

PR #50504 完整报告

vllm-project/vllm

[CI][Fix] Resolved the Ascend NPU test build image fail and add file dependencies

合并时间: 2026-09-02 16:14

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/50504>

执行摘要

- 一句话: 修复 Ascend NPU CI 构建, 优化依赖安装并智能跳过无关测试触发。
- 推荐动作: 此 PR 是一次重要的 CI 基础设施修复与优化, 值得 NPU 相关开发者和 CI 维护者精读。关键的设计决策在于使用 `source_file_dependencies` 实现精细化的测试触发控制, 这平衡了测试覆盖率与 CI 效率。主要的改进空间在于采纳关于固定 uv 版本的安全建议, 以及维护好那份庞大的排除列表。

功能与动机

解决当前 CI 构建失败。根据 PR body 和错误日志, 问题在于安装 `vllm-ascend/requirements.txt` 时, `triton-ascend` 包在默认 PyPI 源中找不到, 导致 Docker 构建失败。PR 的目标是恢复 NPU 测试的绿色状态, 并通过优化减少未来的 CI 耗时和资源浪费。

实现拆解

1. 构建脚本核心修复 (`.buildkite/scripts/hardware_ci/run-npu-test.sh`): 将 Dockerfile 中所有 `pip` 命令替换为 `uv pip`, 并配置 `UV_EXTRA_INDEX_URL` 指向华为云 PyPI 镜像。对于安装 `vllm-ascend` 依赖, 显式使用 `--extra-index-url` 参数。这解决了 `triton-ascend` 包的安装失败问题。同时, `uv` 的使用旨在提升包安装速度。
2. 测试触发条件优化 (`.buildkite/hardware_tests/ascend_npu.yaml`): 为 “Ascend NPU Test” 步骤添加了 `source_file_dependencies` 配置。该配置采用 “包含所有 (`vllm/`) 但排除大量无关路径 (如 `!vllm/platforms/cuda.py`, `!vllm/engine/`, `!tests/` 等)” 的策略, 旨在确保只有影响 NPU 相关代码的变更才会触发此耗时较长的硬件测试。
3. 超时设置调整: 删除了原先硬编码的 `timeout_in_minutes: 20`。根据 Review 讨论, 团队决定先观察 CI 运行的实际时间, 以便后续设置更合理的超时值。
4. 量化方法路径排除补充: 根据 Reviewer wangxiyuan 的建议, 在 `source_file_dependencies` 中补充了更多与 NPU 无关的量化方法路径 (如 `awq`, `gptq` 等), 进一步减少误触发。

关键文件:

- `.buildkite/hardware_tests/ascend_npu.yaml` (模块测试配置; 类别 `test`; 类型 `test-coverage`): 定义了 Ascend NPU 测试步骤的配置, 是实现智能测试触发控制的核心文件。

- `.buildkite/scripts/hardware_ci/run-npu-test.sh` (模块 构建脚本; 类别 `infra`; 类型 `infrastructure`) : 构建 Ascend NPU 测试 Docker 镜像的核心脚本, 直接修复了依赖安装失败问题。

关键符号: 未识别

关键源码片段

`.buildkite/scripts/hardware_ci/run-npu-test.sh`

构建 Ascend NPU 测试 Docker 镜像的核心脚本, 直接修复了依赖安装失败问题。

```
# ... 前面代码 ...
# 原 : pip install -r requirements.txt 失败
# 新 : 使用 uv 并指定 extra-index-url
RUN --mount=type=cache,target=/root/.cache/uv \
    uv pip install -r /workspace/vllm-ascend/requirements.txt \
    --extra-index-url https://mirrors.huaweicloud.com/ascend/repos/pypi \
    --quiet --index-strategy unsafe-best-match

# 安装 vllm-ascend 本身
RUN --mount=type=cache,target=/root/.cache/uv \
    source /usr/local/Ascend/ascend-toolkit/set_env.sh && \
    source /usr/local/Ascend/nnal/atb/set_env.sh && \
    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/Ascend/ascend-toolkit/latest/
    $(uname -i)-linux/devlib && \
    # 同时使用主 PyPI、pytorch CPU 源和华为云源
    uv pip install -e /workspace/vllm-ascend/ \
    --extra-index-url https://download.pytorch.org/whl/cpu/ \
    --extra-index-url https://mirrors.huaweicloud.com/ascend/repos/pypi \
    --quiet --index-strategy unsafe-best-match
# ... 后续代码 ...
```

评论区精华

- 安全与版本锁定: Review 机器人 `depthfirst-app[bot]` 指出直接通过管道执行 `curl` 安装 `uv` 存在供应链风险, 建议锁定特定版本 (如 `https://astral.sh/uv/0.7.13/install.sh`)。此建议未被作者在本次 PR 中采纳, 留下了安全隐患。
- 依赖文件排除范围: `wangxiyuan` 指出需要排除更多量化方法相关的代码路径 (如 `vllm\model_executor\layers\quantization` 下的 `awq/gptq` 等)。作者后续补充了这些排除项。
- 超时策略: `Yikun` 询问是否需要设置超时。作者和 `wangxiyuan` 解释称当前编译耗时长, 具体时间未知, 计划在后续 PR 中根据运行数据确定。决策: 移除超时, 待观察。
 - `uv` 安装器的版本锁定与安全性 (`security`): 作者未在本次 PR 中采纳该安全建议。
 - `source_file_dependencies` 的排除范围补充 (`correctness`): 作者后续提交补充了量化方法路径的排除规则。

- NPU 测试超时设置的去留 (design): 决定先移除固定超时, 待收集运行数据后再在后续 PR 中设置合理值。

风险与影响

- 风险:

1. 供应链安全: 使用未固定版本的 uv 安装脚本 (`curl ... | sh`) 存在风险, 如果 `astral.sh` 服务被入侵, 可能在构建阶段执行恶意代码。
2. 依赖排除列表维护负担: `.buildkite/hardware_tests/ascend_npu.yaml` 中的 `source_file_dependencies` 列表非常长且具体。未来如果新增与 NPU 相关的代码路径 (例如新的 `vllm/platforms/ascend.py`), 需要开发者主动将其添加到包含模式中, 否则变更可能不会触发 NPU 测试, 导致潜在回归未被发现。
3. 工具链引入: 在构建脚本中引入了 uv 作为新的构建依赖。虽然提升了效率, 但也增加了一个外部依赖, 需要确保其在 Docker 构建环境中的稳定性和兼容性。

- 影响:

- 对用户: 直接用户 (开发者) 影响较小, 但修复了 Ascend NPU 平台的 CI, 使得该平台的开发者能获得持续集成反馈, 保障了开发质量。
- 对系统: 修复了 CI 流水线的断裂, 恢复了 NPU 硬件测试。优化后, 未来针对 NPU 无关代码的提交将不会触发此测试, 节省了构建和测试资源。
- 对团队: 为华为 Ascend NPU 的贡献者扫清了障碍。`source_file_dependencies` 的引入模式可能为其他耗时较长的硬件测试 (如某些 GPU 测试) 的智能触发提供参考。
- 风险标记: 供应链安全风险 (uv 未固定版本), 维护负担 (`source_file_dependencies` 列表需持续更新)

关联脉络

- PR #54863 [XPU][CI] Move heavy jobs to nightly test in Intel GPU CI: 同属 CI 优化范畴, 都是将耗时的硬件测试从常规 CI 移出或优化触发条件, 以提升 PR 测试效率。
- PR #52350 [CI] Shard LoRA TP distributed tests: 同属 CI 性能优化, 通过分片并行降低特定测试组的耗时。