

PR #41423 完整报告

vllm-project/vllm

[Bugfix] Fix spawn_new_process_for_each_test silently swallowing test failures

合并时间: 2026-05-06 23:17

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/41423>

执行摘要

- 一句话: 修复 spawn 测试装饰器静默吞掉测试失败
- 推荐动作: 该 PR 是重要的测试基础设施修复, 值得合并。但必须确保与 XPU/ROCm 平台兼容, 需 review follow-up PR #41895。作者已积极解决反馈。建议在合并前对受影响的测试在 CI 中充分运行。

功能与动机

Issue #41415 指出 `spawn_new_process_for_each_test` 装饰器始终通过, 不管测试函数是否失败, 导致测试覆盖率虚高, 潜在 bug 被隐藏。该问题由 Claude 在 PR #36823 评论中首次发现。

实现拆解

1. 重写 `spawn_new_process_for_each_test` 装饰器 (`tests/utils.py`): 从基于 `python -m` 模块执行改为使用 `cloudpickle` 序列化测试函数及其参数, 并通过 `subprocess` 执行内联脚本 (`python -c`) 直接调用函数。子进程捕获 `Skipped` 后正常退出, 其他异常写入 `traceback` 文件并退出码 1; 父进程读取文件后重新抛出 `RuntimeError`。
2. 重构 `TestCudagraphIntegration` 测试类 (`tests/v1/cudagraph/test_cudagraph_dispatch.py`): 由于 `self` 不可序列化, 将 `setup_method`、`_run_and_monitor_call` 提取为模块级函数, `test_capture_replay_bypass_logic` 改为模块级函数, 使用正确的 `cudagraph_capture_sizes=[1,2]` 匹配 token 数。
3. 重构 `test_custom_logitsprocs` (`tests/v1/logits_processors/test_custom_online.py`): 原 `async` 测试依赖不可序列化的 `AsyncOpenAI` client, 改为同步函数, 在函数体内创建 `async` client 并调用 `asyncio.run()`。
4. 新增测试覆盖 (`tests/utils/test_spawn_decorator.py`): 测试正常通过、失败捕获、skip、参数化四种场景。
5. 修复上游不可序列化问题: 识别并修复 `TestCudagraphIntegration` 和 `test_custom_logitsprocs` 中的 `unpicklable` 参数, 确保新装饰器不会抛出 `PicklingError`。

关键文件:

- `tests/utils.py` (模块 工具函数; 类别 `test`; 类型 `test-coverage`; 符号 `spawn_new_process_for_each_test`): 核心实现文件, 重写了 `spawn_new_process_for_each_test` 函数, 解决 bug

- tests/v1/cudagraph/test_cudagraph_dispatch.py (模块 CUDA 图; 类别 test; 类型 test-coverage; 符号 TestCudagraphIntegration, setup_method, _run_and_monitor_call, test_capture_replay_bypass_logic) : 受影响的测试文件, 将 TestCudagraphIntegration 类重构为独立函数以支持 pickle
- tests/utils_/test_spawn_decorator.py (模块 测试装饰器; 类别 test; 类型 test-coverage; 符号 test_spawn_decorator_passing, test_spawn_decorator_failure_is_caught, test_spawn_decorator_skip, test_spawn_decorator_parametrized) : 新增测试文件, 验证装饰器在不同场景下正确工作
- tests/v1/logits_processors/test_custom_online.py (模块 日志处理器; 类别 test; 类型 test-coverage; 符号 test_custom_logitsprocs, _async_main, _run) : 受影响的测试, 因为 async client 不可序列化, 重构为同步并在内部创建 client

关键符号: spawn_new_process_for_each_test, _run_and_monitor_call, test_capture_replay_bypass_logic, test_custom_logitsprocs, test_spawn_decorator_passing, test_spawn_decorator_failure_is_caught, test_spawn_decorator_skip, test_spawn_decorator_parametrized

关键源码片段

tests/utils.py

核心实现文件, 重写了 spawn_new_process_for_each_test 函数, 解决 bug

tests/utils.py: 重写后的 spawn_new_process_for_each_test 装饰器

```
def spawn_new_process_for_each_test(f: Callable[_P, None]) -> Callable[_P, None]:
    """Decorator: 每次测试函数调用都在一个新子进程中运行。
```

使用 cloudpickle 序列化函数, 子进程通过 stdin 接收, 执行后将 traceback 写入临时文件; 父进程读取后重新抛出异常, 避免静默吞没。

