

PR #55305 完整报告

vllm-project/vllm

[Bugfix][Responses] Fix browser.find action type

合并时间: 2026-09-11 20:07

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55305>

执行摘要

- 一句话: 修正 browser.find 动作类型, 修复 Responses 双路径校验崩溃
- 推荐动作: 建议精读, 但阅读重点不在 2 行字面量修改, 而在两处结构性信号: 一是「同一语义在流式与非流式两条路径重复实现」的风险, 可考虑后续把 browser action 的构造抽成共用的工厂函数, 让 action 类型与字段映射只定义一次; 二是新增测试的断言强度不对称 (非流式只断言 pattern、流式断言 url), 若继续迭代 browser.* 系列, 建议补齐非流式路径的 url/cursor 字段断言并考虑加一个最小的 Responses 端点级测试, 用真实 SDK 反序列化输出以验证契约。对只关心修复本身的人, 看 `harmony.py::_parse_browser_tool_call` 与 `streaming_events.py::emit_browser_tool_events` 两处改动即可。

功能与动机

Issue #55295 明确描述了故障链路: Harmony 侧使用 `recipient="browser.find"`, 但 Responses API 的 web search action 规范要求 `type="find_in_page"`; 在上游 main (69cf0559) 上 vLLM 对两条路径都传了 `type="find"`, 于是「Both paths raise a Pydantic ValidationError instead of returning the output item/events」, 报错信息为 `ValidationError: Input should be 'find_in_page'`。Issue 还附上了 `openai-python v2.16.0` 与 `v2.17.0` 的 `response_function_web_search.py` 定义截图, 证明该字面量在多个 SDK 版本中都是强校验的枚举, 属于与 SDK 契约不一致导致的确定性崩溃, 而非偶发问题。

实现拆解

1. 确认故障入口与影响面: 问题不在 Harmony 解析器本身, 而在把 Harmony message 映射为 OpenAI Responses 输出项的转换层。仓库中同一语义存在两条独立实现路径: 非流式的消息转换与流式的事件构造, 二者各自构造 `ActionFind`, 因此必须同步修改, 否则只有一种调用方式被修好。
2. 修复非流式转换路径: 在 `vllm/entrypoints/openai/responses/harmony.py` 的 `_parse_browser_tool_call` 中, `recipient == "browser.find"` 分支的 `ActionFind(..., type="find")` 改为 `type="find_in_page"`。该分支位于 `browser.search / browser.open` 的同级 if-elif 链中, 返回统一的 `ResponseFunctionWebSearch(type="web_search_call")` 对象; 修改后 `json.loads` 失败时的占位降级逻辑与 `status="completed"` 语义都保持不变, 只是让 action 通过 SDK 的 Pydantic 校验。
3. 修复流式事件构造路径: 在 `vllm/entrypoints/openai/responses/streaming_events.py` 的 `emit_browser_tool_events` 中, `function_name == "find"` 分支同样把

`response_function_web_search.ActionFind(type="find")` 改为 `type="find_in_page"`。该 action 会同时挂在 `response.output_item.added` 与后续 `response.output_item.done` 事件的 item 上，因此这一处字面量影响整个 web search 生命周期事件序列（in_progress / searching / completed / done）。

4. 测试配套（两条路径各一个回归用例）：`tests/entrypoints/openai/responses/test_harmony_utils.py` 新增 `test_browser_find_recipient_uses_responses_action_type`，直接构造 `Message.from_role_and_content(...).with_channel("analysis").with_recipient("browser.find")` 并调用 `harmony_to_response_output`；`tests/entrypoints/openai/responses/test_streaming_events.py` 新增 `test_browser_find_uses_responses_action_type`，调用 `emit_browser_tool_events(message, StreamingState())` 并同时断言 `events[0]`（added）与 `events[-1]`（done）两个事件的 `action.type == "find_in_page"`。两个用例都复用了 `{"pattern": "vLLM", "cursor": 42}` 这一输入以贴近 Harmony 的真实参数形态，同时为测试文件补上了 `ResponseFunctionWebSearch`、`Message`、`Role`、`StreamingState`、`emit_browser_tool_events` 等导入。
5. 无配置与部署改动：未新增环境变量、未改依赖版本（未锁定 openai / openai-harmony 版本），也未改文档；修复完全依赖 SDK 现成的字面量定义，因此不引入部署侧动作。

