PR #33663 完整报告

sgl-project/sglang

Fix serving benchmark post-warmup cache flush race

合并时间：2026-08-06 23:27

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33663

执行摘要

- 一句话：修复 serving 基准测试缓存刷新竞态
- 推荐动作：值得精读。flush_server_cache 的分支设计和对"等待服务端空闲"的测试方法（真实 HTTP server 模拟延迟响应）很有借鉴意义，也可作为 benchmark 工具链后续演进的基础。

功能与动机

PR body 指出：Warmup requests can return before the scheduler is fully idle, causing /flush_cache to return 400 and abort the benchmark，该问题由 #32481 暴露——后者为 sglang.bench_serving 引入了 vllm-embedding 后端与 engine-specific post-warmup cache reset。为了公平测量，必须等待调度器完全排空后再清空 prefix cache，否则 warmup 的 prefix 会干扰被测性能，而直接 flush 又会因调度器非空闲而失败。

实现拆解

1. 重构缓存刷新入口：在 python/sglang/benchmark/serving.py 中定义默认超时 _DEFAULT_SGLANG_FLUSH_CACHE_TIMEOUT = 60.0，将 flush_server_cache 改为三分支行为——vllm 前缀走 /reset_prefix_cache 且不传参；sglang 前缀走 /flush_cache 并传 params={"timeout": flush_cache_timeout}；其余后端走 /flush_cache 但不带参数。这样既修复竞态，又避免影响不识别 timeout 的后端。
2. 打通参数链路：benchmark() 新增 flush_cache_timeout 形参，并在 warmup 判定后调用 flush_server_cache(base_url, backend, flush_cache_timeout)。
3. CLI 配置：在 cli_main() 新增 --flush-cache-timeout，使用 _finite_positive_float 校验（拒绝 0/ 负数 / 无穷大），默认 60 秒；run_benchmark 中通过 hasattr 兜底，兼容 SimpleNamespace 直调。
4. 测试配套：test/registered/bench_fn/test_benchmark_datasets_api.py 新增 TestBenchmarkCacheFlush：用 mock 断言 vLLM、SGLang、lmdeploy 的请求形态；用真实 HTTPServer 模拟延迟空闲的调度器，验证 flush_server_cache 会一直等到服务端空闲；新增 CLI 非法 timeout 值拒绝测试；测试基类统一迁移到 CustomTestCase。

关键文件：

- python/sglang/benchmark/serving.py（模块 基准测试；类别 source；类型 core-logic；符号 _DEFAULT_SGLANG_FLUSH_CACHE_TIMEOUT, flush_server_cache,

benchmark, run_benchmark）：核心变更文件，实现可配置的缓存刷新超时，并将 vLLM、SGLang、其他后端分支拆开。

- test/registered/bench_fn/test_benchmark_datasets_api.py（模块 基准测试；类别 test；类型 test-coverage；符号 TestBenchmarkCacheFlush, test_cache_flush_uses_the_backend_specific_request, test_sglang_cache_flush_waits_for_idle, DeferredFlushHandler）：测试配套，覆盖后端特定请求形态与等待空闲行为，并迁移到 CustomTestCase。

关键符号：flush_server_cache, benchmark, run_benchmark, _finite_positive_float, test_cache_flush_uses_the_backend_specific_request, test_sglang_cache_flush_waits_for_idle, test_serving_benchmark_cli_rejects_invalid_flush_cache_timeout

关键源码片段

test/registered/bench_fn/test_benchmark_datasets_api.py

测试配套，覆盖后端特定请求形态与等待空闲行为，并迁移到 CustomTestCase。

```
def test_sglang_cache_flush_waits_for_idle(self):
    '''A busy server can become idle before the benchmark's flush times out.

    用真实本地 HTTP 服务模拟忙碌中的调度器：收到 flush 请求后先挂起，
    等 server_idle 事件再返回 200，从而验证 flush_server_cache 会一直
    等到服务端空闲而不是立即失败。
    '''
    request_received = threading.Event()
    server_idle = threading.Event()

    class DeferredFlushHandler(BaseHTTPRequestHandler):
        def do_POST(self):
            # 解析 query 中的 timeout；0 表示服务端不支持等待
            url = urlparse(self.path)
            timeout = float(parse_qs(url.query).get('timeout', ['0'])[0])
            request_received.set()
            if timeout <= 0:
                status = 400
            else:
                # 有 timeout 则等待“调度器空闲”事件，超时前返回 200
                status = 200 if server_idle.wait(timeout) else 400
            self.send_response(status)
            self.end_headers()

        def log_message(self, format, *args):
            pass # 静默日志，避免污染测试输出

    server = HTTPServer(('127.0.0.1', 0), DeferredFlushHandler)
    server_thread = threading.Thread(target=server.serve_forever, daemon=True)
    server_thread.start()
```

```
base_url = f'http://127.0.0.1:{server.server_port}'

try:
    with ThreadPoolExecutor(max_workers=1) as executor:
        # 在独立线程中执行，因为真实 HTTP 请求是阻塞调用
        flush = executor.submit(flush_server_cache, base_url, 'sglang', 5.0)
        self.assertTrue(request_received.wait(timeout=5))
        self.assertFalse(flush.done())
        server_idle.set() # 模拟调度器变为空闲
        flush.result(timeout=5)
finally:
    server.shutdown()
    server.server_close()
    server_thread.join(timeout=5)
```

评论区精华

本次 PR 没有实质性 review 讨论，维护者 b8zhong 直接批准。CI 过程中 GitHub 机器人提示分支相对 base commit 已 diverged 并要求 rebase，作者随后合并 main 解决冲突，没有引发设计层面的争议。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 默认超时变化：/flush_cache 的默认等待从 10 秒提升到 60 秒。若调度器异常卡死，benchmark 会多等 60 秒才失败，CI 超时预算需要关注。
  - 后端行为分化：非 vllm、非 sglang 后端（如 lmdeploy）不再收到 timeout 参数，这是有意修正，但如果某后端此前依赖该参数，可能产生行为变化。
  - CLI 严格化：--flush-cache-timeout 拒绝 0 和负数，用户无法显式表示“不等待”。
  - 测试可靠性：新集成测试依赖本地 HTTPServer 与线程事件，存在端口占用 / 线程泄漏的表象风险，但代码已用 daemon 线程和 finally 清理。
  - 影响面集中在 sglang.benchmark.serving 工具链，不触碰 SRT 推理核心，回归风险低。
  - 影响：影响所有通过 sglang.bench_serving（含 SGLANG_IS_IN_CI 自动 flush 场景）运行 benchmark 的用户：benchmark 不再因调度器未排空而虚假中断，结果更稳定；新增 CLI 参数向后兼容（默认 60 秒）；测试套件覆盖更全面。团队层面，修复了 #32481 引入的回归，保障 CI 中 embedding benchmark 的可执行性。
  - 风险标记：默认超时 10s 变 60s，非 sglang 后端不再传 timeout，CLI 拒绝 0 超时，测试依赖本地 HTTP 服务

关联脉络

- PR #32481 embedding: centralize capabilities and complete OpenAI compatibility: 本 PR 修复了 #32481 引入的 post-warmup cache flush race；#32481 新增了

vllm-embedding 后端与 engine-specific post-warmup cache reset 逻辑。