PR #34913 完整报告

sgl-project/sglang

[CI] Move the static ratchets back to CPU unit tests

合并时间：2026-08-15 14:32

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34913

执行摘要

- 一句话：静态 ratchet 检查移回 CPU 测试，降低提交延迟
- 推荐动作：值得快速浏览。主要价值不在于功能变更，而在于：(1) 展示了将重量级 AST/ 正则扫描从 pre-commit 迁移到 CPU 测试的执行路径切换模式，包括 register_cpu_ci 的使用和路径解析的适配；(2) ratchet 检查器本身采用“双向精确 pin”设计（超过或低于基线都会失败），这种防护模式在后续引入新检查器时可以直接借鉴。若你负责 CI 或代码健康度治理，建议精读 test_global_config_read_ratchet.py 与 test_server_args_mutation_ratchet.py。

功能与动机

PR body 明确说明：'These AST/regex scans re-parse all of python/sglang/srt on every commit that touches the package (pass_filenames: false, ~22s locally), which is too slow to sit on the local commit path. This moves them back to test/registered/unit/ as CPU tests (partial revert of #34309); the checks and their baselines are unchanged.' 即每次提交只要触及 python/sglang/srt，pre-commit 就会重新解析整个包，本地耗时约 22 秒，严重影响提交体验；因此将检查迁移到 CI 阶段执行。

实现拆解

1. 删除本地静态检查器：移除 scripts/lint/ 下的 check_static_ratchets.py（聚合入口，main() 原本在单进程内按序执行 9 个检查）、check_server_args_mutation_ratchet.py、check_legacy_global_ratchet.py、check_module_state_ratchet.py、check_parallel_adoption_ratchet.py、check_decode_bookkeeping_ownership.py。
2. 迁移到 CPU 测试：将检查器主体移入 test/registered/unit/，改写为 unittest 风格。例如 check_global_config_read_ratchet.py 迁移为 test_global_config_read_ratchet.py，独立函数 check_global_config_read_ratchet() / check_configured_size_call_sites() / check_no_renamed_accessor_imports() 变为 TestGlobalConfigReadRatchet.test_global_field_reads_match_the_baseline() 等；check_decode_bookkeeping_ownership.py 迁移为 test/registered/unit/spec/test_decode_bookkeeping_ownership.py。每个测试文件开头调用 register_cpu_ci(est_time=..., suite='base-a-test-cpu') 注册进 CPU CI 套件。
3. 适配路径解析：由于文件从 scripts/lint/（相对源码根两级）移到 test/registered/unit/（相对源码根四级），原先 Path(__file__).resolve().parents[2] / 'python' / 'sglang' 的路径计算失效，改为 Path(next(iter(sglang.__path__))) 或

Path(next(iter(sglang.srt.__path__)))，依赖已安装包路径而非脚本位置。

4. 调整 pre-commit 配置与文档：.pre-commit-config.yaml 删除对该批检查器的注册（-6 行）；python/sglang/srt/server_args.py 有一处 +1/-1 的调整（疑似同步修正过期注释，材料未给出具体内容）；.claude/skills/sglang-runtime-context/SKILL.md 与 .claude/rules/unit-test-admission.md 同步更新，反映检查现在运行在 CPU 测试中。
5. 基线保持：所有 ratchet 的 baseline 值未变（如进程级配置读为零、get_global_server_args 为 1、set_global_server_args_for_* 为 4），且新测试在"超过基线"之外还检查"低于基线"，强制基线只能随真实进展而降低。

