PR #25197 完整报告

sgl-project/sglang

ci: decouple stage and runner for cuda registry

合并时间：2026-05-14 08:28

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25197

执行摘要

- 一句话：拆分 CI 注册 suite 为 stage 和 runner_config
- 推荐动作：该 PR 展示了如何通过 AST 工具进行大规模安全重构，值得 CI 基础设施维护者精读。建议合并后提醒团队迁移期间避免同时修改注册点。

功能与动机

原 suite 字符串如 stage-c-test-8-gpu-h200 混合了阶段和运行器信息，不便于独立配置。PR 将 register_cuda_ci(..., suite='stage-X-test-Y', ...) 拆分为 stage='stage-X' 和 runner='Y'，提升可读性和可配置性，同时保留对旧 suite 的后向兼容。

实现拆解

1. 数据类改造：在 python/sglang/test/ci/ci_register.py 中，CIRegistry 新增 stage 和 runner_config 可选字段，suite 变为可选，并添加 effective_suite 属性，当 stage 和 runner_config 均设置时自动复合为 f'{stage}-test-{runner_config}'，否则回退到 suite。
2. 注册函数签名更新：register_cuda_ci、register_cpu_ci、register_amd_ci、register_npu_ci 均将 suite 改为可选，新增 stage 和 runner_config 关键字参数（通过 * 分隔）。
3. AST 解析器调整：RegistryVisitor._parse_call_args 支持新的关键字参数，参数列表扩增为 _ALL_PARAMS，验证逻辑允许 suite 为 None 但必须提供 stage+runner_config 之一。
4. 注册点迁移：对 258 个文件中所有 register_cuda_ci 调用，将符合 ^stage-[abc]-test- 模式的 suite 替换为独立的 stage 和 runner_config；nightly-*、stress、weekly-* 等特殊套件仍使用 suite=。
5. 下游消费者更新：test/run_suite.py 中的 filter_tests 和 scripts/ci/utils/compute_partitions.py 从读取 registry.suite 改为读取 registry.effective_suite。
6. 验证：使用 AST 解析器导出迁移前后的 CIRegistry 列表，对比 effective_suite 重建后的条目完全一致，且 compute_partitions.py 输出 JSON 字节相同。

关键文件：

- python/sglang/test/ci/ci_register.py（模块 CI 注册；类别 test；类型 core-logic；符号 CIRegistry, effective_suite, _parse_call_args, register_cuda_ci）：核心变更文件，定义了新的 CIRegistry 数据结构和 effective_suite 属性，改造了所有注册函数和 AST 解析器。

- test/registered/8-gpu-models/test_dsa_models_hisparse.py（模块 Hisparse 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/registered/8-gpu-models/test_dsa_models_mtp.py（模块 MTP 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/registered/debug_utils/test_cuda_coredump.py（模块 Coredump 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/registered/models/test_dummy_grok_models.py（模块 Dummy 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/registered/models/test_ministral3_models.py（模块 Ministral3 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/registered/models/test_ministral4_models.py（模块 Ministral4 测试；类别 test；类型 configuration）：展示 register_cuda_ci 调用从 suite 格式迁移为 stage+runner_config 的典型范例。
- test/run_suite.py（模块 运行套件；类别 test；类型 configuration）：下游消费者，改为使用 CIRegistry 的 effective_suite 属性读取 suite。
- scripts/ci/utils/compute_partitions.py（模块 分区计算；类别 infra；类型 configuration）：下游消费者，改为使用 effective_suite 计算分区。

关键符号：effective_suite, _parse_call_args, register_cuda_ci, register_cpu_ci, register_amd_ci, register_npu_ci

关键源码片段

python/sglang/test/ci/ci_register.py

核心变更文件，定义了新的 CIRegistry 数据结构和 effective_suite 属性，改造了所有注册函数和 AST 解析器。

```
# `_PARAM_ORDER` 保持位置参数兼容，`_KWARG_ONLY` 为新关键字参数
_PARAM_ORDER = ('est_time', 'suite', 'nightly', 'disabled')
_KWARG_ONLY = ('stage', 'runner_config')
_ALL_PARAMS = _PARAM_ORDER + _KWARG_ONLY
_UNSET = object()

@dataclass
class CIRegistry:
    backend: HWBackend
    filename: str
    est_time: float
    # 新增字段，均为可选
    stage: Optional[str] = None
```

```
    runner_config: Optional[str] = None
    # `suite` 保留用于向后兼容（nightly/stress 等）
    suite: Optional[str] = None
    nightly: bool = False
    disabled: Optional[str] = None

    @property
    def effective_suite(self) -> Optional[str]:
        # 当 `stage` 和 `runner_config` 都设置时，自动复合
        if self.stage is not None and self.runner_config is not None:
            return f'{self.stage}-test-{self.runner_config}'
        return self.suite

def register_cuda_ci(
    est_time: float,
    suite: Optional[str] = None,
    nightly: bool = False,
    disabled: Optional[str] = None,
    *,
    stage: Optional[str] = None,
    runner_config: Optional[str] = None,
):
    '''CUDA CI 注册标记（运行时无操作，仅供 AST 解析）。'''
    return None
```

评论区精华

暂无实质性讨论。PR 由作者独立完成，无 review 评论。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 兼容性风险：如果新代码外部的消费者直接访问 CIRegistry.suite 而非 effective_suite，在使用了新参数的注册点会得到 None，导致错误。但 PR 已更新所有内部消费者。
  2. 大量文件变更风险：258 个文件在主线分支可能产生冲突，需要及时合并。
  3. AST 解析器边界：_parse_call_args 假设所有参数为常量，若注册调用包含变量或复杂表达式，解析器会失败，但 CI 注册调用通常使用字面量。 - 影响：对开发者：增加了 stage 和 runner_config 两个清晰维度，便于配置和后续扩展。对运行系统：无直接影响，所有逻辑语义保持不变。对合并协作：由于涉及大量文件，需协调其他分支避免冲突。 - 风险标记：大量文件变更，向下兼容依赖，AST 解析器正确性，验证脚本依赖

关联脉络

- PR #25193 ci: compute matrix partition counts from est_time: 都修改了 scripts/ci/utils/compute_partitions.py，且本 PR 的分区计算依赖 effective_suite 属性。