PR #6243 完整报告

verl-project/verl

[rollout] feat: vLLM Prefill-Decode disaggregated rollout (NIXL + Mooncake wiring)

合并时间：2026-07-08 16:38

原文链接：http://prhub.com.cn/verl-project/verl/pull/6243

执行摘要

- 一句话：vLLM Prefill-Decode 分离 rollout，支持 NIXL 和 Mooncake
- 推荐动作：建议精读此 PR，特别是 ZMQ IPC 地址计算、PD role 分配逻辑和 KV Transfer Config 构建。设计上采用进程内 AsyncLLM（而非子进程 +HTTP）以适应现有 collective_rpc 机制，值得关注。值得注意的是，当前 PD 路径在典型负载下性能低于 colocated，需要明确告知用户预期。

功能与动机

在 vLLM 引擎上实现 Prefill-Decode 分离，为用户提供与 SGLang PD (#6117) 对等的功能。PR body 声明："vLLM counterpart to verl#6117... fills the previously-stubbed vLLMPDReplica so users can pick rollout.name=vllm, disaggregation.enabled=true and route GRPO rollouts across 1 prefill + N decode vLLM engines via NIXL or Mooncake KV transfer."

实现拆解

实现分为以下步骤：

1. 配置层：新增 DisaggregationConfig dataclass（verl/workers/config/disaggregation.py），定义预填 / 解码副本数、传输后端、TP 参数等，并在 RolloutConfig.__post_init__ (verl/workers/config/rollout.py) 中校验。
2. Replica 类解析：修改 verl/workers/rollout/replica.py 中的 get_rollout_replica_class，当 disaggregation.enabled=True 且 rollout=vllm 时返回 vLLMPDReplica（与 SGLang 分支并列）。
3. 核心 Replica 实现：新增 vLLMPDReplica 类（verl/workers/rollout/vllm_rollout/vllm_pd_replica.py），继承 vLLMReplica。重写 launch_servers 方法：通过 Ray 收集每个 worker 的节点 /GPU 信息，计算 GPU 分配，为每个 prefill/decode server 生成独立的 KV Transfer 配置，并异步启动 Ray actor（_spawn_pd_server）。
4. Server 端 PD 路由：修改 vLLMHttpServer（verl/workers/rollout/vllm_rollout/vllm_async_server.py）：构造函数接受 disaggregation_role 和 kv_transfer_config，将 kv_transfer_config 注入 vLLM 引擎参数；新增 set_pd_peer 方法（prefill server 端）注册解码 peers；新增 _select_decode_peer 方法（轮询）和 _pd_dispatch 方法（将一次请求从 prefill 调度至指定 decode server）。

5Traine侧适配：修改ServerAdapter（verl/workers/rollout/vllm_rollout/vllm_rollout.py）计算 PD 感知的 per-replica world size，为每个 trainer rank 分配 role（prefill/decode）和 server index，确保 ZMQ IPC 地址针对每个 actor 唯一（修复 #1）。

6. Bug 修复与测试：修复了 ZMQ 端口冲突、Mooncake 连接池不启用导致的端口耗尽（EADDRNOTAVAIL）和 TP>1 支持；新增 42 个单元测试（tests/workers/rollout/test_vllm_pd_disaggregation_on_cpu.py），覆盖配置验证、replica 类解析、kv_transfer_config 构建。

