# PR #36605 完整报告

sgl-project/sglang

[CI] Graceful teardown for the radix_cache server fixtures

合并时间：2026-08-27 16:36

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36605

## 执行摘要

- 一句话：radix_cache 测试夹具改优雅关闭，修复 H200 CI GPU 未空闲失败
- 推荐动作：实现机械，无需逐文件精读，但值得快速浏览 PR body 的根因分析和统一替换模式。对于维护多 GPU 测试夹具的工程师有参考价值：服务器进程应统一采用"先 SIGTERM 等待、再 SIGKILL 兜底"的关闭方式，让 CUDA context 在用户态释放，而不是把显存回收留给驱动异步处理。建议关注 wait_timeout=60 与 idle 门控 30 秒的取值关系，以及该模式后续是否应推广到 radix_cache 目录之外的其他多卡测试套件。

## 功能与动机

PR body 明确指出：nightly 8 卡 H200 的 KL/HiCache 测试在 setUpClass 阶段报 GPU(s) still not idle after waiting 30s；前一个测试类 SIGKILL 掉 TP8 服务器后，驱动仍在回收约 122 GiB/GPU 显存，错误信息甚至显示调度器进程已消失而显存仍被持有。直接 SIGKILL 让 CUDA context 的释放完全交给内核在进程回收时处理，时间超过下一个测试类的 30 秒空闲门控窗口。该模式此前已在 #32829、#31871 中应用到共享 server fixture，但 radix_cache 目录下这些文件一直未转换；且 #36281 拆分 nightly 文件时把裸 kill 带入了 test_unified_radix_cache_kl_glm52.py，成为本次连锁失败的直接触发点。

## 实现拆解

1. 根因定位：PR body 将问题锁定在 tearDownClass 的裸 SIGKILL 上。radix_cache 测试大多直接调用 kill_process_tree(cls.process.pid)，进程被杀后 CUDA context 与 pinned host memory 的回收完全依赖 GPU 驱动在内核态异步完成；8 卡 H200 上约 122 GiB/GPU 的回收耗时超过下一个测试类的 idle 门控（30 秒），导致 setUpClass 失败。
2. 统一替换关闭语义：在 11 个测试文件中，将 tearDownClass 里的 kill_process_tree(cls.process.pid) 全部替换为 terminate_and_kill_process_tree(cls.process, wait_timeout=60)。该函数来自 sglang.test.test_utils，先向整个进程树发送 SIGTERM，等待进程在用户态完成显存释放和 host memory unpin，超时后再 SIGKILL 兜底；wait_timeout=60 覆盖了 30 秒门控窗口。
3. 覆盖范围：涉及 test/registered/radix_cache 下的全部高负载夹具，包括 test_unified_radix_cache_kl_dsv4.py（3 处 tearDownClass）、test_unified_radix_cache_kl_hybrid_bitexact.py（3 处，含 getattr 防空保护）、test_unified_radix_cache_hicache_pp_kl.py（2 处，含 L3 file backend 临时目录清理）、test_unified_radix_cache_kl_cp.py、test_unified_radix_cache_kl_dcp.py、

test_unified_radix_cache_kl_full.py、test_unified_radix_cache_kl_glm52.py、test_unified_radix_cache_kl_mimo.py、test_unified_radix_cache_kl_swa.py、test_radix_cache_hit.py、test_radix_attention.py。其中 kl_dcp 原本已有 process.wait(timeout=60) 后再 kill 的混合逻辑，本次统一收敛到封装函数，消除重复实现。

4. import 调整：所有文件移除 from sglang.srt.utils import kill_process_tree，改从 sglang.test.test_utils 导入 terminate_and_kill_process_tree，与 popen_launch_server 等测试工具同源。
5. 验证与配套：无源码主路径、配置或 schema 改动；合并者通过 /rerun-group radix_cache 触发 4-gpu-h100 下 5 个测试的重跑；未新增测试用例，属于对既有夹具的可靠性补强。

