PR #22828 完整报告

sgl-project/sglang

ci: enable /rerun-test for multimodal gen PR tests

合并时间：2026-04-22 12:34

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22828

执行摘要

- 一句话：扩展 /rerun-test 命令以支持多模态生成测试的定向重新运行。
- 推荐动作：此 PR 值得 CI 基础设施维护者精读，重点关注 resolve_test_file 函数中的路径解析设计决策，包括多模态目录的集成方式、错误处理机制以及如何平衡新旧测试路径的兼容性。

功能与动机

根据 PR body 的描述，当前多模态生成测试只能通过 /rerun-stage 命令重新运行整个阶段（所有分区），这不够灵活且效率低下。此 PR 旨在启用针对单个测试文件的定向重新运行，例如使用命令如 /rerun-test python/sglang/multimodal_gen/test/server/test_server_a.py，以提高开发者的测试验证效率。

实现拆解

1. 扩展 slash 命令处理器的路径解析逻辑：修改 scripts/ci/utils/slash_command_handler.py 中的 resolve_test_file 函数，新增对多模态测试目录（python/sglang/multimodal_gen/test/）的支持，并引入 detect_multimodal_suite 函数来映射 GPU 数量到 runner 标签。关键变更包括添加多模态路径前缀检查、同时搜索 test/registered/ 和多模态目录以解析裸文件名，并返回 is_multimodal 标志以指示测试类型。
2. 更新 CI 工作流以处理扩散依赖：修改 .github/workflows/rerun-test.yml，新增 install_diffusion 输入参数（默认 "false"），在工作流中根据该参数分支安装步骤：当 install_diffusion == "true" 时，运行 ci_install_dependency.sh diffusion 来安装扩散模型依赖；同时，运行测试时对多模态测试使用 pytest 命令而非 unittest。
3. 测试验证配套：PR body 中列出了详细的测试计划，包括验证路径解析、runner 分发和依赖安装，Issue 评论中显示了作者通过实际命令测试了功能，确保变更正确工作。

关键文件：

- scripts/ci/utils/slash_command_handler.py（模块 CI 脚本；类别 infra；类型 infrastructure；符号 resolve_test_file, detect_multimodal_suite）：这是核心变更文件，负责扩展 /rerun-test 命令的路径解析逻辑以支持多模态测试目录，直接影响命令的可用性和正确性。

- .github/workflows/rerun-test.yml（模块 工作流；类别 infra；类型 infrastructure）：此文件更新了 CI 工作流配置，新增 install_diffusion 输入以正确处理多模态测试的依赖安装和运行命令，确保测试能正确执行。

关键符号：resolve_test_file, detect_multimodal_suite

关键源码片段

scripts/ci/utils/slash_command_handler.py

这是核心变更文件，负责扩展 /rerun-test 命令的路径解析逻辑以支持多模态测试目录，直接影响命令的可用性和正确性。

```
# 关键函数：resolve_test_file，用于解析用户提供的测试文件路径
def resolve_test_file(file_part):
    """
    解析用户提供的文件路径，返回（解析后路径，是否为多模态测试，错误消息）。
    支持多模态测试目录（如 python/sglang/multimodal_gen/test/）和现有 test/registered/ 目录。
    """
    # 定义多模态测试目录和路径到 runner 的映射
    MULTIMODAL_TEST_DIR = "python/sglang/multimodal_gen/test"
    MULTIMODAL_PATH_TO_RUNNER = {
        "2_gpu": "2-gpu-h100",
        "2-gpu": "2-gpu-h100",
    }
    MULTIMODAL_DEFAULT_RUNNER = "1-gpu-h100"

    # 检查是否明确为多模态路径
    multimodal_prefixes = [
        "python/sglang/multimodal_gen/test/",
        "sglang/multimodal_gen/test/",
        "multimodal_gen/test/",
    ]
    for prefix in multimodal_prefixes:
        if file_part.startswith(prefix):
            # 构建完整路径，确保以 python/ 开头
            full_path = (
                file_part
                if file_part.startswith("python/")
                else f"python/sglang/multimodal_gen/test/{file_part[len(prefix):]}"
            )
            if not os.path.isfile(full_path):
                return None, False, f"File not found: `{full_path}`" # 文件不存在时返回错误
            return full_path, True, None # 成功返回多模态路径和标志

    # 现有逻辑：处理 test/registered/ 路径
    if file_part.startswith("test/"):
        file_part = file_part[len("test/") :]
    if "/" not in file_part:
        # 同时搜索 test/registered/ 和多模态目录以处理裸文件名
```

```
        matches = glob.glob(f"test/registered/**/{file_part}", recursive=True)
        mm_matches = glob.glob(f"{MULTIMODAL_TEST_DIR}/**/{file_part}", recursive=True)
        mm_matches = [m for m in mm_matches if os.path.basename(m).startswith("test_")] #
        过滤测试文件

        # 处理匹配结果：唯一匹配时返回，否则报告错误
        if len(matches) == 1 and len(mm_matches) == 0:
            return matches[0][len("test/") :], False, None
        if len(matches) == 0 and len(mm_matches) == 1:
            return mm_matches[0], True, None
        all_matches = matches + mm_matches
        if len(all_matches) == 0:
            return None, False, f"No test file found matching `{file_part}` under `test/registered/`
            or `{MULTIMODAL_TEST_DIR}/`."
        if len(all_matches) > 1:
            match_list = "\n".join(f"- `{m}`" for m in sorted(all_matches))
            return None, False, f"Ambiguous filename `{file_part}` — matched {len(all_matches)}
            files:

    {match_list}

    Please provide the full path, e.g. `/rerun-test {all_matches[0]}`"
        # 其余逻辑省略 ...
```

评论区精华

Review 中仅有一条来自 mickqian 的批准评论，没有实质性讨论或争议点。这表明变更得到了快速认可，可能因为逻辑清晰且与现有 CI 基础设施一致。

- 测试验证与功能确认 (testing): 变更通过了初步测试，功能按预期工作，但未涉及设计或性能的深入讨论。

风险与影响

- 风险：技术风险包括： 1) 路径解析错误：在 slash_command_handler.py 中，新增的多模态路径逻辑可能导致歧义或文件未找到错误，特别是在处理裸文件名时，如果文件同时在 test/registered/ 和多模态目录中存在，会返回错误信息，但已通过错误处理缓解。 2) 依赖安装问题：在 rerun-test.yml 中，新增的 install_diffusion 分支可能导致扩散依赖安装失败，影响测试运行；需要确保 ci_install_dependency.sh diffusion 脚本稳定。 3) 兼容性风险：变更可能影响现有 /rerun-test 命令对其他非多模态测试的行为，但修改保留了原有逻辑，风险较低。
  - 影响：对用户（开发者）的影响：显著提升测试效率，支持针对多模态生成测试的精细重跑，减少等待时间。对系统（CI 流水线）的影响：扩展了 slash 命令的功能，增加了工作流配置的复杂度，但通过分组键（包括 install_diffusion）确保多模态测试被单独批处理，避免干扰其他测试。对团队的影响：促进更快的迭代和调试，特别是在多模态开发场景中。
  - 风险标记：路径解析复杂性，依赖安装分支风险

关联脉络

- PR #22830 ci: enable /rerun-test for nightly test suites: 该 PR 同样扩展了 /rerun-test 命令的功能（用于夜间测试套件），与本 PR 在 CI 基础设施和 slash 命令处理上有直接关联，展示了持续改进 CI 工具能力的趋势。