

PR #2701 完整报告

radixark/miles

feat(ci): declare supported GPU generations on every CUDA test

合并时间: 2026-09-10 13:00

原文链接: <http://prhub.com.cn/radixark/miles/pull/2701>

执行摘要

- 一句话: 为所有 CUDA 测试注册声明支持的 GPU 世代 (hopper/blackwell), 为后续硬件感知调度铺路。
- 推荐动作: 本 PR 是 CI 基础设施的一次重要重构, 虽然不改变运行时行为, 但为硬件感知的 CI 调度奠定了坚实的基础。建议精读 `tests/ci/ci_register.py` 中的 `_check_cuda_hardware` 方法和 `tests/ci/hardware.py` 的设计, 理解其如何通过架构词汇表、自动调度和严格校验来保证元数据的完整性。该设计值得在类似需要声明资源能力的 CI 系统中借鉴。

功能与动机

为支持 Blackwell (B200) 等新硬件的 CI 部署, 需要让 CI 系统感知每个测试实际支持的 GPU 世代。传统做法是从 suite 名推断硬件 (默认为 hopper), 但这对于大多数测试是不准确的, 且会将判断延迟到从未发生的实际运行中。PR body 明确指出: *Required rather than defaulted: a default derived from the suite name would stamp "hopper only" on every existing test, which is false for most of them and defers the real judgement to a sweep that never happens.*

实现拆解

1. 新增架构词汇表与自动调度函数: 在 `tests/ci/hardware.py` 中定义了 `KNOWN_ARCHES` (hopper, blackwell)、`CUDA_STAGE_ARCH` (suite 到默认架构的映射) 和 `auto_arch()` 函数, 用于确定测试在无外部干预时应在哪个架构上运行。
2. 扩展 `register_cuda_ci` 签名与校验: 在 `tests/ci/ci_register.py` 的 `CIRegistry` 数据类中增加 `hardware` 字段; 将 `hardware` 加入 `_VALID_KWARGS`; 在 `RegistryVisitor._parse_call_args` 中调用新增的 `_check_cuda_hardware` 方法。
3. 实现严格的硬件校验逻辑: `_check_cuda_hardware` 方法确保 `hardware` 非空、仅包含已知架构、无重复, 且声明的硬件集中的首选架构 (通过 `auto_arch` 确定) 必须与该测试声明的 `home suite` 所属架构一致, 从而强制“测试栖息地”与“默认运行地”匹配。
4. 批量更新 99 个 CUDA 测试的注册: 根据每个测试实际代码路径, 为其 `register_cuda_ci` 调用添加 `hardware=[...]` 参数。分类为: 4 个仅限 blackwell (nvfp4/mxvp8 内核)、6 个仅限 hopper (fa3 后端、H200 硬件参数)、89 个同时支持 hopper 和 blackwell。

5. 补充测试覆盖：新增 tests/ci/test/test硬件.py 测试架构映射表的完整性与正确性；在 tests/ci/test/test_ci_register.py 中增加多个测试用例，覆盖硬件参数缺失、未知架构、重复架构、未知 suite、home suite 不匹配以及非 CUDA 后端误用 hardware 参数等场景。

关键文件：

- tests/ci/ci_register.py（模块 CI 注册器；类别 test；类型 core-logic；符号 _check_cuda_hardware, _VALID_KWARGS, CIRegistry）：核心变更文件，扩展了 CI 注册框架的 AST 解析器，添加了 hardware 参数的解析和严格校验逻辑。
- tests/ci/hardware.py（模块 硬件词汇表；类别 test；类型 data-contract；符号 KNOWN_ARCHES, CUDA_STAGE_ARCH, auto_arch）：新增文件，定义了 GPU 架构词汇表、suite 到架构的映射表以及自动调度函数，是硬件感知调度的核心数据源。
- tests/ci/test/test_ci_register.py（模块 CI 注册器测试；类别 test；类型 test-coverage；符号 test_cuda_hardware_must_be_non_empty, test_unknown_arch_rejected, test_duplicated_arch_rejected, test_unknown_cuda_suite_rejected）：测试文件，新增了多个测试用例，全面覆盖了 hardware 参数的校验逻辑，确保注册框架的正确性和健壮性。
- tests/ci/test/test硬件.py（模块 硬件词汇表测试；类别 test；类型 test-coverage；符号 test_every_cuda_suite_declares_an_arch, test_every_declared_arch_is_known, test_hopper_is_the_default_arch, test_auto_arch_follows_preference_not_declaration_order）：新增测试文件，验证架构映射表的完整性（所有 suite 都有对应架构）、架构值的合法性以及自动调度函数的正确性。
- tests/e2e/megatron/model_scripts/test_deepseek_v4_flash_4layer_megatron_impl_ci.py（模块 DeepSeek 测试；类别 test；类型 configuration）：代表性测试文件，展示了如何为现有 CUDA 测试注册添加 hardware 参数（示例为 hopper+blackwell）。