关键文件：

- verl/workers/rollout/vllm_rollout/vllm_pd_replica.py（模块 vLLM 引擎；类别 source；类型 core-logic；符号 vLLMPDReplica, init, launch_servers, _collect_cuda_devices）：新增 vLLMPDReplica 类，核心 PD 副本实现，负责 server 启动和 GPU 分配。
- verl/workers/rollout/vllm_rollout/vllm_async_server.py（模块 vLLM 引擎；类别 source；类型 core-logic；符号 set_pd_peer, _select_decode_peer, _pd_dispatch）：修改 vLLMHttpServer 以接受 PD 角色和 KV Transfer 配置，新增 set_pd_peer 和 dispatch 方法。
- tests/workers/rollout/test_vllm_pd_disaggregation_on_cpu.py（模块 测试；类别 test；类型 test-coverage；符号 test_disaggregation_defaults_disabled_and_valid, test_disaggregation_enabled_nixl_accepted, test_disaggregation_all_known_backends_pass_config_validation, test_disaggregation_unknown_backend_rejected）：新增 42 个 GPU-free 单元测试，覆盖配置验证、replica 类解析和 kv_transfer_config 构建。
- verl/workers/rollout/vllm_rollout/vllm_rollout.py（模块 vLLM 引擎；类别 source；类型 core-logic）：修改 ServerAdapter 以感知 PD 布局，计算 per-replica world size 并分配角色。
- verl/workers/rollout/replica.py（模块 Rollout 框架；类别 source；类型 dependency-wiring）：修改 get_rollout_replica_class 以支持 vllm 的 PD 分支。
- verl/workers/config/disaggregation.py（模块 配置；类别 source；类型 configuration）：新增 DisaggregationConfig dataclass，定义 PD 配置项和后端校验逻辑。

关键符号：vLLMPDReplica.init, vLLMPDReplica.launch_servers, vLLMPDReplica._build_kv_transfer_config, vLLMPDReplica._spawn_pd_server, vLLMHttpServer.set_pd_peer, vLLMHttpServer._select_decode_peer, vLLMHttpServer._pd_dispatch, ServerAdapter.init, get_rollout_replica_class

关键源码片段

verl/workers/rollout/vllm_rollout/vllm_pd_replica.py

新增 vLLMPDReplica 类，核心 PD 副本实现，负责 server 启动和 GPU 分配。

```
class vLLMPDReplica(vLLMReplica):
    def __init__(
        self,
        replica_rank: int,
```

```
        config: RolloutConfig,
        model_config: HFModelConfig,
        gpus_per_node: int = 8,
        is_reward_model: bool = False,
        is_teacher_model: bool = False,
        name_suffix: str = "",
    ):
        super().__init__(...)
        disagg = self.config.disaggregation
        assert disagg.enabled, "vLLMPDReplica requires rollout.disaggregation.enabled=True"

        # 当前版本仅支持 NIXL 和 Mooncake 传输后端
        if disagg.transfer_backend not in ("nixl", "mooncake"):
            raise NotImplementedError(
                f"vLLMPDReplica supports transfer_backend in ('nixl', 'mooncake'); "
                f"got {disagg.transfer_backend!r}."
            )
        if disagg.prefill_replicas != 1:
            raise NotImplementedError(f"prefill_replicas=1 only (got {disagg.prefill_replicas})")

        self._n_prefill = disagg.prefill_replicas
        self._n_decode = disagg.decode_replicas

        # 默认 decode TP 与 prefill TP 相同
        self._prefill_tp = self.config.tensor_model_parallel_size
        self._decode_tp = (
            disagg.decode_tensor_model_parallel_size
            if disagg.decode_tensor_model_parallel_size is not None
            else self._prefill_tp
        )

        # 计算 PD 所需 GPU 总数，并校验不超过节点可用的 GPU 数量
        pd_world_size = self._prefill_tp + self._n_decode * self._decode_tp
        if pd_world_size > gpus_per_node:
            raise NotImplementedError(
                f"PD replica needs {pd_world_size} GPUs but gpus_per_node={gpus_per_node}; "
                "single-node only in this revision."
            )
        if self.config.data_parallel_size != 1:
            raise NotImplementedError(f"data_parallel_size=1 only (got {self.config.data_parallel_
            size})")
        if self.config.pipeline_model_parallel_size != 1:
            raise NotImplementedError(...)

        self.world_size = pd_world_size
        self._prefill_servers: list[ActorHandle] = []
        self._decode_servers: list[ActorHandle] = []
```

