PR #2538 完整报告

radixark/miles

release: miles version release workflow

合并时间：2026-08-15 10:57

原文链接：http://prhub.com.cn/radixark/miles/pull/2538

执行摘要

- 一句话：新增 CI 门控的版本发布工作流，冻结依赖三元组并发布官方镜像。
- 推荐动作：值得精读。本 PR 展示了多个高价值设计决策：用 commit status 做跨工作流门控、cadence 覆盖隔离 release 与 nightly 基线、retag 规避镜像清理周期、cherry-pick 与镜像层一致性约束。发布维护者应重点阅读 docs/ci/04-release.md 与 release-tag.yml 的 force 逃生门；普通贡献者需要了解 bot-cherry-pick.yml 的限制。

功能与动机

PR body 明确说明：官方发布必须保留通过 CI 的精确 Miles、SGLang、Megatron-LM 源码组合，而此前 CI 无法针对任意 release ref 运行全量发布范围，也无法在分支切割、hotfix、打 tag、镜像发布之间携带冻结的三元组，维护者缺少一条可复现、受支持的发布路径。

实现拆解

1. CI 策略层：新增 release cadence 与 cadence 覆盖

- tests/ci/ci_policy.py：新增 RELEASE_CADENCE 并纳入 CI_CADENCES；resolve_policy 中 release 与 weekly 一样全范围、admit_nightly_tests=True、bypass_fastfail=True，但 write_baseline 保持 False（冻结 SHA 的 release 运行写基线会污染 nightly 对比）；resolve_workflow_inputs 新增 cadence_override 参数并优先于触发器推断，避免被调用工作流继承调用方 event_name 时静默降级为 regular 范围。
- .github/workflows/pr-test.yml：新增 workflow_call 输入 ref、cadence（默认 release）、image_tag；resolve-ci-policy 任务故意不 checkout inputs.ref（防 CodeQL 缓存投毒）；concurrency group 改为字面量前缀 pr-test- 并加入 inputs.ref，解决被调用时 github.workflow 解析为调用方名称导致 CUDA/ROCm 互相取消的问题；各 stage 透传 ref。
- pr-test-rocm.yml 与可复用 _run-ci*.yml 同步透传 ref，保证 release 分支 smoke 也测试正确 commit。

关键文件：

- .github/workflows/release-branch-cut.yml（模块 发布编排；类别 infra；类型 infrastructure）：发布流程的编排入口：切分支、冻结依赖、retag 镜像、调度全量 CI、打 commit status。

- .github/workflows/release-tag.yml（模块 标签门控；类别 infra；类型 infrastructure）：发布门控核心：校验 setup.py 版本、要求 release-ci commit status 为 success，创建 tag 并触发镜像构建。
- .github/workflows/release-docker.yml（模块 镜像发布；类别 infra；类型 infrastructure）：官方镜像构建与发布：从 release-lock.json 读取 SGLang/Megatron SHA 构建 cu13/cu12 镜像。
- tests/ci/ci_policy.py（模块 CI 策略；类别 test；类型 core-logic；符号 resolve_policy, resolve_workflow_inputs, RELEASE_CADENCE）：CI 策略核心：新增 release cadence 与 cadence_override，保证 release 运行不写性能基线。
- .github/workflows/bot-cherry-pick.yml（模块 热修挑选；类别 infra；类型 infrastructure）：热修复流程的关键约束：拒绝镜像层文件改动，保证冻结镜像与发布镜像一致性。
- .github/scripts/release/write_release_lock.py（模块 依赖冻结；类别 infra；类型 data-contract；符号 main）：定义 release-lock.json 的 schema 与写入逻辑，是依赖冻结的单一事实来源。
- .github/workflows/pr-test.yml（模块 CI 编排；类别 infra；类型 infrastructure）：PR CI 主工作流被改为支持 workflow_call 与 release cadence，变更涉及 concurrency、输入透传等核心机制。
- docs/ci/04-release.md（模块 发布文档；类别 docs；类型 documentation）：发布流程的操作手册，维护者按此从版本 bump 走到镜像验证。
- tests/ci/test/test_run_suite.py（模块 策略测试；类别 test；类型 test-coverage）：CI 策略锁测试：更新 concurrency 表达式并覆盖 release cadence 的语义。
- .github/scripts/release/bump_miles_version.py（模块 版本脚本；类别 infra；类型 core-logic；符号 main）：setup.py 版本 bump 脚本，版本号与 tag 一致性的第一道校验。
- .github/workflows/bot-bump-miles-version.yml（模块 版本提交；类别 infra；类型 infrastructure）：自动化版本 bump 与开 PR 的机器人工作流。

