PR #36413 完整报告

sgl-project/sglang

[CPU][CI]: fix a few issues that cause XEON CI failures

合并时间：2026-08-27 13:59

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36413

执行摘要

- 一句话：修复 XEON CI 失败：SPR 容器重构、缓存挂载、AMX 参数调整
- 推荐动作：值得 CI/ 基础设施维护者精读。重点学习：如何根据 sglang 的内存预留机制（psutil 读宿主内存、cgroup 不生效）推导出容器编排方案；以及 review 如何通过 "hiding the real dependency" 的判断把一次不正确的测试补丁拦截下来并最终回退。对 DeepSeek V4 共享专家融合逻辑感兴趣的人，可关注 get_parallel().override(moe_ep_size=1) 这个测试辅助模式的用法。

功能与动机

PR 的目标直指 XEON CI 的持续失败。PR body 列出需要修复的失败用例（test_deepseek_v4_shared_expert_fusion.py 的 6 个 UT、AMX 后端的 MMLU 与 FP8 延迟测试）；在 issue 评论中，Xia-Weiwen 解释 deepseek UT 'is failing in CI. AI did not find the guilty commit. It's probably due to some framework changes that are not directly related to this UT'，并确认失败发生在 Xeon 而非 CUDA。三个 commit 记录了一步步定位的根因：GNR 容器的离线 MMLU 运行时从 people.eecs.berkeley.edu 下载 data.tar，URL 现在返回 HTTP 403（302 -> forbidden）；mem-fraction-static 0.2 在双 socket 并行 CI 下仍被 KV-cache 预留逻辑推高内存导致 OOM；SPR 盒子上的两个 socket-pinned 容器因 sglang 不遵守 cgroup --memory 限制而耗尽宿主内存。

实现拆解

1. SPR 容器模型重构（.github/workflows/pr-test-xeon.yml）：
   - 根因：sglang 通过 psutil 读取宿主内存自动计算 mem_fraction_static（约 0.88），不遵守 cgroup --memory 限制，每台 SPR 盒子上的两个 socket-pinned 容器都会预留约 195GB 的 KV 池，并行运行直接耗尽宿主 RAM 触发 OOM。
   - 变更：matrix 从 part_size=4 / part_lo+part_hi 的 socket 划分，改成 3 台 SPR 盒子各跑一个 full-machine 容器，part_id=0/1/2、part_size=3 的三分片；每盒只构建一次镜像，单容器运行。
   - 影响：单容器内存预算回到盒子物理内存内，不再依赖 cgroup 生效；测试并行度由 4 降至 3。
2. GNR 容器挂载 sgl_eval 缓存：

- 根因：GNR job 中 TestIntelAMXAttnBackend.test_mmlu 运行 sgl-eval 的离线 MMLU，运行时从 people.eecs.berkeley.edu 下载 data.tar，该 URL 已返回 HTTP 403，导致任务必败。
- 变更：docker run 增加 -v $HOME/.cache/sgl_eval:/root/.cache/sgl_eval，让容器复用宿主机已下载的评测数据。
- 影响：MMLU 测试变为离线可用，不再依赖外网可达性。

3. intel_amx 注意力后端测试内存参数调整（backend a/b）：
   - 根因：mem-fraction-static=0.2 时，KV-cache 配置器按 (1 - mem_fraction) * pre_model_load_memory 预留 slack，并发容器加载模型的瞬时内存下降会触发预留抬升，最终 OOM。
   - 变更：test_intel_amx_attention_backend_a.py 的 test_latency_default_model 装饰器与 TestDPAttention.setUpClass 启动参数，以及 test_intel_amx_attention_backend_b.py 的 test_latency_fp8_qwen 与 test_latency_fp8_moe_model 装饰器，全部从 0.2 改为 0.3。
   - 影响：KV 预留比例收缩，模型（含 FP8 量化与 MoE 模型）在 CI 并发环境下不再因内存峰值失败；这是测试专用调参，不代表部署建议。
4. DeepSeek V4 shared expert fusion 单测的尝试与回退：
   - 最初为修复 Xeon 上失败的 6 个 UT，用 SimpleNamespace(world_size=1) 直接 patch parallel_state._MOE_EP；review 指出 V4 的 shared_experts_fusion_disable_reason 并不读 EP，且 setUp() 已通过 get_parallel().override(moe_ep_size=1) 覆盖该场景，patch 反而掩盖真实依赖；最终该文件的全部改动被 revert，保留测试原始语义并避免重复逻辑。
5. 配套说明：无软件包、schema 或部署改动；全部变更集中于 CI workflow 与 CPU 测试参数，测试配套即本 PR 主体。