关键文件：

- `vllm/entrypoints/openai/responses/harmony.py`（模块 响应转换；类别 source；类型 core-logic；符号 `_parse_browser_tool_call`, `ActionFind`）：非流式转换入口：`_parse_browser_tool_call` 是 `browser.find` → `ActionFind` 的唯一构造点，本次把 `type` 从 "find" 改为 "find_in_page"，直接决定了非流式 Responses 请求是否还会抛 `Pydantic ValidationError`。
- `vllm/entrypoints/openai/responses/streaming_events.py`（模块 流式事件；类别 source；类型 core-logic；符号 `emit_browser_tool_events`, `ActionFind`）：流式事件构造入口：`emit_browser_tool_events` 在 `find` 分支构造 `ActionFind` 并挂到 `added/done` 事件上，是 issue 中「streaming event builder」那条崩溃路径的修复点。
- `tests/entrypoints/openai/responses/test_harmony_utils.py`（模块 单元测试；类别 test；类型 test-coverage；符号 `test_browser_find_recipient_uses_responses_action_type`）：为非流式转换路径补回归测试 `test_browser_find_recipient_uses_responses_action_type`，直接锁定 `harmony_to_response_output` 对 `browser.find` 的输出类型。
- `tests/entrypoints/openai/responses/test_streaming_events.py`（模块 单元测试；类别 test；类型 test-coverage；符号 `test_browser_find_uses_responses_action_type`）：为流式事件路径补回归测试 `test_browser_find_uses_responses_action_type`，同时断言 `added` 与 `done` 两个事件，覆盖旧代码上会抛错的实际调用链。

关键符号：`_parse_browser_tool_call`, `emit_browser_tool_events`, `harmony_to_response_output`, `test_browser_find_recipient_uses_responses_action_type`, `test_browser_find_uses_responses_action_type`

关键源码片段

`vllm/entrypoints/openai/responses/harmony.py`

非流式转换入口：_parse_browser_tool_call 是 browser.find → ActionFind 的唯一构造点，本次把 type 从 "find" 改为 "find_in_page"，直接决定了非流式 Responses 请求是否还会抛 Pydantic ValidationError。

```
def _parse_browser_tool_call(message: Message, recipient: str) -> ResponseOutputItem:
    """把 Harmony 的 browser.* 工具调用转换成 Responses 的 web_search_call 输出项。"""

    # content 由调用方从 message.content 中取出；Harmony 把工具参数放在正文里。
    # 解析失败时降级为占位文本，保证响应结构仍然合法，而不是直接抛异常中断请求。
    try:
        browser_call = json.loads(content.text)
    except json.JSONDecodeError:
        logger.warning(
            "Invalid JSON in browser tool call, using error placeholder: %s",
            content.text,
        )
        json_retry_output_message = (
            f"Invalid JSON args, caught and retried: {content.text}"
        )
        browser_call = {
            "query": json_retry_output_message,
            "url": json_retry_output_message,
            "pattern": json_retry_output_message,
        }

    # 按 recipient 分派到 OpenAI SDK 的 action 联合类型。每个 action 的 type
    # 都是 Pydantic 的 Literal 字段，取值必须与 SDK 定义逐字一致，否则在构造对象
    # 这一步就会抛 ValidationError，这正是 Issue #55295 的根因。
    if recipient == "browser.search":
        action = ActionSearch(
            query=f"cursor:{browser_call.get('query', '')}", type="search"
        )
    elif recipient == "browser.open":
        action = ActionOpenPage(
            url=f"cursor:{browser_call.get('url', '')}", type="open_page"
        )
    elif recipient == "browser.find":
        # 本次修复点：旧值 "find" 会被 SDK 拒绝，正确字面量是 "find_in_page"。
        # 注意这里取的是 url 字段，而流式路径取的是 cursor 字段，两处字段名并不一致。
        action = ActionFind(
            pattern=browser_call.get("pattern", ""),
            url=f"cursor:{browser_call.get('url', '')}",
            type="find_in_page",
        )
    else:
        raise ValueError(f"Unknown browser action: {recipient}")

    # 统一包裹为 web_search_call 输出项，action 的合法与否决定整个 item 能否构造成功
    return ResponseFunctionWebSearch(
```

```

        id=f"ws_{random_uuid()}",
        action=action,
        status="completed",
        type="web_search_call",
    )

