PR #34186 完整报告

sgl-project/sglang

[CI] Key scheduled CUDA suites by runner_config instead of hand-written jobs

合并时间：2026-08-10 07:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34186

执行摘要

- 一句话：CUDA 夜测迁移至 runner_config 注册式调度，统一走 _pr-test-stage.yml
- 推荐动作：值得 CI 基础设施负责人精读：该 PR 展示了如何通过 registry-keyed 注册把分散的手写调度收敛为单一执行路径，scheduled 输入作为 per-commit 与夜间行为差异的开关设计干净，token 隔离与超时推导的注释含实证依据。对一般功能开发者，只需了解测试注册格式变更即可，不影响业务代码。关注点建议放在迁移完整性和超时预算变化上，可在合并后观察一个周期的 nightly 结果确认无静默丢失。

功能与动机

PR body 明确指出："Replaces the hand-written nightly/weekly job list with registry-keyed stages, so scheduled suites run through the same _pr-test-stage.yml path as per-commit CI." 此前 nightly 工作流中每个 job 手写 runs-on、安装步骤、分区矩阵、超时参数，与 per-commit CI 的 runner_configs 体系割裂，测试若要上线一台新机型必须同时修改 workflow 与 suite 注册两处。改为 registry-keyed 后，测试通过 register_cuda_ci 声明 runner_config 即可自动落入对应机器的夜间套件，workflow 文件不再需要按 job 维护。

实现拆解

1. 注册契约迁移：批量改写测试文件中的 register_cuda_ci 调用，从 suite="nightly-8-gpu-b200", nightly=True 形式改为 stage="nightly", runner_config="8-gpu-b200"。涉及 test/registered/ 下数十个文件（如 test_dsa_glm52_cache_layer_split.py、test_custom_all_reduce.py、8-gpu-models/test_glm52_fp8.py 等），nightly 标识不再由 flag 表达，而是由 stage 名称承载。
2. suite 命名与校验逻辑调整（test/run_suite.py）：NIGHTLY_SUITES 中 CUDA 分支从 19 个手写 suite 名缩减为 7 个 nightly-test-{runner_config} 形状的条目；filter_tests 的合法性校验从只查 NIGHTLY_SUITES 或 PER_COMMIT_SUITES 之一，改为查三个字典的并集（_valid_suites_by_backend()），因为 CUDA 夜测 suite 仅凭名称选择、不再依赖 --nightly。
3. workflow 重构（.github/workflows/nightly-test-nvidia.yml、weekly-test-nvidia.yml、_pr-test-stage.yml）：夜间工作流删除了约 550 行手写 job，改为按 runner_config 动态调用 _pr-test-stage.yml；_pr-test-stage.yml 新增 scheduled 输入，用于切换夜间行为：--timeout-from-est-time 逐文件超时、完整 jit kernel 测试网格（

SGLANG_JIT_KERNEL_RUN_FULL_TESTS=1）、IS_H200 服务器启动上限放宽、metrics 收集上传、以及默认串行 shard（除非 full_parallel=true）。precision baseline 所需环境变量也在此步骤导出。

4. 超时推导（python/sglang/test/ci/ci_utils.py）：新增 derive_timeout_per_file(est_time) = max(est * 1.5, est + 1800)，run_unittest_files 的 timeout_per_file 参数改为 Optional[float]，为 None 时按每个文件的 est_time 推导预算；run_suite.py 新增 --timeout-from-est-time 开关。
5. 配套清理：scripts/ci/utils/slash_command_handler.py 删除 _LEGACY_SUITE_TO_RUNNER_CONFIG 映射，旧式单字符串 suite= 注册不再可通过 /rerun-test 调度，错误消息引导重新注册；scripts/ci/stage_models_overrides.json 相应删除 12 个 legacy suite 的 label 映射；docs/docs/references/nightly_precision_regression.mdx 同步更新 token 变量名与 workflow 说明。

