

PR #2501 完整报告

radixark/miles

Start dynamic worker ports above Ray's worker range

合并时间: 2026-09-04 13:49

原文链接: <http://prhub.com.cn/radixark/miles/pull/2501>

执行摘要

- 一句话: 将动态端口分配起始值从 15000 调整至 20000, 并新增异常处理测试用例。
- 推荐动作: 此 PR 值得精读, 原因有二: 1) 它包含了一个重要的基础性调整 (端口范围), 虽然改动量小, 但影响核心路径; 2) 评审中的讨论提出了一个超越当前 PR 范围的、关于系统可配置性和兼容性的关键设计问题, 对理解架构演进方向有参考价值。建议在合并后, 后续跟进评审员提出的“将 Miles 端口范围设为可配置并验证其与 Ray 范围不重叠”的建议。

功能与动机

虽然没有明确的 Issue, 但变更的核心动机源于解决一个潜在的端口冲突风险: Ray 的 worker 进程默认使用 10002-19999 范围的端口。原先动态分配器从 15000 开始, 可能落在或接近该范围的上端, 存在竞争条件。将其调整至 20000 并提取为命名常量, 是为了提供更清晰、更安全的默认行为, 并为进一步的可配置性打下基础。Review 评论也直接指出了此变更对兼容外部 Ray 集群 (如 Slurm 部署) 的潜在影响。

实现拆解

1. 核心配置调整: 在 `miles/utils/workers/addr_allocator.py` 中, 新增模块级常量 `_DYNAMIC_PORT_START = 20000`, 并将 `PortAllocator.alloc` 方法中获取默认起始端口的硬编码值 15000 替换为该常量。此举提升了代码可读性, 并将端口范围决策集中管理。
2. 测试夹具增强: 在 `tests/fast/utils/workers/conftest.py` 中新增 `patch_ray_get_failure` 测试夹具。该夹具通过 `monkeypatch` 将 `ray.get` 替换为一个总是抛出 `RuntimeError` 的函数, 用于模拟端口探测请求提交成功但在结果检索阶段失败的场景。
3. 测试用例扩展: 在 `tests/fast/utils/workers/test_addr_allocator.py` 中进行三项关键补充: a. 将验证未见节点默认起始端口的断言从 15000 更新为 20000, 以匹配源码变更。 b. 新增 `test_a_skipped_candidate_range_advances_from_the_returned_block` 测试, 验证当端口块被占用时, 游标能正确地从实际分配到的端口加上块大小开始推进。 c. 新增两个使用 `patch_ray_get_failure` 夹具的测试, 分别验证: `ray.get` 失败时错误被正确抛出, 且节点游标状态不变; 对于从未见过的节点, 探测失败不会为其创建游标记录。这些测试覆盖了资源分配器的关键异常处理路径。

关键文件:

- `miles/utils/workers/addr_allocator.py` (模块 端口分配器; 类别 source; 类型 core-logic; 符号 `_DYNAMIC_PORT_START, alloc`): 核心源码变更, 定义了动态端口分配的起始常

量，是本次调整的入口。

- tests/fast/utils/workers/test_addr_allocator.py (模块 端口分配器; 类别 test; 类型 test-coverage; 符号 test_a_skipped_candidate_range_advances_from_the_returned_block, test_a_failed_probe_does_not_advance_the_node_cursor, test_a_failed_probe_does_not_create_a_cursor_for_an_unseen_node, test_alloc_starts_from_the_base_port_on_an_unseen_node) : 核心测试文件, 新增并修改了多个测试用例以覆盖新的默认值和关键的异常处理路径。
- tests/fast/utils/workers/conftest.py (模块 测试夹具; 类别 test; 类型 test-coverage; 符号 patch_ray_get_failure, _raise) : 新增了用于模拟 ray.get 调用失败的测试夹具, 是实现异常路径测试覆盖的基础设施。

