_weight_transfer.py（模块 权重传输；类别 source；类型 core-logic；符号 _init_socket, _cleanup）：在 BucketedWeightSender 的 _init_socket 和 _cleanup 方法中添加了 IPC 套接字文件清理逻辑，防止重启时出现 'Address already in use' 错误，并根据 review 反馈改进了错误处理。
- verl/trainer/constants_ppo.py（模块 训练配置；类别 source；类型 configuration；符号 PPO_RAY_RUNTIME_ENV）：在 PPO_RAY_RUNTIME_ENV 中添加了 AMD RCCL 必需的环境变量 HSA_NO_SCRATCH_RECLAIM，解决了 FSDP 初始化失败问题。
- verl/trainer/ppo/ray_trainer.py（模块 训练器核心；类别 source；类型 core-logic；符号 dump_generations）：修复了 JSON 序列化错误，为 json.dumps 添加 default=str 参数以处理 numpy.bool 类型。
- verl/tools/sandbox_fusion_tools.py（模块 工具脚本；类别 source；类型 core-logic；符号 init_execution_pool）：为 ExecutionWorker 添加 name 和 get_if_exists 选项，防止同一训练会话中重复创建 Actor。
- verl/workers/rollout/vllm_rollout/vllm_async_server.py（模块 异步服务器；类别 source；类型 core-logic）：设置了 VERL_REPLICA_RANK 环境变量，确保 vLLM 工作进程能继承正确的副本 rank。
- verl/experimental/fully_async_policy/config/fully_async_ppo_trainer.yaml（模块 实验配置；类别 config；类型 configuration）：修复 Hydra 配置搜索路径，支持可编辑安装。

关键符号：_get_zmq_handle, init, _init_socket, _cleanup, _dump_generations, init_execution_pool

关键源码片段

verl/workers/rollout/vllm_rollout/utils.py

修改了 vLLMColocateWorkerExtension 和 vLLMOmniColocateWorkerExtension 的 _get_zmq_handle 方法，将句柄从基于 GPU UUID 改为基于 (replica_rank, local_rank)，这是解决 ROCm 通信不匹配的核心改动之一。

```
def _get_zmq_handle(self) -> str:
    """Get ZMQ handle for communication.
```

```
    Uses replica_rank + local_rank to form handle so it matches the sender side
    regardless of CUDA_VISIBLE_DEVICES differences, and avoids collisions
    when multiple replicas share the same node.
    """
    replica_rank = os.environ.get("VERL_REPLICA_RANK", "0") # 从环境变量获取副本
    rank，默认为 "0"
    return f"ipc:///tmp/rl-colocate-zmq-replica-{replica_rank}-rank-{self.local_rank}.sock" #
    基于副本 rank 和本地 rank 构建句柄
```

verl/workers/rollout/vllm_rollout/vllm_rollout.py

修改了 VLLMRollout 类的 __init__方法，重构了 ZMQ 句柄生成逻辑，使用 replica_rank 和 node-local rank 替代 GPU UUID，并添加了详细注释解释原因。

```
def __init__(self, config, replica_rank=-1):
    # ... 省略 rank 计算等初始化代码 ...
    # Use replica_rank + node-local rank to form ZMQ handle instead of GPU UUID,
    # because CheckpointEngineWorker and vLLM worker may see different GPU UUIDs
    # when CUDA_VISIBLE_DEVICES differs between processes (common on ROCm/AMD).
    # Must use node-local rank (not rollout_rank) so it matches vLLM worker's
    # local_rank on every node. Include replica_rank to avoid collisions when
    # multiple replicas share a node.
    local_rank = self.rollout_rank % local_world_size # 计算节点本地 rank
    self.zmq_handle = f"ipc:///tmp/rl-colocate-zmq-replica-{self.replica_rank}-rank-{local_rank}.
    sock" # 构建新句柄
```

verl/workers/rollout/vllm_rollout/bucketed_weight_transfer.py

在 BucketedWeightSender 的 _init_socket 和 _cleanup 方法中添加了 IPC 套接字文件清理逻辑，防止重启时出现 'Address already in use' 错误，并根据 review 反馈改进了错误处理。

```
def _init_socket(self):
    """Initialize ZMQ REQ socket and bind."""
    if self.zmq_handle.startswith("ipc://"):
        ipc_path = self.zmq_handle[len("ipc://") :] # 提取 IPC 文件路径
        try:
            os.remove(ipc_path) # 尝试删除残留文件
        except OSError: # 捕获所有文件系统错误（包括 PermissionError）
            pass # 忽略错误，避免进程崩溃
    self.socket = self.zmq_context.socket(zmq.REQ)
    self.socket.bind(self.zmq_handle)
```