verl/workers/rollout/vllm_rollout/vllm_async_server.py

修改 vLLMHttpServer 以接受 PD 角色和 KV Transfer 配置，新增 set_pd_peer 和 dispatch 方法。

```
    def _select_decode_peer(self) -> ActorHandle:
        # 轮询选择解码 peer（与 vllm-project/router 默认策略一致）
        idx = getattr(self, "_pd_peer_idx", 0)
        peer = self._pd_decode_peers[idx % len(self._pd_decode_peers)]
        self._pd_peer_idx = idx + 1
        return peer

    async def _pd_dispatch(
        self,
        prompt_ids: list[int],
        prompt_token_ids: ...
    ) -> TokenOutput:
        import uuid # 注意：此处违反 PEP-8，应在文件顶部导入
        # ... 构造 KVTransferParams，包含 sid（session ID）等
        params = {
            "sender_type": "prefill",
            "receiver_type": "decode",
            "enable_receivers": False,
            "sid": str(uuid.uuid4()),
        }
        decode_server = self._select_decode_peer()
        return await decode_server.generate.remote(
            prompt_ids,
            ...
            kv_transfer_params=params,
        )
```

评论区精华

Review 中讨论了几个关键问题：

- actor 名称前缀不一致：xiazhahe 指出 vllm_pd_replica.py 中启动的 server actor 使用 f"vllm_pd_server_{...}" 名称，而 vllm_rollout.py 中 _execute_method 使用的 _get_server_name_prefix() 可能不匹配，导致 ray.get_actor 失败。协作作者 yasumira 回应已修复。
- socket 端口竞争：gemini-code-assist 指出在 vllm_pd_replica.py 中，prefill_sock 在 actor 创建后才关闭，可能导致子进程绑定端口时发生竞争。建议尽早关闭。该问题为开放状态。
- 局部导入违反 PEP-8：gemini-code-assist 指出 _pd_dispatch 方法内部导入了 uuid 模块，应移至文件顶部。
  - actor 名称前缀不匹配导致 ray.get_actor 失败 (correctness): yasumira 确认已修复，匹配 actor 名称。
  - prefill_sock 端口竞争 (correctness): 评论后未见进一步修复，状态保持开放。
  - 局部导入 uuid 违反 PEP-8 (style): 未在后续讨论中回应，可能已被接受或修改。

风险与影响

- 风险：
  1. 核心路径变更风险：修改了 vLLMReplica 和 ServerAdapter 的初始化逻辑，可能影响普通 colocated 模式。需确保回归测试覆盖。
  2. 分布式兼容性：PD 模式当前仅支持单节点（single-node）、DP=1、PP=1，未来多节点扩展可能带来架构冲突。
  3. Mooncake 后端稳定性：Mooncake 长期运行下仍可能出现连接失败（类似 issue #23272），虽通过启用连接池缓解，但未彻底解决。
  4. 性能倒退误解：Bench 数据显示 PD 路径比 colocated 慢 10-18%，用户可能误以为此功能是性能倒退；需在文档中明确说明适用场景（大模型 / 高并发）。 - 影响：对用户：新增 vLLM PD 路径，需配置 disaggregation.enabled=true 和相应后端。单节点用户可尝试；多节点暂不支持。对系统：增加了 vLLMPDReplica、DisaggregationConfig 等新组件，模块间耦合加深（config→replica→server→adapter）。对团队：需同时维护 NIXL 和 Mooncake 两个传输后端，测试矩阵扩大；上线前需验证 Mooncake 稳定性。
- 风险标记：核心路径变更，分布式兼容性，Mooncake 稳定性，性能倒退风险

关联脉络

- PR #6117 [rollout] feat: SGLang Prefill-Decode disaggregated rollout: SGLang PD 的对应实现，PR 6243 基于其 API 和配置设计，填补 vLLM 侧 stub。
- PR #1749 [RFC]: Optimizing Mooncake TCP Transport for High-Concurrency Throughput: Mooncake 传输层的 RFC，PR 6243 在 Mooncake 集成中引用了该优化方向。
- PR #23272 [Bug] PD disaggregation + Mooncake: sustained load causes KV transfer failures: SGLang 社区报告的 Mooncake 长期运行问题，PR 6243 通过启用连接池缓解了类似问题。