关键文件：

- .github/workflows/nightly-test-nvidia.yml（模块 夜测编排；类别 infra；类型 infrastructure）：核心重构对象：550 行手写 job 缩减为 180 行 registry-keyed 薄壳，按 runner_config 动态复用 _pr-test-stage.yml，是消除两套 CI 路径重复实现的关键。
- test/run_suite.py（模块 套件调度；类别 test；类型 test-coverage；符号 NIGHTLY_SUITES, filter_tests, run_a_suite, main）：suite 注册契约的核心载体：NIGHTLY_SUITES 改为 nightly-test-{runner_config} 形状，filter_tests 校验逻辑从二分改为三字典并集，并新增 --timeout-from-est-time 开关。
- python/sglang/test/ci/ci_utils.py（模块 CI 工具；类别 test；类型 test-coverage；符号 derive_timeout_per_file, run_unittest_files）：新增 derive_timeout_per_file 与 run_unittest_files 的 per-file 超时推导能力，是夜间 suite 混合快慢测试时避免一刀切超时预算的关键支撑。
- python/sglang/test/precision_baseline_store.py（模块 基线存储；类别 test；类型 test-coverage；符号 _store_token）：精度基线存储的写 token 从 HF_TOKEN 系列隔离为独立环境变量 SGLANG_PRECISION_HF_TOKEN，避免遮蔽 runner 的 gated-model 读 token，属于安全相关的关键配套。
- scripts/ci/utils/slash_command_handler.py（模块 CI 命令；类别 infra；类型 infrastructure；符号 _extract_legacy_suites, detect_suite）：删除 _LEGACY_SUITE_TO_RUNNER_CONFIG 映射，旧式单字符串 suite 注册不再可被 /rerun-test 调度，是本次迁移的收尾动作，决定了遗留注册项的可运维性。
- .github/workflows/_pr-test-stage.yml（模块 测试管线；类别 infra；类型 infrastructure）：统一执行路径的承载者：新增 scheduled、job_timeout_minutes 输入，按 schedule 切换完整 jit 网格、IS_H200、HF_HUB 超时、metrics 收集与 precision baseline 环境导出。
- .github/workflows/weekly-test-nvidia.yml（模块 周测编排；类别 infra；类型 infrastructure）：周测工作流同步迁移到 registry-keyed 模式，通过与 nightly 相同的 check-changes + stage 调用结构验证该模式的通用性。

关键符号：derive_timeout_per_file, filter_tests, run_unittest_files, _store_token, detect_suite, _extract_legacy_suites

关键源码片段

test/run_suite.py

suite 注册契约的核心载体：NIGHTLY_SUITES 改为 nightly-test-{runner_config} 形状，filter_tests 校验逻辑从二分改为三字典并集，并新增 --timeout-from-est-time 开关。

```
# test/run_suite.py
# CUDA 夜间套件不再按功能手写名字，而是按 runner_config 组织：
# 测试通过 register_cuda_ci(stage="nightly", runner_config=...) 声明归属，
# stage 名称承载节奏，不再需要 nightly=True flag。
def filter_tests(
    ci_tests: List[CIRegistry], hw: HWBackend, suite: str, nightly: bool = False
) -> List[CIRegistry]:
    ci_tests = [
        t
        for t in ci_tests
        if t.backend == hw and t.effective_suite == suite and t.nightly == nightly
    ]

    # 合法性校验需要三个字典的并集，而不是只看 per-commit 或 nightly 半边：
    # CUDA nightly suite 直接按名称选取，不依赖 --nightly flag。
    if suite not in _valid_suites_by_backend().get(hw, set()):
        print(f"Warning: Unknown suite {suite} for backend {hw.name}")

    enabled_tests = [t for t in ci_tests if t.disabled is None]
    skipped_tests = [t for t in ci_tests if t.disabled is not None]
    return enabled_tests, skipped_tests
```

python/sglang/test/ci/ci_utils.py

新增 derive_timeout_per_file 与 run_unittest_files 的 per-file 超时推导能力，是夜间 suite 混合快慢测试时避免一刀切超时预算的关键支撑。

```
# python/sglang/test/ci/ci_utils.py
# 慢速方差在很大程度上是加性的（冷 HF cache、慢的 server 启动），
# 单纯乘系数会在两端都欠配：test_encoder_dp 实际跑 200-426s，
# 却曾以 1.5x 预算（765s）超时到 1185s；test_lora_deepseek_v3_base_logprob_diff
# （est 1800）恰好落在 1.5 * est 上。因此每个文件在比例预算之上
# 再叠加同一份绝对松弛。
DERIVED_TIMEOUT_SLACK = 1800.0
DERIVED_TIMEOUT_FACTOR = 1.5


def derive_timeout_per_file(est_time: float) -> float:
    """根据单个文件的 est_time 推导超时上限，取比例与绝对两者的较大值。"""
    est = float(est_time)
    return max(est * DERIVED_TIMEOUT_FACTOR, est + DERIVED_TIMEOUT_SLACK)
```

评论区精华

该 PR 没有实质 review 评论（0 条 review comments；2 条 issue comments 分别为 Mintlify 预览机器人通知和作者触发的 /tag-and-rerun-ci）。有价值的讨论内嵌在代码注释与提交演进中：

1) ci_utils.py 注释解释了超时公式的动机——"Slow-run variance is largely additive (cold HF cache, slow server launch), so the multiplier alone under-provisions at both ends"，并给出 test_encoder_dp 曾以 1.5x 预算（765s）实际耗时超过 1185s 的实证，因此采用比例加绝对松弛的 max(est*1.5, est+1800) 组合； 2) precision_baseline_store.py 的 _store_token docstring 说明 token 隔离理由——"Deliberately not HF_TOKEN: that name already carries the runner's gated-model read token, so writing the store token there would shadow it and turn every gated model on the job into a 401"； 3) 提交 revert gpu idle wait; fix stale suite name and --nightly help 与 restore IS_H200 and HF_HUB timeouts; widen derived timeout; fix metrics needs 显示作者在自测中对策略做了多轮收敛（如放宽推导超时、恢复 H200 启动上限与 HF_HUB 超时）。