关键文件：

- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_dsv4.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：本 PR 的核心现场：DSV4 Flash + HiCache 的 TP4 多服务器夹具，包含 3 处 tearDownClass，全部从 kill_process_tree 换成 terminate_and_kill_process_tree；8 卡 H200 显存回收问题在该类上最严重。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_hybrid_bitexact.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：包含 3 处 tearDownClass，且使用 getattr(cls, 'process', None) 防空保护；文件顶部的 KL 回归守卫说明（#34184/#29792）是理解该夹具价值的关键上下文。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_hicache_pp_kl.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：PP 2x2 场景（Qwen3-32B + HiCache），2 处 tearDownClass；L3 file backend 夹具依赖先终止进程再删除临时目录的顺序，本次改动保证进程退出后才清理。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_glm52.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：PR body 点名该文件由 #36281 从 test_unified_radix_cache_kl_nightly.py 拆分而来，并把裸 kill 带了过来，是本 PR 的直接触发点之一。
- test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_dcp.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：原本已有 process.wait(timeout=60) 后 kill 的混合逻辑，本次统一收敛到 terminate_and_kill_process_tree，消除重复实现并保证进程树整体被优雅关闭。
- test/registered/radix_cache/test_radix_cache_hit.py（模块 测试夹具；类别 test；类型 test-coverage；符号 tearDownClass）：同时注册 CUDA、AMD、CPU 三个 CI 套件的最小单卡夹具，是这次关闭模式推广的代表，影响面覆盖三套 runner 配置。

关键符号：tearDownClass, terminate_and_kill_process_tree

## 关键源码片段

### test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_dsv4.py

本 PR 的核心现场：DSV4 Flash + HiCache 的 TP4 多服务器夹具，包含 3 处 tearDownClass，全部从 kill_process_tree 换成 terminate_and_kill_process_tree；8 卡 H200 显存回收问题在该类上最严重。

```
# 关键改动：tearDownClass 统一改用优雅关闭。此前这里直接 kill_process_tree(cls.process.pid)，
# 相当于 SIGKILL，服务器来不及在用户态释放 CUDA context 与 pinned host memory。
from sglang.test.test_utils import (
    DEFAULT_URL_FOR_TEST,
    CustomTestCase,
    is_in_ci,
    popen_launch_server,
    terminate_and_kill_process_tree, # 先 SIGTERM，等待进程退出，再对残留进程树 SIGKILL
)


class TestUnifiedDeepSeekV4FlashHiCache(UnifiedRadixTreeTestMixin, CustomTestCase):
    """DeepSeek V4 Flash FP8 + HiCache + UnifiedRadixCache。"""

    @classmethod
    def setUpClass(cls):
        cls.model = DSV4_FLASH_MODEL
        cls.base_url = DEFAULT_URL_FOR_TEST
        cls.process = popen_launch_server(
            cls.model,
            cls.base_url,
            timeout=DSV4_FLASH_LAUNCH_TIMEOUT,
            other_args=cls._server_args(),
            env={
                'SGLANG_DSV4_FP4_EXPERTS': '0',
                'SGLANG_ENABLE_UNIFIED_RADIX_TREE': '1',
            },
        )
        cls.input_ids = get_input_ids(cls.model, num_samples=18)

    @classmethod
    def tearDownClass(cls):
        # wait_timeout=60：给服务器最多 60 秒在用户态完成显存释放与 unpin，
        # 避免 8 卡 H200 上驱动回收 122 GiB/GPU 显存超过 30 秒空闲门控窗口。
        terminate_and_kill_process_tree(cls.process, wait_timeout=60)
```

