PR #23644 完整报告

sgl-project/sglang

[AMD] Fix nightly version tag selection

合并时间：2026-04-24 17:39

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23644

执行摘要

- 一句话：修复 AMD 夜间 Docker 版本标签排序问题
- 推荐动作：此 PR 是基础设施维护类变更，逻辑相对简单，但 review 中提出的路径鲁棒性问题值得关注。建议在后续的 PR 中跟进修复路径问题，使用 $BASH_SOURCE 构建相对路径。对于想了解 SGLang CI 流水线的读者，此 PR 展示了如何统一版本标签选择逻辑。

功能与动机

AMD 夜间 Docker 构建需要选择最新的发布版本标签，但原来的排序方式将 post 和稳定版本排在 rc 之后，导致错误的版本被选中。PR 通过引入共享的 PEP 440 版本比较工具来解决此问题。

实现拆解

1. 修改 GitHub Actions 工作流文件

- 文件: .github/workflows/release-docker-amd-nightly.yml 和 .github/workflows/release-docker-amd-rocm720-nightly.yml
- 变更: 将原来通过 git tag -l 'v[0-9]*' --sort=-v:refname 获取最新版本标签的方式，替换为调用 Python 脚本 python tools/get_version_tag.py --tag-only，该脚本实现了 PEP 440 兼容的版本比较逻辑。
- 原因: --sort=-v:refname 按 ASCII 顺序排序，导致 v0.5.7post1 排在 v0.5.7rc2 之后，而 PEP 440 规定 post 版本应高于 rc 版本。
- 影响: 确保稳定版和 post 发行版标签优先于 rc 候选版。

2. 在 Nightly 工作流中添加 Python 环境配置

- 文件: 同上两个工作流文件
- 变更: 在调用 Python 脚本之前添加了 actions/setup-python@v5 步骤，设置 Python 版本为 3.10。
- 原因: 原来的工作流没有 Python 环境，无法执行 Python 脚本。添加此步骤确保脚本可以正常运行。

3. 更新 AMD CI 容器启动脚本

- 文件: scripts/ci/amd/amd_ci_start_container.sh 和 scripts/ci/amd/amd_ci_start_container_disagg.sh
- 变更: 将原来类似的 git tag 命令替换为 python3 python/tools/get_version_tag.py --tag-only || true。
- 原因: 保持一致性，使得容器启动时选择的版本标签与 Docker 构建使用相同的逻辑。使用 || true 避免 Python 不可用时脚本失败。

4. 测试验证

- 执行了 python3 test/registered/unit/tools/test_get_version_tag.py 验证共享工具的正确性。
- 对 shell 脚本进行了语法检查（bash -n）。

关键文件:

- .github/workflows/release-docker-amd-nightly.yml（模块 GitHub 工作流；类别 infra；类型 infrastructure）：AMD 夜间 Docker 构建的主工作流，修改了版本标签获取方式并添加了 Python 环境设置。
- .github/workflows/release-docker-amd-rocm720-nightly.yml（模块 GitHub 工作流；类别 infra；类型 infrastructure）：另一个 AMD 夜间 Docker 工作流，做了与上一个工作流相同的修改。
- scripts/ci/amd/amd_ci_start_container.sh（模块 CI 脚本；类别 infra；类型 infrastructure）：AMD CI 容器启动脚本之一，使用共享工具获取版本标签。但存在路径依赖问题。
- scripts/ci/amd/amd_ci_start_container_disagg.sh（模块 CI 脚本；类别 infra；类型 infrastructure）：另一个 AMD CI 容器启动脚本，与上一个脚本做了相同的修改。

关键符号：get_version_tag.py (未直接修改，通过 --tag-only 参数使用)

关键源码片段

scripts/ci/amd/amd_ci_start_container.sh

AMD CI 容器启动脚本之一，使用共享工具获取版本标签。但存在路径依赖问题。

```
# scripts/ci/amd/amd_ci_start_container.sh（片段）
SGLANG_VERSION="v0.5.5"
# 从 origin 获取最新标签以确保最新
if git fetch --tags origin; then
  # 使用共享工具，使稳定版 /post 版本排在 rc 版本之上
  VERSION_FROM_TAG=$(python3 python/tools/get_version_tag.py --tag-only || true)
  if [ -n "$VERSION_FROM_TAG" ]; then
    SGLANG_VERSION="$VERSION_FROM_TAG"
    echo "Using SGLang version from git tags: $SGLANG_VERSION"
  fi
fi
# 注意：此路径依赖仓库根目录执行，否则会失败
```

评论区精华

Review 中 gemini-code-assist[bot] 提出了一个关于路径鲁棒性的问题：

> 当前 Python 脚本的路径是相对于工作目录的，如果脚本从其他目录（如 scripts/ci/amd/）执行，将无法找到脚本。之前的 git tag 命令可以在仓库的任何子目录下工作。建议使用相对于脚本位置的路径。

该评论使用了 $(dirname "${BASH_SOURCE[0]}")/../../../python/tools/get_version_tag.py 作为替代方案。

结论: 该建议未在最终提交中被采纳。PR 的合并者 bingxche 在完成审核后直接将 PR 合并，未回应此建议。目前 scripts/ci/amd/ 目录下的脚本仍然依赖从仓库根目录执行。

- Python 脚本路径鲁棒性 (correctness): 建议未被采纳。当前实现仍然依赖从仓库根目录执行。

风险与影响

- 风险：
  1. 路径依赖性（中风险）：scripts/ci/amd/amd_ci_start_container.sh 和 amd_ci_start_container_disagg.sh 中直接使用 python3 python/tools/get_version_tag.py，如果用户不从仓库根目录执行脚本，将导致失败。review 中已指出此问题但未修复。
  2. Python 依赖风险（低风险）：这些脚本假设系统已安装 Python3，如果环境中没有 Python，脚本可能失败（虽然使用 || true 缓解）。
  3. 工具行为变更风险（低风险）：共享工具 get_version_tag.py 的行为必须与预期一致。如果该工具未来被修改而未考虑此处的使用场景，可能导致版本选择错误。 - 影响：影响范围: AMD 夜间 Docker 构建工作流和 AMD CI 容器启动脚本。影响程度: 中等。修复影响 AMD 平台的自动化发布流程，确保选择正确的版本标签。对于手动执行 CI 脚本的开发人员，可能存在路径问题。用户影响: 无直接用户影响，属于基础设施改进。团队影响: AMD 相关的开发和 CI 团队。
- 风险标记：脚本路径依赖，缺少 Python 环境回退

关联脉络

- PR #23607 [AMD] upd local registry address: 同为 AMD CI 基础设施的 PR，修改了同一批文件（amd_ci_start_container.sh 和 amd_ci_start_container_disagg.sh），属于同一功能线。