PR #34483 完整报告

sgl-project/sglang

[AMD] CI: cut two setup cycles from the AMD multimodal-gen lanes

合并时间：2026-08-21 13:01

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34483

执行摘要

- 一句话：AMD 多模态 CI 削减 4 个 setup 周期，约省 8 runner- 小时
- 推荐动作：建议精读本 PR 以及其基础提交（如 run_suite.py 分片分配与平衡逻辑），因为它展示了 CI 编排中基于实测数据做资源重组的完整思路：先量化 setup 成本，再选择“折叠 / 收敛”而不是简单增加超时，并保持失败隔离与独立超时。值得关注的设计决策是 if: (success() || failure()) 的失败隔离手法，以及“分片数量可由调度器自由选择”的弹性做法。

功能与动机

PR body 指出：在 run 31443692177 中，AMD 多模态任务每个 job 需承担 27–97 分钟的容器拉取和 23–98 分钟的依赖安装，总计每 job 100–155 分钟固定成本，而实际测试仅占约 5.6 小时对比 setup 约 15.8 runner- 小时，约 74% 开销被浪费；作为对照，CUDA 的依赖安装仅需 27 秒。因此作者希望去掉整个 setup 周期，而不是继续优化矩阵调参。

实现拆解

1. 将 unit 套件折叠进 1-GPU 分片：在 .github/workflows/pr-test-amd.yml 与 pr-test-amd-rocm720.yml 的 multimodal-gen-test-1-gpu-amd / -rocm720 任务 part 0 中新增 Run diffusion unit tests 步骤，if: matrix.part == 0 && (success() || failure()) 保证 diffusion 步骤失败也执行，并设 timeout-minutes: 30 独立预算。
2. 2-GPU 套件由 3 分片改为 2 分片：将 multimodal-gen-test-2-gpu-amd / -rocm720 的 part: [0, 1, 2] 改为 [0, 1]，并把 run_suite.py --total-partitions 3 改为 --total-partitions 2。依据同一 run 的逐用例耗时（14/26/21 分钟健康工作），两分片各约 30 分钟用例时间，即使 flux 重试仍在 180 分钟超时内。
3. 移除独立 unit 任务：删除 multimodal-gen-unit-test-amd / -rocm720 整个 job 以及其在 target_stage_select、needs 中的条目。
4. 更新覆盖率报告：在 scripts/ci/utils/ci_coverage_report.py 中更新注释，说明 unit 套件现作为 multimodal-gen-test-1-gpu-amd part 0 的步骤在 ROCm 上运行，从而保留 "unit": ("CUDA", "AMD") 映射。
5. 配套修复：合入的提交还包含为 2-GPU 分片配置传递预计算分区分配、确保 standalone 文件在用例失败后继续运行、以及为每个分片数量测试完整套件调度的辅助改动（run_suite.py 相关），这些改动是提前为本次分片收敛打的基础。

关键文件：

- .github/workflows/pr-test-amd.yml（模块 CI 工作流；类别 infra；类型 infrastructure）：AMD 主 CI 工作流：折叠 unit 套件到 1-GPU part 0、2-GPU 分片 3→2 并更新 total-partitions、删除独立 unit job 及其 target_stage 条目。
- .github/workflows/pr-test-amd-rocm720.yml（模块 CI 工作流；类别 infra；类型 infrastructure）：ROCm 7.2 对应工作流，与 pr-test-amd.yml 保持同步修改，是 PR 实际验证的 lane。
- scripts/ci/utils/ci_coverage_report.py（模块 覆盖率脚本；类别 infra；类型 infrastructure）：更新 unit 套件覆盖矩阵的来源注释，说明其已在 ROCm 上作为 1-GPU job 的 step 运行，保持 coverage 报告准确。

关键符号：未识别

评论区精华

PR 内仅作者在 Issue/PR 评论中记录验证情况，无实质性 review 讨论。作者在评论中说明了 run-ci 无法覆盖本 PR 修改的 jobs：因为 check-changes 的 multimodal_gen 输出为 false，除非改动该路径的代码否则 1-GPU/2-GPU 任务在 gate 中会被跳过，因此只能通过手动 dispatch ROCm 7.2 lane 来验证；最终 run 31638833841 和 31660972923 的结果确认 unit fold 工作正常（diffusion 失败后单元步骤仍执行并通过 1360 个测试），且 2-GPU 分片在超时内完成。此外作者指出了已知的 ROCm flux 缺陷 #34351/#34352 导致的重试放大问题，是当前最大时间消耗项，但本 PR 有意不处理。

- run-ci 无法覆盖被修改的 CI jobs (question): 作者通过手动 dispatch ROCm 7.2 lane 来验证修改，run 31638833841 和 31660972923 确认单元折叠与分片收敛有效。
- 2-GPU 分片超时余量不足 (performance): 作者决定不提高超时，因为更大的 cap 会使坏 job 占用稀缺的 mi300 runner 更久；当前 split 是安全的。

风险与影响

- 风险：风险主要集中 CI 基础设施层面： 1) 2-GPU 分片从 3 片减到 2 片后，每个分片承载的测试更多，若 flux 用例恢复或新增用例可能挤爆 180 分钟 step 超时（验证中 shard 0 已到 149 分钟，接近上限）；作者明确不提高超时以避免长时间的 runner 占用，但未来若某分片超时会导致 2-GPU 测试覆盖率下降。 2) unit 套件折叠进 1-GPU part 0 后，若 if: matrix.part == 0 && (success() || failure()) 在 Actions 中表现异常（如失败后不触发），会静默失去 unit 覆盖。 3) 修改了 ci_coverage_report.py 的注释，若该脚本后续解析这些注释或逻辑依赖任务名，会产生误导。 4) 删除独立 unit job 会让 AMD 覆盖矩阵在 dashboard 上消失，除非 coverage 报告正确映射。整体风险受限于 CI 编排，不涉及推理核心代码。
  - 影响：影响是 CI 层面的效率和资源消耗：AMD lane 上每次完整 PR 运行的 job 数从 8 个降至 5 个，预计节省约 4 个 setup 周期，约 8 runner- 小时 / 次 PR run；同时减少了对稀缺 mi300 runner 的占用（避免每个 job 单独拉镜像和装依赖）。对用户在模型推理能力上没有直接影响，但对 AMD 开发者体验有显著改善——验证时间更短、PR gate 更快。团队需要留意 2-GPU 分片在 flux 修复后的负载变化，以及 unit 步骤折叠后失败时的可观测性。

- 风险标记：CI 超时风险，资源密集型基础设施，缺少自动化回归测试覆盖 CI 改动

关联脉络

- PR #35764 [AMD][CI] Fix ROCm 7.0's dead apt index fail the MORI dependency install: 同为 AMD CI 稳定性改进，关注依赖安装阶段的可靠性，与本 PR 的 setup 成本优化同属 AMD CI 体验改善线路。
- PR #35654 [AMD] Retry transient network failures in ROCm Dockerfile curl fetches: 处理 ROCm Dockerfile 中 curl 网络重试，与本 PR 讨论的容器拉取耗时问题同属 AMD CI setup 成本范畴。
- PR #34204 [AMD] CI: 相关 gate 调整（PR 中提到的 ROCm 7.2 lane 成为 PR gate 的变更）: PR body 提到 ROCm 7.2 lane 是 #34204 之后的 PR gate，本 PR 的验证就是基于该 gate 执行的。