关键文件：

- test/registered/unit/test_global_config_read_ratchet.py（模块 配置读取检查；类别 test；类型 rename-or-move；符号 _parsed_modules, _field_reads, _configured_size_call_sites, TestGlobalConfigReadRatchet）：迁移后最核心的静态 ratchet 测试，包含进程级配置读取的 AST 扫描逻辑与双向基线 pin。
- scripts/lint/check_static_ratchets.py（模块 静态检查；类别 source；类型 deletion；符号 main）：原聚合入口，负责在单进程内按序运行全部静态 ratchet 检查；删除它即切断了 pre-commit 的本地执行路径。
- test/registered/unit/spec/test_decode_bookkeeping_ownership.py（模块 记账归属；类别 test；类型 rename-or-move；符号 TestDecodeBookkeepingOwnership, test_bookkeeping_sites_match_owner_allowlist, test_spec_v2_draft_workers_do_no_scheduler_bookkeeping）：从 scripts/lint/check_decode_bookkeeping_ownership.py 迁移，保留对调度器记账时钟归属和 spec-v2 draft worker 行为约束的检查。
- test/registered/unit/test_server_args_mutation_ratchet.py（模块 参数变更；类别 test；类型 test-coverage；符号 TestServerArgsMutationRatchet, test_out_of_pipeline_mutations_match_the_baseline）：新增测试文件，承载迁移后的 server_args 变更检查，逻辑自包含，体现双向基线 pin 的典型写法。
- .pre-commit-config.yaml（模块 提交钩子；类别 config；类型 configuration）：删除对该批静态 ratchet 检查器的本地 hook 注册，是本次执行路径切换的关键配置。
- python/sglang/srt/server_args.py（模块 服务参数；类别 source；类型 core-logic）：存在一处 +1/-1 的源码调整，疑似同步修正过期注释（材料未展开具体内容），涉及运行时配置解析模块需谨慎确认。
- .claude/skills/sglang-runtime-context/SKILL.md（模块 辅助文档；类别 docs；类型 documentation）：同步更新 ratchet 检查执行位置的说明，避免误导后续开发者。

关键符号：TestGlobalConfigReadRatchet._check, TestGlobalConfigReadRatchet.test_global_field_reads_match_the_baseline, TestServerArgsMutationRatchet.test_out_of_pipeline_mutations_match_the_baseline, TestDecodeBookkeepingOwnership.test_bookkeeping_sites_match_owner_allowlist, TestDecodeBookkeepingOwnership.test_spec_v2_draft_workers_do_no_scheduler_bookkeeping, TestLegacyGlobalRatchet.test_legacy_accessor_call_sites_match_the_baselines, TestModuleStateRatchet.test_global_statements_match_the_pins, check_static_ratchets.main

关键源码片段

test/registered/unit/test_global_config_read_ratchet.py

迁移后最核心的静态 ratchet 测试，包含进程级配置读取的 AST 扫描逻辑与双向基线 pin。

```
"""Ratchet guard: 进程级配置读取只能减少。

``get_server_args()`` 返回已发布的 ``ServerArgs`` —— 单进程启动记录。
配置决策应改用命名空间访问器（``get_exec()`` 等），它们携带解析后的值。
"""

#  (_collect / _INERT_DYNAMIC_READS / _DIRECT_BASELINE 等常量定义在同文件上方,
# 此处聚焦扫描与测试封装)

def _field_reads():
    """返回 (direct, alias) 两类进程级配置字段读取。

    direct 是 ``get_server_args().field`` 形态；alias 是
    ``sa = get_server_args()`` 之后的 ``sa.field`` / ``getattr(sa, 'field')``。
    """
    direct, alias = [], []
    for path in sorted(_PACKAGE_ROOT.rglob('*.py')):
        rel = path.relative_to(_PACKAGE_ROOT).as_posix()
        # 归属模块（runtime_context / server_args / arg_groups）自身
        # 是发布方，读取不受此 ratchet 约束
        if rel.startswith(_SLOT_OWNERS):
            continue
        try:
            tree = ast.parse(path.read_text())
        except SyntaxError:
            continue
        inert = frozenset(name for path_, name in _INERT_DYNAMIC_READS if path_ == rel)
        module_direct, module_alias = _collect(rel, tree, inert)
        direct += module_direct
        alias += module_alias
    return direct, alias


class TestGlobalConfigReadRatchet(CustomTestCase):
    def _check(self, kind, reads, baseline):
        # 双向精确 pin：超过基线 = 新增进程级配置读取；
        # 低于基线 = 有人删除调用但没有同步下调基线，会掩盖后续回退
        if len(reads) > baseline:
            self.fail(
                f'{kind} process-global config field reads grew: {len(reads)} > '
                f'baseline {baseline}.'
            )
        if len(reads) < baseline:
            self.fail(
```

```
                f'{kind} process-global config field reads shrank: {len(reads)} < '
                f'baseline {baseline}. Lower the baseline in this file to lock '
                'in the progress.'
            )

    def test_global_field_reads_match_the_baseline(self):
        direct, alias = _field_reads()
        self._check('direct', direct, _DIRECT_BASELINE)
        self._check('alias-form', alias, _ALIAS_BASELINE)
```

