PR #43864 完整报告

vllm-project/vllm

[Bugfix] Exclude Ray DP from #42585's deferred port allocation

合并时间：2026-05-28 23:55

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/43864

执行摘要

- 一句话：修复 Ray DP 多 API-server 场景因端口分配导致挂起的 bug
- 推荐动作：值得精读以理解不同后端对地址分配机制的限制。设计清晰：后续若增加新后端，需类似评估其对延迟分配的兼容性。测试设计精巧，避免 GPU 依赖，可重复运行。

功能与动机

PR #42585 引入的内核分配端口机制未对 Ray DP 后端做隔离。Ray DP 的 actor 通过 pickle 接收地址，且 _perform_handshakes 形同虚设，导致 driver 通过 gather_actual_addresses 报告的真实地址无法到达 actor，actor 一直连接端口 0 而挂起。该问题由 Nemotron 团队在 TP1DP8 和 TP1DP16 部署中发现并内部报告。

实现拆解

1. 判断 Ray DP 后端：在 vllm/entrypoints/cli/serve.py 中通过 parallel_config.data_parallel_backend == "ray" 获得 is_ray_dp 变量。
2. 修改 defer_api_server_ports：将原来 not rust_frontend_path 改为 not (rust_frontend_path or is_ray_dp)，让 Ray DP 和 Rust 前端一样使用 driver 侧预分配。
3. 条件化 gather_actual_addresses：只有 not is_ray_dp 时才调用 gather_actual_addresses 并更新 addresses，因为 Ray DP 的 actor 启动时已持有预分配地址，无需后期修正。
4. 新增回归测试：在 tests/v1/engine/test_core_engine_actor_manager.py 中添加 test_ray_dp_addresses_resolved_before_actor_creation、_bind_and_report_worker、_make_vllm_config_ray_dp_multinode 等辅助，使用 Ray local_mode 和 CPU-only placement groups 模拟多节点环境，验证 driver 端口预分配以及 actor 侧地址快照均为真实端口。
5. 测试覆盖确认：CI 运行测试，保证当前 main 上失败，修复后通过。

关键文件：

- vllm/entrypoints/cli/serve.py（模块 入口服务；类别 source；类型 core-logic）：修正延迟端口分配逻辑，添加 Ray DP 后端隔离
- tests/v1/engine/test_core_engine_actor_manager.py（模块 引擎测试；类别 test；类型 test-coverage；符号 get_addresses, _bind_and_report_worker, ray_context_dp2, _make_vllm_config_ray_dp_multinode）：新增端到端回归测试，验证 Ray DP 地址预分配

正确性

关键符号：test_ray_dp_addresses_resolved_before_actor_creation, _bind_and_report_worker, _make_vllm_config_ray_dp_multinode, ray_context_dp2, get_addresses, create_dp_placement_groups

关键源码片段

vllm/entrypoints/cli/serve.py

修正延迟端口分配逻辑，添加 Ray DP 后端隔离

```
# vllm/entrypoints/cli/serve.py (partial)
from vllm.v1.engine.utils import get_engine_zmq_addresses

# Defer port allocation to the child's bind() to avoid TOCTOU, except
# for Rust front-end and Ray DP, which can't see the post-bind rebind
# (CLI-arg subprocess / pickled-into-actor snapshot respectively) and
# so pre-allocate driver-side -- reintroducing the original race only
# there.
is_ray_dp = parallel_config.data_parallel_backend == "ray"
addresses = get_engine_zmq_addresses(
    vllm_config,
    num_api_servers,
    defer_api_server_ports=not (rust_frontend_path or is_ray_dp),
)

with launch_core_engines(...) as ...:
    ...
    if not is_ray_dp:
        # Forward each child's bound endpoints to the engine handshake
        # (runs on ``with`` exit). Skipped for Ray DP, where addresses
        # are pre-allocated above and Ray actors already hold them.
        actual_inputs, actual_outputs = (
            api_server_manager.gather_actual_addresses()
        )
        addresses.inputs = actual_inputs
        addresses.outputs = actual_outputs
```

评论区精华

无 review 评论。PR 作者在 body 中说明了 bug 来源（#42585 未对 Ray DP 做隔离），且由内部 Nemotron 团队报告。njhill 批准合并，无额外讨论。

- 暂无高价值评论线程

风险与影响

- 风险：Ray DP 路径回退到 driver 侧预分配端口，重新引入 #42585 试图消除的 TOCTOU 竞争（driver 探测端口和子进程绑定之间的窗口）。但该竞争仅在 Ray DP 多节点多

API-server 场景下存在，且原有行为即是如此，并非新增风险。单节点和多进程 DP 继续受益于 #42585 的修复。对 serve.py 的控制流改动影响所有多 API-server 启动路径，但逻辑简单，回归概率低。测试覆盖了 Ray DP 路径，但缺少对 Rust 前端（也是预分配路径）的显式回归测试。

- 影响：修复 Ray DP 后端 num_api_servers > 1 多节点部署的确定性挂起问题。无副作用。测试在 CI 中运行，但需要 Ray 依赖，可能不在所有 CI 环境下执行。
- 风险标记：Ray DP 路径重新引入 TOCTOU 竞争

关联脉络

- PR #42585 Kernel-assigned-at-bind ports for API-server sockets: 引入延迟端口分配机制，本 PR 修复其对 Ray DP 的未隔离问题。