关键文件：

- .github/workflows/pr-test-xeon.yml（模块 CI 工作流；类别 infra；类型 infrastructure）：CI 工作流是本次修复的主体：重构 SPR 容器模型（双 socket-pinned 容器 → 单全机容器、分片 4→3）以解决 sglang 不遵守 cgroup 内存限制导致的 OOM，并为 GNR 容器挂载 sgl_eval 缓存解决 MMLU 下载 HTTP 403。
- test/registered/cpu/test_intel_amx_attention_backend_a.py（模块 AMX 测试；类别 test；类型 test-coverage；符号 test_latency_default_model, TestDPAttention.setUpClass）：intel_amx 默认模型与 DP 测试的内存参数调整：mem-fraction-static 0.2→0.3，缓解并发模型加载时 KV-cache 预留导致的 OOM；是 XEON CI 内存问题修复的一部分。
- test/registered/cpu/test_intel_amx_attention_backend_b.py（模块 AMX 测试；类别 test；类型 test-coverage；符号 test_latency_fp8_qwen, test_latency_fp8_moe_model）：FP8 量化模型（Qwen FP8 与 MoE）延迟测试同样调整 mem-fraction-static 到 0.3；这两个用例曾单独在 SPR2 job 中被点名失败。

关键符号：test_latency_default_model, TestDPAttention.setUpClass, test_latency_fp8_qwen, test_latency_fp8_moe_model

关键源码片段

test/registered/cpu/test_intel_amx_attention_backend_a.py

intel_amx 默认模型与 DP 测试的内存参数调整：mem-fraction-static 0.2→0.3，缓解并发模型加载时 KV-cache 预留导致的 OOM；是 XEON CI 内存问题修复的一部分。

```
class TestIntelAMXAttnBackend(CustomTestCase):
    # mem-fraction-static 从 0.2 提到 0.3：KV-cache 配置器按
    # (1 - mem_fraction) * pre_model_load_memory 预留 slack，
    # 0.2 在双 socket 并行 CI 下、对端容器并发加载模型导致
    # 可用内存瞬时低于阈值时仍会 OOM。
    @intel_amx_benchmark(
        extra_args=["--batch-size", "4", "--mem-fraction-static", "0.3"],
        min_throughput=40,
    )
    def test_latency_default_model(self):
        return DEFAULT_MODEL_NAME_FOR_TEST


class TestDPAttention(CustomTestCase):
    @classmethod
    def setUpClass(cls):
        cls.model = DEFAULT_MLA_MODEL_NAME_FOR_TEST
        cls.base_url = DEFAULT_URL_FOR_TEST
        other_args = [
            "--trust-remote-code",
            "--disable-radix-cache",
            "--attention-backend",
            "intel_amx",
            # DP 2 路启动同样提高预留比例，避免并发加载峰值
            "--mem-fraction-static",
            "0.3",
            "--disable-overlap-schedule",
            "--tp",
            "2",
            "--enable-dp-attention",
            "--dp",
            "2",
        ]
```

test/registered/cpu/test_intel_amx_attention_backend_b.py

FP8 量化模型（Qwen FP8 与 MoE）延迟测试同样调整 mem-fraction-static 到 0.3；这两个用例曾单独在 SPR2 job 中被点名失败。

```
class TestIntelAMXAttnBackendQuant(CustomTestCase):
    # FP8 模型测试同样把 mem-fraction-static 提到 0.3。
    # 0.2 在 CI 并发加载时会被 KV 预留逻辑推过内存预算
    # （见 commit 3614c66 的说明）。
    @intel_amx_benchmark(
```

```
        extra_args=["--batch-size", "4", "--mem-fraction-static", "0.3"],
        min_throughput=150,
    )
    def test_latency_fp8_qwen(self):
        return DEFAULT_MODEL_NAME_FOR_TEST_QWEN_FP8

    @intel_amx_benchmark(
        extra_args=["--batch-size", "4", "--mem-fraction-static", "0.3"],
        min_throughput=50,
    )
    def test_latency_fp8_moe_model(self):
        return DEFAULT_MODEL_NAME_FOR_TEST_FP8_WITH_MOE
```

