PR #47478 完整报告

vllm-project/vllm

[ROCm][CI] Adding Rust parity

合并时间：2026-07-07 06:05

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47478

执行摘要

- 一句话：ROCm 平台 CI 增加 Rust 前端测试覆盖
- 推荐动作：该 PR 展示了如何为异构平台（ROCm）扩展 CI 覆盖，值得 CI 基础设施团队和有跨平台需求的开发者阅读。脚本中的根目录回退技巧也值得借鉴。

功能与动机

Adds ROCm AMD CI coverage for Rust frontend cargo style, clippy, and tests on the MI250 pool. It also adds MI300 Rust frontend parity jobs for OpenAI coverage, serve admin coverage, core correctness, tool use, and distributed paths. Dockerfile.rocm now includes the Rust sources, toolchain file, and unzip dependency needed for these jobs. The cargo CI helper can also resolve the repo root when the image workspace does not include git metadata.

实现拆解

1. 在 CI 配置中添加 Rust 作业：在 .buildkite/test-amd.yaml 中新增 14 个 CI 任务。MI250 池（无 GPU）上包含两个作业：Rust Frontend Cargo Style + Clippy 和 Rust Frontend Cargo Tests，两者均依赖 rust/、rust-toolchain.toml 和辅助脚本。MI300 池上新增 12 个作业，覆盖 OpenAI 端点（chat completion、serve admin、logprobs）、核心正确性、工具使用和分布式测试，部分设置为 optional: true 以降低阻断风险。
2. 改进脚本的根目录解析：在 .buildkite/scripts/run-rust-frontend-cargo-ci.sh 中，当 git rev-parse --show-toplevel 因镜像工作区缺少 git 元数据而失败时，通过脚本所在路径推导仓库根目录，确保 cargo 命令能在任意环境下找到正确的 Cargo.toml。
3. 更新 Docker 镜像：在 docker/Dockerfile.rocm 中，从构建阶段复制 rust/ 目录和 rust-toolchain.toml 两个阶段（构建和发布），并安装 unzip 包（Rust 工具链安装所需），确保测试环境具备完整的 Rust 构建依赖。

关键文件：

- .buildkite/test-amd.yaml（模块 CI 配置；类别 config；类型 configuration）：新增 14 个 CI 任务，包括 MI250 的 cargo 静态检查 / 测试和 MI300 的集成测试，是实现 Rust 前端 CI 覆盖的核心配置。
- .buildkite/scripts/run-rust-frontend-cargo-ci.sh（模块 CI 脚本；类别 other；类型 core-logic）：增强根目录解析逻辑，当 git 不可用时回退到脚本相对路径，确保 CI 脚本健

壮性。

- docker/Dockerfile.rocm（模块 Docker 构建；类别 infra；类型 infrastructure）：复制 Rust 源码和工具链文件，安装 unzip 依赖，为 CI 测试环境提供必要的 Rust 构建支持。

关键符号：未识别

关键源码片段

.buildkite/scripts/run-rust-frontend-cargo-ci.sh

增强根目录解析逻辑，当 git 不可用时回退到脚本相对路径，确保 CI 脚本健壮性。

```
#!/bin/bash
# run-rust-frontend-cargo-ci.sh
# 在执行 cargo 操作之前确保 ROOT_DIR 正确设置

if ROOT_DIR="$(git rev-parse --show-toplevel 2>/dev/null)"; then
    # 若 git 可用，直接获取仓库根目录
    :
else
    # 若镜像环境没有 git 元数据，则根据脚本路径推导根目录
    SCRIPT_DIR="$(cd -- "$(dirname -- "${BASH_SOURCE[0]}")" && pwd -P)"
    ROOT_DIR="$(cd -- "${SCRIPT_DIR}/../.." && pwd -P)"
fi

cd "$ROOT_DIR"
# 确保 cargo 输出有颜色
export CARGO_TERM_COLOR="${CARGO_TERM_COLOR:-always}"

echo "Running in directory: $ROOT_DIR"
echo "Mode: $MODE"
```

docker/Dockerfile.rocm

复制 Rust 源码和工具链文件，安装 unzip 依赖，为 CI 测试环境提供必要的 Rust 构建支持。

```
# Dockerfile.rocm 片段

# 在构建阶段复制 Rust 代码和工具链文件
COPY --from=build_vllm ${COMMON_WORKDIR}/vllm/rust /rust
COPY --from=build_vllm ${COMMON_WORKDIR}/vllm/rust-toolchain.toml /rust-toolchain.toml

# 在发布阶段同样复制
COPY --from=build_vllm_wheel_release ${COMMON_WORKDIR}/vllm/rust /rust
COPY --from=build_vllm_wheel_release ${COMMON_WORKDIR}/vllm/rust-toolchain.toml /rust-
toolchain.toml

# 安装 unzip，Rust 工具链安装需要此包
RUN apt-get update -q -y && apt-get install -q -y --no-install-recommends \
    unzip \
    && rm -rf /var/lib/apt/lists/*
```

评论区精华

无实质讨论，PR 由 khluu 直接批准。仅有一条 claude[bot] 自动评论提示从 fork 发起，需维护者手动触发审查。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. CI 时间增加：新增 14 个作业，最长达 180 分钟，可能延长整体 CI 流水线时间。但部分作业设为 optional 或 no_gpu，明确标注 # TBD，表明尚未完全稳定。
  2. 镜像体积膨胀：Dockerfile.rocm 复制 Rust 源码和工具链文件，增加镜像大小，但增量可控。
  3. 脚本健壮性：根目录回退逻辑在 git 不可用时生效，但若 SCRIPT_DIR 解析失败仍可能导致后续命令异常。 - 影响：面向团队：ROCm 维护者和 Rust 前端开发者可更早发现回归，确保 Rust 代码在 AMD GPU 上的正确性。影响范围限于 CI 基础设施，不改变运行时行为。影响程度中等偏低。 - 风险标记：CI 作业扩展，镜像体积增加

关联脉络

- 暂无明显关联 PR