## test/registered/radix_cache/unified_radix_tree/test_unified_radix_cache_kl_hybrid_bitexact.py

包含 3 处 tearDownClass，且使用 getattr(cls, 'process', None) 防空保护；文件顶部的 KL 回归守卫说明（#34184/#29792）是理解该夹具价值的关键上下文。

```
# 该文件为 KL 回归守卫测试（Inkling 模型 + UnifiedRadixTree），
# 三个测试类的 tearDownClass 都统一改为优雅关闭。以其中一个类的关闭逻辑为例：

# —— 以下位于某个 CustomTestCase 子类内部 ——
```

```
@classmethod
def tearDownClass(cls):
    # setUpClass 失败时 process 可能不存在，先防空再优雅关闭，
    # 避免 tearDownClass 反而掩盖原始失败。
    if getattr(cls, 'process', None) is not None:
        terminate_and_kill_process_tree(cls.process, wait_timeout=60)
```

## 评论区精华

Review 层面没有实质评论，ispobock（合并者）直接 APPROVED，无未解决疑虑。Issues 评论记录了 rerun 过程：ispobock 第一次执行 /rerun-group test/registered/radix_cache/unified_radix_tree被bot以未知组名拒绝（并列出已知groups），随后改为 /rerun-group radix_cache 成功触发 4-gpu-h100 上 5 个测试的重跑。核心方案讨论集中在 PR body 中：显式选择"先 SIGTERM 再 SIGKILL"、复用 #32829/#31871 已建立的 terminate_and_kill_process_tree 模式，而不是缩短 idle 门控或串行化测试类。

- /rerun-group 组名纠错与 rerun 结果 (other): 使用 radix_cache 组成功重跑 5 个测试；PR 最终通过并合并。
- teardown 方案与合并决策 (design): 方案被批准，无未解决疑虑；合并者以路由方式触发重跑验证。

## 风险与影响

- 风险：
  1. teardown 最长阻塞 60 秒：若服务器在 SIGTERM 后迟迟不退出（如 TP8 大模型关闭慢），tearDownClass 可能增加最多 60 秒等待；但此前 30 秒门控已导致失败，60 秒等待换取显存确定性释放，整体更稳，且 SIGKILL 兜底保证不会无限阻塞。
  2. process 空值保护不统一：test_unified_radix_cache_kl_hybrid_bitexact.py 使用 getattr(cls, 'process', None) 防空，而 dsv4、hicache_pp 等文件直接访问 cls.process；若 setUpClass 中途失败，tearDownClass 可能抛 AttributeError 掩盖原始错误（base 版本同样如此，未新增风险）。
  3. 依赖共享工具函数：terminate_and_kill_process_tree 位于 sglang.test.test_utils，其语义变化会影响全部 11 个测试文件；但统一封装也带来单点演进的好处，与 #36589 中 kill_process_tree 默认等待 reap 的方向一致。
  4. 无运行时、性能或安全影响：变更不触及 python/sglang/srt 下任何推理路径。 - 影响：影响范围限定在 CI 测试基础设施：11 个 radix_cache/HiCache 测试文件的夹具关闭方式，对推理服务、API 和用户无影响。受益面是 nightly 8 卡 H200 以及同类多卡 runner 的测试稳定性，消除了因显存回收超时导致的虚假失败和后续测试类级联跳过；同时为团队提供了可复制的多 GPU 测试夹具关闭模式，后续新增 radix_cache 测试可直接沿用 terminate_and_kill_process_tree。 - 风险标记：测试基础设施变更，teardown 最长阻塞 60 秒，process 空值保护风格不统一

## 关联脉络

- PR #36589 fix: kill_process_tree waits for the reap by default: 同属进程终止与资源回收主题：该 PR 修改 kill_process_tree 默认等待 reap，本 PR 则显式使用 terminate_and_kill_process_tree（先 SIGTERM 再 SIGKILL），两者目标一致、互为补充。
- PR #35791 [Radix Cache] Add test-only TreeCore inspector for shared backend tests: 同属 radix cache 测试基建方向，说明 unified-radix-cache 测试套件仍在扩展；本 PR 提升的夹具可靠性是该套件持续运行的基础。