PR #23576 完整报告

sgl-project/sglang

[diffusion] CI: minor refactor CI

合并时间：2026-04-24 08:48

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23576

执行摘要

- 一句话：简化扩散测试用例组织结构，合并列表并重命名文件
- 推荐动作：该 PR 属于常规维护重构，不涉及功能变更。建议阅读 gpu_cases.py 和 diffusion_case_parser.py 的变更，了解测试组织方式的新约定。"1-gpu-b200" 测试套件的文件引用已更新，注意保持同步。

功能与动机

简化扩散 CI 的测试用例组织方式，减少碎片化列表，使开发者更容易添加和查找测试用例。

实现拆解

1. 合并测试用例列表（python/sglang/multimodal_gen/test/server/gpu_cases.py）：将 ONE_GPU_CASES_A、ONE_GPU_CASES_B、ONE_GPU_CASES_C 合并为 ONE_GPU_CASES，并在文件末尾使用 += 将 ModelOpt 用例追加到 ONE_GPU_CASES 中；同理，将 TWO_GPU_CASES_A 和 TWO_GPU_CASES_B 合并为 TWO_GPU_CASES，并移除原有的拼接语句。
2. 重命名 ModelOpt 列表（gpu_cases.py）：将 ONE_GPU_CASES_C 改为 ONE_GPU_MODELOPT_CASES，更清晰地表明它包含 ModelOpt（量化）测试用例。
3. 重命名测试文件（test_server_c.py → test_server_b200.py）：将 B200 专用的测试文件更名，并更新其导入语句使用新的列表名 ONE_GPU_MODELOPT_CASES。
4. 更新配套配置（testcase_configs.py、run_suite.py）：修改文档注释中的列表引用，以及 run_suite.py 中 FILE_SUITES 的条目，将 test_server_c.py 替换为 test_server_b200.py。
5. 增强用例解析器（scripts/ci/utils/diffusion/diffusion_case_parser.py）：新增 visit_AugAssign 方法以处理 += 赋值，新增 _extract_case_ids 方法支持变量引用和二进制加法操作，并在 _extract_case_ids_from_list 中处理 ast.Starred 解包，最后添加去重逻辑以保持列表顺序。

关键文件：

- python/sglang/multimodal_gen/test/server/gpu_cases.py（模块 测试用例；类别 test；类型 test-coverage）：核心变更文件，合并了所有扩散测试用例列表并重命名了 ModelOpt 列表

- scripts/ci/utils/diffusion/diffusion_case_parser.py（模块 CI 脚本；类别 infra；类型 infrastructure；符号 visit_AugAssign, _extract_case_ids_from_list, _process_aug_assignment, _extract_case_ids）：增强解析器以支持 += 赋值和变量引用，确保新结构能被正确解析
- python/sglang/multimodal_gen/test/server/test_server_b200.py（模块 测试脚本；类别 test；类型 rename-or-move）：文件重命名并更新导入，与 gpu_cases.py 的新列表名对齐
- python/sglang/multimodal_gen/test/server/testcase_configs.py（模块 测试配置；类别 test；类型 test-coverage）：更新文档注释中的列表名称引用
- python/sglang/multimodal_gen/test/run_suite.py（模块 运行脚本；类别 test；类型 test-coverage）：更新文件引用以匹配重命名后的测试文件

关键符号：visit_AugAssign, _process_aug_assignment, _extract_case_ids, _extract_case_ids_from_list

关键源码片段

python/sglang/multimodal_gen/test/server/gpu_cases.py

核心变更文件，合并了所有扩散测试用例列表并重命名了 ModelOpt 列表

```
# python/sglang/multimodal_gen/test/server/gpu_cases.py
# 合并前：ONE_GPU_CASES_A, ONE_GPU_CASES_B, ONE_GPU_CASES_C 三个独立列表
# 合并后：统一为 ONE_GPU_CASES，并在末尾追加 ModelOpt 用例
ONE_GPU_CASES: list[DiffusionTestCase] = [
    # === Text to Image (T2I) ===
    DiffusionTestCase("qwen_image_t2i", ...),
    DiffusionTestCase("qwen_image_t2i_cache_dit_enabled", ...),
    # ... 原有 A 和 B 的所有用例 ...
    DiffusionTestCase("flux_2_image_t2i_upscaling_4x", ...),
    # === Text to Video (T2V) ===（原来在 B 中）
    DiffusionTestCase("wan2_1_t2v_1.3b", ...),
    # ...
]
# 平台条件追加（原来分别追加到 A/B）
if not current_platform.is_hip():
    ONE_GPU_CASES.append(DiffusionTestCase("hunyuan3d_shape_gen", ...))
    ONE_GPU_CASES.append(DiffusionTestCase("turbo_wan2_1_t2v_1.3b", ...))
if current_platform.is_hip():
    ONE_GPU_MODELOPT_CASES = []
else:
    ONE_GPU_MODELOPT_CASES = [
        _make_modelopt_ci_case("flux1_modelopt_fp8_t2i", ...),
        # ...
    ]
# 使用 += 合并（简化前是通过 [*A, *B, *C] 拼接）
ONE_GPU_CASES += ONE_GPU_MODELOPT_CASES

# 2-GPU 同理合并
```

```
TWO_GPU_CASES = [
    DiffusionTestCase("wan2_2_i2v_a14b_2gpu", ...),
    # ... 原来 A 和 B 的所有用例 ...
    DiffusionTestCase("flux_2_ti2i_multi_image_cache_dit", ...),
]
TWO_GPU_CASES = _with_default_num_gpus(TWO_GPU_CASES, 2)
```

