PR #27135 完整报告

sgl-project/sglang

[codex] Fix adaptive metrics test flake

合并时间：2026-06-04 10:07

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27135

执行摘要

- 一句话：修复自适应推测测试的随机性失败
- 推荐动作：该 PR 适合合入以解决 flaky test 问题，但建议跟踪该测试后续是否仍能有效检测 gauge 更新机制回归。如果出现相关 bug，可考虑实现 Codex 建议的重试 / 轮询方案，在容忍延迟的同时保留严格验证。

功能与动机

测试 test_adaptive_metrics_exposed 存在随机失败。原因为 Prometheus gauge 在周期性的解码统计日志中发射，单次抓取可能观察到上 shift 前的旧值。PR body 明确指出："The test assumed Prometheus gauge values were synchronized with /server_info immediately after one extra decode. These gauges are emitted during periodic decode stats logging, so a single scrape can still observe the previous valid adaptive config."

实现拆解

该 PR 仅修改一个测试文件 test/registered/spec/eagle/test_adaptive_speculative.py，共 8 行变更（+5/-3）。

1. 放宽 gauge 断言：在 test_adaptive_metrics_exposed 方法中，将 self.assertEqual(steps, 3.0) 替换为 self.assertIn(steps, {1.0, 3.0})，将 self.assertEqual(draft_tokens, 4.0) 替换为 self.assertIn(draft_tokens, {2.0, 4.0})。这样，即使抓取到上 shift 前的初始配置值（steps=1, draft_tokens=2），测试也不会失败。
2. 保持上 shift 状态验证：上 shift 确认仍通过 self.assertEqual(state["speculative_num_steps"], 3) 在 /server_info 接口上进行，确保上 shift 确实发生。
3. 无其他代码修改：测试的其他部分（如 _drive_upshift、_generate、_scrape_metric 辅助方法）均未改动。

关键文件：

- test/registered/spec/eagle/test_adaptive_speculative.py（模块 测试；类别 test；类型 test-coverage）：唯一修改文件，包含测试 flaky 修复：将 gauge 断言从严格相等改为集合包含，以容忍抓取到旧配置值。

关键符号：test_adaptive_metrics_exposed

关键源码片段

test/registered/spec/eagle/test_adaptive_speculative.py

唯一修改文件，包含测试 flaky 修复：将 gauge 断言从严格相等改为集合包含，以容忍抓取到旧配置值。

```
    def test_adaptive_metrics_exposed(self):
        """After an upshift, the adaptive current-state gauges are scrapeable."""
        state = self._drive_upshift()
        self.assertEqual(state["speculative_num_steps"], 3, f"Never upshifted: {state}")
        # One more decode so the reporter emits a fresh logging interval.
        self._generate(HIGH_ACCEPT_PROMPT)

        steps = self._scrape_metric("sglang:spec_num_steps")
        draft_tokens = self._scrape_metric("sglang:spec_num_draft_tokens")

        # 使用 assertIn 替代 assertEqual，允许 gauge 值为初始配置值（1.0、2.0）
        # 或上 shift 后的值（3.0、4.0），从而避免因抓取周期差异导致的 flaky。
        self.assertIn(steps, {1.0, 3.0}, "spec_num_steps gauge has unexpected value")
        self.assertIn(
            draft_tokens,
            {2.0, 4.0},
            "spec_num_draft_tokens gauge has unexpected value",
        )
```

评论区精华

- 自动化 Codex 评论（P2）：建议使用重试 / 轮询机制直到 gauge 值与上 shift 状态匹配，而非放宽断言。认为接受初始配置值会掩盖“gauge 从未更新”的回归。该评论未获回复或解决。
- Gemini Code Assist 评论：描述变更内容，但未提供具体反馈。
- 审核人 Qiaolin-Yu：直接批准，无额外讨论。

当前方案以最小改动修复随机失败，但牺牲了对 gauge 更新机制的严格检验。

- 放宽 gauge 断言是否削弱测试 (testing)：未采纳；当前合并者批准了简单放宽方案。

风险与影响

- 风险：
  - 测试有效性降低：放宽断言后，如果 Prometheus gauge 更新机制被破坏（例如永远不发射新值），测试仍可通过，因为初始配置值在允许集合中。这可能导致回归延迟发现。
  - 无生产代码风险：变更仅限测试文件，不影响任何生产逻辑、配置或部署。
  - 低回归风险：该 PR 只修改一个测试方法中的两行断言，变更简单且局部，引入新 bug 的可能性极低。
- 影响：
  - 用户（开发者 /CI）：消除了 test_adaptive_metrics_exposed 的随机失败，CI 流水线更稳定，减少误报。但降低了该测试对 gauge 更新机制的覆盖强度。

- 系统：无影响，因为未修改任何运行时组件。
- 团队：快速修复 flaky test 的惯用做法。团队需权衡是否后续加入重试逻辑以恢复测试严格性。
- 风险标记：测试有效性降低

关联脉络

- 暂无明显关联 PR