PR #26422 完整报告

sgl-project/sglang

[CI] /rerun-test: support glob wildcard patterns

合并时间：2026-05-28 12:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26422

执行摘要

- 一句话：CI /rerun-test 支持 glob 通配符模式
- 推荐动作：值得精读，特别是 expand_glob_spec 的实现和安全约束的设计，可以作为 CI 工具开发的参考。也可关注去重逻辑的演进。

功能与动机

PR Body 明确指出：/rerun-test already accepts multiple files and groups them by dispatch shape, but a wildcard arg silently fails today…When rerunning a whole family of tests, hand-enumerating every file is tedious.

实现拆解

1. 判断 glob 模式：新增 _is_glob_pattern 函数，检查参数是否包含 *、?、[等元字符。
2. 展开通配符：新增 expand_glob_spec 函数，根据是否包含路径分隔符采用不同 glob 搜索策略，并过滤出符合 test_*.py 约束的文件。
3. 集成到命令处理：修改 handle_rerun_test 函数，在解析参数前调用 expand_glob_spec 进行展开，展开后的文件流经原有分组分发流程。
4. 去重机制：第 4 次提交中改为基于解析后的文件路径去重，而非原始字符串，解决了显式文件与 glob 模式重叠导致重复分发的问题。
5. 约束与兼容：不允许 glob 模式携带 ::test 选择器；普通参数保持原有唯一匹配行为。

同时更新了 .claude/skills/ci-workflow-guide/SKILL.md 文档，描述了新用法。

关键文件：

- scripts/ci/utils/slash_command_handler.py（模块 CI 脚本；类别 infra；类型 infrastructure；符号 _is_glob_pattern, expand_glob_spec, _under_test_root, handle_rerun_test）：核心实现文件，新增 glob 展开和去重逻辑
- .claude/skills/ci-workflow-guide/SKILL.md（模块 技能文档；类别 docs；类型 documentation）：文档更新，描述 glob 用法与命令语法

关键符号：_is_glob_pattern, expand_glob_spec, handle_rerun_test, handle_rerun_group

关键源码片段

scripts/ci/utils/slash_command_handler.py

核心实现文件，新增 glob 展开和去重逻辑

```
import glob
import os

# 定义 glob 元字符元组
_GLOB_METACHARS = ("*", "?", "[")

def _is_glob_pattern(file_part):
    """检查 file_part 是否包含 glob 元字符"""
    return any(ch in file_part for ch in _GLOB_METACHARS)

def expand_glob_spec(file_part):
    """
    将通配符模式展开为匹配的测试文件列表（仓库相对路径）。

    若模式包含路径分隔符（"/"），则从仓库根目录和 test/ 目录分别 glob；
    否则（裸模式）则在 test/registered/ 和多模态测试目录下递归搜索。
    结果经过过滤：仅返回位于已知测试根目录下、以 test_ 开头、.py 结尾的文件。

    Returns (文件列表, 错误信息)，成功时错误信息为 None。
    """
    pat = file_part
    # 若模式以 test/ 开头，则剥离，因为后面会拼接
    if pat.startswith("test/"):
        pat = pat[len("test/"):]

    matches = set()
    if "/" in pat:
        # 路径模式：从仓库根目录和 test/ 下 glob（支持 ** 递归）
        for base in (".", "test"):
            matches.update(glob.glob(os.path.join(base, pat), recursive=True))
    else:
        # 裸模式：在已知测试根目录下递归搜索
        for root in ("test/registered", MULTIMODAL_TEST_DIR):
            matches.update(glob.glob(os.path.join(root, "**", pat), recursive=True))

    def _under_test_root(path):
        """判断 path 是否位于允许的测试根目录下"""
        return path.startswith("test/registered/") or path.startswith(
            MULTIMODAL_TEST_DIR + "/"
        )

    files = sorted(
        {
            os.path.normpath(p)
            for p in matches
            if os.path.isfile(p)
```

```
            and os.path.basename(p).startswith("test_")
            and p.endswith(".py")
            and _under_test_root(os.path.normpath(p))
        }
    )
    if not files:
        return [], (
            f"No test files matched wildcard `{file_part}` under "
            f"`test/registered/` or `{MULTIMODAL_TEST_DIR}/` "
            f"(patterns only match files named `test_*.py`)."
        )
    return files, None
```

评论区精华

在 PR 评论中，作者测试了多种 glob 模式（如 test_*backend*.py、test_intel_amx_attention_backend_[ab].py 等），验证了匹配效果。合著者 ch-wan 测试了显式文件与 glob 混合的场景，发现了重复分发问题，作者在第 4 次提交中通过基于解析后身份去重解决了该问题。核心设计讨论围绕安全约束：只有 test/registered/ 和 multimodal test 目录下且文件名以 test_ 开头的文件才被匹配，防止误匹配非测试文件。

- 显式文件与 glob 模式重叠导致重复分发 (correctness): 作者在第四次提交中改为基于解析后的文件身份去重，而非基于原始字符串，解决了该问题。
- glob 通配符的安全约束 (design): 该设计被接受，无争议。

风险与影响

- 风险：主要风险：glob 模式可能匹配过多文件，导致 CI 资源消耗增加；通配符可能被恶意利用来触发大量测试。但已通过限制测试根目录和文件名模式缓解，且匹配结果通过 skip_permission_check 保护。另外，修改了 handle_rerun_test 函数签名（增加 command_label 参数），可能影响调用方，经检查所有调用点已更新。
- 影响：影响范围：主要影响使用 /rerun-test 命令的开发者和 CI 系统。开发效率提升，可以快速重试一组测试，无需逐个列举。CI 回复评论现在会带上原始命令标识，便于区分多个并发命令。对现有非通配符用法完全向后兼容，无用户端影响。
- 风险标记：glob 可能匹配大量文件，向后兼容需验证，函数签名变更影响调用

关联脉络

- 暂无明显关联 PR