关键符号：resolve_policy, resolve_workflow_inputs, strip_run_ci_prefix, write_release_lock.main, bump_miles_version.main

关键源码片段

tests/ci/ci_policy.py

CI 策略核心：新增 release cadence 与 cadence_override，保证 release 运行不写性能基线。

```
# release 与 weekly 同范围，但绝不写性能基线
RELEASE_CADENCE = "release"
CI_CADENCES = frozenset({REGULAR_CADENCE, NIGHTLY_CADENCE, WEEKLY_CADENCE,
RELEASE_CADENCE})
```

```
def resolve_policy(cadence: str, raw_labels: set[str]) -> RunPolicy:
    """根据显式 cadence 与 raw labels 解析选择与 fast-fail 行为。"""
    if cadence not in CI_CADENCES:
        raise ValueError(f"Unknown CI cadence {cadence!r}")
```

```
    if "nightly" in raw_labels and cadence != NIGHTLY_CADENCE:
        raise ValueError("The nightly workflow label requires cadence='nightly'")

    requested = strip_run_ci_prefix(raw_labels) & set(KNOWN_LABELS)
    # weekly 与 release 都包含全部注册标签；release 仅在不写基线这点上不同
    if "run-ci-all" in raw_labels or cadence in {WEEKLY_CADENCE, RELEASE_CADENCE}:
        scope = set(KNOWN_LABELS)
    elif cadence == NIGHTLY_CADENCE:
        scope = set(KNOWN_LABELS) - {"long", "ft-long"}
    elif "run-ci-image" in raw_labels:
        scope = set(KNOWN_LABELS) - {"long", "ft-short", "ft-long"}
    else:
        scope = set()
    full_cadences = {NIGHTLY_CADENCE, WEEKLY_CADENCE, RELEASE_CADENCE}
    return RunPolicy(
        cadence=cadence,
        include_labels=frozenset(scope | requested),
        admit_nightly_tests=cadence in full_cadences,
        bypass_fastfail=cadence in full_cadences or "bypass-fastfail" in raw_labels,
        # 冻结 SHA 的 release 运行写基线会污染 nightly 对比
        write_baseline=cadence in {NIGHTLY_CADENCE, WEEKLY_CADENCE},
    )


def resolve_workflow_inputs(
    event_name: str, schedule: str, pr_labels_json: str, cadence_override: str = ""
) -> WorkflowPolicy:
    """适配 GitHub 触发器事实为稳定策略输出。

    `cadence_override` 显式携带 workflow_call 的 cadence（如 release 分支切割请求
    全量 release 运行），必须优先于触发器推断：被调用的工作流继承调用方的
    event_name，若按触发器推断会把 release 运行静默降级为 regular 范围。
    """
    if cadence_override:
        cadence, raw_labels = cadence_override, ()
    elif event_name == "pull_request":
        raw_labels = _canonical_pr_labels(pr_labels_json)
        cadence = NIGHTLY_CADENCE if "nightly" in raw_labels else REGULAR_CADENCE
    elif event_name == "schedule":
        cadence, raw_labels = SCHEDULE_POLICIES[schedule]
    elif event_name == "workflow_dispatch":
        cadence, raw_labels = REGULAR_CADENCE, ()
    else:
        raise ValueError(f"Unsupported PR Test trigger: {event_name}")
    run_policy = resolve_policy(cadence, set(raw_labels))
    return WorkflowPolicy(
        cadence=cadence,
        raw_labels=raw_labels,
        bypass_fastfail=run_policy.bypass_fastfail,
```

