PR #41573 完整报告

vllm-project/vllm

[ROCm][CI] Stabilize ROCm shutdown and distributed compile CI

合并时间：2026-05-10 11:47

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41573

执行摘要

- 一句话：稳定 ROCm 关闭流程和分布式编译 CI
- 推荐动作：此 PR 是维护性修复，值得快速合并。对于 ROCm 开发者，建议关注关闭测试中的超时断言是否仍然导致不稳定，并可考虑采纳 gemini-code-assist[bot] 的建议加入缓冲。无需深入精读。

功能与动机

此 PR 解决了两个特定的夜间 CI 故障：mi355_1: Entrypoints Integration (API Server openai - Part 2) 和 mi300_2: Distributed Compile Unit Tests (2xH100-2xMI300)。在 PR 描述中，作者指出需要稳定 ROCm 环境下的关闭流程，因为进程退出在 HIP/RCCL/ 原生扩展拆卸时可能花费额外时间；同时需要纠正 AMD 编译作业中的测试路径和排除不兼容的 CUDA 测试。

实现拆解

1. 测试关闭逻辑重构：在 tests/entrypoints/openai/completion/test_shutdown.py 的 test_abort_timeout_exits_quickly 函数中，将原有的忙等轮询循环（for _ in range(20): if proc.poll()...）替换为 proc.wait(timeout=max_exit_time)。这个 proc.wait 是阻塞调用，会等待进程终止或超时，更加可靠且避免了忙等。
2. 引入 ROCm 特定的超时阈值：通过 _IS_ROCM 标志对退出超时时间进行条件判断，设为 4.0 秒（ROCm）或 2.1 秒（非 ROCm）。注释说明 ROCm 下进程退出后，HIP/RCCL/ 原生扩展拆卸可能花费额外时间，因此需要更长的容忍时间。
3. 调整 AMD CI 编译流水线：在 .buildkite/test-amd.yaml 中，移除了 Distributed Compile Unit Tests 步骤中的 test_sequence_parallelism.py（仅 CUDA 测试），并将 test_tp2_ar_rms.py 的路径从 tests/compile/passes/distributed/ 更正为 tests/compile/fusions_e2e/。同时，将 tests/compile/fusions_e2e/ 添加到了 source_file_dependencies 中，以确保路径变更后触发条件正确。

关键文件：

- tests/entrypoints/openai/completion/test_shutdown.py（模块 测试；类别 test；类型 test-coverage；符号 test_abort_timeout_exits_quickly）：重构了关闭测试逻辑，用 proc.wait 替代轮询循环，并引入 ROCm 特定超时时间（4.0 秒），是 PR 的主要改动点。

- .buildkite/test-amd.yaml（模块 构建配置；类别 config；类型 configuration）：调整了 AMD CI 编译作业的测试命令和依赖路径，移除不兼容的 CUDA 测试，并修正 tp2_ar_rms 路径。

关键符号：test_abort_timeout_exits_quickly

关键源码片段

tests/entrypoints/openai/completion/test_shutdown.py

重构了关闭测试逻辑，用 proc.wait 替代轮询循环，并引入 ROCm 特定超时时间（4.0 秒），是 PR 的主要改动点。

```
# tests/entrypoints/openai/completion/test_shutdown.py

# 替换原有的轮询循环，使用 proc.wait 等待进程退出
# 针对 ROCm 平台，最大退出时间设为 4.0 秒（容纳 HIP/RCCL 拆卸开销）
max_exit_time = 4.0 if _IS_ROCM else 2.1

try:
    # 阻塞等待进程退出，最多等 max_exit_time 秒
    proc.wait(timeout=max_exit_time)
except subprocess.TimeoutExpired:
    # 超时后强制杀死进程并等待完成
    proc.kill()
    proc.wait(timeout=5)
    pytest.fail("Process did not exit after SIGTERM with abort timeout")

exit_time = time.time() - start_time
# 断言退出时间小于阈值，防止某些情况下计时误差导致不稳定
assert exit_time < max_exit_time, (
    f"Default shutdown took too long: {exit_time:.1f}s"
)
```

评论区精华

1. 关于轮询循环历史：评审者 DarkLight1337 询问为什么原来使用循环而不是 proc.wait，njhill 表示不确定，但认为改用 proc.wait 没问题。这表明原始实现可能没有特定原因，采用新方案是合理的。
2. 关于超时断言：gemini-code-assist[bot] 提出断言 assert exit_time < max_exit_time 可能因 Python 解释器或 OS 调度的轻微开销而导致不稳定的失败，建议在断言中加入 0.5 秒的缓冲。这是一个有效的工程建议，但目前尚未采纳。

- 使用 proc.wait 替代轮询循环的合理性 (design): 一致认为 proc.wait 更合适，无反对意见。
- 超时断言潜在的时序不稳问题 (correction): 未采纳，但该建议合理，可作为后续改进。

风险与影响

- 风险：

1. 回归风险（低）：关闭测试的更改主要影响 ROCm 平台，非 ROCm 平台的超时时间仍为 2.1 秒，且使用了标准的 proc.wait，回归概率低。
2. 稳定性风险（低）：如评审所提，断言 *exit_time* < max_exit_time 可能在 ROCm 环境下因计时精度问题偶尔失败，但 max_exit_time 本身已考虑了额外消耗，实际风险较小。建议加入 0.5 秒缓冲作为后续改进。
3. CI 覆盖风险：移除了 test_sequence_parallelism.py（仅 CUDA），AMD 编译测试中不再覆盖此路径，但该测试本身不适用于 ROCm，因此是合理变更。 - 影响：影响范围：仅影响 ROCm 平台的 CI 流水线和关闭测试，不涉及任何核心服务逻辑或用户功能。影响程度：低。修复了两个夜间故障，提高了 ROCm CI 的稳定性和可靠性，同时减少了因不兼容测试导致的误报。 - 风险标记：测试计时可能不稳定

关联脉络

- 暂无明显关联 PR