```

vllm/entrypoints/openai/responses/streaming_events.py

流式事件构造入口：emit_browser_tool_events 在 find 分支构造 ActionFind 并挂到 added/done 事件上，是 issue 中「streaming event builder」那条崩溃路径的修复点。

```

def emit_browser_tool_events(
    previous_item: HarmonyMessage,
    state: StreamingState,
) -> list[StreamingResponsesResponse]:
    """为 browser.* 工具调用（web search 系列）生成流式事件。"""

    # recipient 形如 "browser.find", 去掉前缀得到具体动作名；
    # 参数同样来自 Harmony 正文里的 JSON 字符串。
    function_name = previous_item.recipient[len("browser.")]
    parsed_args = json.loads(previous_item.content[0].text)
    action = None

    if function_name == "search":
        action = response_function_web_search.ActionSearch(
            type="search",
            query=parsed_args["query"],
        )
    elif function_name == "open":
        action = response_function_web_search.ActionOpenPage(
            type="open_page",
            # TODO: translate to url
            url=f"cursor:{parsed_args.get('cursor', '')}",
        )
    elif function_name == "find":
        # 与 harmony.py 的非流式路径保持一致：SDK 要求 ActionFind.type 为
        # "find_in_page", 否则构造 added / done 事件时会直接抛 Pydantic 校验错误。
        # 这里读取的是 cursor 字段，而非流式路径读取的是 url 字段。
        action = response_function_web_search.ActionFind(
            type="find_in_page",
            pattern=parsed_args["pattern"],
            # TODO: translate to url
            url=f"cursor:{parsed_args.get('cursor', '')}",
        )
    else:
        raise ValueError(f"Unknown function name: {function_name}")

    state.current_item_id = f"tool_{random_uuid()}"
    events: list[StreamingResponsesResponse] = []

```

```

# added 事件中携带的 action 必须已通过校验，否则整条事件流在首个事件就断掉
events.append(
    ResponseOutputItemAddedEvent(
        type="response.output_item.added",
        sequence_number=-1,
        output_index=state.current_output_index,
        item=response_function_web_search.ResponseFunctionWebSearch(
            # TODO: generate a unique id for web search call
            type="web_search_call",
            id=state.current_item_id,
            action=action,
            status="in_progress",
        ),
    )
)
events.append(
    ResponseWebSearchCallInProgressEvent(
        type="response.web_search_call.in_progress",
        sequence_number=-1,
        output_index=state.current_output_index,
        item_id=state.current_item_id,
    )
)
# 函数后半段还会追加 searching / completed 与 output_item.done 事件；
# 新增测试断言 events[-1] (done) 同样携带 find_in_page action，确保整条链路一致。

```

tests/entrypoints/openai/responses/test_streaming_events.py

为流式事件路径补回归测试 test_browser_find_uses_responses_action_type，同时断言 added 与 done 两个事件，覆盖旧代码上会抛错的实际调用链。

```

def test_browser_find_uses_responses_action_type():
    """Both streaming output items must use the Responses find action type."""
    # 用 Harmony 的真实调用形态构造输入：analysis 频道 + browser.find 收件人
    message = (
        Message.from_role_and_content(
            Role.ASSISTANT, '{"pattern": "vLLM", "cursor": 42}'
        )
        .with_channel("analysis")
        .with_recipient("browser.find")
    )

    events = emit_browser_tool_events(message, StreamingState())

    # 同时取首尾事件：added 事件在旧代码上就会抛 ValidationError，
    # done 事件复用同一 action 语义，一并断言可防止只修好前半段的情况。
    added, done = events[0], events[-1]
    assert added.type == "response.output_item.added"
    assert done.type == "response.output_item.done"
    for event in (added, done):

```

```
assert isinstance(event.item, ResponseFunctionWebSearch)
assert event.item.action.type == "find_in_page"
assert event.item.action.pattern == "vLLM"
# 流式路径从 cursor 字段拼 url, 因此这里期望 cursor:42
assert event.item.action.url == "cursor:42"
```

评论区精华

本 PR 没有产生任何内联 review 评论 (review_comments_count = 0), 讨论主要围绕校验、CI 与合并流程:

1. 自动化 review 缺失: [claude\[bot\]](#) 明确说明「This pull request is from a fork — automated review is disabled. A repository maintainer can comment [@claude review](#) to run a one-time review.」, 即本次合并没有经过自动化深度审查, 人工审查也止步于批准。
 2. 维护者批准: [chaunceyjiang](#) 给出 APPROVED, 正文只有一句「Thanks」, 随后多次 [/ci run](#) (Buildkite #88036 / #88066 / #88247), 最终由他完成合并——典型的低风险小修复快速通道。
 3. 作者自证测试有效性: 作者 [nicole-lihui](#) 在评论中写道「Tested both the non-streaming and streaming paths, including the added and done events. Both regression tests fail on the old code, and all 94 tests across the two files pass with the fix.」——这句话同时覆盖了「新测试在旧代码上确实红」与「修复后文件内全绿」两个关键信息, 弥补了缺少 reviewer 复核测试的空白。
 4. 提交历史反映环境摩擦而非返工: 3 个 commit 中只有 1 个是功能提交, 另外 2 个是 merge main; 其中 [f54db10d](#) 的消息为「chore: merge main for CI build fixes」并带有 Codex 协作者标记, 说明反复的 CI 触发 / 失败来自主干 CI 环境问题, 而不是本 PR 逻辑被反复推翻。
- 流式与非流式两条路径必须同步修复 (testing): 两处字面量同步改为 "find_in_page", 两个新测试在旧代码上均失败、修复后两文件共 94 个测试全通过。
 - fork PR 缺少自动化深度 review (other): 代码质量主要依赖作者自证与维护者信任, 维护者 [chaunceyjiang](#) 以「Thanks」批准并手动多次触发 CI。
 - CI 反复失败导致多次 merge main (other): 合并动作由 CI 环境问题驱动, 而非代码逻辑返工, 最终通过并合并。