scripts/ci/utils/diffusion/diffusion_case_parser.py

增强解析器以支持 += 赋值和变量引用，确保新结构能被正确解析

```
# scripts/ci/utils/diffusion/diffusion_case_parser.py
class CaseCollector(ast.NodeVisitor):
    # ...
    def visit_AugAssign(self, node: ast.AugAssign):
        """处理 += 这类赋值，用于合并列表"""
        self._process_aug_assignment(node.target, node.op, node.value)
        self.generic_visit(node)

    def _process_aug_assignment(self, target, op, value):
        """执行 += 的语义：将右侧case ID追加到左侧列表"""
        if not isinstance(target, ast.Name) or not isinstance(op, ast.Add):
            return
        rhs_case_ids = self._extract_case_ids(value)
        if rhs_case_ids is None:
            return
        lhs_case_ids = self.cases.get(target.id, [])
        # 合并并保留顺序
        self.cases[target.id] = [*lhs_case_ids, *rhs_case_ids]

    def _extract_case_ids(self, node):
        """通用提取：支持列表字面量、变量引用、列表加法"""
        if isinstance(node, ast.List):
            return self._extract_case_ids_from_list(node)
        if isinstance(node, ast.Name):
            return list(self.cases.get(node.id, [])) # 引用之前解析过的变量
        if isinstance(node, ast.BinOp) and isinstance(node.op, ast.Add):
            left_ids = self._extract_case_ids(node.left)
            right_ids = self._extract_case_ids(node.right)
            if left_ids is None or right_ids is None:
                return None
            return [*left_ids, *right_ids]
        return None

    def _extract_case_ids_from_list(self, node: ast.List):
        """从列表字面量中提取case ID"""
        case_ids = []
        for elt in node.elts:
            if isinstance(elt, ast.Starred):
                # 处理 *list 展开
```

```
                starred = self._extract_case_ids(elt.value)
                if starred:
                    case_ids.extend(starred)
                continue
            case_id = self._extract_case_id_from_call(elt)
            if case_id:
                case_ids.append(case_id)
        return case_ids
    # 后续还添加了去重逻辑：保留首次出现顺序
```

评论区精华

只收到一个来自 gemini-code-assist[bot] 的评论，建议将 ONE_GPU_CASES += ONE_GPU_MODELOPT_CASES 改为显式列表连接 ONE_GPU_CASES = ONE_GPU_CASES + ONE_GPU_MODELOPT_CASES，以避免就地修改可能带来的副作用。该建议未被采纳，PR 已合入当前实现。

- 使用 += vs 显式列表连接 (design): PR 合入时未采纳建议，保留了 += 写法。

风险与影响

- 风险：低风险。变更仅限于测试基础设施和配置，不涉及核心推理逻辑。主要风险在于：
  - 如果 diffusion_case_parser.py 的解析逻辑存在边界情况（如嵌套 AugAssign），可能导致 CI 解析失败。
  - 测试文件的移动和重命名可能影响历史 CI 配置或外部脚本引用。
  - 测试用例合并后，若某个用例被重复添加，去重逻辑可能隐藏问题。
  - 影响：影响范围：仅影响扩散（diffusion）测试套件的维护者和 CI 运行。 影响程度：低。开发者添加新测试用例时只需操作 ONE_GPU_CASES 或 TWO_GPU_CASES，不再需要关心多个子列表。重命名和文件移动带来短暂的混乱，但整体简化了流程。
- 风险标记：测试配置变更，文件重命名

关联脉络

- 暂无明显关联 PR