)

评论区精华

三条核心 review 讨论全部围绕发布安全性与可复现性：

1. CodeQL bot 指出 release-docker.yml 未显式限制 GITHUB_TOKEN 权限，最终补上 permissions: contents: read。
2. guapisolo 的 P1 指出 release-tag.yml 用 run 创建时间判断 CI 新旧会误拒首次 cut run（run 创建早于 lockfile commit），最终改为 release-branch-cut.yml 最后通过 finalizer 在 tested SHA 上打 release-ci commit status，release-tag 改为校验该 status。
3. guapisolo 的 P1 指出 workflow_call 继承 caller 事件导致 docker-paths 不识别 Docker 变更，CI 会用旧冻结镜像而发布构建新镜像，最终通过 bot-cherry-pick.yml 拒绝所有触碰镜像层文件的 cherry-pick 来闭环。

- release-docker.yml 缺少显式 permissions (security)：已在工作流顶部补充显式 permissions: contents: read，guapisolo 在最终审批中再次提醒 'need handle permission issue'。
- release-tag 门控不能用 run 创建时间判断 (correctness)：改为 release-branch-cut 的 finalizer 在 tested SHA 上打 release-ci commit status，release-tag 检查该 status，彻底消除时间推断问题。
- Docker 热修复后 CI 镜像与发布镜像不一致 (correctness)：在 bot-cherry-pick.yml 中拒绝所有触碰镜像层文件的提交，requirements.txt 例外；镜像层修复必须走新版本。
- 最终审批中的权限处理提醒 (security)：release-docker.yml 补充 permissions: contents: read 后关闭。

风险与影响

- 风险：
  1. release-tag.yml 的门控完全依赖 release-ci commit status：若 finalizer 未打上或 status 被覆盖，会阻塞或误放行 tag；force=true 是设计好的审计逃生门，但存在滥用风险。
  2. release-docker.yml（PR body 已指出）手工 workflow_dispatch 时 version 与 ref 不交叉校验，可能构建出与 tag 不一致的镜像。
  3. release-branch-cut.yml 通过 retag 规避 docker-build 的 dev 标签清理，但依赖 Docker Hub API 与 buildx 的可用性；若 dev 镜像 tag 规则变化，release 镜像可能找不到。
  4. bot-cherry-pick.yml 只拒绝 docker/ 下的镜像层文件，requirements.txt 等运行时依赖变更仍可 cherry-pick，未来若构建引入新的镜像层来源需要同步更新黑名单。
  5. concurrency 字面量前缀依赖所有调用方保持命名一致，新增可复用工作流时容易踩坑。 - 影响：对维护者：发布路径从手工操作变为可审计、可重复的 CI 流程，分支切割、CI 验证、打 tag、镜像发布之间有明确门控与证据链。对开发者：hotfix 提交策略受限（镜像层改动只能走新版本），cherry-pick 流程被机器人接管。对 CI 资源：release cadence 复用 weekly 全量范围，每次发布触发完整 CUDA/ROCm 套件，资源开销大但

仅在发布时发生。对运行时与训练代码零影响。 - 风险标记：门控依赖 commit status，镜像层 hotfix 受限，手工 dispatch 版本 /ref 不校验，concurrency 字面量前缀易失效，release cadence 资源消耗大

关联脉络

- PR #2564 release: bump miles version to 0.1.0: 首个通过本发布流程落地的版本 bump，验证 bot-bump-miles-version.yml 产出的 PR 形态。
- PR #2499 feat(ci): add weekly full-suite cadence: 引入 cadence 策略与 CI 套件编排，本 PR 在其基础上新增 release cadence 并复用其策略解析框架。
- PR #2480 [AMD CI] Enable verified ROCm tests: ROCm CI 套件在本 PR 中被接入 release 流程，pr-test-rocm.yml 透传 ref 并使用 release cadence。