风险与影响

- 风险:
 1. 双路径一致性风险 (本次已同步, 但结构上易再次漂移): 非流式的 `_parse_browser_tool_call` (harmony.py) 与流式的 `emit_browser_tool_events` (streaming_events.py) 是两套独立 if-elif 分支, action 字面量、字段来源都各写一遍。本次两处都改对了, 但同样的 `browser.search` / `browser.open` 分支仍存在重复定义, 未来 SDK 若再次调整枚举 (如 `openai-python` 从 2.16.0 到 2.17.0 就变动过 `response_function_web_search` 的定义), 仍需人工保证两处同步。

1. 参数取值不一致可能仍导致语义缺失：非流式路径构造 `url=f"cursor:{browser_call.get('url', '')}"`，流式路径构造 `url=f"cursor:{parsed_args.get('cursor', '')}"`。但新增的非流式测试输入是 `{"pattern": "vLLM", "cursor": 42}` 且只断言了 `action.pattern`，没有断言 `action.url`；流式测试则断言 `url == "cursor:42"`。这意味着当 Harmony 真的传 `cursor` 时，非流式路径可能产出空的 `cursor` URL，而现有测试无法拦住这类偏差——这是一个尚未被本 PR 覆盖的潜在正确性问题。
2. 兼容性方向：修复把 `"find"` 换成 `"find_in_page"` 是单向收紧到 SDK 规范。如果某些下游客户端此前依赖旧的字面量（实际上旧值会被 SDK 拒绝，无法真正透出），不会出现行为回退；但若部署环境使用极旧版本的 OpenAI SDK（其 `ActionFind` 仍接受 `"find"`），需要确认类型注解是否兼容——PR 未对该版本边界做说明或约束。
3. 测试层级：新增用例是转换层 / 事件构造层的单元测试，能精确定位回归，但没有覆盖到 Responses HTTP 端点的端到端路径（请求 → SSE 事件流 → 客户端 SDK 反序列化），因此对「SDK 能否成功解析 vLLM 输出」这一最终目标只能做间接保证。- 影响：
 1. 对用户：直接修复了使用 Harmony 系模型（如 `gpt-oss` 类支持 `browser.find` 的场景）时 Responses API 报 `Pydantic ValidationError` 的问题，同时解除了流式与非流式两种调用方式的封锁；受影响用户此前会拿到错误而不是 `web_search_call` 输出项或事件流。
4. 对系统：改动局限在 `vllm/entrypoints/openai/responses/` 目录下的两个转换函数，不涉及调度器、KV cache、模型执行或编译路径，运行时行为除该 `action` 的 `type` 字面量外完全不变，回归面很窄。
5. 对团队：这是前端协议对齐类修复的又一个样本——vLLM 的输出结构必须紧跟 OpenAI SDK 的 Pydantic 定义。本次由于缺少 fork PR 的自动化 review，测试真伪主要依赖作者自证，提醒维护者在这类「SDK 契约」补丁上应关注是否有端到端或 SDK 反序列化层面的验证。- 风险标记：SDK 枚举强耦合，双路径重复实现，非流式 `url` 字段可能为空，仅单元级测试

关联脉络

- PR #56369 [Frontend][last/N] Move all non-OpenAI content out of the OpenAI folder.: 同属 `entrypoints/openai` 目录的前端重构线，涉及 `vllm/entrypoints/openai` 下的路由与对应 `tests/entrypoints/...` 测试搬迁；理解 Responses 目录当前结构演进时会与该重构发生文件路径层面的冲突。
- PR #56260 [Bugfix][Rust Frontend][Renderer] Align DeepSeek tool-call arguments with deepseek-recipe: 同为「工具调用参数 / 类型与上游规范对齐」的修复：Rust 前端把 DeepSeek tool-call 参数改为原样回退以对齐官方 recipe，与本 PR 把 `browser.find` 的 `action type` 对齐 OpenAI SDK 枚举属于同一类协议契约修复脉络。