"""

```
@functools.wraps(f)
```

```
def wrapper(*args: _P.args, **kwargs: _P.kwargs) -> None:
```

```
    # 创建临时 traceback 文件
```

```
    with tempfile.NamedTemporaryFile(delete=False, suffix=".tb", mode="wb") as tmp:
```

```
        tb_file = tmp.name
```

```
    try:
```

```
        payload = cloudpickle.dumps((f, args, kwargs, tb_file))
```

```
        # 子进程内联脚本: 反序列化、执行函数、捕获异常
```

```
        child_script = (
```

```
            "import sys, cloudpickle, traceback\n"
```

```
            "try:\n"
```

```
            "    from _pytest.outcomes import Skipped\n"
```

```
            "except ImportError:\n"
```

```
            "    class Skipped(BaseException): pass\n"
```

```
            "f, args, kwargs, tb_file = "
```

```
            "cloudpickle.loads(sys.stdin.buffer.read())\n"
```

```
            "try:\n"
```

```

        "    f(*args, **kwargs)\n"
        "except Skipped:\n"
        "    sys.exit(0)\n"
        "except BaseException:\n"
        "    open(tb_file, 'w').write(traceback.format_exc())\n"
        "    sys.exit(1)\n"
    )
    repo_root = str(VLLM_PATH.resolve())
    env = os.environ.copy()
    env["PYTHONPATH"] = repo_root + os.pathsep + env.get("PYTHONPATH", "")
    result = subprocess.run(
        [sys.executable, "-c", child_script],
        input=payload, capture_output=True, env=env,
    )
    if result.returncode != 0:
        tb = ""
        if os.path.exists(tb_file) and os.path.getsize(tb_file) > 0:
            with open(tb_file) as fp:
                tb = fp.read()
        else:
            tb = result.stderr.decode()
        raise RuntimeError(
            f"Test {f.__name__} failed (exit code {result.returncode}):\n{tb}"
        )
    finally:
        if os.path.exists(tb_file):
            os.unlink(tb_file)
    return wrapper

```

评论区精华

多个 reviewer 参与讨论：

- gemini-code-assist指出旧函数体残留在模块级导致 IndentationError（已修复），以及子进程未处理 Skipped 异常（已修复）。
- ProExpertProg要求增加 skip 和 parametrize 场景的测试，并确保装饰器直接用在 test_* 函数上（已添加）。
- jikunshang指出移除 mp.set_start_method('spawn') 后 XPU/ROCm 平台因 fork 限制出现 RuntimeError，确认通过 #41895 恢复。
- ProExpertProg最后说 "Nice work! Now let's see what this actually means for the tests that weren't running..."，认可修复并期待后续暴露的失败。
- dangling old function body 导致 IndentationError (correctness): 作者确认并修复
- 缺少对 pytest.skip 的处理 (correctness): 作者修复：在 child_script 中先捕获 Skipped 并 exit(0)
- 测试 function 装饰位置和 import 导入 (design): 作者修正为导入实际装饰器并直接装饰 test_* 函数

- 移除 `mp.set_start_method('spawn')` 对 XPU/ROCm 的影响 (correctness): 作者通过 follow-up PR #41895 恢复 `mp.set_start_method('spawn')` 以维持兼容性
- 暴露的隐藏测试失败: unpicklable 参数 (correctness): 作者将类方法改为函数, 将 `async client` 改为同步内部创建, 解决序列化问题

风险与影响

- 风险:
 - 跨平台兼容风险: 移除 `mp.set_start_method('spawn')` 使用 `subprocess`, 在 XPU/ROCm 上出现 `Cannot re-initialize XPU in forked subprocess`。已通过 #41895 恢复。
 - 序列化依赖风险: 新装饰器要求函数和参数可序列化。之前因装饰器静默通过, 部分测试使用了不可序列化对象 (如 `AsyncOpenAI client`、`self`) , 现在会暴露为 `PicklingError`。作者已修复两个主要案例, 但其他使用者需注意。
 - 暴露隐式失败: 之前静默通过的测试现在真实失败, 可能导致 CI 短期不稳定, 但这是期望的改进。需逐个甄别是真正问题还是平台兼容问题。
 - 影响范围: 所有使用 `@spawn_new_process_for_each_test` 和 `@create_new_process_for_each_test('spawn')` 的测试, 包括 CUDA Graph 集成测试和自定义 logits processor 测试。
 - 影响: 正面: 测试基础设施可靠性大幅提升, CI 失败可追溯, 之前隐藏的 bug 将被暴露。负面: 短期可能因暴露的失败导致 CI 红色, 需要投入修复。对团队开发流程影响: 装饰器使用者需确保函数和参数可序列化; XPU/ROCm 平台需要额外兼容措施。
 - 风险标记: 跨平台兼容风险, 依赖序列化能力, 隐藏测试失败暴露, 测试基础设施变更

关联脉络

- PR #41415 [bug] `spawn_new_process_for_each_test` decorator broken: 该 issue 是当前 PR 的直接动机, 报告了装饰器永远通过的 bug。
- PR #41895 [Bugfix] Restore `mp.set_start_method('spawn')` for XPU/ROCm compatibility: 为了解决本 PR 引入的 XPU/ROCm 兼容问题而提的 follow-up PR, 将恢复 `spawn start method`。