- 超时推导公式的设计动机 (design): 采用 max(est * 1.5, est + 1800) 组合，并让 run_unittest_files 在 timeout_per_file=None 时按每个文件自己的 est_time 推导预算；后续提交 widen derived timeout 进一步放宽。
- precision 基线 token 与 gated-model 读 token 的隔离 (security): 新增独立环境变量 SGLANG_PRECISION_HF_TOKEN，workflow 导出步骤、HfStoreConfig.from_env 报错信息与文档全部同步切换。
- legacy suite 的 /rerun-test 可调度性回退 (design): 接受该回退：所有可调度的 CUDA suite（per-commit 与 scheduled）统一走 runner_configs.yml 单一来源，遗留注册项降级为非可调度并提示迁移。
- nightly 与 per-commit 行为差异的开关设计 (design): 以 scheduled: true 作为统一开关，配套 job_timeout_minutes、run_timeout_minutes 区分 job 级与 run 级超时，避免在 per-commit 路径上引入夜间副作用。

风险与影响

- 风险： 1) 测试静默丢失风险：94 个文件的批量迁移中，若某个 register_cuda_ci 的 stage/runner_config 与 run_suite.py 中 NIGHTLY_SUITES 条目不匹配，测试将得不到合法 suite 校验保护（CUDA 之外的 backend 仍被 _SUITE_CHECKED_BACKENDS 排除在校验外），可能静默不跑； 2) --nightly 语义变化：CUDA 夜测 suite 改为按名称选取，--nightly flag 仅对 AMD/CPU/NPU 生效，旧脚本若沿用 --nightly + 新 suite 名组合可能选不到测试； 3) /rerun-test 能力回退：删除 _LEGACY_SUITE_TO_RUNNER_CONFIG 后，尚未迁移的 legacy suite（如 GB300 系列、部分 AMD）无法再通过 slash 命令单独重跑，排障效率下降； 4) 超时策略变化：derive_timeout_per_file 对短测试（est < 3600s）会给到明显更大的预算（est+1800），可能延长失败反馈时间，而对 est=0 的测试（如 test_custom_all_reduce 夜测注册 est_time=110）则相对安全；长测试（如 test_glm52_fp8 的 est_time=2880）所得预算低于旧版 --timeout-per-file 18000，存在超时风险； 5) 令牌环境变量切换：precision 基线从 HF_TOKEN_PRECISION_STORE/HF_TOKEN 切换到 SGLANG_PRECISION_HF_TOKEN，若 _pr-test-stage.yml 的导出步骤与 _store_token 读取不一致（如 secrets 未注入），基线读写会立即失败。

- 影响：影响范围集中在 CI 基础设施：所有 CUDA nightly/weekly 测试的调度路径（约 40+ 个测试文件、7 个 runner_config 机器池）从手写 workflow 迁移到统一 stage 管线，新增测试只需一次 register_cuda_ci 注册即可自动进入夜间队列，新增机器池只需在 runner_configs.yml 声明。对普通开发者，影响主要体现为测试注册 API 的格式变化（suite+nightly → stage+runner_config）；对 CI 维护者，消除了 nightly 与 per-commit 两套实现长期漂移的维护负担，并统一了分区计算、rust-ext 复用、metrics 上报等行为。团队协作上，后续调整夜间测试的机器选择、超时、并行度都只需改注册项或 _pr-test-stage.yml，不再需要编辑大型 workflow 文件。
- 风险标记：涉及 CI 核心调度路径，94 文件批量迁移存在静默丢失风险，legacy suite 失去 /rerun-test 能力，超时预算策略变化，token 环境变量切换

关联脉络

- PR #34147 [AMD] [CI] Register the DeepSeek-V4-Pro-DSpark MI35x nightly job so its suite actually runs: 同为夜间 CI 编排修复（AMD nightly job 从未真正运行），与本 PR 统一夜间调度路径的主题一致，验证了 registry 覆盖不足会导致测试静默不跑的问题。
- PR #34081 config: business code no longer reads the published ServerArgs: 同属 " 收敛到单一来源 " 的架构方向（配置读取统一到 runtime_context），与本 PR 将夜间 job 收敛到 runner_configs.yml 的单一事实来源理念一脉相承。
- PR #34133 config: derive the runner's DCP topology from its ParallelState: 同为 " 去掉重复推导、统一派生 " 的重构模式，与本 PR 消除 nightly/weekly 手写 job 重复实现的思路一致。