关键符号：register_cuda_ci, _check_cuda_hardware, auto_arch

关键源码片段

tests/ci/ci_register.py

核心变更文件，扩展了 CI 注册框架的 AST 解析器，添加了 hardware 参数的解析和严格校验逻辑。

```
# tests/ci/ci_register.py

def _check_cuda_hardware(self, func_name: str, suite: str, hardware: list) -> None:
    """Validate `hardware` and tie it to the declared home stage.

    The home stage must sit on the test's first supported arch, so "where
    this test lives" and "where it runs by default" cannot disagree.
    """
    # 必须声明至少一个架构
    if not hardware:
        raise ValueError(
            f"{self.filename}: hardware in {func_name}() must list at least one arch "
            f"from {list(KNOWN_ARCHES)}"
        )
```

```

# 不能包含未知架构
unknown = [arch for arch in hardware if arch not in KNOWN_ARCHES]
if unknown:
    raise ValueError(
        f"{self.filename}: unknown arches {unknown} in {func_name}(); "
        f"valid arches: {list(KNOWN_ARCHES)}"
    )
# 不能重复声明
if len(set(hardware)) != len(hardware):
    raise ValueError(f"{self.filename}: duplicated arch in {func_name}() hardware={hardware}")
# 声明的 suite 必须存在于映射表中
if suite not in CUDA_STAGE_ARCH:
    raise ValueError(
        f"{self.filename}: unknown CUDA suite {suite!r} in {func_name}(); "
        f"valid suites: {sorted(CUDA_STAGE_ARCH)}"
    )
# 关键校验: suite 所属架构必须与 hardware 首选架构一致
home_arch, first = CUDA_STAGE_ARCH[suite], auto_arch(hardware)
if home_arch != first:
    raise ValueError(
        f"{self.filename}: suite {suite!r} is {home_arch} but hardware={hardware} "
        f"runs on {first} by default; a test's home stage must be on its first "
        f"supported arch"
    )

```

tests/ci/hardware.py

新增文件，定义了 GPU 架构词汇表、suite 到架构的映射表以及自动调度函数，是硬件感知调度的核心数据源。

```
# tests/ci/hardware.py
```

```

# 已知 GPU 架构，顺序即为自动调度优先级
KNOWN_ARCHES: tuple[str, ...] = ("hopper", "blackwell")

```

```

# 每个 CUDA suite 所属的默认架构
CUDA_STAGE_ARCH: dict[str, str] = {
    "stage-b-2-gpu-h200": "hopper",
    "stage-c-2-gpu-h200": "hopper",
    "stage-c-4-gpu-h200": "hopper",
    "stage-c-8-gpu-h100": "hopper",
    "stage-c-8-gpu-h200": "hopper",
    "stage-c-8-gpu-b200": "blackwell",
}

```

```

def auto_arch(hardware: list[str]) -> str:
    """The arch a test runs on when nothing asks for a different one."""
    supported = set(hardware)
    for arch in KNOWN_ARCHES:

```

```
if arch in supported:
    return arch
raise ValueError(f"no known arch in {sorted(supported)}")
```

