

PR #52367 完整报告

vllm-project/vllm

[CI/Build] Use file rendezvous for UniProc loader fixtures

合并时间: 2026-08-29 11:53

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/52367>

执行摘要

- 一句话: 测试夹具改用文件 rendezvous, 消除端口竞争
- 推荐动作: 该 PR 是一个小而聚焦的改进, 建议精读以了解测试基础设施中避免端口竞争的模式。值得关注的设计决策是使用文件存储替代 TCP rendezvous, 这符合 issue 的原则。

功能与动机

该 PR 是关联 issue #51275 的一部分, 旨在解决测试基础设施中的端口竞争问题。issue 中明确了规则: 'the process that binds chooses the port at bind time, and publishes the actual endpoint over an existing control path'。PR body 指出这两个夹具以 rank 0 和 world size 1 运行, 故不需要网络 TCPStore, 使用文件存储可消除不必要的端口竞争。这是对 issue 中 Category I (probe-then-bind) 问题的具体修复, 与 #50999 的单节点 uniproc/mp (将 TCP rendezvous 替换为 file://) 方向一致。

实现拆解

1. 修改 tests/model_executor/model_loader/runai_streamer_loader/conftest.py: 将导入从 get_distributed_init_method(get_ip(), get_open_port()) 改为 get_file_store_init_method(), 并更新 RunaiDummyExecutor._init_executor() 中的 distributed_init_method 赋值。
2. 修改 tests/model_executor/model_loader/tensorizer_loader/conftest.py: 同样将 DummyExecutor._init_executor() 中的导入和赋值替换为 get_file_store_init_method()。
3. 未修改生产代码, 仅测试夹具。
4. 验证: 运行 applicable pre-commit hooks, 并通过 32 种唯一文件存储初始化方法在 CPU/Gloo 进程组上完成了渲染测试。

关键文件:

- tests/model_executor/model_loader/runai_streamer_loader/conftest.py (模块 测试配置; 类别 test; 类型 test-coverage): 将 RunaiDummyExecutor 的 TCP rendezvous 替换为文件存储
- tests/model_executor/model_loader/tensorizer_loader/conftest.py (模块 测试配置; 类别 test; 类型 test-coverage): 将 DummyExecutor 的 TCP rendezvous 替换为文件存储

关键符号: `get_file_store_init_method`, `RunaiDummyExecutor._init_executor`,
`DummyExecutor._init_executor`

关键源码片段

`tests/model_executor/model_loader/runai_streamer_loader/conftest.py`

将 `RunaiDummyExecutor` 的 TCP rendezvous 替换为文件存储

```
# tests/model_executor/model_loader/runai_streamer_loader/conftest.py
from vllm.utils.network_utils import get_file_store_init_method
from vllm.v1.executor import UniProcExecutor
from vllm.v1.worker.worker_base import WorkerWrapperBase
```

```
# 该 dummy executor 用于测试 RunAI streamer,
# 不能使用 vllm_runner fixture (会派生 worker 进程, patch 不生效)
```

```
class RunaiDummyExecutor(UniProcExecutor):
```

```
    def _init_executor(self) -> None:
```

```
        # 单进程 (rank 0 / world size 1) 无需 TCP rendezvous,
        # 改用文件存储避免端口竞争
        distributed_init_method = get_file_store_init_method()
```

```
        local_rank = 0
```

```
        rank = 0
```

```
        is_driver_worker = True
```

```
        device_info = self.vllm_config.device_config.device.__str__().split(":")
```

```
        if len(device_info) > 1:
```

```
            local_rank = int(device_info[1])
```

```
        worker_rpc_kwargs = dict(
```

```
            vllm_config=self.vllm_config,
```

```
            local_rank=local_rank,
```

```
            rank=rank,
```

```
            distributed_init_method=distributed_init_method,
```

```
            is_driver_worker=is_driver_worker,
```

```
        )
```

```
        self.driver_worker = WorkerWrapperBase()
```

```
        self.collective_rpc("init_worker", args=([worker_rpc_kwargs],))
```

```
        self.collective_rpc("init_device")
```

`tests/model_executor/model_loader/tensorizer_loader/conftest.py`

将 `DummyExecutor` 的 TCP rendezvous 替换为文件存储

```
# tests/model_executor/model_loader/tensorizer_loader/conftest.py
from vllm.utils.network_utils import get_file_store_init_method
from vllm.v1.executor import UniProcExecutor
from vllm.v1.worker.worker_base import WorkerWrapperBase
```

```

class DummyExecutor(UniProcExecutor):
    def _init_executor(self) -> None:
        """Initialize the worker and load the model."""
        self.driver_worker = WorkerWrapperBase(rpc_rank=0)
        # 单进程场景同样改用文件存储，避免端口竞争
        distributed_init_method = get_file_store_init_method()
        local_rank = 0
        # 若指定设备索引，则覆盖 local rank
        device_info = self.vllm_config.device_config.device.__str__().split(":")
        if len(device_info) > 1:
            local_rank = int(device_info[1])
        rank = 0
        is_driver_worker = True
        kwargs = dict(
            vllm_config=self.vllm_config,
            local_rank=local_rank,
            rank=rank,
            distributed_init_method=distributed_init_method,
            is_driver_worker=is_driver_worker,
        )
        self.mm_receiver_cache = None
        self.collective_rpc("init_worker", args=([kwargs],))
        self.collective_rpc("init_device")

    def shutdown(self):
        if hasattr(self, "thread_pool"):
            self.thread_pool.shutdown(wait=False)

```

评论区精华

review 中主要讨论了 CI 覆盖情况。aoshen02 询问 'Could you confirm the CI does cover these tests?', yu-xin-c 回应确认两个文件均在测试套件中，属于 model_executor 作业，并提供了 pytest 命令。另一次讨论涉及 Buildkite CI 状态，yu-xin-c 指出 CI 保持绿色，包括 NVIDIA model-executor 作业和 AMD MI300 镜像。最终 aoshen02 和 jeejeelee 均批准了该 PR。

- CI 覆盖确认 (question): 确认覆盖，CI 运行通过。
- CI 状态跟进 (other): CI 通过，可以合并。

风险与影响

- 风险：风险极低：仅修改测试夹具，不影响生产代码。文件存储使用临时文件，可能对运行环境的文件系统有依赖，但该变更已在 CI 中验证。潜在风险是文件存储可能导致测试运行速度稍慢或文件清理问题，但鉴于 fixture 规模，影响可忽略。
- 影响：影响范围限于测试基础设施，消除测试中的端口竞争，提高测试稳定性。对用户无影响，对团队而言，此变更与 #51275 的长期目标一致。
- 风险标记：测试基础设施改动，低风险

关联脉络

- PR #51275 [RFC]: Race-free port management: pick ports at bind time, publish over an existing channel: 本 PR 直接实现该 RFC 中 Category I 的修复方向。
- PR #50999 [RFC]: Replace TCP rendezvous with file:// for single-node uniproc/mp: 推测与本 PR 同为系列改动，但由于材料未提供具体信息，此关联基于上下文推断。