评论区精华

mingfeima 对 deepseek UT 修改的价值反复质疑并最终促成回退，是本次最有价值的交锋：

- "why we need this one?"（mingfeima）→ "This UT is failing in CI. AI did not find the guilty commit. It's probably due to some framework changes that are not directly related to this UT."（Xia-Weiwen）→ "failing which CI? Xeon or CUDA?" → "Xeon"。
- 代码 review："do not patch parallel_state._MOE_EP with SimpleNamespace(world_size=1). V4 shared_experts_fusion_disable_reason does not read EP; You can do get_parallel().override(moe_ep_size=1)... Wrap every classmethod gate test is hiding the real dependency."（mingfeima）
- 随后 mingfeima 发现 setUp() 已统一做 override："just noticed that here: setUp() already applies get_parallel().override(moe_ep_size=1) for every test. Do we need to update this file?" —— 结论是不需要，最终整个文件改动被 revert。
- CI 层面，mingfeima 指出 "The modified FP8 benchmarks still fail in the Xeon SPR2 job with no parseable benchmark output. Please identify the subprocess failure or revert this memory change"，Xia-Weiwen 确认该问题由同一 PR 中 Mingxu 的 SPR 容器重构修复。
- deepseek UT 修改是否必要 (question): 确认 UT 在 Xeon CI 上失败，动机成立；但最终改动被 review 判定为不必要而回退。
- 不要 patch _MOE_EP，改用 override (design): 作者接受建议（"Thanks. Updated."），修改改为直接复用 override 模式，随后整个文件改动被 revert。
- setUp 已覆盖，无需改文件 (question): 结论是不需要改文件；最终该文件的全部改动被 revert。
- SPR2 上 FP8 benchmark 仍无输出 (testing): Xia-Weiwen 确认该问题由同一 PR 中 Mingxu 的 SPR 容器重构修复（"This is fixed in this PR by Mingxu."）。

风险与影响

- 风险：
  - 内存问题只是缓解而非根治：0.2→0.3 针对 CI 并发场景调整，若模型权重更大或并发加载瞬时峰值更高仍可能 OOM；且这是测试专用值，与生产部署的 mem_fraction_static

建议无关。

- SPR 并行度下降与单点风险：分片从 4 降到 3，整体测试时长可能增加；每台 SPR 盒子单容器承载 1/3 测试池，单台机器故障会导致 CI 覆盖明显缺失。
- 依赖宿主机缓存：挂载 ~/.cache/sgl_eval 后，MMLU 数据的正确性与新鲜度取决于宿主机缓存，缓存不完整或过期时测试可能读到旧数据或仍失败。
- DeepSeek UT 问题未根治：回退 test_deepseek_v4_shared_expert_fusion.py 改动后，若该 UT 在 Xeon 环境仍因框架原因失败，后续 CI 仍会报红，需要另找根因。
- 工作流改动无单元测试覆盖，YAML 语法错误只能在真实 runner 上暴露。

- 影响：
  - 用户侧：无功能性 API 或模型行为变化，纯 CI/ 测试基础设施。
  - 系统侧：XEON CI（GNR + SPR 盒子）稳定性显著提升：MMLU 不再依赖外网下载、内存 OOM 消除、容器模型简化。
  - 团队侧：维护者需要知晓 SPR 容器从 socket 划分为整机划分的策略变化，以及测试参数与 CI 拓扑的耦合（mem-fraction 与容器内存机制相关）。
  - 影响程度：中低，属于工程基建，不涉及线上推理路径。
  - 风险标记：CI 拓扑变更，内存参数临时调优，依赖宿主机缓存，深层根因未根治，测试并行度下降

关联脉络

- PR #36605 [CI] Graceful teardown for the radix_cache server fixtures: 同为 CI 稳定性修复：radix_cache 测试夹具由强杀改为优雅关闭，避免 CI GPU 未空闲失败；与本 PR 一样属于 CI 资源 / 进程生命周期治理。
- PR #36589 fix: kill_process_tree waits for the reap by default: kill_process_tree 默认等待回收，修复 GPU 资源泄漏；与本 PR 关注的 CI 内存 / 进程资源管理问题同源。
- PR #36639 [CI] Fix Q8KV8 sparse prefill test fixture: 同为 CI 测试夹具修复，属于同一批 CI 稳定性维护脉络。