评论区精华

1. IPC 套接字清理的错误处理：gemini-code-assist[bot] 指出，仅捕获 FileNotFoundError 在多用户环境中存在风险，残留套接字文件可能因权限问题导致 PermissionError，进而使训练进程终止。建议改为捕获更宽泛的 OSError。作者在第二次提交中采纳了此建议，将 except FileNotFoundError 改为 except OSError。
2. Ray Actor 全局名称的潜在冲突：gemini-code-assist[bot] 警告，硬编码的全局名称 "sandbox-execution-pool" 在多租户 Ray 集群中可能导致碰撞，使新作业意外连接到现有池。

作者 xiaohong42 解释，Verl 训练作业通过 ray.init() 创建独立的 Ray 集群，因此跨作业的命名空间冲突不会发生；硬编码名称旨在同一训练会话内重用 Actor，与同文件中 TokenBucketWorker 的模式一致。

3. AMD环境变量的作用：wuxibin89询问HSA_NO_SCRATCH_RECLAIM环境变量的用途。xiaohong42 详细解释，该变量是 AMD RCCL 在 MI300X GPU 上的必需设置，用于控制 GPU 暂存内存回收，避免 FSDP 初始化失败。

- IPC 套接字清理的错误处理 (correctness): 作者在第二次提交中采纳建议，将 except FileNotFoundError 改为 except OSError，以更优雅地处理文件系统错误。
- Ray Actor 全局名称的潜在冲突 (design): 作者认为当前设计是合理的，未作修改，因为 Verl 的作业隔离机制降低了风险。
- AMD 环境变量的作用 (question): 通过解释澄清了该变量的必要性和平台特异性。

风险与影响

- 风险：
  1. 平台兼容性风险：新增的 HSA_NO_SCRATCH_RECLAIM 环境变量是 AMD 特有的，在 NVIDIA 或 Ascend 平台上会被忽略，但需确保不会意外干扰其他硬件。
  2. ZMQ 句柄逻辑变更风险：ZMQ IPC 句柄从基于 GPU UUID 改为基于 (replica_rank, local_rank)，虽然解决了 ROCm 不匹配问题，但需确保在所有部署场景（如单节点、多节点、多副本）下都能正确匹配，避免通信失败。
  3. IPC 套接字清理风险：清理残留套接字文件时捕获 OSError 可能掩盖其他文件系统问题，但权衡后认为防止重启失败更重要。
  4. Ray Actor 重用风险：硬编码的 Actor 名称在同一 Ray 集群内可能导致意外重用，但作者解释 Verl 的作业隔离机制降低了此风险。
  5. 测试覆盖不足：由于改动针对 ROCm 特定运行时行为，无法在 CI 中无 AMD 硬件的情况下充分测试，可能隐藏平台特定 bug。
- 影响：
  1. 用户影响：使完全异步 FSDP2 训练能够在 AMD ROCm 平台（如 MI300X）上稳定运行，扩展了硬件支持范围。用户无需修改代码，所有改动为内部实现细节。
  2. 系统影响：提升了跨平台兼容性，ZMQ 句柄逻辑的改进也使多节点、多副本部署更健壮。环境变量和 JSON 序列化的修复是通用改进，对所有平台有益。
  3. 团队影响：为后续在 AMD 硬件上开展训练任务铺平道路，减少了平台特定的调试开销。
  - 风险标记：跨平台兼容性，核心通信路径变更，缺少硬件特定测试

关联脉络

- PR #5967 [fully_async] fix: Add Mindspeed Patch for Async Training on Ascend NPUs: 类似地，该 PR 修复了在特定硬件（Ascend NPU）上完全异步训练的问题，涉及环境变量和补丁应用，与本 PR 的 AMD 平台修复有相似之处。
- PR #6052 [fully_async] fix: avoid blocking ray.get inside async actor methods: 同属完全异步训练模块的 bugfix，关注异步通信和事件循环问题，与本 PR 的 ZMQ 通信改进相关。