关键符号: alloc

关键源码片段

miles/utils/workers/addr_allocator.py

核心源码变更, 定义了动态端口分配的起始常量, 是本次调整的入口。

```
import logging
from dataclasses import dataclass, field

import ray

logger = logging.getLogger(__name__)

# 将动态端口起始值定义为模块级常量, 便于集中管理和未来可配置化。
# 调整至此值旨在避开 Ray 自身 worker 进程的默认端口范围 (通常为 10002-19999) 。
_DYNAMIC_PORT_START = 20000

@dataclass
class PortAllocator:
    _next_port_of_ip: dict[str, int] = field(default_factory=dict)

    def alloc(self, actor, *, node_ip: str, consecutive: int = 1) -> int:
        # use small ports to prevent ephemeral port between 32768 and 65535.
        # also, ray uses port 10002-19999, thus we avoid near-10002 to avoid racing condition
        # 注意: 注释中提到的 Ray 端口范围可能已过时, 需结合实际集群配置验证。
        start_port = self._next_port_of_ip.get(node_ip, _DYNAMIC_PORT_START)
        port: int = ray.get(
            actor._get_free_port_block.remote(
                start_port=start_port,
                count=consecutive,
            )
        )
        self._next_port_of_ip[node_ip] = port + consecutive
        return port
```

评论区精华

评审员 guapisolo 指出了一个关键设计风险：硬编码的 20000 起始值假设了 Ray 的默认 worker 端口范围。然而，Ray 的 Slurm 集成示例使用的端口范围是 20002-29999，这意味着 Miles 的动态端口起点可能与 Ray 的范围重叠。在并发启动时，两者可能同时选中相同端口，导致 rollout 单元失败。评审员建议：

"Please make the Miles range configurable and validate it is disjoint from the external Ray min/max/list..."

这个评论将此次简单调整提升到了系统兼容性和配置健壮性的层面，指出当前实现仍然存在一个未解决的设计缺陷。

- 硬编码端口范围与外部 Ray 集群兼容性 (design): 提出了一个超越当前 PR 范围的设计问题：Miles 的动态端口范围应当可配置以适配不同的 Ray 部署环境。此问题在 PR 中未解决。

风险与影响

- 风险：主要风险（来自评审）：硬编码的 `_DYNAMIC_PORT_START = 20000` 与可变的外部 Ray 集群配置（如 Slurm 模式下的 20002-29999）存在端口冲突风险。如果 Miles 和 Ray 的端口范围重叠，且两者同时探测同一端口，会导致绑定失败，影响 rollout 进程启动。
次要风险：起始端口的数值变更（从 15000 到 20000）是一个潜在的破坏性配置变更，需要确保所有依赖此默认值的部署和测试都已适配。测试覆盖降低的风险：新增的测试用例显著提升了对异常路径和边界情况的覆盖，特别是 `ray.get` 失败时的状态一致性检查，这有助于降低未来相关变更引入回归 bug 的风险。
- 影响：对系统：此变更影响所有使用 `PortAllocator` 进行动态端口分配的组件，主要是 worker 提供和 rollout 模块。调整了默认端口范围，旨在减少与 Ray 基础设施端口冲突的可能性。对团队：这是一个相对小范围但重要的维护性变更。它通过提取常量改善了代码清晰度，并通过增强测试提升了该模块的可靠性。评审讨论揭示了需要进一步讨论的架构决策（端口范围可配置化），可能引发后续的优化 PR。对用户：对于使用默认配置的用户，端口分配行为将发生变化（起点更高），理论上能提高与 Ray 进程的共存安全性。对于使用自定义 Ray 部署（如 Slurm）的用户，此变更可能引入或暴露端口冲突问题，需要关注。
- 风险标记：配置硬编码，潜在端口冲突

关联脉络

- PR #2049 Add a worker provider abstraction over RayWorkerManager: 两者都涉及 Miles 的 worker 管理和地址分配机制。PR#2049 引入了工作者提供者抽象层，可能与 PR#2501 改进的 PortAllocator 存在功能协作关系。
- PR #2050 Stop checking actor liveness while waiting for a tcp port: 两者都涉及 worker 端口就绪探测的逻辑。PR#2050 简化了探测接口，而 PR#2501 调整了探测的端口范围起始值。