评论区精华

1. 关于 `hardware` 参数是否应为必填：PR 作者在 body 中解释了选择必填而非默认值的权衡——默认值会导致大多数测试被错误标记为 'hopper only'，并掩盖了真正的架构判断工作。
 2. 关于几个测试的错误分类：作者指出，仅凭 grep 信号（如 NVFP4）会错误分类 `test_inkling.py`（实际在 H200 上运行）和 `test_mxfp8_quantizer.py`（等待 H200 参考路径而非 Blackwell 硬件），这强调了人工逐案判断的必要性。
 3. 关于 `stage-c-8-gpu-b200` 的历史遗留问题：PR 提到该 suite 已在测试文件中存在数月但无对应 workflow job，本次验证逻辑关闭了这个漏洞。
 4. 无功能变更：PR 明确声明 No behaviour change，因为 `hardware` 参数直到后续 PR（第 4/5 部分）才会被实际读取用于调度。
- 硬件参数是否必填的权衡 (design): 采纳必填方案，强制开发者在注册时明确声明测试的硬件能力。
 - 测试硬件分类的准确性 (correctness): 对 99 个 CUDA 注册进行了人工审查和正确分类。
 - 无功能变更声明 (design): 本 PR 仅建立元数据基础设施，不改变现有 CI 行为。
 - 合并 main 分支以修复发现阶段阻塞 (correctness): 合并了 main 分支，将新测试的 `hardware` 声明纳入本次 PR。

风险与影响

- 风险：
 1. 语法变更风险：修改了 `register_cuda_ci` 函数的必填参数，导致所有现有调用点必须更新，否则会在 CI 发现阶段报错。这是有意为之的破坏性变更，通过本次 PR 批量修复。
 2. 元数据不一致风险：如果某个测试的 `hardware` 声明与其实际代码路径不匹配（例如声明支持 ['blackwell'] 但实际使用了 Hopper 专用内核），将导致测试在错误的硬件上运行或失败。PR 通过严格校验和人工审查降低了此风险。
 3. CI 发现阶段失败风险：如 Issue 评论所指出，本次 PR 合并后，新增的 CUDA 测试若缺少 `hardware` 声明，将导致整个 CI 发现过程失败，阻塞所有 CPU 和 GPU 运行。已通过 3774ed4 提交解决。
 4. 长期维护风险：未来新增 CUDA 测试时，开发者必须正确声明硬件架构，增加了注册的认知负担和出错可能。但这是获得清晰硬件感知能力所必需的权衡。
- 影响：
 1. 对 CI 系统：为所有 CUDA 测试建立了明确的硬件能力元数据，是后续实现基于架构的智能 CI 路由（如将仅限 Blackwell 的测试自动分配到 B200 runner）的必要前提。
 2. 对测试工程师：增加了注册新 CUDA 测试时的一个必须考虑的维度（硬件架构），但提供了更清晰的测试能力文档。
 3. 对基础设施：使 CI 系统能够更精确地匹配测试与硬件资源，避免在错误的硬件上运行测试或浪费稀缺硬件资源（如将支持两者的测试错误地默认调度到 B200）。

4. 对代码库：通过严格的校验逻辑，防止了 suite 名与架构声明不匹配等“静默错误”的长期积累。 - 风险标记：CI 注册器 API 破坏性变更，元数据正确性依赖人工审查，后续 PR 强依赖本 PR 基础设施

关联脉络

- PR #2854 [AMD] docker: retire the rocm700 variants, add a rocm10-mi35x variant: 同为 CI 基础设施演进，涉及硬件支持（AMD ROCm）的更新，与本 PR 的硬件感知 CI 主题相关。
- PR #2038 [feat] Enable THD packed-sequence training for DeepSeek-V4-Flash, with context parallelism: 本 PR 中 test_deepseek_v4_flash_4layer_megatron_impl_ci.py 的 hardware 声明与此 PR 的功能相关，涉及 DeepSeek-V4 的硬件路径。
- PR #2876 ci(e2e): Harbor trials on real e2b sandboxes through the full rollout path (manual for now): 近期 CI 测试增强 PR，同样关注 CI 基础设施的完善和测试覆盖，与本 PR 的重构目标一致。