test/registered/unit/test_server_args_mutation_ratchet.py

新增测试文件，承载迁移后的 server_args 变更检查，逻辑自包含，体现双向基线 pin 的典型写法。

```
"""Ratchet guard: server_args 在解析管线外的变更只能减少。

``ServerArgs.__post_init__`` 返回后，实例携带解析完成的配置，
解析管线（``server_args.py`` / ``arg_groups/``）是唯一能计算它的地方。
"""

from sglang.test.ci.ci_register import register_cpu_ci

# 注册进 CPU CI 套件，est_time 是预估耗时
register_cpu_ci(est_time=5, suite='base-a-test-cpu')

import re
import unittest
from pathlib import Path

import sglang
from sglang.test.test_utils import CustomTestCase

# 依赖已安装包路径，适配 test/registered/unit/ 的目录层级
_SGLANG_ROOT = Path(next(iter(sglang.__path__)))

# 匹配 server_args 属性赋值 / setattr 的正则；(?![=}]) 排除 == 比较和 f-string {x=}
_MUTATION_PATTERNS = [
    re.compile(r'\bserver_args\.[a-z0-9_]+\s*=(?![=}])'),
    re.compile(r'\bsa\.[a-z0-9_]+\s*=(?![=}])'),
    re.compile(r'get_(?:global_)?server_args\(\)\.[a-z0-9_]+\s*=(?![=}])'),
    re.compile(
        r'setattr\(\s*(?:[\w.]+\.)?(?:server_args|sa|get_(?:global_)?server_args\(\))\s*,'
    ),
]

# 解析管线本身允许变更；multimodal_gen 的 ServerArgs 是另一个契约外的类
_EXCLUDED = ('srt/server_args.py', 'srt/arg_groups', 'multimodal_gen')
_BASELINE = 0
```

```
class TestServerArgsMutationRatchet(CustomTestCase):
    def test_out_of_pipeline_mutations_match_the_baseline(self):
        count = 0
        for path in sorted(_SGLANG_ROOT.rglob('*.py')):
            rel = path.relative_to(_SGLANG_ROOT).as_posix()
            if rel.startswith(_EXCLUDED):
                continue
            source = path.read_text()
            count += sum(len(p.findall(source)) for p in _MUTATION_PATTERNS)
        # 双向精确 pin：多一个违规或少一个（基线未同步下调）都会失败
        if count > _BASELINE:
            self.fail(
                f'server_args mutations outside the resolution pipeline grew: '
                f'{count} > baseline {_BASELINE}.'
            )
        if count < _BASELINE:
            self.fail(
                f'server_args mutations outside the resolution pipeline shrank: '
                f'{count} < baseline {_BASELINE}. Lower the baseline to lock in.'
            )


if __name__ == '__main__':
    unittest.main()
```

评论区精华

该 PR 没有任何 review 评论（review_comments_count = 0）。唯一交互是作者在 Issue 评论中触发 /tag-and-rerun-ci 重新跑 CI。因此没有评审交锋可提炼；变更由作者自行合并。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 本地防护时机后移：检查从 pre-commit 移到 CI，开发者提交时不再收到即时失败反馈，违规代码可能在推送后 CI 阶段才暴露，反馈回路变长。
  - 路径解析依赖已安装包：Path(next(iter(sglang.__path__))) 要求测试环境能 import sglang；若安装布局异常，扫描根目录会漂移甚至找不到包，导致检查失效或误报。
  - 扫描成本转移：全包 AST/ 正则解析依然存在，只是从本地 22 秒移到 CPU 测试；CI 的 base-a-test-cpu 套件总耗时增加，但可并行执行。
  - server_args.py 的 +1/-1 改动内容未在材料中展开，属于低风险但需要确认的残留点。
- 影响：
  - 对开发者：本地提交不再被约 22 秒的全包扫描拖慢，开发体验改善；但静态 ratchet 违规的反馈从提交前移到 CI 阶段。

- 对 CI：新增 6 个 CPU 测试文件注册到 base-a-test-cpu 套件，CI 总耗时增加（每个检查约 5-8 秒，可并行调度缓解）。
- 对代码库：scripts/lint/ 精简，ratchet 检查集中到 test/registered/unit/，符合仓库“静态约束作为 CPU 测试”的组织趋势。
- 风险标记：本地防护时机后移，路径解析依赖安装包，扫描成本转移至 CI

关联脉络

- PR #34864 [CI] Path-gate Rust workspace tests in lint: 同为 CI 路径优化：通过路径门控减少无关变更触发的检查，与本 PR 缩短开发者反馈回路的目的一致。